IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 11 June 2024, accepted 22 June 2024, date of publication 2 July 2024, date of current version 29 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3422211

SURVEY

Decentralized and Distributed Learning for AIoT: A Comprehensive Review, Emerging Challenges, and Opportunities

HANYUE XU[1,2], KAH PHOOI SENG[1,3], (Senior Member, IEEE), LI MINN ANG[3], (Senior Member, IEEE), AND JEREMY SMITH[2], (Member, IEEE)

[1]XJTLU Entrepreneur College (Taicang), Xi'an Jiaotong–Liverpool University, Taicang 215400, China
[2]Department of Electrical Engineering and Electronics, University of Liverpool, L69 3BX Liverpool, U.K.
[3]School of Science, Technology and Engineering, University of the Sunshine Coast, Petrie, QLD 4502, Australia

Corresponding author: Kah Phooi Seng (Jasmine.Seng@xjtlu.edu.cn)

ABSTRACT The advent of the Artificial Intelligent Internet of Things (AIoT) has sparked a revolution in the deployment of intelligent systems, driving the need for innovative data processing techniques. Due to escalating data privacy concerns and the immense volume of data produced by IoT devices, decentralized and distributed learning methods that are rapidly replacing traditional centralized learning play a pivotal role. As AIoT systems become increasingly ubiquitous, the accompanying computational and storage demands necessitate a departure from conventional paradigms towards more scalable, distributed, and decentralized architectures. This paper delves into the background of AIoT, with a particular focus on the evolution of distributed and decentralized learning mechanisms that operate without the need for centralized data collection, thus aligning with the General Data Protection Regulation (GDPR) for enhanced data privacy. The various distributed and decentralized learning strategies are the focus of this paper that facilitate collaborative model training across multiple AIoT nodes, thereby not only improving the performance of the AIoT system but also mitigating the risks of data concentration. The review further explores the adaptability of AI algorithms in these distributed settings, assessing their potential to optimize system performance and learning efficacy. The paper concludes with some use cases and lessons learned for decentralized and distributed learning in various AIoT areas.

INDEX TERMS Artificial intelligent Internet of Things, distributed learning, split federated learning, decentralized learning, artificial intelligence, graph-based learning.

I. INTRODUCTION

In recent years, with the rapid development of the Internet of Things (IoT) and the massive growth of data generated by devices, artificial intelligence (AI) algorithms have been proposed to empower IoT systems with the ability to process and analyze data to provide more accurate and extensive services and decisions. The vast amount of data generated in IoT systems also provides an excellent opportunity for training AI models. This integrated technology combining AI and IoT systems is known as the artificial intelligent Internet of Things (AIoT), which has achieved unprecedented success in areas such as smart power grids [1], transportation [2], and smart cities [3]. Artificial intelligence algorithms, especially in the field of deep learning, bring highly accurate decision-making and analysis to the service. However, its huge computing overhead and a large number of data storage resource requirements have also become a challenge for AIoT. Traditional distributed learning plays a crucial role in AIoT, which can parallelize data and models to distribute in different edge devices in the AIoT system, reducing the computing and storage pressure on a single AIoT device.

The General Data Protection Regulation (GDPR) is the European Union's privacy regulation for personal data, and the protection and supervision of personal information has reached an unprecedented level. Collected data from AIoT

The associate editor coordinating the review of this manuscript and approving it for publication was Byung-Seo Kim.

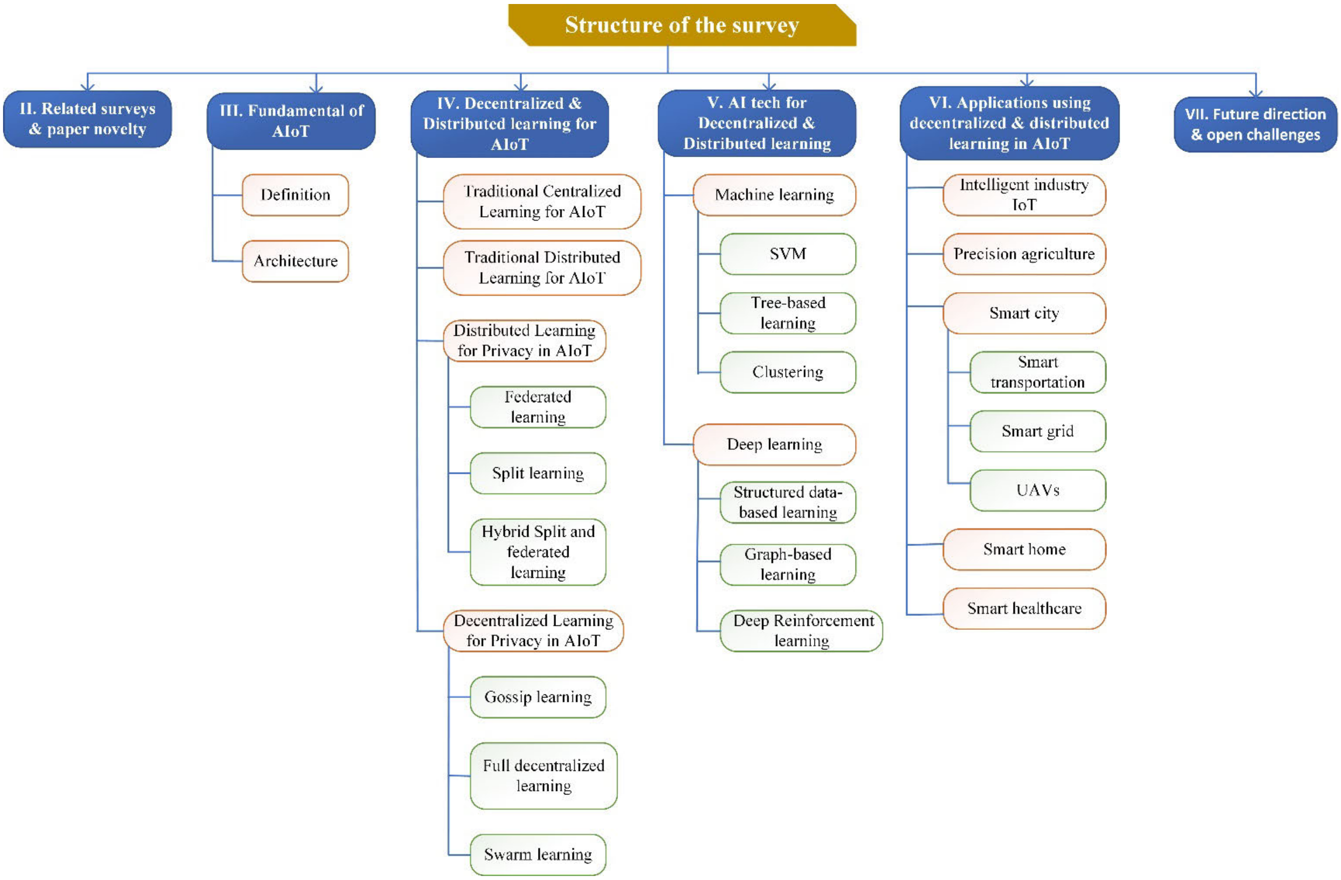
Structure of the survey
II. Related surveys & paper novelty
III. Fundamental of AIoT
Definition
Architecture
IV. Decentralized & Distributed learning for AIoT
Traditional Centralized Learning for AIoT
Traditional Distributed Learning for AIoT
Distributed Learning for Privacy in AIoT
Federated learning
Split learning
Hybrid Split and federated learning
Decentralized Learning for Privacy in AIoT
Gossip learning
Full decentralized learning
Swarm learning
V. AI tech for Decentralized & Distributed learning
Machine learning
SVM
Tree-based learning
Clustering
Deep learning
Structured data-based learning
Graph-based learning
Deep Reinforcement learning
VI. Applications using decentralized & distributed learning in AIoT
Intelligent industry IoT
Precision agriculture
Smart city
Smart transportation
Smart grid
UAVs
Smart home
Smart healthcare
VII. Future direction & open challenges

FIGURE 1. **Roadmap for this survey paper.**

will often contain personal privacy information, such as the camera on the door lock in the smart home or the body status information collected by smart health devices. Therefore, the traditional centralized learning and data parallel distributed learning on the cloud have data privacy vulnerabilities and are no longer suitable for AIoT model learning. It is replaced by emerging distributed and decentralized learning paradigms based on data privacy, such as federated learning, swarm learning, and split learning. The emerging distributed learning solves the challenge of collaborative learning between different edge nodes in AIoT without sharing data by designing model parameter aggregation algorithms. As a subset of distributed learning, decentralized learning removes the central server that is easy to leak data privacy, and only requires communication between edge nodes to complete the training and update of AI models. It has been applied to scenarios requiring higher data privacy, such as the Internet of Medical (IoM) [4] and the Internet of Drones (IoD) [5].

As the decision brain in AIoT, the applicability of AI algorithms in distributed and decentralized learning frameworks is particularly critical. As a result, researchers are working to propose a number of different distributed and decentralized learning frameworks to support more machine learning and deep learning models. Deep learning models, graph-based learning models, reinforcement learning models, and traditional machine learning models such as decision trees, SVM, clustering, etc., have been extended frameworks for collaborative learning without sharing data in the AIoT field. In addition, AI algorithms can also be used as an optimization solution to solve the challenges of distributed and decentralized learning. For example, reinforcement learning can optimize the bias of non-IID data to federated learning by intelligently selecting client devices to participate in each round of federated learning [6].

A. CONTRIBUTIONS OF THIS SURVEY

In this survey, we focus on the emerging fusion of distributed and decentralized learning paradigms with the artificial intelligent Internet of Things. This article will discuss how distributed and decentralized learning can be effectively used to address the challenges of data silos in GDPR-affected AIoT and how artificial intelligence algorithms can adapt and empower architectures for distributed and decentralized learning. For ease of discussion, we identify and classify the content of these emerging convergence paradigms into the basic framework of AIoT, the evolution and challenges of AIoT machine learning architectures (e.g., federated learning, split learning, swarm learning), and the empowered and adaptation of AI algorithms to distributed and decentralized learning, shown in Figure 1. The main contributions of this paper are summarized as the following points:

- This paper focuses on the latest research work by searching databases such as IEEE Xplore and Scopus, covering more than 170 reference papers, and by means of summary tables in different sections to help researchers more clearly understand the latest technologies of the current fusion paradigm, existing challenges, and research opportunities in the field.
- The review covers emerging paradigms such as split federated learning and personalized federated learning use cases and their optimization directions. Compared to the field of distributed learning in IoT, there are many more papers using the emerging distributed paradigm of federated learning discussions. We attempt to provide balanced coverage of different distributed and decentralized learning.
- This paper covers a wide range of AI-enabled technologies in distributed and decentralized learning. The deployment, optimization, and enablement of machine learning, deep learning, graph-based learning, and reinforcement learning architectures in AIoT systems are analyzed and discussed.
- This review starts from the basic architecture of AIoT and analyzes the application methods of distributed and decentralized learning in its different layers. Different optimization schemes based on the limitations of AIoT architecture are discussed. Finally, we propose future research directions in this emerging field and identify some open challenges that may stimulate research thinking.

B. OUTLINE OF THIS SURVEY

The remainder of this article is organized as follows. Section II introduces the recent related works in this area and emphasizes the novelty of our article. Section III introduces the basic architecture of AIoT and discusses the functions of distributed and decentralized learning in different layers. Section V provides a comprehensive overview of the general concepts and frameworks of distributed and decentralized learning in AIoT, compares the advantages and disadvantages between them, and discusses lessons learned and optimized use cases. Section VI reviews specific works and discusses adaptability and enablement in AIoT systems from four broad AI techniques: machine learning, deep learning, graph-based learning, and reinforcement learning, which are also classified as shown in Table 7. In Section VI, we present some relevant use cases and application areas for distributed and decentralized learning. We discuss the challenges to be addressed and the vision for the future in section VII. Finally, Section VIII draws the conclusion of the article. A list of key acronyms and abbreviations used throughout the paper is shown in Table 1.

TABLE 1. List of key acronyms.

Acronyms	Definitions
AIoT	Artificial intelligence Internet of Things
IoT	Internet of things
AI	Artificial intelligence
Non-IID data	Non-independent and identically distributed data
FL	Federated learning
SL	Split learning
SFL	Hybrid split and federated learning
DML	Distributed machine learning
DFL	Decentralized federated learning
CFL	Centralized federated learning
PFL	Personalized federated learning
ML	Machine learning
PCA	Principal component analysis
SGD	Stochastic gradient descent
SVM	Support vector machine
DT	Decision Tree
CNN	Convolutional Neural Network
DNN	Deep Neural Network
RNN	Recurrent Neural Networks
LSTM	Long-short Term Memory
GAN	Generative Adversarial Networks
DRL	Deep reinforcement learning
DQN	Deep Q-Network
MLP	Multi-layer perceptron
UAV	Unmanned Aerial Vehicle

II. RELATED SURVEYS AND PAPER NOVELTY

The utilization of distributed and decentralized learning in the context of the Internet of Things (IoT) is advancing, with numerous scholars examining existing work from various viewpoints and providing insights into the future trajectory of this field, as depicted in Table 2. Among the most prominent distributed learning frameworks is federated learning, and recent research has been primarily focused on addressing the privacy challenges associated with federated learning in the IoT domain. For instance, in [7], the authors examine the transition from centralized to distributed learning, compare these two architectures, and propose a taxonomy of federated learning. However, this study lacks a comprehensive review of distributed learning from an IoT perspective. Similarly, [8] conducts a comprehensive survey on the algorithms, frameworks, and technologies implementing federated learning on IoT architecture and discusses the privacy advantages of federated learning over traditional distributed learning. Nevertheless, it lacks an in-depth analysis of the latest federated learning, such as personalized federated learning and fully decentralized federated learning.

References [9] and [10] offer a comprehensive review of federated learning for these two categories, complementing the previous review. Yet, the examination of federated learning in the context of AIoT represents only one facet

TABLE 2. Summary of related works on reviewing distributed and decentralized learning.

Ref.	Years	Contribution	Limitation and scope for improvement	AIoT?	Main focus
[7]	2020	This paper extensively studies federated learning architecture, design, and deployment and proposes a taxonomy of emerging aspects.	Only research on federated learning in the IoT field is examined, and the latest research is not included. It can be improved to investigate more decentralized federated learning frameworks.	✓	FL
[8]	2021	This paper conducted a comprehensive survey on the algorithms, frameworks, and technologies that implement federated learning on IoT architecture and discussed the privacy advantages of federated learning.	A review of the latest frameworks for federated learning is not detailed enough. It can be improved by summarizing the latest frameworks, such as personalized federated learning.	✓	FL
[9]	2023	This paper summarizes the existing methods of DFL and interprets DFL from different architectural perspectives.	Can be improved by discussions on detailed description of the DFL early AIoT field of research.	✓	DFL
[10]	2022	Personalized federated learning is reviewed in detail, and existing research is classified according to personalized techniques.	It is a supplement to the Federated Learning Review. It can be improved to review all types of federated learning and focus on personalized federated learning.	✓	FL
[11]	2022	This paper summarizes the latest progress in distributed training and inference of the combination of pervasive computing and artificial intelligence in the field of IoT from the perspective of algorithms and systems.	Only a survey of AI distributed learning concerning resource constraints is conducted, which can be improved to review more of the latest frameworks under resource constraints.	✓	Distributed learning
[12]	2023	This article focuses on the security and privacy challenges posed by distributed machine learning and provides a comprehensive overview of the various defense mechanisms.	It only focuses on distributed learning, which can be improved to review the privacy of decentralized learning systematically.	✓	FL, Distributed learning
[13]	2021	A comprehensive survey of recent DML technologies applied to and supported by wireless communication networks is conducted.	Lack of the latest distributed machine learning architectures such as the latest split learning. It can be improved to explain the opportunities for distributed and decentralized learning in wireless.	✓	FL, Distributed learning
[14]	2021	The extensive research on edge computing in AIoT system architecture is investigated, and the application of edge computing algorithms in AIoT is presented.	Decentralized learning is regarded as the challenging direction of edge computing. It can be improved to investigate the studies of decentralized learning deeply.	✓	Edge computing
[15]	2022	This paper reviews the research on decentralized deep learning and provides a taxonomy for summarizing threat models in Decentralized deep learning.	A brief mention of SL, without much of an overview of IoT. It can be improved to explain the structure of distributed learning in detail.	✓	FL, SL
[16]	2023	The behavior of centralized, distributed, and distributed architectures is discussed from the perspective of Edge nodes.	There is no detailed review of distributed learning architecture and its research and application in IoT, which can be improved.	✓	edge computing (SL, FL)
[17]	2023	Reviews emerging FL, SL, and SFL distributed learning approaches and their applications in healthcare.	Only research in the medical field. It can be improved to review more applications of distributed frameworks in other fields.	✘	FL, SL, SFL
[18]	2022	Progress in federated learning and split learning is reviewed, and the areas where the two approaches combine are mentioned in SFL.	FL is referred to in general terms. It can be improved to provide a comprehensive overview of federal learning.	✓	SFL
[19]	2023	This paper provides a comprehensive review of distributed machine learning, focusing on a summary of current synchronization and aggregation methods and a review of the limitations of current technologies.	Not much has been elaborated about the application areas and limitations of DML in IoT, which can be improved.	✓	FL, SL

of the crucial topics to be investigated in this survey. For example, in [7], the authors investigate the evolution from centralized to distributed learning, compare these two architectures, and propose a taxonomy of federated learning. However, it lacks a comprehensive review of distributed learning from an IoT perspective. Reference [8] conducted a comprehensive survey on the algorithms, frameworks, and technologies that implement federated learning on IoT architecture and discussed the privacy advantages of federated learning compared with traditional distributed learning. However, it lacks the framework to investigate the latest federated learning, such as personalized federated learning, full decentralized federated the previous review. However, federated learning in the context of AIoT is only one part of the important topics we will investigate in this survey. We will also focus on other recent innovations in distributed and decentralized learning algorithms. References [9] and [10] provides a comprehensive review of federated learning.

Existing surveys on privacy issues related to distributed learning have approached the topic from various novel angles. The works in [11], [12], and [13] mainly focus on reviewing existing research on distributed learning, covering federated learning, machine learning, and communication for privacy. The authors of [12] review and discuss the different types of privacy and security attacks based on the distributed paradigm and propose different defense mechanisms to deal with these attacks. The authors in [13] reviewed the applicability, architecture, computational efficiency, and communication cost of distributed machine learning in wireless communication networks, and a variety of countermeasures are introduced to protect the security and privacy of distributed systems. Meanwhile, [11] summarizes the latest progress of distributed training and inference of the combination of pervasive computing and artificial intelligence in the field of IoT. However, these surveys only focus on the general distributed system architecture and do not provide a comprehensive and in-depth summary of the more recent distributed architecture, especially for split learning. Our survey aims to fill the gap by comprehensively reviewing the latest distributed and decentralized learning research at the layer of AI and IoT fusion.

Different from the above papers, the authors of [14], [15], and [16] review distributed and decentralized learning from the perspective of edge computing. Reference [14] investigated existing research on AIoT on edge computing and reviews multiple edge computing algorithms and decentralized learning as a solution to edge computing challenges. From the perspective of multi-access edge computing, [15] outlined the obstacles from the perspective of multi-access edge computing [16] discussed the behavior of centralized, decentralized, and distributed architectures from the perspective of edge services and reviewed performance metrics for these frameworks. Although the above existing papers have reviewed deep learning and distributed learning, they have mainly limited their discussion to federated learning, while other recent decentralized and distributed learning discussions have been less. Our paper will strengthen the investigation and discussion of other recent frameworks at different layers of AIoT and compare the commonalities and differences among these frameworks

Finally, as the latest review, the author conducted an in-depth review of AI-based decentralized learning in [17], [18], and [19]. Specifically, these papers covered the latest decentralized frameworks, such as swarm learning, split learning, and split federated learning. The investigation in [17] provides different frameworks for emerging federated learning, split learning, and federated split learning and compares the advantages and disadvantages between different algorithms. However, decentralized learning in the AIoT field is not the focus of this paper; it only focuses on areas related to smart healthcare. The review content in [18] and [19] is considered to be the most recent survey in our investigation direction, because they review the frameworks of distributed and decentralized learning and their aggregation algorithms from an IoT perspective. However, they do not fully study decentralized learning in the AI-driven IoT framework. In [18], the author proposed an in-depth survey of the framework of decentralized learning, without the constraints under the IoT architecture, such as sensor layer Medina and transport layer; these partitioning strategies have a lot to do with the design and aggregation of decentralized learning. Our paper represents a comprehensive survey of decentralized and distributed learning at all the different layers of the IoT framework driven by AI models.

III. FUNDAMENTALS OF ARTIFICIAL INTELLIGENT INTERNET OF THINGS

Artificial intelligent Internet of Things is an ecosystem of collaborative working between artificial intelligence technology and Internet of Things systems. Among them, IoT is the infrastructure of AIoT, which is able to implement the acquisition of large amounts of sensor data to provide data sources and the operation of the entire ecosystem. AI is the analysis and decision-making core of AIoT, which is responsible for mining potential information from massive data to enhance the perception and decision-making ability of IoT systems and realize intelligent interconnection. Therefore, the main framework of AIoT is based on IoT systems, but there will be some changes.

AIoT architecture utilizes a cloud-edge-end architecture similar to the Internet of Things, which can also be defined as cloud-fog-end architecture. As shown in Figure 2, the end layer contains a large number of interconnected sensors and actuators distributed over a wide area to sense their surroundings and execute the decision instructions received by the AI. The edge layer has multiple edge nodes to expand the computing and storage capabilities of the devices at the end layer, which can process, aggregate, and calculate the collected data locally, reducing the dependence on the central server. The fog layer is located between the edge layer and the cloud server and is usually deployed near the cellular tower to reduce the data transmission distance of the AIoT system and shorten the corresponding time to make decisions as soon as possible. The cloud is the central server, with massive computing, storage, and bandwidth, which can coordinate the edge layer and the fog layer to assist them in making deeper AI decisions. The AIoT layers correspond to distributed and decentralized learning frameworks as follows.

A. END LAYER

As the AIoT system perceives the external hub, the end layer is mainly responsible for collecting the surrounding data and performing small computing tasks or data preprocessing. For example, smartwatches, smartphones, and Raspberry PI devices are all end-layer devices. Devices at the end layer usually have some computing and storage capabilities, but due to their primary task being to collect data and execute decisions, these resources are limited. In order to utilize these computing resources and reduce the network bandwidth of AIoT systems, researchers leverage distributed learning and decentralized learning to collaborate on multiple resource-limited devices to improve the overall learning capability.

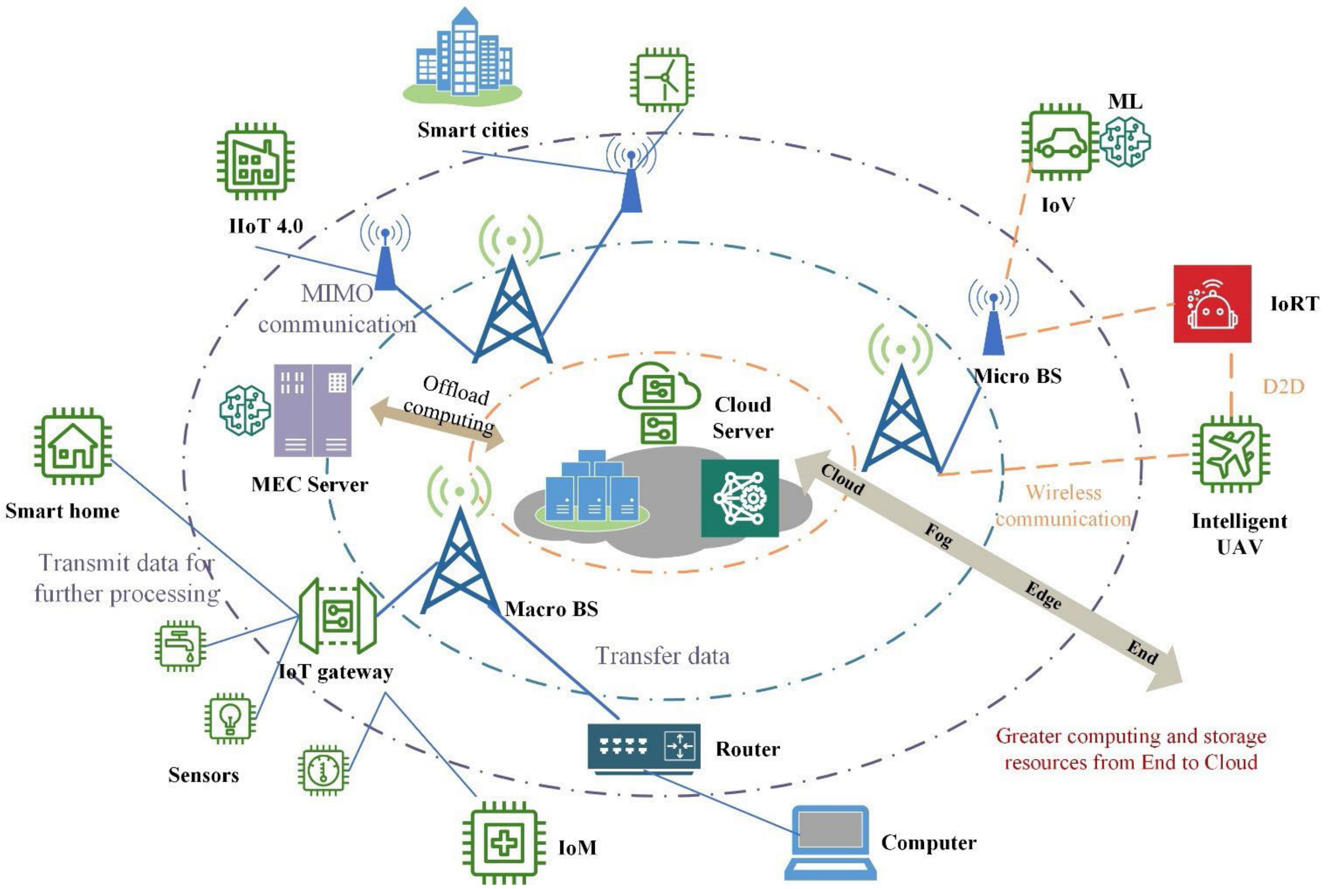

FIGURE 2. Basic architecture of AIoT.

The data collected by the end layer is not always complete and correct, and there may be data missing or outliers. Grammenos et al. [20] proposed a principal component analysis (PCA) method with differential privacy using the emerging distributed framework of federated learning in resource-constrained devices. Briguglio et al. [21] enhanced unsupervised federated PCA by proposing a federated learning version of supervised PCA that is more suitable for end-layer devices with limited memory and improved runtime. Furthermore, Narayanamurthy et al. [22] also studied the introduction of the subspace learning method of federated power method in wireless communication mode to further deal with channel noise and related noise of local sparse data. In this way, the influence of the disturbance data collected by the end layer on the performance of the AIoT system is reduced.

B. CLOUD LAYER

The cloud layer is particularly important in AIoT systems, especially within the framework of the distributed learning paradigm. It can be regarded as the AI decision-making brain in the AIoT system, with massive computing and storage capabilities to provide scalable computing and storage resources for the AIoT system to assist the end layer, edge layer, and fog layer in making deeper intelligent decisions. In distributed learning, the global model of the AIoT system is often trained in the cloud due to its powerful capabilities, so the performance of its AI algorithm is the focus of central distributed learning research. More specifically, the algorithm of model parameter aggregation will affect the final result of distributed learning, and the details will be discussed in Section IV-C. In addition, a variety of different machine learning models have been proposed to build on cloud-based distributed learning, as described in Section V.

C. EDGE LAYER

The edge computing layer is closer to the device side of the end layer and has fast reaction ability, but it cannot cope with a large number of calculations and storage occasions. Edge computing enables end-layer devices to respond quickly without accessing the cloud, reducing transmission overhead and improving fast response times. It is the focus of distributed and decentralized learning and is responsible for data caching, distributed machine learning and collaboration, load balancing, and data privacy protection. In an architecture without clouds, the edge layer can also be built in a decentralized learning framework through the coordination of multiple edge nodes. In addition, the edge layer is responsible for authentication, authorization, offloading, and communication with the cloud. It runs through the AIoT system like a human spine. Therefore, the research on distributed learning and decentralized learning based on edge computing is the most popular.

The ability to process real-time data is the biggest difference between the edge layer and the cloud layer, in addition to being resource-limited. Zhu and Jin [23] proposed a real-time federated system RT-FedEvoNAS to update the global model based on the subnets of each generation of sampling and training without computing through clouds, which accelerated the model convergence and realized real-time data processing. For wearable edge sensors, Nandy and Xhafa [24] proposed Fed-ReMECS, a federated learning model for real-time emotion classification based on multimodal flows. Real-time distributed and decentralized systems that rely on AIoT edge layers can address emergency scenarios with data-sharing constraints, such as health detection of wearable devices, home fire detection, and so on.

D. FOG LAYER

The fog layer is an extension and expansion of the cloud, located in a local area network or gateway, which is not made up of powerful servers but of huge peripherals that are weaker and more dispersed and located outside of a large data center. It contains various edge nodes, which are closer to the edge device than the cloud and have more storage and computing resources than the edge layer. The fog layer is more widely distributed geographically and has a greater range of mobility, which makes it suitable for today's growing number of smart devices that do not require a lot of computing, especially for some real-time and streaming applications that are sensitive to time delays. Similar to edge computing, fog computing is a distributed computing paradigm that reduces the dependence on the cloud through real-time information processing and collaborative learning of multiple nodes.

The fog layer optimizes some aspects of the distributed and decentralized learning framework in the architecture of AIoT systems. Due to its geographic distribution in multiple locations, it is closer to edge devices than clouds, reducing the network traffic required for model parameter transmission in distributed learning. Rajagopal et al. [25] introduced a distributed learning framework based on data privacy, FedSDM, to realize faster real-time processing of intelligent IoT medical data by integrating a cloud-fog-edge framework. Compared with clouds, the fog layer has more nodes that can train the global model collaboratively, thus reducing the aggregation times of distributed learning models and making them converge faster.

Saha et al. [26] proposed a fog-supporting federated learning framework, FogFL, with the fog layer acting as the global model aggregator for each round of communication between the edge and the cloud, and its fog node reducing the energy consumption of edge device communication without affecting the convergence rate of the model. Third, the fog layer can know the operation of each edge node faster than the cloud layer, solve the situation of distributed learning stagnation faster, and optimize the ability to allocate resources [27]. Moreover, on edge nodes with limited resources, the fog layer can also provide storage and computing resources to achieve effective model offloading. Sethi and Pal [28] proposed a federated learning technology based on reinforcement learning, FedDOVe, to calculate the unloading ratio and improve the load balancing of the vehicle internet of things (IoV) by finding the best association between the vehicle edge node and the fog layer.

E. PHASE SUMMARY OF ARTIFICIAL INTELLIGENCE INTERNET OF THINGS LAYERS

The survey work in this section reveals the following key points and insights for the fundamental concept and architectures for AIoT:

- AIoT architecture is characterized by a cloud-edge-end or cloud-fog-edge-end architecture. Distributed learning paradigms are mostly designed around clouds, providing computing and storage resources for the training and computation of AI models in AIoT. However, due to the single-point instability and communication delay problems of the cloud, edge computing is emerging.
- The edge layer is located closer to the terminal device, replacing some computing and communication work of the cloud, helping to deal with real-time tasks, freeing the AIoT system's dependence on the cloud. Therefore, edge computing has become the most common method of decentralized learning.
- Fog computing extends the function of the cloud, providing more computing and storage resources than the edge layer, and its current work is more inclined to optimize resource allocation and reduce communication latency and other auxiliary work.

IV. EVOLUTION OF AIOT MACHINE LEARNING ARCHITECTURE

This section provides a detailed overview of the evolution of machine learning models in AIoT architectures from centralized learning to distributed learning to more recent decentralized learning architectures due to data privacy security concerns, as shown in Figure 3. Moreover, this section also summarizes the emerging distributed and decentralized learning frameworks, as shown in Table 6.

A. TRADITIONAL CENTRALIZED LEARNING FOR AIOT

With the increase of IoT devices, the amount of data that can be collected becomes enormous, and a single device is no longer enough to meet the needs of IoT application scenarios. The use of machine learning models in AI decision-making requires more data volumes to support its services. The centralized learning paradigm is the first architecture to be considered, collecting data from different edge devices to the cloud and building machine learning models. Users can request the cloud through API to get the service trained by

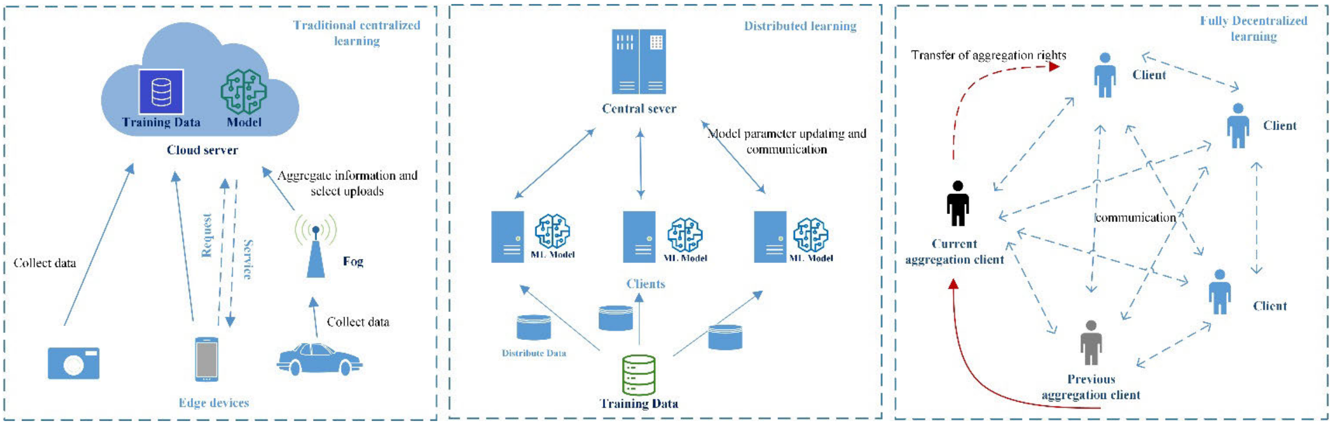
Traditional centralized learning
Training Data
Model
Cloud server
Aggregate information and select uploads
Collect data
Request
Service
Fog
Collect data
Edge devices
Distributed learning
Central sever
Model parameter updating and communication
ML Model
ML Model
ML Model
Clients
Distribute Data
Training Data
Fully Decentralized learning
Transfer of aggregation rights
Client
Client
communication
Current aggregation client
Client
Previous aggregation client

FIGURE 3. Architecture of centralized learning, distributed learning, and decentralized learning.

the cloud model. The left side of Figure 3 demonstrates centralized learning in the IoT framework. The most typical cloud service providers are Amazon Web Services, Google Cloud, and Microsoft Azure, which have large-scale data storage and efficient computing power [29]. However, in the paradigm of centralized cloud computing, the server is far away from the data collection and the end users who need the service, so there is extremely high communication latency. The proposed fog computing paradigm meets the needs of IoT scenarios that require strict communication delay limits, such as the Internet of Vehicles [30] and wearable devices [31]. Vehicle fog of centralized learning, as an example, is shown in Figure 3. The fog node is closer to the data source than the cloud server and can filter, such as by aggregating device messages, to reduce the pressure on the core network.

While this paradigm can collect more data to train deep learning models to produce smarter services, this data may come from the privacy of the users. This data may include confidential company data (e.g., company financial data, hospital patient data) or personal information (e.g., intimate photos, health records), which could lead to a potential risk if compromised by a cloud provider. Data privacy protection in a centralized learning framework can be extended to cloud-based privacy protection because if data in the cloud is compromised, then the privacy of the client will also be violated. The main existing cloud privacy protection systems can be divided into three types: cryptography method, data perturbation, and information concealment.

Cryptography methods mainly study the construction of key management mechanisms, homomorphic cipher schemes, and obfuscation methods, which are suitable for privacy protection and different from traditional data encryption and decryption. The representative technology is homomorphic encryption [32]. It enables the cloud server to calculate the ciphertext data without decrypting it so that the plaintext of the ciphertext is calculated accordingly. Jiang et al. [33] store medical data on a public cloud server framework and train machine learning models using homomorphic encryption techniques. Busom et al. [34] used homomorphic encryption technology to solve the data privacy problem of smart meters. Dowlin et al. [35] proposed the CryptoNets framework to enable learned neural networks to train input data based on homomorphic encryption.

The data perturbation mechanism aims to remove the correlation between different private data, introduce noise into the original data, and achieve privacy protection through data anonymization (e.g., K-anonymity [36], L-diversity [37], T-proximity [38], etc.), and prevent cluster analysis, crowdsourcing computing, deep learning, and other big data analysis. The most common approach is to adopt differential privacy (DP) [39]. The purpose of differential privacy is to ensure that any individual in the data set or out of the data set has little influence on the final published query results. Its mathematical definition can be expressed as $\Pr[M(x) \in S] \leq e^{\varepsilon} \Pr\left[M\left(x'\right) \in S\right]$, where the smaller ε is, the higher the level of privacy protection. Differential privacy protection can be realized by adding appropriate interference noise to the return value of the query function. The commonly used techniques are the Laplacian mechanism and the exponential mechanism. For example, Rubinstein et al. [40] proposed a differential private support vector machine (SVM) learning mechanism in which Gaussian noise was added to the output classifier in the model, and the prediction results were similar to ordinary SVM. Zhang and Zhu [41] designed a dynamic differential privacy mechanism based on ADMM to analyze the impact of privacy on the accuracy of the model's output results. The differential privacy protection method plays a very important role in the centralized learning framework, so it is also extended to the edge computing distributed learning framework [42]. This will be discussed in detail later.

The method of information concealment is used to protect metadata and transfer metadata in a changing form, and the corresponding restoration control parameters should be separated from the information itself for storage and transmission. Digital watermarking is one of the representative methods, which directly imprints some identification information (that is, digital watermarking) into the digital carrier (including multimedia, documents, software, etc.) but does not affect the use value of the original carrier and is not easy to be perceived or noticed by human perceptual systems (such as visual or auditory systems). Cao et al. [43] proposed a heterogeneous cloud platform based on CPU and FPGA to achieve high scalability and generalization of digital watermarking programs to make cloud computing more private. Digital watermarking technology can also be combined with other privacy technologies. For example, Dong et al. [44] used homomorphic encryption technology and digital watermarking technology to protect the copyright of images stored in the cloud server and to safely detect the dishonest behavior of the cloud. Moreover, to solve the response delay problem of watermarking in cloud computing, Cheng et al. [45] introduced edge computing technology and proposed an image digital watermarking method combined with homomorphic encryption, which is superior to the original privacy protection scheme.

While there are already a number of privacy technologies that address data privacy in centralized learning frameworks, as summarized in Table 3, the data collected comes from edge devices in different geographies and is, therefore, subject to different data privacy regulations. Moreover, there are structural drawbacks to centralized learning. The first is the response delay of the cloud server because the data needs to be transmitted for a long time to reach the cloud. The second is the cost of data transmission because a lot of data upload and offload takes a certain amount of time, and the network transmission is not free. To overcome these challenges, the machine learning in the AIoT framework can be changed on a physical level.

TABLE 3. Cloud privacy method for centralized learning.

Privacy type	Protect objects	Methods	Study
Cryptography method	Computing privacy	homomorphic encryption	[33] [34] [35]
Data perturbation	Data privacy	K-anonymity, L-diversity, T-proximity, Differential privacy	[40] [41] [42]
Information concealment	Communication privacy	Digital watermarking, the fusion of other privacy techniques	[43] [44] [45]

B. TRADITIONAL DISTRIBUTED LEARNING FOR AIOT

Compared with traditional centralized learning, distributed learning can solve the problem of the computational amount of the machine learning model being too large and the training data pair and the model is too large. Distributed learning utilizes data distributed across multiple devices for machine learning or deep learning to improve model performance and scale to larger training data and larger models. As shown in the middle of Figure 3, the training data is divided into disjoint data fragments and sent to each client. The client conducts model training locally and sends the gradient or model parameters to the central server, which aggregates the received parameters. Both synchronous and asynchronous distributed stochastic gradient descent (SGD) algorithms are suitable for distributed learning. There are three paradigms for distributed learning architecture, with different algorithms and models adapting to different paradigms, but the basic framework is the same, as shown in Figure 4. These paradigms are as follows:

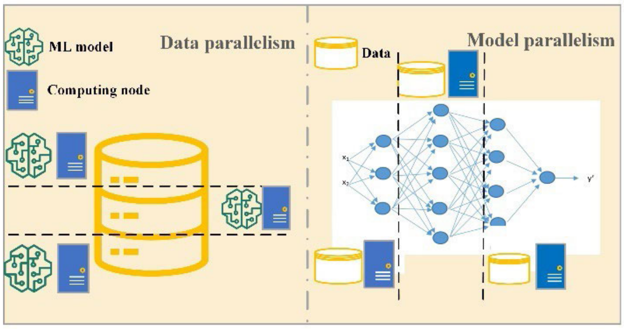

FIGURE 4. Data and model parallelism in distributed learning.

1) DATA PARALLELISM

The training data is distributed across multiple clients, and the same model is trained locally on the client. When each compute node completes a model update, the model parameters calculated by each client are transferred to the central server for aggregation operations to update the final model. The main purpose of data parallelism is to avoid the limitation of how much training data cannot be stored in a single client and to reduce the time cost of model computation.

2) MODEL PARALLELISM

A model is divided into parts (e.g., layers of a DNN model are divided into different parts) and distributed among different clients. The results of the calculation are transferred between clients to generate the final model. This paradigm is used to address the memory capacity limitations of machine learning models.

3) HYBRID PARALLELISM

The hybrid parallel paradigm combines the properties of model parallelism and computational parallelism. Data parallelization is used between clients, and model parallelization is used for a single client.

The architecture of distributed learning determines that it has privacy issues due to data sharing among multiple clients and the synchronization of parallel training of models. Therefore, there have been some advances in distributed learning privacy protection. For example, Xie et al. [46] proposed a privacy-protecting near-end gradient algorithm for a model of asynchronously updated distributed learning tasks. In view of the geographical distribution environment of data, Vulimiri et al. [47] proposed to protect data privacy by restricting the location of data transmission. Zhou et al. [48] formulated the special privacy of geographically distributed data centers as an optimization problem with multiple constraints and proposed a solution on Azure.

C. DISTRIBUTED LEARNING FOR PRIVACY IN AIOT

Although there has been a lot of work on the use of privacy protection technology to protect the privacy of distributed learning, it still cannot meet the privacy provisions of many laws and regulations; the fundamental reason is that the central server has too much control over data and computing. As a result, recently proposed new distributed learning architectures such as federated learning (FL), split learning (SL), and hybrid split federated learning (SFL) have been popular in terms of data privacy. Compared with basic distributed machine learning, they have absolute control over the data, and the central server cannot directly or indirectly manipulate the data on the compute node, and the compute node can stop computing and communication at any time and exit the learning process. In this section, we will explore the distributed strategies and concepts underlying FL, SL, and SFL.

1) FEDERATED LEARNING

Federated learning is a distributed machine learning framework with privacy protection, which aims to allow decentralized participants to collaborate on machine learning model training without sharing private data with other participants. In addition to decentralized local users, federated learning participants can also be multiple enterprises facing the dilemma of data silos, where they have independent databases but cannot share them with each other. Federated learning ensures the security and privacy of data by designing encrypted parameter passing instead of remote data transmission during training. Google [49] first applied federated learning to Gboard (Google Keyboard), combined user terminal devices, trained local models using local data of users, and then aggregated and distributed model parameters in the training process to achieve the goal of accurately predicting the next word. As shown in Figure 5, the key steps for basic federated learning are as follows:

1) **Client selection:** The central server samples from a set of clients that meet the eligibility requirements. The conditions selected may include whether the device is idle or a channel condition for transmission. For example, the mobile phone is selected as the client. To avoid affecting device users, the selected device must be idle and have network transmission conditions.
2) **Model initialization and transfer:** The central server selects the appropriate machine learning model based on the required task and initializes the model parameters. The communication rounds with the client are then set up, and the initial model is transferred to the client.
3) **Client local computation and update**: Each selected device computes locally the updates to the model received from the central server by executing a training program.
4) **Model aggregation and selection**: The central server collects the aggregate of updated device models and computes the next version of the global model. The machine learning process is implemented by minimizing a loss function that calculates updates based on each batch of data on the client t. Therefore, the final result can be calculated by using the weighted average method for the calculated loss function $f_i(w)$ on each client, i.e.,

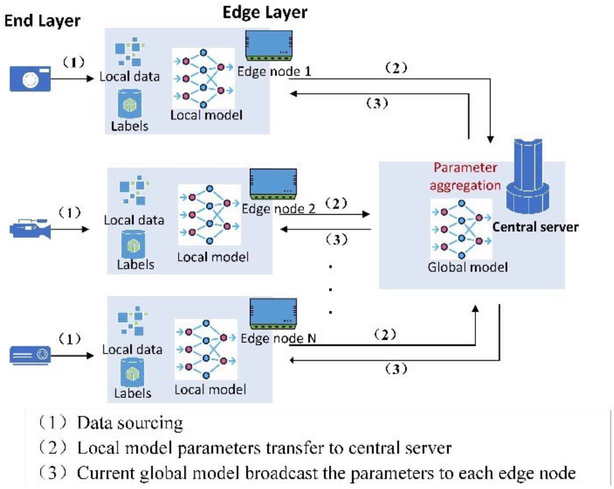

FIGURE 5. Architecture of federated learning for AIoT.

$$F_k\,(w) = 1/n_k \sum\nolimits_{i\in P_k} f_i(w) \tag{1}$$

$$f\,(w) = \min(\sum\nolimits_{k=1}^{K} n_k/nF_k\,(w)) \tag{2}$$

where n_k is the number of batches on each client and $f_i(w)$ is the updates calculated based on the data of each batch on the client. After the central server aggregates the parameters and completes the global model push, it broadcasts the new global model again to the selected client and repeats the 3) and 4) steps until the model converges.

FedSGD [50] algorithm has provided the basic architecture for federated learning, but because edge devices have absolute control over themselves, there are still some problems in centralized federated learning that need to be optimized. The aggregation algorithm is the focus of centralized federated learning steps. Table 6 compares the aggregation optimization algorithms commonly used at present. For the different challenges faced by federated learning, the classic optimization algorithm often used as a baseline is as follows:

2) SYSTEMS HETEROGENEITY

Differences in communication and computing power exist between devices. In FedAvg [50], selected clients train the same epoch locally. Although the author points out that raising the epoch can effectively reduce communication costs, with a larger epoch, many devices may fail to complete the training on time, and those clients that fail to complete the training will be dropped. Either dropping this part of the client model directly or using this part of the unfinished model to aggregate will have a bad effect on the convergence of the final model and cause the parameter deviation of the model. To alleviate this problem, Scaffold [51] proposes a strategy of constantly correcting the model update direction in the local training stage, that is, adding a patch item at each update step to prevent the update direction from going wrong. It also reduces traffic by maintaining a control variate in each round. This control variable can be adjusted according to gradient changes in the global model, making communication between each device more compact. The algorithm not only overcomes the problem of device heterogeneity using the variance reduction technique but also reduces the communication cost without sacrificing model performance. Both algorithms use synchronous methods to implement federated learning, but global synchronization is difficult due to limited computing capacity and battery time. FedAsync [52] is an asynchronous federated optimization algorithm that introduces a hybrid hyperparameter that adaptively controls the trade-off between convergence speed and variance reduction to solve the error strategy of asynchronous algorithms. Asynchronous training is relatively more flexible and shows the advantage of being more general in the case of discrete and heterogeneous delays in the system without waiting for other devices to participate in global aggregation.

3) STATISTICAL HETEROGENEITY

The data of different users cannot be distributed independently. Since the parameters of the local client training model will not only deviate greatly from the parameters trained by other clients but also from the parameters of the server global model, it is necessary to ensure that the update of the local client model cannot deviate too much from the server global model. FedAVGM [53] makes an improvement on the basis of FedAvg. It introduces momentum to update weight w on the server side, which improves the training effect of the FedAvg algorithm on non-independent, equally distributed data. In addition, this paper also proposes a non-IID data generation method in FL based on Diliclet distribution. FedProx [54] solves this problem by adding a proximal term to the optimization goal, which causes the model to pay more attention to local model weights that are close to the global model weights when updated. In this way, the negative effects of non-independent data and device heterogeneity can be reduced, and the model performance can be improved. Unlike the FedProx algorithm, the core idea of FedNova [55] is to use a normalized average method to eliminate target inconsistencies while maintaining fast error convergence. In each iteration, local training is performed on each device, and the local model parameters are normalized. Then, the normalized parameters are sent to a central server. A central server collects parameter updates from all devices and normalizes them to reduce target inconsistencies. The server then distributes the updated model parameters back to the devices so that training can continue in the next iteration. The advantage of FedNova is that it is able to eliminate target inconsistencies due to data heterogeneity while maintaining fast error convergence.

4) LIMITED COMMUNICATION

The communication delay between all compute nodes is high, and the network stability is poor, which requires a high fault tolerance mechanism. FedBoost [56] solves the problem of communication limitations by combining federated learning with an Ensemble algorithm. Unlike working with gradient compression, it reduces the cost of server-to-client and client-to-server communication. In addition to optimizing the communication limits of federated learning, this algorithm also ensures the privacy of the transmission process. Generally, the algorithm completes SGD locally on the client to improve communication efficiency. FedBR [57] uses label-agnostic pseudo-data to improve the performance of heterogeneous data from the perspective of data heterogeneity. Because FedBR does not require labeling of pseudo-data or large pseudo-data sets, communication costs are reduced. Communication efficiency has always been a major focus in the optimization of federated learning. Therefore, many algorithms will take into account the communication efficiency of the algorithm while optimizing other problems, including [50], [51], [53], and [54].

In addition to the three major shortcomings mentioned above, federated learning also extends the problem of model heterogeneity due to the different task requirements of each client. To address these heterogeneity challenges, an effective approach is to personalize the device, data, and model levels to mitigate heterogeneity and obtain high-quality personalized models for each device, i.e., personalized federated learning. To address these heterogeneity challenges, personalized federated learning is proposed to mitigate heterogeneity and obtain high-quality personalized models for each device by personalizing the device, data, and model levels [58].

Mansour et al. [59] implemented personalized prediction of the model by incorporating contextual features. The data interpolation technology has the characteristics of low communication cost and data security protection. Transfer learning [60] can also serve as a framework for personalized federated learning to build personalized models for new users by aggregating models from different environments and fine-tuning them. Multi-task personalized federated learning [61] aims to use the commonality and difference between tasks to learn together and determine which layer should be shared by discriminating the correlation between tasks so as to improve the generalization ability of the model. Meta-learning [62] is a learning-to-learn learning method, which makes the model results universal to all kinds of tasks through training. Strategies that are insensitive to tasks and have strong generalization ability can be learned through meta-learning, which is suitable for application in personalization. FedDK [63] algorithm introduced knowledge distillation technology to a federated learning framework and circulated knowledge between each client to form a personalized model for each group by transferring knowledge distillation from the local model to global distillation. In order to solve the non-IID problem, FedRep [64] proposed to use the base layer to learn the dimensionality reduction representation of global feature representation between data, based on the concept that data usually has global feature representation and the statistical heterogeneity between clients or tasks is mainly concentrated on labels. In order to alleviate the impact of non-IID on model training, personalization is realized by using a personalization layer as the unique local head of each client. Hanzely and Richtárik [65] proposed a new gradient descent method, LLGD, that seeks trade-offs between global and local models so that each device can individually train private data without communication. Each client should learn not a single global model but a mixture of the global model and its own local model.

5) SPLIT LEARNING

Split Learning (SL) is a distributed model training scheme. The core idea is to split the network structure, with each device only retaining a part of the network and the sub-network structure of all devices, forming a complete network model. In the training process, different devices only perform forward or reverse calculations on the local network structure. The results are passed on to the next device [66]. As with federated learning, split learning ensures that local data does not leave the local device and, therefore, effectively reduces the risk of data privacy leakage.

Depending on the location of the data and labels, the SL framework comes in a variety of configurations, as shown in Figure 6. Simple vanilla split learning is the most classic architecture, which consists of only one client node and one server node. Take the DNN model as an example. The DNN model has divided into two subnetworks: the client node retains the local training data and the subnetwork with the input layer, and the server node is responsible for receiving and calculating the gradient of the second submodel according to the forward results and labels of the client node and feeding back to the client [67]. However, the label information in the server may be shared with the client, so there will be data privacy issues. The U-shaped structure solves this problem. In this architecture, both the client and the server hold the final layer of the neural network, which communicates twice through forward and backward calculations so that the client can locally calculate the gradient based on the complete forward calculation result and label [68]. The latest architecture of distributed learning is a vertically partitioned data architecture for multi-party heterogeneous data scenarios [69]. Multiple clients compute part of the model of their own local training data, and the server's cut layer is responsible for concatenating and completing the remaining forward calculation on the server side. The reverse calculation process is the opposite of the forward calculation. However, in a vertically partitioned data architecture, the client's label information is shared with the server.

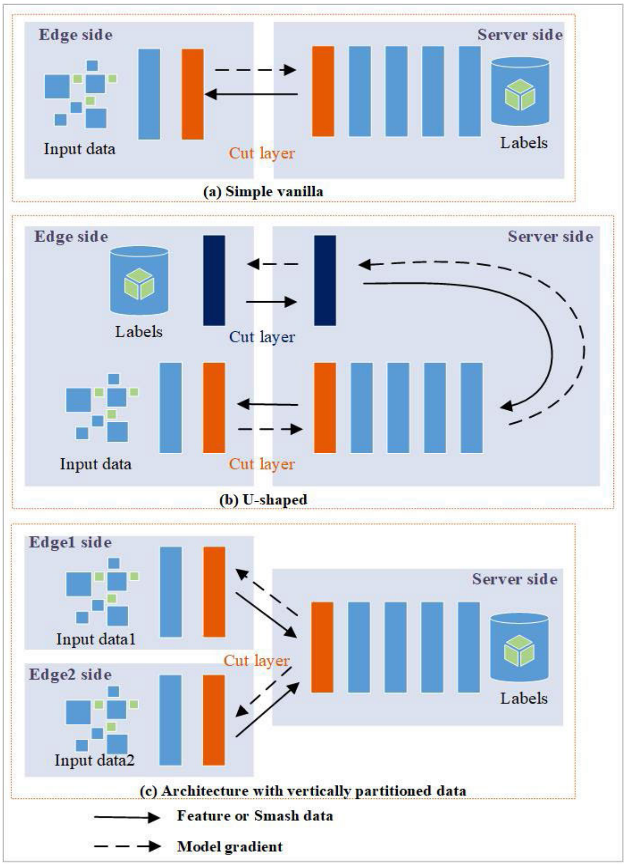

FIGURE 6. Architecture of classical split learning.

Based on the split learning of the vertical data partitioning architecture, there are three variants: Extended vanilla SL, SL for multi-task, and multi-hop SL. [70]. The Extended vanilla SL architecture is a multi-party heterogeneous data Client that cuts the layer and sends it to another client before sending it to the Server for processing. The SL for multi-task architecture is used to solve different supervised learning tasks. The multi-modal data from different clients are spliced in the cut layer of forward calculation in the client and sent to multiple servers, respectively, and each server completes a model training. The multi-hop SL framework organizes multiple clients and a Server training model in a sequential manner. The first Client sends the cut layer forward to the second Client, and the last client sends the cut layer forward to the Server to complete the remaining forward calculation.

Split learning has two advantages over federated learning; one is that neither the edge device nor the cloud has full control over the complete model and is, therefore, more secure. Second, it can realize the heterogeneity of the edge-cloud model, which is more conducive to the training of low-resource equipment. SL solves the problem of edge device computing resource limitation in FL, but it also leads to higher communication costs. Because the model is split into two sub-models, frequent forward and backward computational interactions between the sub-models can lead to additional communication overhead. Singh et al. [71] compared the communication costs of federated and split learning and showed that SL is more efficient than FL in terms of communication efficiency as the parameters of the client and model are increased. However, as the amount of local data increases, federal learning performs better. In addition, Gao et al. [72] built FL and SplitNN frameworks on the actual IoT edge device Raspberry Pi to compare their learning performance and device computing overhead. The experimental results show that SL has better learning performance than FL under unbalanced data distribution, while federation has better learning performance under non-IID data distribution. In the case of limited communication bandwidth, FL has a lower communication overhead than SL. Table 4 shows distributed learning options in different scenarios.

TABLE 4. Distributed learning options in different scenarios.

Scenarios	SL	FL
Model parameter	large	Middle or small
Clients	large	Middle or small
Training data	Middle or small	large
Data distribution	Unbalanced data	Extreme non-IID data

In order to solve the problem of excessive communication cost of split learning, there has been much research to optimize SL. Since the underlying logic of SL is to exchange frequent interactions between clients for low computing overhead on edge devices, balancing computing overhead and communication frequency is the key to SL optimization. Chopra et al. [73] proposed that AdaSplit reduces the communication cost of algorithms by reducing the communication load and communication frequency. AdaSplit's framework divides the training into local and global stages. In the global stage, the server does not pass the model gradient to the client, and the client needs to use the local loss function for training so as to eliminate the client's dependence on the server gradient and reduce the communication load. The framework also introduces a UCB policy to prioritize communication with clients who need to improve performance, thereby reducing the frequency of communication. The authors also propose the 3C-score framework, which includes the computational overhead, communication cost, and collaboration performance of SL to help evaluate later split learning algorithms. Furthermore, Ayad et al. [74] introduced an autoencoder and an adaptive threshold mechanism to an IoT system with limited resources. In terms of split learning, AE was added to the neural network to reduce the amount of data sent by the client in the forward computation, and an adaptive threshold mechanism was added to track the gradient to reduce the updated amount of post-feedback communication.

Most of the current research aims to reduce the load of split learning communication by compressing the transmitted information. Zheng et al. [75] compared different information compression methods on SL, including cut layer size reduction, top-k sparsification, quantization, and L1 regularization. A stochastic top-k sparsification method is proposed, which is superior to other sparsification methods in model convergence, generalization, and compression ratio. It reduces the communication cost of SL while ensuring the accuracy of training. Hsieh et al. [76] proposed a quasi-orthogonal batch compression method based on circular convolution for SL (C3-SL) and features in high-dimensional space to compress the features on the split learning cut layer to represent dimensions, thereby reducing the communication cost of feature transmission and 2.25 times the computational overhead compared with the original dimension compression.

Chen et al. [77] optimized the problem of low communication efficiency from the architecture of SL. In this paper, a loss-based asynchronous training framework is proposed. When the loss of the model is greater than a certain value, the client model will be updated, thus reducing the communication frequency of split learning. Moreover, the model gradient of server transmission is further quantified to reduce the communication load of the asynchronous training framework and reduce the communication cost.

6) HYBRID SPLIT FEDERATED LEARNING

As mentioned above, both federated learning and split learning have their own shortcomings as distributed learning paradigms. The main disadvantages of federated learning are that it is difficult to train a complete large-scale model with limited resources at the edge device side, and there are security risks in accessing both local and global models equally between client and server [78]. The disadvantage of split learning is that only one client interacts with the Server at the same time while the other clients are idle, so the problem of communication overhead exists [79].

Hybrid split Federated Learning (SFL) is proposed to combine the advantages of two distributed learning methods and optimize the shortcomings of each. SplitFed [80] is the classic architecture of SFL, as shown in Figure 7. In the framework, multiple clients compute in parallel as resource-constrained edge devices and connect directly to the central server and the Fed server, where the Fed server is the server used to execute the FedAvg algorithm on the local model of multiple clients. Each Client performs parallel forward calculation based on its own data and sends its own cut layer to the Main Server. Then, the Main Server performs parallel forward and reverse calculations corresponding to each Client. Finally, the Main Server sends the corresponding cut layer to each Client. The Main Server uses FedAvg to aggregate the gradients of local multi-part models. After receiving the gradient information sent by the Main Server, each Client carries out the backpropagation process of the local model, and then all the clients upload the gradient of the local model to the Fed Server for FedAvg calculation. Finally, the aggregation results are synchronized with each client to update the local model. The SplitFed framework has similar model accuracy and better model convergence compared to SL.

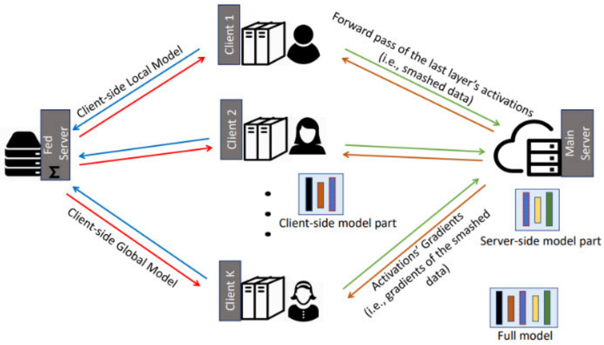

FIGURE 7. Classic architecture of hybrid split federated learning [80].

There are also two variants of SplitFed called splitFedv1(SFLV1) and splitFedv2. The difference between the two depends on the match between the server-side and the client-side [80]. In the splitfedv1 algorithm, the server-side corresponds to the client side one by one, and the client-side model executes in parallel and uses FedAvg aggregation to wait for the global server-side model. splitFedv2 (SFLV2) abandons the Fedavg part, improves the accuracy of the model by randomly selecting the client order and updating the model with each forward and backward propagation, and also avoids the problem of catastrophic forgetting in splitfedv1. SplitFedv3 [81] was also proposed to help the SFL avoid the problem of catastrophic forgetting. Its client sub-network corresponds to each client one by one, and the server sub-network is an average version, thus avoiding catastrophic forgetting problems caused by sequential training of client datasets. The SFLG framework [82] promoted SFL by combining splitfedv1 and splitfedv2 to realize the function of flexibly selecting different numbers of server terminals according to the resources and number of edge servers.

The paper also evaluates the learning performance, computing overhead, and communication overhead of FL, SL, and SFL under heterogeneous data distribution on a real-world IoT edge device, Raspberry Pi, shown in Table 5. On IID and balanced data sets, the convergence rate of the SL model is better than that of the FL model, but SL will have an unstable learning curve affected by the number of clients, and the performance of SFLv1 is close to FL. Moreover, the learning performance of SL is more affected by the number of clients under the unbalanced data distribution. Under non-IID data, FL has better learning performance than SL, and the performance of SFL is also close to FL, so it is proved that combining FL and SL optimizes the problem that the client cannot retain contribution under non-IID data of SL. The experiment also shows that when training small models on a limited number of edge devices, the training time and communication overhead of SL is worse than that of FL, but SFL achieves less training time under the same overhead.

TABLE 5. Different distributed learning performance in Raspberry Pi.

Scenarios	SL	FL	SFL
Training time			√
Communication overhead		√	
Used memory	√		√
Temperature	√		√
Peak power	√		√

The basic framework of SFL is a combination of model splitting in SL and client parallelism in FL. Therefore, the categories of SFL can be classified based on model decomposition types and parallel training strategies. Zhang et al. [83] demonstrated that different model decomposition strategies lead to increased transmission costs and time-to-model convergence, thereby reducing the optimization effect of SFL. As a part of the SL algorithm, model decomposition directly affects the overall training cost and data privacy of SFL, which can be divided into static decomposition and dynamic decomposition. SplitFed learning splits the model according to an assumed split point. There are also works on decomposition under specific models. The FESTA [84] architecture considers a specific model architecture, Vision Transformer (ViT), as a deep learning model for image processing in the SFL framework. ViT consists of three parts: the head for extracting features, the body for feature dependencies, and the tail for feature mapping, so it is easier to be decomposed in the SL part. Tian et al. [85] conducted model segmentation for the BERT model in the FedBert framework. The embedded layer and header layer of the BERT model have less computing load and can be divided into resource-constrained edge devices for training, while the transformer layer is computationally intensive and needs to be decomposed into important servers with higher computing power for training and aggregation.

However, static decomposition methods lack the adaptability of other models to the generalization and system heterogeneity of IoT. FedSyL [86], HFSL [87], and ARES [88] frameworks can select split points suitable to minimize the cost of training per round by adaptive analysis of the computational overhead of client and model training. For example, the HSFL framework utilizes the context bandit learning algorithm (LinUCB [89]) to find an optimal split point for each client in each training round to adaptively offload part of the model training from the client to the server in SFL, thereby reducing training latency. It is an end-edge-cloud architecture layered structure, so it also addresses the impact of heterogeneous devices on the SFL framework, shown in Figure 8. computational overhead, transmission costs, etc., in the SFL framework. FedLite framework [90] reduced the communication overhead of SFL with as little loss of model accuracy as possible by clustering the training data and only transmitting the clustering centroid to the server using the method of quantifying. GSFL architecture [91] grouped clients and shared servers in the group for parallel training, thus avoiding the process of sequential training for all clients and reducing the cost of training time. Furthermore, Yin et al. [92] combined split decision, bandwidth, and computational resource optimization into a multi-objective problem and proposed a GAN-driven algorithm to find a solution. A parallel scheme without label sharing is designed to reduce client idleness in the split. Han et al. [93] designed an auxiliary network for the model based on the idea of local-loss-based training [94], which can use the output of the cut layer to calculate the local loss function instead of processing the loss function at the model output so that the client can update the model without receiving the model gradient. The client does not need to wait for gradients, and the server does not need to transmit gradients, thus reducing the communication overhead and training latency of SFL.

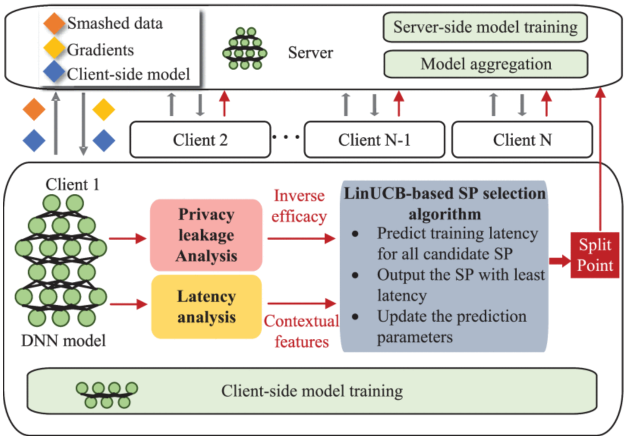
Smashed data
Gradients
Client-side model
Server
Server-side model training
Model aggregation
Client 2
Client N-1
Client N
Client 1
DNN model
Privacy leakage Analysis
Latency analysis
Inverse efficacy
Contextual features
LinUCB-based SP selection algorithm
Predict training latency for all candidate SP
Output the SP with least latency
Update the prediction parameters
Split Point
Client-side model training

FIGURE 8. Centralized HSFL framework [75].

D. DECENTRALIZED LEARNING FOR PRIVACY IN AIOT

Decentralized learning is a paradigm in distributed learning, which uses distributed accounting and storage, but there is no centralized node, and the rights and obligations of any node are equal, as shown in the right of Figure 3. The data blocks in the system are jointly maintained by nodes with maintenance functions in the whole system, and any node stops working will not affect the overall operation of the system. Since the clients can train the model together without the need to trust the center, data privacy is better than centralized distributed learning because the global model is not centralized in one place, but different clients perform parameter aggregation operations at each stage, so it is not vulnerable to attacks or model update backward inference operations. Furthermore, due to every node is equal, the risk of a single point of failure is avoided. However, full decentralization also leads to higher communication costs and higher computing resource requirements. This is not very friendly to resource-limited edge devices.

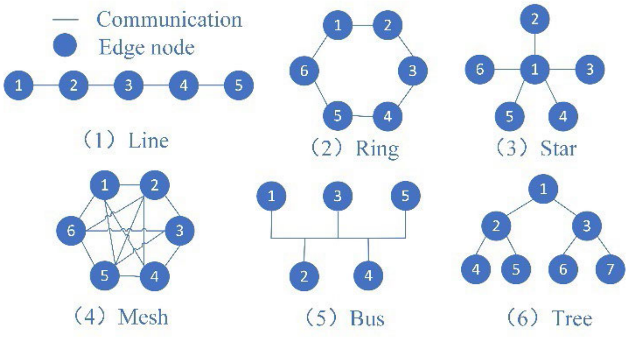

FIGURE 9. Network topology of decentralized learning.

As shown in Figure 9, there are many network topologies for decentralized learning, including Line, Ring, Mesh, Star, Bus, Tree, and Hybrid, of which the ring topology is the most commonly used. As shown in the right side of Figure 3, it represents a decentralized learning architecture that applies the sequential model transfer strategy in ring topology. The initial client C_1 trains the local data model in the given order and transmits the model parameters to the subsequent client C_2. The client C_2 aggregates the parameters calculated by the client C_3 and trains the local data model, and then transmits the model parameters to the subsequent client C_4. The remaining clients loop in sequence until the model converges.

Another model-sharing strategy for ring topology is node selection based on dynamic probability. This strategy aims to dynamically select a client based on probabilistic considerations to act as a ''server'' at a certain stage to aggregate model parameters of other clients and train the local model. In each parameter aggregation stage, a client is selected based on the probability of performing an aggregation operation, knowing that the termination condition is met, which may include the limit of global communication rounds or the accuracy rate of the model. This section will discuss some of the most prominent decentralized learning frameworks.

1) GOSSIP LEARNING

Gossip learning [104] is a random peer-to-peer architecture for clients to share and learn knowledge by gossip protocol. Each client initializes a local model and periodically transmits it to the selected node in the network. The selection of nodes adopts gossip-based peer sampling. The selected nodes aggregate operations by averaging model parameters and updating the model with local data. This mechanism makes the model walk randomly in the network and update at the access node. Hegedus et al. [105] compared gossip learning and centralized federated learning and considered compression techniques applicable to the two methods. The results show that gossip learning is superior to centralized federated learning in scenarios where data is evenly distributed on nodes. Experiments prove that it can be a new alternative to centralized federated learning under communication optimization in the future.

As a decentralized learning framework, gossip learning also has the challenge of high communication costs. References [95], [96], and [97] proposed different random optimization algorithms based on gossip to accelerate the convergence of the global model, reduce iteration, and improve the communication efficiency of gossip learning. In [96], Choco-SGD considers a decentralized optimization setting where each node computes the gradient of its local function and applies a compression operator to the gradient, thereby reducing the communication load. Nodes communicate with their neighbors, including sending and receiving compressed gradients. The convergence rate of the algorithm for strongly convex targets is $O(1/(nT) + 1/(T\delta^2\omega)^2)$, where T is the number of iterations, δ is the characteristic gap of the connection matrix and the first term $O(1/(nT))$ in the rate is equivalent to a centralized baseline with precise communication.

2) FULL DECENTRALIZED FEDERATED LEARNING

Decentralized Federated Learning (DFL) [106] is derived from federated learning and has no central server for aggregating model updates. Instead, nodes (clients) communicate directly with each other to share information and update their models. Its learning process involves peer-to-peer communication between participating nodes, with each node updating its model based on its own data as well as models or gradients it may receive from its neighbors. Its architecture is more flexible than basic federated learning and reduces the risk of data privacy because it does not need to be centralized on a central server. In the case that the client needs to communicate with the central node several times, DFL can also improve the communication efficiency of model convergence. Furthermore, gossip learning, as a communication protocol-based decentralized learning, can also be combined with DFL to reduce the network bandwidth of message transmission [107].

In addition to some of the challenges shared with centralized federated learning, there are some specific challenges with DFL. Since there is no centralized server to manage clients in DFL, it may lead to confusion in communication among clients with heterogeneous systems. For example, in the sequential model transfer policy pair topology, the client can only wait for the model parameters from the previous client, and this dependence can lead to deadlock in the network. To solve this problem, paper [98] proposed a method for clients to request other clients in advance before aggregation so as to know the status of other clients and prevent network stagnation. In addition, the lack of a central server also leads to issues with aggregation fairness. Lack of incentives can cause clients to choose to only get trained models from other clients without providing their own local data and computing resources. This behavior will affect the trust between clients and make them reluctant to provide their own knowledge. Kang et al. [99] proposed an incentive mechanism that quantifies the reliability of clients by assigning credit scores so that clients with higher credit scores (knowledge contributions) reap greater rewards.

Because of the peer-to-peer communication strategy of DFL, blockchain technology can be easily built on it to protect data privacy. Blockchain is a new application mode of computer technology that includes distributed data storage, peer-to-peer transmission, consensus mechanisms, encryption algorithms, etc. It uses cryptography to ensure the security of data transmission and access and realizes the establishment of trust and acquisition of rights between different nodes through a consensus mechanism. Referencces [100], [101], and [102] proposed to integrate blockchain technology into DFL's distributed framework to enhance data privacy. Its essence is to achieve data privacy and the security of a decentralized learning framework through blockchain technology architecture. From the aspect of data security, the client's local data is protected from external attacks through the chain data structure. From the aspect of model gradient or parameter transmission, it uses cryptographic methods, such as differential privacy, to protect the security of parameter transmission and access. From the perspective of the DFL incentive mechanism, trust between nodes is established through a consensus algorithm to prevent the occurrence of free-riding behavior in clients.

3) SWARM LEARNING

Group learning [108] is a decentralized learning framework based on blockchain technology with inherent features of global collaboration and knowledge sharing. It does not have a dedicated central server, but each swarm edge node builds an independent model, retains local private data, and shares model parameters in the swarm network, as shown in Figure 10. Data privacy and security in the Swarm network are maintained by the Ethereum blockchain, where smart contracts enable the network to select nodes identified with appropriate authorization measures to perform parameter merging each time synchronization stops. Swarm learning builds a middle layer and an application layer on each node to integrate machine learning platforms, blockchains, models, and Swarm API for execution in heterogeneous edge nodes. Since swarm learning builds blockchain technology, differential privacy algorithms, functional encryption, or cryptographic transfer learning methods can all be an extension of its data privacy protection. This technology was first applied in the field of clinical machine learning with strict data privacy. In [109], authors applied multiple datasets of cancer histopathology to verify the effectiveness and interpretability of swarm learning training with cross-regional participants, and the results showed that swarm learning was superior to most locally trained models and comparable to the performance of models trained on combined datasets.

TABLE 6. Summary of emerging distributed and decentralized learning.

Architecture	Ref.	Year	algorithm	Type	Other method	Problem solving
Distributed learning	[50]	2016	FedAvg	FL	\	Unbalanced data, statistical heterogeneity
	[51]	2020	Scaffold		\	Systems heterogeneity, statistical heterogeneity, communication traffic
	[52]	2020	FedAsync		\	Systems heterogeneity, stragglers
	[53]	2019	FedAvgM		\	statistical heterogeneity
	[54]	2020	FedProx		\	Systems heterogeneity, statistical heterogeneity
	[55]	2020	FedNova		\	Systems heterogeneity, statistical heterogeneity
	[56]	2020	FedBoost		\	Communication cost
	[57]	2023	FedBR		\	Communication cost, statistical heterogeneity
	[59]	2020	clustering + interpolation	PFL	Adding User Context	Communication cost, data security
	[60]	2023	FedPos		Transfer Learning	statistical heterogeneity
	[61]	2022	MTFL		Multi-task Learning	Systems heterogeneity, statistical heterogeneity, personalized model accuacy
	[62]	2020	Per-Fed		Meta-Learning	statistical heterogeneity
	[63]	2023	FedDK		Knowledge Distillation	statistical heterogeneity
	[64]	2021	FedRep		Base + Personalization Layers	statistical heterogeneity, Cross-Device
	[65]	2020	LLGD		Mixture of Global and Local Models	Communication cost, statistical heterogeneity
	[73]	2021	AdaSplit	SL	3C Score	Bandwidth consumption, Systems heterogeneity, resource-limited device
	[74]	2021	AutoEncoder		Adaptive threshold mechanism	Communication cost
	[75]	2023	randomized top-k sparsification		\	Communication load
	[76]	2022	C3-SL		Batch compression	Communication load
	[77]	2021	Quantization		\	Communication load
	[80]	2022	SFLV1 SFLV2	SFL	PixelDP	Training time, parameter privacy, resource-limited device
	[81]	2020	SFLv3		\	“Catastrophic forgetting” problem
	[82]	2021	SFLG		\	Communication cost
	[83]	2023	FSL_PA		\	Transmission delay, data privacy
	[84]	2021	FESTA		\	Network bandwidth
	[85]	2022	FedBert		\	resource-limited
	[88]	2022	ARES		\	resource-limited, Systems heterogeneity
	[90]	2022	FedLite		activation quantization	Communication cost
	[91]	2023	GSFL		\	Training delay, cross-device
	[92]	2023	HSFL		GAN	Split decision, bandwidth, computing resources
	[93]	2022	Local-loss-based training		\	latency and communication efficiency
Decentralized learning	[95]	2021	Gossip-PGA	Gossip learning	\	speed of convergence
	[96]	2019	Choco-SGD		\	Communication load
	[97]	2021	Decentralized SGD		Adaptive method	statistical heterogeneity
	[98]	2019	BrainTorrent	DFL	\	data privacy. statistical heterogeneity
	[99]	2019	multiweight subjective logic model		Blockchain	single-point-of-failure, accidental or malicious modification
	[100]	2021	BESIFL		Blockchain	Malicious node
	[101]	2023	LPBFL		Blockchain	Light weight framework, data privacy
	[102]	2018	ModelChain		Blockchain	single-point-of-failure, accidental or malicious modification
	[103]	2022	SDRL	Swarm learning	Actor-critic strategy optimize	Dynamic environment, data privacy

Zhu et al. [103] proposed a deep reinforcement learning SDRL based on swarm learning designed for robotic manipulation in dynamic and complex environments, proving that swarm learning can be applied not only in the clinical

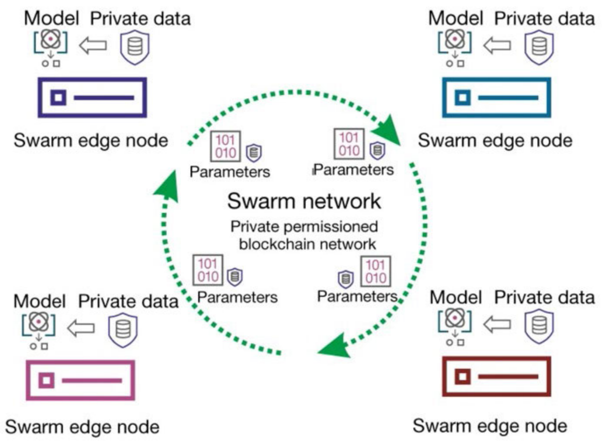

FIGURE 10. Swarm network [103].

field. SDRL implements collaborative learning of multiple robot agents using group learning methods so that each agent contributes to and benefits from the shared learning experience. Actor-critic strategy optimization has also been added to the framework to allow each robot to evaluate its behavior (Actor) and associated rewards (Critic), thus facilitating a more nuanced learning process. In the swarm network, robots share their learning experiences (loss function gradient) with each other through a blockchain network (Ethereum V2.0 platform) to achieve collaborative learning. Experimental results show that the SDRL method enhances the learning process of multiple agents. In addition, the more agents involved, the faster the learning.

E. PHASE SUMMARY OF DECENTRALIZED AND DISTRIBUTED LEARNING ARCHITECTURES FOR AIOT

The survey work in this section summarized the different frameworks and architectures of distributed learning and decentralized learning, and investigate the approaches used by recent works to address the challenges in this field. This section reveals the following key points and insights for decentralized and distributed learning for AIoT:

- The evolution from traditional centralized learning to emerging decentralized learning reveals the need for data privacy and security in AIoT systems that no longer rely on a single server for computing and storage. Instead, multiple tasks are accomplished through collaborative learning by multiple edge devices.
- Although there has been a lot of research on federated learning to make it easier to deploy on AIoT systems by optimizing aggregation algorithms, there are still scenario limitations. Split federated learning combines the benefits of split and federated learning, reducing communication costs while being easy to deploy on resource-constrained edge devices.
- Decentralized learning is a large branch of distributed learning that maximizes the performance of each edge device by removing a central server. It solves many of the problems of distributed learning due to the central server, but there is less research on it.

V. DISTRIBUTED LEARNING AND DECENTRALIZED LEARNING FOR AI TECHNOLOGY IN AIOT

In AIoT, artificial intelligence algorithms are used to process and analyze large-scale data generated by the Internet of Things. AI technology is centralized, and it focuses on model accuracy, computational overhead, and memory. The edge devices of the Internet of Things generally lack computing resources, are heterogeneous, and have data-sharing constraints, so the construction of AI algorithms on IoT devices is a challenge. Not only that, but real-time or near real-time execution of these computationally intensive solutions is also a challenging problem for AIoT [110]. This requires AI to be distributed and decentralized as a means of dividing data, models, and policies into smaller parts to satisfy device training costs, communication latency, and privacy constraints. Conversely, AI also empowers distributed and decentralized learning architectures to optimize their performance in AIoT. Therefore, this section is divided into three subsections from the perspective of AI algorithms to discuss the optimization of AI technology in IoT by distributed and decentralized learning frameworks and the role of AI in these frameworks. Table 7 shows a summary of the distributed and decentralized AI technologies discussed in this section.

A. MACHINE LEARNING BASED FRAMEWORKS

Machine learning is a subset of AI algorithms and the most mainstream and lightweight algorithm in AIoT systems. This chapter mainly discusses the machine learning that requires distributed and decentralized learning in AIoT systems and is the most generalized: support vector machine, tree-based learning, and clustering.

1) SUPPORT VECTOR MACHINE

Support vector machine (SVM) is a class of supervised learning algorithms for classification and regression analysis. The main goal of SVM is to find a hyperplane in n-dimensional space with logarithmic data points for obvious classification, and it is also possible to perform nonlinear classification using kernel methods. In [111], the authors studied the optimization of SVM called FedSVM with linear cores using a hierarchical federated learning approach to reduce SVM communication overhead at the edge-fog-cloud layer and enhance the privacy security of industrial IoT data. As shown in Figure 11, FedSVM is used for predictive maintenance (PM), where the main task at the fog stage of its framework is to identify the hyperplane that can effectively isolate failure data from health data for all equipment on the plant site. The hyperplane of fog level identification uses federal average aggregation at the cloud level to enhance the privacy of edge data, and the specific sub-gradient descent mechanism is adopted in the parameter updating process at fog level for loss function $f(w)$ can be represented as

$\partial f\ (w)\ =\ \frac{1}{D}\sum_{i=1}^{N}\partial f_i(w)+2\lambda w$ to improve the model to improve the accuracy and reliability, where D is the data sample from edge device, and λ represented a regularizer added to the local loss function. The average accuracy of the FedSVM algorithm can exceed 85%, and since the fog server does not have to wait for the edge device to update the parameters, the model runs in FedSVM in a quarter of the normal federated learning framework.

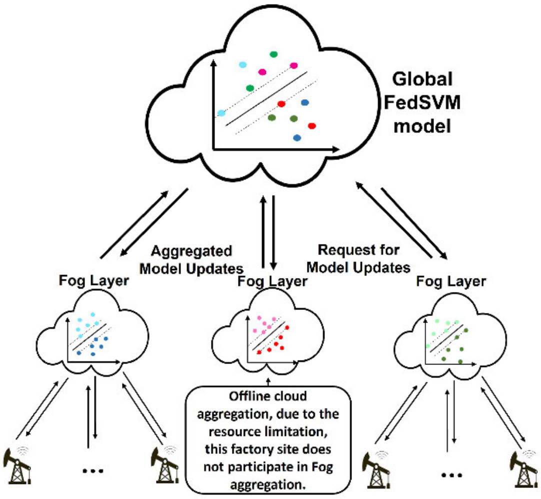
Global FedSVM model
Aggregated Model Updates
Request for Model Updates
Fog Layer
Fog Layer
Fog Layer
...
...
Offline cloud aggregation, due to the resource limitation, this factory site does not participate in Fog aggregation.

FIGURE 11. FedSVM architecture [111].

FedSVM is a centralized federated learning framework that is difficult to adapt to spatially dispersed medical data that requires higher data privacy. However, the FedEHR approach [112] developed a decentralized, federated learning architecture based on FedSVM for the Medical Internet of Things domain (IoMT) using a Soft-margin L1-regularised sparse Support Vector Machine (sSVM) Classifier, which is designed to process large amounts of medical data to accurately predict and diagnose heart disease. The Sparse characteristic of the classifier allows it to focus on the most important features, improving interpretability and efficiency in processing complex heart health data. Clustering Primitive Double Splitting (cPDS) method This iterative technique is used to manage the decentralization of healthcare data. It combines both primitive and dual aspects to ensure fast and accurate convergence to solve large-scale sSVM optimization problems. In decentralized setting comprising m devices, cPDS method for sSVM can be expressed as $\min\limits_{\theta,\theta_o}\sum_{j=1}^{m}k_i(\theta,\theta_0)+0.5\psi_j\left\|\theta_j\right\|_2^2+\rho_j\left\|\theta_j\right\|_1$, where $k_i(\theta,\theta_0)$ is kernel function in SVM model to represent the hyperplane $k_i\left(\theta,\theta_0\right)\ =\ \max\left\{0,1-\ell_i(\varphi_i^{\tau}\theta+\theta_0)\right\}$ denotes a hinged loss function related to sample i. ψ and ρ indicates the co-efficients for penalty, and $\|\theta\|_1$ represent sparsity. In addition, the paper optimizes the information transmission of the framework and adopts the round-robin algorithm to allocate communication resources evenly and reduce transmission costs. Compared to GA-SVM [113], FedSVM, and SVM, FedEHR can achieve an accuracy of nearly 99% after 1000 rounds of communication, which is higher than other models.

The previous frameworks all use the nature of federated learning to protect data privacy. Hsu et al. [114] proposed a federated learning system PPFL for detecting Android malware to introduce secure multi-party computing (SMPC) to protect data privacy for the SVM model. In the initialization phase, PPML randomly selects a large prime Q from the federated learning group that is required in the locally generated SMPC so that the random value of each client is unknown to other clients and servers to ensure the security of the client model. The client uses additional key sharing to protect the parameters of its local model from being accessed and, sends the parameters to the server to aggregate the encryption parameters using SMPC, and then sends the updated global model back to the client for decryption and update. PPFL is not affected by the number of clients, and the model accuracy is close to 94%; the communication load size is only 145KB, and the communication time is very short at 0.912 ms.

SVM can not only solve the problem of linear separability but also map the data in the high-dimensional space by transforming the kernel function for linear separability so it can represent most linear models. Because of the versatility and good generalization of kernel functions, it is suitable for personalized federated learning to change the corresponding kernel functions according to the needs of clients. Moreover, since SVM works by mapping data to high-dimensional space for classification, it is also suitable for building on nodes with sparse data in distributed or decentralized AIoT environments. However, the data distribution and communication overhead of multiple clients under distributed or decentralized is a big challenge for SVM. In AIoT, due to data privacy protection, there will be data heterogeneity and imbalance, so SVM makes it difficult to choose the right kernel function, and it is more sensitive to data imbalance than other machine learning algorithms, which will affect the final model accuracy. Since SVM model convergence requires more communication times, communication overhead is also a big challenge for the SVM model.

2) TREE-BASED LEARNING

Tree-based learning (DT) is a tree-structured machine learning algorithm that selects the partition with the greatest entropy reduction in a dataset. In this model, internal nodes represent data attributes, branches represent feature value conditions, and hierarchical feature differentiation addresses classification and regression problems. The unique structure necessitates a distinct architecture for decentralized and distributed learning. Specially for Gradient boosting decision tree (gbdt) [115] is a decision tree-based ensemble learning technique, typically using decision trees, to minimize predefined loss functions.

Li et al. [116] designed SimFL, a horizontal federated learning framework for designing a federated environment with loose privacy constraints in a gradient-boosting decision tree, to solve the problem of inefficient and insufficient model accuracy due to the use of secret sharing and homographic encryption methods to protect data privacy. This FL framework has two stages: preprocessing and training. In the preprocessing, participants first use a randomly generated locally sensitive hash (LSH) function $\left\{F_k(x_i^m)\right\}_{k=1,2,\cdots L}$ to calculate the hash value such that the hash values of two adjacent points are highly equal, and the hash values of two non-adjacent points are highly unequal, so that there are infinite input data for the same hash value, where $x_i^m \in I_m$is instance in each participates P_m. Therefore, when the hash value is broadcast, other participants cannot infer the input data, thus achieving the effect of protecting the data. In the training, each participant uses the similarity information together to train several trees one by one. Once a tree is built in one party, it is sent to the other parties to update the gradient. In the SimFL, the computer overhead is $O\left(2NL + Nd\right)$, the communication overhead is $8T\left[N + \left(2^D - 1\right)\left(M - 1\right)\right]$ (assume there are T trees, M parties, N total instances, and L hash functions), and the test error is much lower than TFL [89], [117].

The aforementioned framework applies to parties with horizontally distributed data, while there are limitations in scenarios where the data is divided by characteristics between different parties called vertical federations. Cheng et al. [118] considered this situation and proposed a SecureBoost algorithm based on XGBoost [119], which is an optimization of GBDT. In SecureBoost, participants are divided into Active Party (with labels) and Passive Party (with data). Passive Party is passed to the Active party data pair $\left[party\left(id\right), feature_{id}\left(k\right), threshold_{id}\left(v\right), G^i_{\{kv\}}, H^i_{\{kv\}}\right]$ which represent the party i to ensure the distribution of data features and the value of sub-points is not leaked. Since there are many parties involved in federated learning, it is necessary for the parties to calculate the sum of derivatives of different features at different loci and data transfer, record the best features, and save the values of the best loci on their respective platforms to generate lookup tables for the subsequent prediction process. During the training process, the information that the Active party can grasp includes the sample space on the nodes in the splitting process, the number of features owned by each party, the specific derivative value, which party the splitting nodes in the regression number come from, and the splitting threshold of their own characteristics, while Passive party can only grasp the sample space and the splitting threshold. This ensures the privacy of the tree's information.

Similarly, under the GBDT model, Yamamoto et al. [120] developed the eFL-Boost algorithm, which is different from SimFL in that eFL-Boost selects one of the data owners to build the tree and uses the tree structure and weights instead of instance and weighted gradients. eFL-Boost constructs decision trees in two stages: local tree structure determination and global leaf weight calculation, shown in Figure 12. In the first stage, the selected participants apply the local data set to build the tree structure, and each data owner calculates the gradient sum of each leaf and sends it, thus reducing the communication cost of the framework. In the global leaf weight calculation, since the leaf weight is directly related to the GBDT output, eFL-Boost aggregates the local gradient calculates the global leaf dimension, and adds the completed tree to the global model, thus reducing the accuracy loss of the model. In terms of data privacy, eFL-Boost has a lower risk than TFL and F-GBDT-G [121] because the inference of data mainly comes from the change of gradient variance in the leaf nodes, while each tree built-in eFL-Boost is based on global distribution and thus has reduced privacy risk.

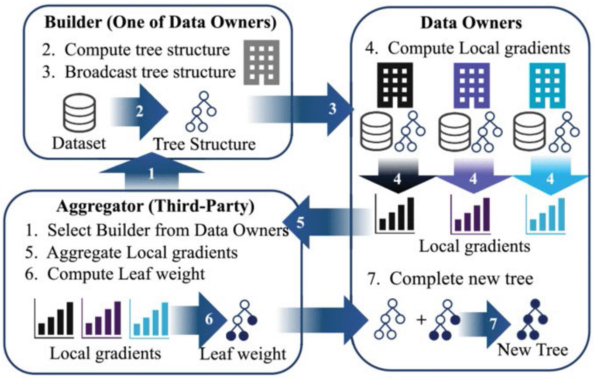
Builder (One of Data Owners)
2. Compute tree structure
3. Broadcast tree structure
Dataset
2
Tree Structure
3
Data Owners
4. Compute Local gradients
4
4
4
Local gradients
7. Complete new tree
1
Aggregator (Third-Party)
1. Select Builder from Data Owners
5. Aggregate Local gradients
6. Compute Leaf weight
5
Local gradients
6
Leaf weight
+
7
New Tree

FIGURE 12. **eFL-Boost architecture [120].**

AdaBoost or Gradient Boost are both tree-based machine learning but are also ensemble learning approaches that aim to improve the performance of a single model by combining the output of multiple weak models. Each weak model focuses on a different aspect of the data and combines their predictions together to create more accurate and robust models. However, most AIoT devices are resource-limitation edge devices, so it is difficult to train the models with large computational costs. Therefore, the tree model-based Boost algorithm is suitable for deployment in AIoT distributed or decentralized environments, running weak models (local tree models) with low computational overhead on edge devices and aggregating them to create more powerful global tree models by means of data privacy protection such as transmission parameters. From the perspective of edge device computing overhead, data privacy, and processing heterogeneous data, tree-based learning is suitable for building distributed or decentralized AIoT environments. However, tree-based learning methods are sensitive to the distribution of training data, so dealing with non-IID data due to distributed or decentralized data is a challenge.

3) CLUSTERING

Distributed and decentralized learning is commonly used when labels are available. However, in tasks where labels are unavailable or difficult to obtain due to strict data-sharing restrictions, unsupervised learning offers a solution. Clustering, a typical unsupervised learning method, aims to divide data into clusters based on specific characteristics, ensuring that points within the same cluster are more similar to each other than to those in different clusters.

Based on Lloyd's K-means clustering algorithm, Dennis et al. [122] propose an unsupervised federated learning k-FED and find that statistical heterogeneity (often seen as a challenge in supervised learning) can be advantageous in the context of federated clustering. K-means is one of the clustering algorithms that updates the initialized random centroid by calculating the average of all points assigned to the cluster and reassigning data points to divide into multiple clusters. The paper establishes two types of separation hypotheses in heterogeneous environments: active separation and inactive separation of devices that contain and do not contain two cluster points, respectively. The paper's analysis is based on two types of separation assumptions in heterogeneous environments: active separation and inactive separation of devices that contain and do not contain two cluster points, respectively. For active cluster pairs, the separation requirement can be defined as $\|\mu\left(T_r\right), \mu\left(T_s\right)\|^2 \geq c\sqrt{\frac{m_0(\Delta_r+\Delta_s)}{2}}$, where $\mu\left(T_r\right)$ and $\mu\left(T_s\right)$ are the mean of clusters T_r, T_s, Δ_r and Δ_s are associate cluster-specific quantities, c is constant value. For inactive cluster pairs, separation requirement is $\|\mu\left(T_r\right), \mu\left(T_s\right)\|^2 \geq 10\sqrt{m_0(\Delta_r + \Delta_s)}$, which is weaker than for active pair. If every active cluster pair meets the active interval requirement and every inactive cluster pair meets the inactive interval requirement, then the data points in the federated learning framework can be efficiently classified by k-FED, proving that k-FED can work effectively under the weak cluster separation requirement in heterogeneous networks. The paper also demonstrates K-FED's communication efficiency and ability to benefit from statistical heterogeneity under the FEDMINST and Shakespeare datasets [123].

Similarly, for statistical heterogeneity, Lubana et al. [124] developed a cluster-based self-supervised learning method, Orchestra, to divide client data into distinguishable clusters to solve the data heterogeneity in federated learning and resource limitations of edge devices and introduced representation learning to transform client raw data into feature spaces that capture underlying patterns or structures in the data. Orchestra clusters locally based on the client's data representation and aggregates on the server using the local centroid for global clustering, quantifying the separation between clusters using "δ, inter-Cluster mixing". The relationship between clustering quality and model generalization performance can be expressed as $\varepsilon\left(f\right) < \zeta_x + O(2\delta + (G-1)\delta^2)$, where ζ_x used to measure the similarity of the latent variable of the two clusters in the distribution x and O is a constant that depends on the size of dataset. δ affects the performance of the model, so it needs to be minimized. Because in Orchestra's architecture, the client only needs to perform data processing and clustering locally, and only the cluster centroid is exchanged between the clients, it is suitable for resource-limited device deployment and has low communication overhead. The experiment demonstrated Orchestra's robustness to statistical heterogeneity, number of customers, participation rate, local era, and communication efficiency, and its performance improved with increasing data heterogeneity.

AI algorithms can not only be built in a distributed and decentralized environment to protect data privacy but also enable distributed and decentralized learning in an optimized way to reduce the loss of model accuracy. Sattler et al. [125] proposed cluster Federated Learning (CFL) to address the limitations of federated learning in data heterogeneity, with the aim of grouping clients with similar data distributions into clusters, allowing for multi-task learning, where each cluster has a model suitable for its specific data distribution. The CFL will cluster clients based on the cosine similarity of the gradient update, which can be expressed as *cosine_similarity* $\left(\nabla_i, \nabla_j\right) = \frac{\nabla_i \cdot \nabla_j}{\|\nabla_i\|\|\nabla_j\|}$, where ∇_i, ∇_j are the gradient updates from clients i and j, respectively. Based on these similarities, clients are clustered so that clients in the same cluster have high cosine similarity in their gradient updates, indicating similar data distribution. The model of each client cluster is refined to better suit the specific data characteristics of that cluster, thus minimizing interference from the disparate distribution of data. The clustering and refining process of the CFL is recursive, as shown in Figure 13, until all clients within each cluster are reached consistently. Experiments show that the accuracy of the model gradually increases with the cluster splitting of clients, and the accuracy of the five-layer convolutional neural network on the CIFAR-10 dataset can reach nearly 65%.

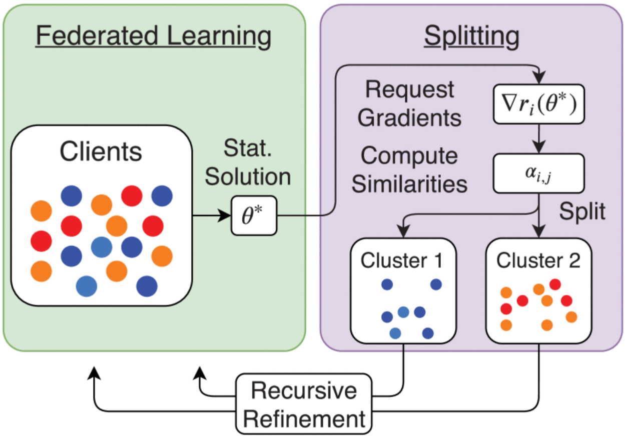

FIGURE 13. CFL dividing the customer groups by recursively [125].

Clustering can be used as a representative algorithm for unsupervised machine learning. Because it is based on the principle of grouping entities that exhibit a high degree of similarity, it is possible to form client clusters with similar data classifications or subsets of global data with more homogeneity in each cluster. Based on this feature, clustering algorithms can effectively solve statistical heterogeneity in distributed or decentralized environments by modeling different clients more accurately. Clustering algorithms are also an option for designing personalized federated learning, where models can be fine-tuned for the unique data distribution of each cluster and thus potentially achieve better performance than generalized models. From a data privacy perspective, data or client clustering can be implemented without major changes to the distributed protocol, so it maintains the privacy-protecting features of federated learning. However, overly homogenous clustering can result in models that perform well on cluster-specific data but poorly on more diverse or invisible data, leading to overfitting. In addition, the wrong clustering may cause the model to be poorly optimized for the actual data distribution of the customer, thus reducing the model performance.

B. DEEP LEARNING BASED FRAMEWORK

Deep learning is a subset of AI algorithms that can model and automatically learn from data using multi-layered neural networks and thus perform well on large and complex datasets. Due to the architecture of deep learning being more complex than machine learning, it is more widely built-in distributed or decentralized learning architectures. This section will discuss structured data-based deep learning, generative adversarial, graph-based deep learning, and deep reinforcement learning.

1) STRUCTURED DATA-BASED LEARNING

Convolutional neural networks (CNN) and recurrent neural networks (RNN) are the most widely used deep learning algorithms, and even Long short-term memory (LSTM) is considered an extension of RNN. They have multiple layers and are used to process large, structured data, such as images, speech, text, or time series, so the computational overhead and parameters of the model are very large, and it is difficult to build globally on the edge of IoT devices.

When dealing with sequence data such as text, learning models of RNN-based architectures are often used. Abedi et al. [126] proposed a FedSL framework combining federated learning and split learning to split recurrent neural networks to process multi-segment sequence data distributed in the client. As shown in Figure 14, in order to process sequentially partitioned data distributed among multiple clients while maintaining privacy, the paper separates the RNN from the hidden layer and splits the network into two subnetworks at the sequence segment split points (denoted as τk and $\tau k+1$). The left subnetwork is trained using data from client$k$, and the right subnetwork is trained using data from client l. The right subnetwork needs to activate the hidden state of the left subnetwork and vice versa the loss gradient. This exchange facilitates forward and backward propagation, gradient calculation, and parameter updating in the two subnetworks without the need to share raw data or complete model parameters. After local training, the subnetwork is sent to the federation server. These subnetworks are aggregated to form a global model, similar to the operation of federated learning. While maintaining data privacy, FedSL can achieve higher accuracy than Fedavg with fewer communication rounds.

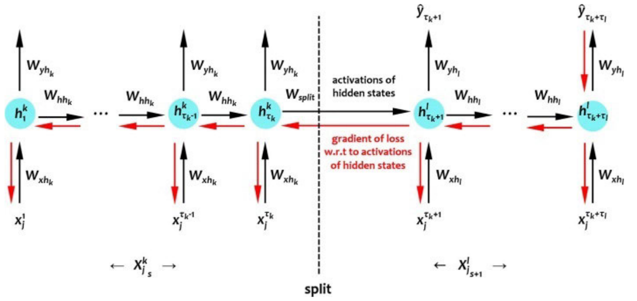

FIGURE 14. Split RNN [126].

However, traditional RNNS have limitations, especially when dealing with long sequences, which are prone to problems such as disappearing gradients and explosions. Long short-term memory (LSTM) networks, as special RNNS, overcome the gradient disappearance problem with more complex internal structures, including gates that regulate the flow of information. These gates (forget gates, input gates, and output gates) allow the LSTM to retain or discard information for a long time, making it more efficient for long sequences of data. In [127], authors proposed LSTMSPLIT, which combines the LSTM network with partition learning for sequential time series data classification and applies differential privacy technology to add noise to the output of interested parties to ensure data privacy. The LSTM network is divided into clients and servers at the specified layer (split layer) and can update the configuration with different weights, both centralized and decentralized, as shown in Figure 15. Compared to split-1DCNN, LSTMSPLIT has a higher time complexity (LSTM itself has a higher time complexity), but it has a higher accuracy, reaching 98.5% on the ECG dataset.

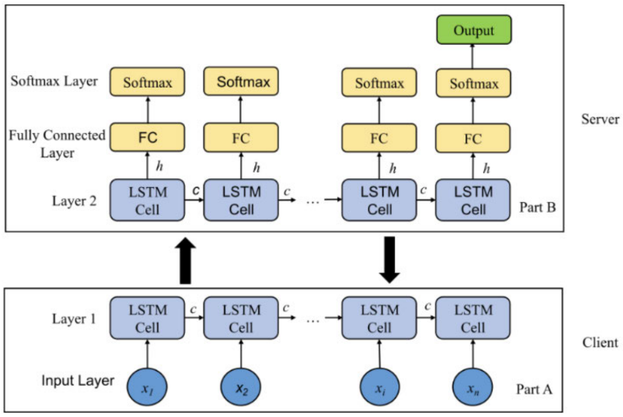

FIGURE 15. SL architecture of the LSTM network [127].

In the aspect of image recognition, the CNN-based model performs better. Ayad et al. [128] proposed a split learning framework based on a convolutional neural network variant Time Convolutional network model (TCN) and took advantage of semi-supervision for classifying ECG records in IoM. This paper introduces the autoencoder (AE) in split learning architecture, which includes the encoder on the client side and the decoder on the server side. The encoder learns to create a smaller latent vector from the split layer output, which is then sent to the server. The server's decoder uses this latent vector to approximate the original encoder input. The aim of AE is to reduce data transmission during training. The authors also use an adaptive threshold mechanism (ATM) to control the number of gradient updates sent to the client during backpropagation to further optimize communication efficiency. The system performs similarly to the original split learning system while reducing the communication overhead between server and client by 71% and client computing by 46.5%. Since the model size of the deep convolutional neural network model hinders the training of resource-constrained edge devices, He et al. [129] proposed a federated learning algorithm FedGKT for group knowledge transfer training, which aims to keep benefit from both Fedavg and SL. FedGKT can transfer knowledge from many compact CNNS trained at the edge to large CNNS trained on a cloud server through knowledge distillation, exchanging hidden features instead of exchanging in FedAvg. The entire model inherits the benefits of SL with reduced communication bandwidth. Compared to edge training using FedAvg, FedGKT uses 9 to 17 times less computational power and requires 54 to 105 times fewer parameters.

Generative adversarial network (GAN) training is divided into two stages. The first is to train the discriminator, which is used to distinguish real samples from fake samples. The task of the training generator is to create the same data as the real sample. The two models work together against training to make the resulting data more accurate. The reason for the poor training performance of the distributed learning model is the lack of data, and the role of GAN can solve this problem. In [130], authors proposed the PerGED-GAN framework to solve the systematic and statistical heterogeneity of federated learning by generating adversarial networks for personalized federated learning. The local model trained in the previous step for each client in the PerGED-GAN is regarded as a discriminator in the GAN to train the generator network and use it to generate new datasets. The Central server aggregates and redistributes samples of generated data collected from customers, ensuring that customers learn from each other without directly sharing data or models. When the customer's model architecture and data distribution changed significantly, the PerFED-GAN approach showed a significant improvement in average test accuracy (42%). Similarly, Wu et al. [131] also proposed a split-federation learning FedCG using GAN. The novelty of FedCG is that it shares the client generator instead of the extractor with the server to aggregate the shared knowledge of the client, improve model performance, and keep the feature extractor local for privacy.

2) GRAPH-BASED LEARNING

Graph-based learning encompasses machine learning techniques that utilize graph structures to model complex relationships and interactions between data nodes. In this context, nodes represent entities such as clients, individuals, or sensors, and edges represent relationships or interactions between these entities. This architecture is particularly suitable for modeling complex interactions in IoT systems and for adapting to changes in IoT device nodes to maintain an accurate network representation. Additionally, graph-based learning effectively represents nodes and relationships in distributed and decentralized learning, facilitating the construction of client relationships and optimizing the challenges posed by system heterogeneity. Authors [132] proposed a unified framework for federated learning applied to graph neural networks, which is shown in Figure 16, and divided graph learning into three distributed and distributed types based on real-world data sets and application scenarios: graph-level setting, subgraph-level setting, and graph node-level setting. The graph-level setup is primarily aimed at large and diverse data sets that may constitute GNN training, but that cannot be shared directly across silos. Subgraph-level setting scenarios where the client holds only a portion of the global data. The graph-level setting is an edge device where local data may be sensitive, and only its k-hop neighbors can see it.

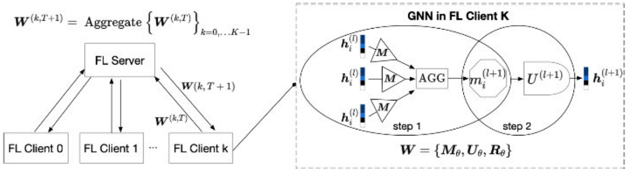

FIGURE 16. Formulation of federated graph neural network [132].

In graph level, Xie et al. [133] proposed a graph Cluster Federated Learning (GCFL) framework to deal with non-independent co-distributed graph structures and node features in federated learning Settings that standard federated learning methods such as FedAvg cannot handle. Due to the graph structure and features of the graph data being different, gradients in the graph neural network (GNN) can capture the differences in the graph structure and features. Therefore, GCFL can realize client dynamic clustering by using the gradient of client transmission. Clustering occurs when a general federated learning method approaches a static point, the criteria for which can be expressed as $\delta_{mean} = \frac{|G_i|}{G} \Delta\theta_i < \varepsilon_1$, where $\Delta\theta_i$ represents the gradient transmitted by client i, ε_1 is used to decide the static point. When the maximum norm of transmitted gradients exceeds $\delta_{max} = \max(\Delta\theta_i) > \varepsilon_2$, the clusters are split. In each cluster, servers' aggregate gradients by cluster, and the goal of each client is to find the optimal model parameters that are close to the true solution. For graph data distributed across 30 clients, GCFL compared Fedavg and FedProx to improve accuracy by 13.27%.

As graph data becomes larger and larger, it is difficult to collect and store on a single-edge device, so a distributed learning architecture based on subgraphs is proposed. At the subgraph level, Zhang et al. [134] proposed FedSage, a federated learning architecture that uses graph neural network model-GraphSage combined with the FedAvg algorithm to enable multiple edge devices to hold a subgraph of a larger graph without directly sharing the graph data. The article also proposed FedSage+, extending FedSage by adding the missing neighbor generator NeighGen. NeighGen solves the challenge of missing links between local subgraphs by generating such links and repairing subgraphs, which consists of two parts: an encoder H_e and a generator H_g, shown in Figure 17. H_e computes node embeddings, while H_g includes a linear regression model *dGen* and a feature generator *fGen*. *dGen* predicts the number of missing neighbors, and *fGen* generates the feature vectors of these neighbors. Experiments show that FedSage+ has generalization, and it can also achieve 86% accuracy on Cora's data set in the case of missing nodes. Also, at the subgraph level, the authors [135] introduced FED-PUB on GNN, which emphasizes personalized learning of subgraphs within different communities. Unlike GraphSage, it uses function embedding for similarity estimation and personalized sparse masks to customize the learning process for each subgraph without explicitly focusing on missing links. In contrast, under the Cora PubMed dataset, the accuracy of FED-PUB is better than that of FedGNN [136], and the GCFL and Fedsage+ mentioned in the article. Both approaches propose novel solutions to the different challenges posed by the distributed nature of subgraph data and the inherent complexity of graph structure information.

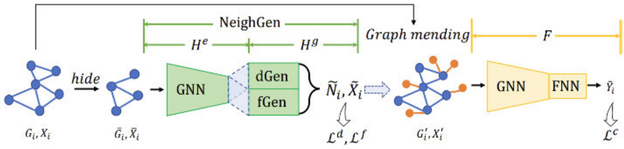

FIGURE 17. Missing neighbor generation and node classification [134].

The architecture of distributed and distributed learning can also be represented in a graph structure, where each client represents nodes in the graph and does not necessarily own the graph data. At the node level, Meng et al. [137] propose a split federated learning method, CNFGNN, for processing spatiotemporal data generated by IoT edge devices. The CNFGNN model unraveled the modeling of node-level temporal dynamics and server-level spatial dynamics. At each node (client), encoder-decoder models extract temporal features from local data and make predictions. The central server uses a graph neural network (GNN) to propagate these extracted node time features and output node embeddings that encapsulate the relationship information between nodes. To capture complex spatial dynamics, CNFGNN uses GNN to generate node embeddings containing all node relationship information. The central server collects the hidden state of all nodes as input to the GNN. The split learning structure is designed to minimize the amount of data that needs to be transferred between nodes and servers. The experiment also demonstrates the applicability of CNFGNN in real traffic prediction scenarios.

Distributed learning can also directly process graph data generated by edge devices. Pei et al. [138] propose D-FedGNN, a graph data federation learning architecture that does not rely on a centralized server, for scenarios where privacy and data ownership are critical. D-FedGNN operates in a fully decentralized manner. Each client independently trains its GNN model using its local data, and the GNN model is responsible for processing the client's private graph data. In this paper, a decentralized parallel stochastic gradient descent (DP-SGD) method is proposed to update the model parameters of each client in a decentralized environment. The model updating process in the client involves forward computation through the graph convolution layer, which can be expressed as $H = \delta(D \cdot X \cdot W)$, where δ is active function, D is the adjacency matrix encoding edge information, X is the node embedding and W is the weight matrix of the GNN. The training process in D-FedGNN involves alternating between client-side local model update and aggregation phases, an approach that reduces communication overhead. In the experiment, it takes less time than FedGraphNN to run 110 federated model aggregation.

3) DEEP REINFORCEMENT LEARNING

Reinforcement learning (RL) involves agents interacting with a dynamic environment, focusing on sequential decision-making problems. Agents choose actions based on current environmental conditions, which then change as a result of these actions. Agents receive rewards based on their decisions and the resulting environmental changes, described by the Markov decision process (MDP). Deep reinforcement learning (DRL) integrates deep learning's perception capabilities with RL's decision-making abilities, employing deep neural networks to learn the Q-value function, thus addressing challenges with large state and continuous action spaces. In distributed learning, local client learning can be seen as the agent's strategy in DRL, with information transmission representing agent interaction. Deep reinforcement federated learning (DRFL) is particularly popular for optimizing issues in federated learning.

Reinforcement learning can optimize the bias of devices to participate in each round of federated learning, from non-IID data to federated learning, by intelligently selecting clients. In [6], authors proposed the FAVOR control framework to solve the bias caused by non-IID data and accelerate convergence by using reinforcement learning to actively select the optimal subset of devices in each round of communication. In addition, a Deep Q network (DQN) based reinforcement learning agent is proposed for device selection in FL, where the state space includes the weights of the global model and the model weights of each client device, and the operation space is simplified to select one of N devices per turn, shown in Figure 18. Moreover, the FAVOR framework has been evaluated on the MNIST, FashionMNIST, and

CIFAR-10 datasets, demonstrating a significant reduction in the number of communications required for training compared to FedAvg. On this basis, Meng et al. [139] proposed that FedRLCS improved the greedy strategy and action space of the dual DQN (DDQN) algorithm and introduced the top-p sampling strategy into the algorithm. This method can select some optimal clients for model aggregation, thus significantly reducing the communication rounds required for federation learning convergence. FedRLCS achieves target accuracy with 58% and 64% fewer communication rounds than FedAvg.

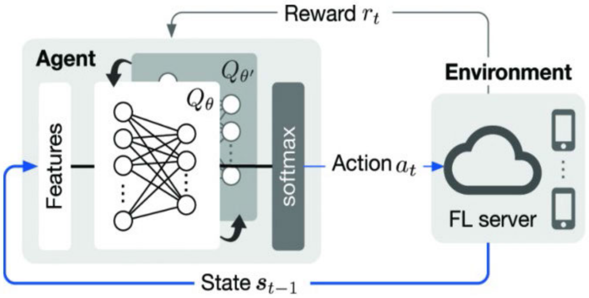

FIGURE 18. DDQN agent interacts with the FL server [6].

Reinforcement federated learning can also solve the resource allocation problem in distributed learning to improve the overall efficiency of the system. In [140], the authors propose a concurrent joint reinforcement learning (CFRL) divided into three phases: policy creation, policy execution, and model updating. The edge host creates a resource allocation policy and shares it with the server, and the server regenerates the resource allocation policy. According to these policies, tasks are assigned or unloaded, and rewards are allocated accordingly. Both edge hosts and servers update their local DQN with the rewards they receive. Compared with standard DRL, CFRL improves the overall resource utilization of the client, especially in the later stages of learning. Similarly, Nguyen et al. [141] used the Markov decision process (MDP) model to describe its resource allocation problem and carried out adaptive strategies through DQN, a variant of Q-Learning. The difference is that this article's DQN is used to optimize the model owner's decisions on energy and channel selection. Compared with traditional Q learning, greedy algorithm, and random algorithm, the DQN strategy shows a faster convergence rate and higher reward.

Deep reinforcement learning also provides new solutions to data privacy issues in distributed and decentralized learning. In [142], the authors provide an FRL architecture in which each agent works on its own independent IoT device, sharing the gradient of the loss function and transmitting mature policy model parameters to other agents. This scheme solves the security problem related to training control strategy due to scalability. Actor-critic proxima policy Optimization (Actor-Critic PPO) is incorporated into each agent as the main method of the framework, which ensures that updates to the policy model do not deviate significantly from the previous policy while ensuring data privacy, resulting in smoother updates and more stable learning. Similarly, in [143], the authors developed the FL2S mechanism, a hierarchical asynchronous FL framework. It leverages DRL to select participants with sufficient computing power and high-quality datasets to ensure reliable data sharing by sharing local data models rather than source data. FL2S improves the efficiency of the client-server federated learning architecture by selecting high-quality data nodes with powerful computing power. A deep deterministic strategy gradient (DDPG) method is used, which involves two neural networks: an online network (actor network) and a target network (critic network) using experiential playback to ensure state independence. Both prove that deep reinforcement learning can provide data privacy in data sharing and processing for distributed learning frameworks while improving processing efficiency and learning speed.

In [144] and [145], this paper presents the offload optimization of deep reinforcement learning on distributed learning. The Fedadapt algorithm [144] mentioned above uses reinforcement learning methods, including proximal policy optimization, to determine which deep neural network layers can be offloaded to the server to adapt to changing network bandwidth during parameter transmission. In [145], The authors consider task offloading strategies and resource allocation strategies in a fog computing environment using DRL, specifically the Advantage Actor-Critic (A2C) algorithm, where a network of actors (responsible for generating action) and a network of critics (responsible for evaluating an Actor's performance and directing his or her next moves). At each time step, the edge route receives the offload request and makes the decision. This paper adopts a multi-agent approach to simplify the problem. The complete offload decision action is decomposed into three sub-actions, and different rewards are designed for three different sub-actions, including the total cost of the offload decision, the upload delay of channel resource allocation, and the processing delay of fog node resource allocation. The cost of the A2C strategy over the random strategy was reduced from about 1.1 seconds to 0.2 seconds, an 81% reduction.

C. PHASE SUMMARY OF AI ALGORITHM FOR DISTRIBUTED LEARNING AND DECENTRALIZED LEARNING

The survey work in this section provides a summary of different AI techniques used in distributed learning as shown in Table 7 and outlines the challenges each AI technology faces in solving IoT system issues through distributed learning. This section reveals the following key points and insights for AI architectures and algorithms in decentralized and distributed learning:

- Based on the data sharing constraint, AI algorithms need to adapt to different decentralized and distributed learning frameworks according to their own architectures, and some AI algorithms will lose their original

TABLE 7. Summary of AI technology for distributed and decentralized learning.

Architecture	Ref	Years	AI tech	Framework	Main focus	Dataset
Machine learning	[111]	2022	SVM, LSTM	CFL	Collaborative prediction from edges, fog and clouds for Industry 4.0 anomaly detection	CMAPSS
	[112]	2023	SVM	DFL	Predicting heart disease in electronic health records	Real-word EHR dataset
	[114]	2020	SVM	CFL	Android malware detection	NICT's Android malware dataset
	[116]	2020	DT Boost	DFL	Replaced the traditional encryption operation and improves the model accuracy	LIBSVM website
	[117]	2018	DT Boost	CFL	Safe aggregation of different regression trees	GSS, Integrated public use of micro data, HSKC
	[118]	2021	GT Boost	CFL	Industrial applications such as credit risk analysis.	GiveMeSomeCredit, uciml
	[120]	2022	DT Boost	DFL	Reduces privacy risks of tree structure aggregation	Credit, Breast, Biodeg,German,Magic
	[122]	2021	Clustering	CFL	Statistical and systematic heterogeneity of federated learning	FEMNIST, Shakespeare
	[124]	2022	Clustering	CFL	Unsupervised parameter tuning	CIFAR-10/-100 datasets
	[125]	2021	Clustering	PFL	Distributed multi-task optimization	MNIST, CIFAR-10
Deep learning	[126]	2020	RNN	SL	Multisegmented sequences of vital signs processed from patients in different hospitals	MNIST, eICU dataset, FMNIST
	[127]	2022	LSTM	SFL	Efficiently process sequential time series data and reduce client computing burden	ECG dataset, HAR dataset
	[128]	2023	TCN	CFL	Private ECG classification system	PTB-XL
	[129]	2020	CNN	CFL	Edge devices with limited resources	CIFAR-10/-100, CINIC-10
	[130]	2022	GAN, DNN	PFL	Client personalized design model architecture	CIFAR-10/-100
	[131]	2022	GAN, DNN	SFL	Solve the problem of sharing gradient leakage of data privacy	FMNIST, CIFAR10, Digit5, Office-Caltech, DomainNet
	[132]	2021	GNNs	CFL	Construction of GNN in federal learning framework	SIDER, BACE, CLINTOX, BBBP, TOX21, CIAO, EPINIONS, CORA, CITESEER, PUBMED, DBLP
	[133]	2021	GNNs	CFL	Heterogeneity of structure and features among graphs	MUTAG, BZR, COX2, DHFR, PTC_MR, AIDS, NCI1, ENZYMES, DD, PROTEINS, COLLAB, IMDB-BINARY, IMDB, MULTI
	[134]	2021	GraphSage	CFL	Lack of cross-subgraph links in systems that share graph data	Cora, Citesser, PubMed, MSAcademic
	[135]	2023	GNNs	CFL	Heterogeneity of subgraphs composed of different communities	Cora, Citesser, PubMed, Amazon-Computer, Amazon-Photo, ogbn-arxiv
	[137]	2021	GNNs	SFL	Modelling complex spatio-temporal dependencies for prediction	PEMS-BAY, METR-LA
	[138]	2021	GNNs	DFL	Graph neural network setting in a decentralized architecture	MoleculeNet
Reinforcement learning	[6]	2020	DRL, DQN	CFL	Optimize communication rounds and statistical heterogeneity	MNIST, FMNIST, CIFAR-10
	[139]	2023	DRL, DNN, DQN	CFL	Optimal client selection policy from the non-IID dataset	CIFAR-10、CIFAR-100、NICO、Tiny ImageNet
	[140]	2021	DRL	CFL	Resource allocation strategy	Numerical simulation dataset
	[141]	2019	DRL, DQN	CFL	Optimal decisions on the Energy and the channels without any a priori network knowledge	Numerical simulation dataset
	[142]	2020	DRL, MLP	CFL	Collaborative learning of multiple agents	Numerical simulation dataset
	[143]	2021	DRL	CFL	The selection strategy for optimal participants	Numerical simulation dataset
	[144]	2022	DNN, DRL, Clustering	SFL	Address the challenges of computing heterogeneity and network bandwidth variations	CIFAR-10, Numerical simulation dataset
	[145]	2021	DRL	CFL	Optimize task offloading and resource allocation policies	Numerical simulation dataset

model performance, so the original algorithms need to be optimized.

- AI algorithms can also optimize existing frameworks for decentralized and distributed learning, reducing statistical heterogeneity and system consistency, especially for reinforcement learning.

- The architecture of deep learning results in a huge amount of computation and is difficult to apply to resource-limited edge devices, so the combined framework of split learning and federated learning in DNNs needs to be considered.
- The graph, subgraph, and node levels of graph-based learning have shown their performance in distributed and decentralized learning, but large-scale graph data has not been well implemented.

VI. APPLICATION OF DISTRIBUTED LEARNING AND DECENTRALIZED LEARNING IN THE AIOT SYSTEM

This section will discuss five applications utilizing decentralized and distributed learning fusion in artificial intelligence Internet of things: (1) intelligent industrial Internet of things; (2) precision agriculture; (3) smart cities; (4) smart home; (5) smart healthcare. Moreover, more detailed applications are also described in these categories. Table 8 shows a summary of applications using decentralized and distributed learning for AIoT.

TABLE 8. Summary of applications using distributed and decentralized learning in AIoT.

Focus Area	Year	Study	Cloud	Edge	Fog
Intelligent Industrial Internet of Things	2021	[146]	✓	✓	
	2019	[147]	✓	✓	
	2020	[148]	✓	✓	
	2022	[149]	✓	✓	
Precision Agriculture	2022	[150]	✓	✓	
	2023	[151]	✓	✓	
	2022	[152]	✓	✓	
Smart City (Smart Transportation)	2023	[153]		✓	
	2022	[154]		✓	✓
	2020	[155]	✓	✓	
Smart City (Smart Grid)	2021	[156]	✓	✓	
	2022	[157]	✓	✓	
Smart City (UAVs)	2021	[158]		✓	
	2023	[159]		✓	
Smart Home	2020	[160]	✓	✓	
	2020	[161]	✓	✓	
	2022	[162]		✓	
Smart Healthcare	2022	[163]		✓	
	2022	[164]		✓	
	2018	[70]	✓	✓	
	2022	[165]	✓	✓	

A. INTELLIGENT INDUSTRIAL INTERNET OF THINGS

The Industrial Internet of Things (IIoT) is to integrate all kinds of acquisition and control sensors or controllers with perception and control capabilities into all aspects of the industrial production process through IoT perception and communication technology, thereby improving production efficiency and reducing product costs and resource consumption. However, since the data information of IIoT is usually related to the safe production of engineering manufacturing and the fault diagnosis of machines, traditional centralized learning methods are no longer suitable for intelligent IIoT, which needs to protect the privacy of industrial data. Therefore, decentralized and distributed learning as an emerging collaborative learning paradigm for data privacy protection has been utilized and optimized by researchers to develop more effective solutions in this area.

Zhang et al. [146] introduced a three-tier federated learning architecture of device-edge-cloud in a distributed intelligent IIoT network and utilized deep reinforcement learning to dynamically select participating devices to rationally allocate computing resources in the network. The key to this approach is the "Federation reinforcement" (RoF) based on deep multi-agent reinforcement learning. Classical replay techniques for efficient learning and convergence are formulated through the actor-critic network architecture and off-policy formula so that RoF schemes can execute the decision to make the best device selection and resource allocation on edge servers. In addition, this method also combines elements such as maximum entropy to improve the exploration and stability of reinforcement learning. This paper presents a federal learning framework based on reinforcement learning that has the potential to minimize evaluation losses while complying with latency and energy constraints in an intelligent IIoT environment.

Industry 4.0 is an emerging concept that combines multiple technologies to solve data-driven industrial problems. Meng et al. [147] proposed a privacy-enhanced, non-interactive federated learning framework, PEFL, to prevent adversaries from exploiting shared parameters to disrupt industrial applications. PEFL implements differential privacy and encryption by using a distributed Gaussian mechanism to perturb local gradient vectors before aggregation operations, and only the server can properly decrypt them to protect participants' data privacy. This approach also ensures that the aggregator is forget-safe, preventing adversaries from leaking data from the intelligent IIoT network by acquiring local gradients and sharing parameters. While providing a high degree of data privacy, PEFL guarantees 97.5% accuracy and low communication overhead when implementing convolutional neural networks.

Robotic systems are also the domain of the Industrial Internet of Things, which automates the work of multiple robots on an industrial assembly line by managing them. Majcherczyk et al. [148] proposed a decentralized federated learning framework, Flow-FL, that can be applied to machine-connected teams for collective learning. Flow-FL is data-driven in the arrangement of learning rounds, and the global state of the framework is maintained in shared memory based on the Gossip protocol. Specifically, the weights of the global model are stored in the shared memory, and when one of the robots has collected enough training data, the model weights are extracted from the distributed shared memory to build a local model. The effectiveness of Flow-FL is verified by the problem of agent trajectory prediction in robot systems. In addition, Ho et al. [149] also studied a joint deep reinforcement learning framework based on the federated learning FedAvg aggregation algorithm. In this paper, proximal policy optimization (PPO) is adopted to realize the task scheduling of an automated warehouse with heterogeneous autonomous robot systems. Compared with other distributed learning algorithms, the performance of average queue length is improved in this method.

B. PRECISION AGRICULTURE

Precision agriculture (also called Smart agriculture) is a modern agricultural technology that combines IoT systems and AI algorithms to make precise decisions and planning by analyzing and learning from large amounts of agricultural data, such as soil characteristics, climate change, etc., which is collected from IoT sensors. This system has been widely used in land use efficiency, quality inspection, and farm and resource management to minimize labor and cost and improve agricultural production efficiency [166]. Collaborative communication technology and machine learning models have been used to predict agricultural productivity and reduce production risks, but most of these data exist in government departments or individual farmers who are unwilling to share data, so traditional distributed learning has challenges.

Due to weather data, soil data, and crop data being decentralized, crop productivity forecasting requires collaborative learning from multiple participants. T et al. [150] developed a federal learning framework based on deep residual network regression models such as ResNet-16 and ResNet-28 to predict soybean yield in a decentralized environment using the FedAvg algorithm. The paper uses a dataset containing three types of data, including weather data, soil composition, and crop management data. The data is distributed across different clients and is partitioned horizontally, so clients all use the same set of features to distribute the data. Based on the performance indexes of MSE, RMSE, MAE, and r, this method performs better than traditional data-centralized training.

Idoje et al. [151] designed an ultra-tuned federal average model for smart farms, which aims to build a smart farm network by adopting a multi-label agricultural data set, with climate data as the independent variable and crop type as the label, to predict the type in the farm. In this paper, the Gaussian naive Bayes classifier model is adopted and built in a decentralized platform duet. Local data is trained by edge nodes on the farm, and updated weights are sent to the aggregator to complete model convergence. The proposed model evaluated various harmonic average values with crop category as the label, and the optimal harmonic average was generated by the FedAvg model.

Pest diseases have also been a major problem affecting agricultural productivity. As traditional pest detection is faced with problems such as uneven and insufficient crop data and diversity of pests and diseases, Deng et al. [152] proposed a fast regional convolutional neural network (R-CNN) based on federated learning technology to solve diseases and pests in orchards. The R-CNN network in the framework was replaced by ResNet-101 to prevent problems with gradient dispersion and gradient explosions during training and to ensure the original structure of small-size targets (pests). In this paper, the distributed computing paradigm based on federated learning can realize a shared model that integrates the data advantages of all parties in the case of data isolation. The FedAvg algorithm is improved by adding a restriction to prevent the large difference between the local model and the global model, and a fixed period is set to obtain the optimal solution in convergence speed and communication cost. The accuracy of the improved distributed computing model in multiple pest detection can reach 89.34%, and the target detection training speed is increased by 59% compared with the benchmark.

C. SMART CITIES

With the surge in urban population, urbanization in various countries is faced with many challenges, such as traffic jams, waste of resources, and urban planning. As a kind of AIoT concept, smart cities are proposed by integrating IoT devices and sensors to collect multiple types of data about different areas, such as vehicle flow, wastewater discharge, smart grid, and other data, and learn and analyze these data through AI technology to provide more effective decision-making proposals for city managers. Traditional centralized learning methods that rely on cloud computing have been unable to adapt to the diversified expansion of devices in smart cities while emerging decentralized and distributed learning realize decentralized smart city applications and ensure that data privacy is not leaked. This section will summarize the applications of smart cities from three aspects: intelligent transportation, smart grid, and Unmanned Aerial Vehicle (UAV) management.

1) SMART TRANSPORTATION

Intelligent transportation brings the information of the sensors of the traffic control system and the information of the vehicle to the edge of the Internet of Things network to achieve collaborative training of the global model without damaging the personal information of the vehicle and improve the efficiency of traffic scheduling. Xu et al. [153] designed an asynchronous federated learning scheme, DBAFL, based on the dynamic scaling factor of blockchain to learn traffic conditions over time to achieve intelligent public transportation, helping drivers improve driving safety and fuel utilization efficiency. DBAFL is built by the Bus and Roadside Unit (RSU) and introduces a new committee-based consensus algorithm that periodically selects new committees from the RSU based on the hash value of the latest block to defend against DDoS attacks. The experiment of DBAFL on heterogeneous devices proves that DBAFL has a low time cost and good performance in terms of privacy protection.

The Internet of Vehicles (IoV) is also a part of intelligent transportation, which realizes functions such as autonomous driving through collaborative learning of multiple vehicles. Xie et al. [154] utilized the federal learning framework FedSNN in the networking of vehicles to enable multiple vehicles to cooperate in training traffic sign recognition tasks and introduced a spike neural network (SNN) based on neural receptive field to achieve higher accuracy recognition by extracting information from the pixels and spatial dimensions of traffic signs. In FedSNN, connected vehicles use the local traffic sign data set to train the local model of SNN and then upload it to the neighboring RSU node for model parameter aggregation and loop until the model converges. Similarly, Liu et al. [155] proposed a federated learn-based gated recurrent unit neural network algorithm (FedGRU) for collaborative learning in vehicle networking for traffic flow prediction.

2) SMART GRID

A smart grid is a branch of smart cities that utilizes artificial intelligence to learn real-time data collected by IoT sensors deployed on the grid to provide more stable, cost-effective, and safe power regulation decisions for the grid. Due to the privacy of power resources, AIoT, based on the emerging decentralized and distributed learning framework, has been widely used in smart grids. Su et al. [156] propose a hierarchical federated learning framework with edge-cloud collaboration and employ Deep Q networks (DQN) of deep reinforcement learning to push the best training strategy to the user's local model based on multidimensional user privacy information and state space. The proposed framework also adds incentives to prevent free riders from taking more of the profits learned in the network. The paper uses simulators to demonstrate that federal deep reinforcement learning in smart grids demonstrates the possibility of collaborative learning between multiple users without sharing data.

The security of the grid is also a constant consideration. Ashraf et al. [157] developed a federal learning approach FedDP for data privacy, using a federal vote classifier (FVC) that is based on majority pass consensus to select traditional machine learning methods such as SVM, KNN, and RF for detecting energy theft in smart grids. FedDP consists of a theft detection station (TDS) and a central server (CS), where TDS is a low-power device that obtains real-time data of energy utilization from smart meters for storage and is responsible for uploading the local newly connected model parameters to CS, and CS is responsible for collecting the model parameters of TDS to train the global model. Compared with the existing model, FedDP has the highest accuracy, which can reach 91.67%, and is suitable for small edge nodes with limited computing resources.

3) UAV MANAGEMENT

Unmanned Aerial Vehicles can also be called drones in various areas of the smart city and play roles such as the distribution of goods, vehicle flow monitoring, and so on. Since the drone swarm is decentralized, it needs to communicate with each other and learn collaboratively to complete the task, so distributed learning and decentralized learning are suitable for the Internet of drones (IoD). Donevski et al. [158] investigated the problem of distributed node scheduling transmission for unmanned aerial vehicles (UAVs) and proposed a federated learning framework for UAV orchestration. In this framework, continuous convex programming and deep reinforcement learning are used to solve the complex trajectory planning optimization problem of UAV static nodes and improve the complex node arrangement of UAVs. For the path planning problem of UAVs, Gad et al. [159] proposed a federated learning algorithm based on knowledge distillation, which uses soft labels to reduce the communication overhead between UAVs and uses a self-organized map (SOM) algorithm to represent the topology of UAVs nodes, so as to generate the best UAVs path planning for intelligent monitoring of sparsely populated areas.

D. SMART HOME

The smart home is a subset of pervasive computing, which uses communication technology and intelligent control technology to connect small devices in the home (e.g., lighting systems, audio and video equipment, air conditioning control, security systems, network appliances, etc.) together to form the Internet of Things network, establishing an environment full of computing and communication capabilities. The integration of AIoT technology into smart home is to learn the user's preferences through environmental perception and user preference analysis so as to adapt to adjust the user's living environment, thereby improving the intelligence and comfort of living. Each perceptron in the smart home involves the user's personal privacy, so in order to solve the problem of data privacy, the emerging distributed and decentralized learning framework provides a better outcome solution for AIoT technology.

In response to the problem that AIoT in smart homes may lead to user privacy disclosure or cyber-attacks, Yu et al. [160] proposed a multi-task federated learning framework, LoFTI, to learn customized context-aware strategies from multiple smart homes to prevent cyber threats, such as unconventional automatic window opening. LOFTI builds historical data sets by collecting IoT access and context information records from edge nodes of the smart home, where edge nodes extract key features from the data set to capture contextual access patterns. The learning framework will use federated multi-task learning to build machine learning models and will be trained through federated learning patterns to achieve situational awareness and avoid abnormal access to the smart home from outside. LOFTI's false positive rate is 49.5% lower than the most advanced whole-home learning.

Lee et al. [161] proposed a federated learning model based on deep reinforcement learning consisting of a local home energy management system (LHEMS) and a global server (GS) to intelligently manage the energy consumption of multiple smart homes with household appliances, solar photovoltaic systems, etc. Each LHEMS in the framework uses the energy consumption data based on the global model to build a local model, and its GS, as an aggregator, uses the FedSGD algorithm to build a global model. The framework proposed in this paper is suitable for many households with different electrical parameters and comfort requirements.

The consumer Internet of Things (CIoT) is a branch of the smart home and is an Internet network composed of the Internet of Things terminals used by the consumer home. Access to the cellular network is its main mode of communication, so there will be more restrictions on data sharing. For cross-island CIoT devices in smart homes, Rasti-Meymandi et al. [162] proposed a new personalized graph Federation IoT learning framework GFIoTL based on graph filtering and a new graph signal processing (GSP) aggregation rule called G-Fedfilt. GFIoT can not only aggregate the gradient of the device but also customize its special model according to the needs of each device. The essence of the G-Fedfilt aggregation algorithm is to consider domain-specific and domain-independent gradient updating. It uses a graph filter to aggregate the model parameters of edge devices and incorporates the FedAvg aggregation algorithm in a special case to make federated learning adjustably personalized. Different from other personalized federated learning, GFIoTL considers the relationship of edge devices on the graph network, and the aggregation rules can also cluster edge devices based on the connectivity of the graph. Compared with the traditional FedAvg, the classification accuracy of GFIoTL is improved by 3.99%, and the communication efficiency is higher under the condition of system heterogeneity.

E. SMART HEALTHCARE

Smart healthcare can also become the Internet of Medical Things (IoMT), collecting data through the use of responsive devices or sensors to check and monitor a patient's physical state to improve the accuracy of detection and the efficiency of diagnosis. With the proliferation of patients and the diversification of treatment methods, the data generated by IoMT increases, and AIoT learns various data through AI technology to help doctors make auxiliary decisions and shorten treatment time, providing a convenient medical environment for patients. However, large and diverse data is difficult to find in a single healthcare institution, and patient data is subject to strict privacy restrictions, making it difficult to collect and share. Therefore, the emerging decentralized and distributed learning technology solves this problem well and is applied in many fields of IoMT, including medical imaging diagnosis, wearable medical monitoring, and online medical systems.

The decentralized learning framework is most widely used in smart health because of its stricter constraint on data sharing. Lian et al. [163] propose a privacy-enhanced decentralized, federated learning system, DEEP-FEL, that allows different medical devices to learn collaboratively without sharing raw data. DEEP-FEL is a decentralized architecture of layered ring topology that utilizes edge servers to communicate and store data and build local training models to exchange and aggregate model parameters with other institutions. The RingAVG algorithm is designed to optimize the aggregation algorithm of the ring topology and update the local model by receiving and aggregating model parameters from different medical institutions. During the communication process, manual perturbation is also added to DEEP-FEL to enhance the privacy protection of model parameters. On the data sets of skin cancer and COVID-19 scans, the proposed framework has good performance in terms of communication efficiency and privacy protection.

Similarly, Tedeschini et al. [164] have proposed a fully decentralized federated learning framework for the diagnosis of medical imaging. The proposed framework is based on the consensus-driven Federal Average method (CFA), which enables the full decentralization of point-to-point communication links by enabling healthcare sites to directly send local model parameters to authenticated other healthcare sites without a parameter server (PS). This paper also uses message queue-based Telemetry Transfer (MQTT) transport protocol to realize real-time local model parameter exchange between heterogeneous medical nodes and proposes a set of optimized information embedded in the payload of MQTT to represent the real-time learning process in each period. The predictive performance and real-time performance of the proposed framework are confirmed by a real-world case of brain tumor segmentation.

Split learning, as a distributed learning framework that can conduct collaborative learning in multiple modes of patient data, also has good effects in smart healthcare. Vepakomma et al. [70] proposed splitNN, a split-learning framework, for the task of training local deep learning networks in each healthcare department and conducting multi-party collaborative training of the same global model by passing the parameters of the local model. SplitNN can effectively reduce the parameters of model training and make the IoMT network more lightweight. Furthermore, Yoo et al. [165] proposed a new split-learning concept, "multi-site split-learning," to realize a global medical image classification model shared and cooperated by multiple hospitals under the protection of data privacy. The proposed framework includes multiple hospitals and a server, and the hospital only needs to train the first hidden layer and transmit the confidential feature map to the server so that the patient's private information is retained locally. Medical data on CT scans, X-ray bone scans, and cholesterol levels of COVID-19 patients are used to demonstrate the performance of the proposed framework in privacy protection and medical imaging diagnosis.

VII. OPEN CHALLENGES AND OPPORTUNITIES

This section presents various open challenges encountered in decentralized and distributed learning for AIoT. The challenges captured in this work include privacy security protection, real-time collaborative learning, incentive mechanisms, multimodal distributed and decentralized learning, heterogeneous challenges and opportunities for management, network, and data sharing in AIoT.

A. PRIVACY SECURITY PROTECTION

Although decentralized learning paradigms such as federated learning and split learning are designed to prevent the leakage of data privacy, data privacy, and system security protection are still the challenges in the inference of models and the transmission of parameters. 1) In the phase of transmitting model gradients or parameters, the attacker will deduce against local or global model parameters to obtain information from the client. At present, the defense method adds noise to the transmitted parameters through encryption technology, such as differential privacy, at the expense of model convergence efficiency and accuracy in exchange for data privacy protection. 2) In the model inference phase, an attacker will steal training data or local models, such as Byzantine attacks and backdoor attacks, to affect the performance of the global model or sub-task. In [167], authors protect the privacy and security of distributed systems by reducing the variance of stochastic gradient as a means of joint variance reduction of stochastic gradient descent, which is robust in Byzantine attacks. Therefore, developing more lightweight privacy or encryption algorithms and more generalized robust distributed and decentralized learning is a direction worth studying.

In addition, although encryption technology protects the privacy and security of parameters during transmission, it still faces some challenges. Some encryption techniques, such as differential privacy, add different levels of noise to the parameters. This noise can lead to a decrease in model accuracy, and for devices with limited resources, it is difficult for their computing resources to handle parameters injected with noise. On the other hand, even if some encryption techniques are lossless, they will still increase the communication overhead of distributed learning. Therefore, designing powerful privacy protection platforms may become an opportunity for distributed learning in the future direction.

B. REAL-TIME COLLABORATIVE LEARNING

Real-time is a challenge for distributed and decentralized learning for AIoT. Distributed and decentralized learning usually uses synchronous protocols for model aggregation, which makes it difficult to adapt to rapidly changing environmental information and heterogeneous hardware settings. In [168], authors proposed a hierarchical pace control framework to coordinate the overall training progress in a federated learning system. Therefore, it is a solution to introduce an adaptive asynchronous aggregation protocol for consuming real-time streaming data.

Another challenge to achieving real-time collaborative learning is the deployment and implementation of AI models in IoT devices. Therefore, in addition to the improvement of communication protocols, the improvement of acceleration technology and deep learning models is also a future direction. From the perspective of hardware adaptation of AI models, acceleration techniques such as pruning, and quantification can reduce the pressure on storage and computing resources that exist when the model is deployed on the device side. This frees most devices from their dependence on cloud servers, enabling real-time collaborative training. But this affects the accuracy of the AI model. From the perspective of hardware, the development of lightweight hardware to support the training of AI models is also one of the future directions.

C. INCENTIVE MECHANISM

Participating in distributed and decentralized learning consumes computing resources, hogs network bandwidth, and shortens the life of edge devices. Most popular distributed training algorithms use small batches of random gradient descent, which in actual training requires waiting for the slowest device in each synchronization batch, resulting in random optimization of full synchronization tends to be slow, i.e. subject to the ''lag effect'', which is more pronounced in heterogeneous networks. At present, most research assume that all terminal devices can participate in distributed learning unconditionally, but the actual situation is that ''selfish'' end devices will not provide enough resources to participate in distributed learning, which will affect the training efficiency of distributed learning. Therefore, the lack of adequate incentives to incentivize clients to bear these costs and contribute is a challenge at this stage.

In order to solve this problem, the managers of the system should consider the establishment of an incentive mechanism from two angles: 1) Evaluate the contributions of each participant; 2) Give participants reasonable rewards. He et al. [169] introduced contract theory as an incentive mechanism for federated learning, designing contracts by analyzing the cost and data label distribution differences of participants' devices. Game theory is also one of the future directions of incentive mechanism in distributed learning. Different from the incentive based on the fair mechanism, it is based on the untrustworthy risk of encouraging more devices to contribute their resources through different rewards. Research on incentive-driven distributed and decentralized learning is an important direction in the future, which can build more efficient AIoT systems.

D. MULTIMODAL DISTRIBUTED AND DECENTRALIZED LEARNING

Although there has been a lot of research on decentralized learning and distributed learning in structured data such as images, audio, text, etc., the application of utilizing multimodal data flows in AIoT has not yet been explored. However, in AIoT system, the data generated by different sensors and devices usually have different modes, such as tactile, visual, and auditory, so it is crucial to establish a multi-modal distributed learning framework. Feng et al. [170] proposed FedMultimodal as a benchmark for federated learning in multimodal learning, covering five representative multimodal applications. Therefore, the study of distributed and decentralized learning algorithms on multi-modes to achieve extreme data heterogeneity is a future direction in the field of AIoT.

Current research on multimodal distributed learning mainly uses the following methods: 1) A new representation space is established before the final decision level to mix the representation of data of different modes. 2) Extract the representation from different modes and send it to the server to align the different modes [171]. These two methods have privacy and security problems from the client and server levels respectively. Therefore, developing more secure distributed and distributed learning models and improving their robustness are also future research directions.

E. HETEROGENEOUS CHALLENGE

As mentioned in the article, statistical heterogeneity and system heterogeneity are the most challenging issues in distributed and decentralized learning for AIoT. These challenges are due to changes in client device hardware conditions (CPU, memory), network connection (3G, 4G, 5G, WiFi), and power supply (battery power), each device in the distributed learning network may have different storage, computing, and communication capabilities. Network and device limitations can cause only a few devices to be active at a time. Moreover, devices usually generate and collect data on the network in different distribution ways, and the amount and characteristics of data across devices may vary greatly, so the data in the federal learning network is non-independent and identically distribute. At present, the mainstream machine learning algorithms are mainly established based on the assumptions of IID data. Therefore, heterogeneous non-IID data features pose great challenges to modeling, analysis and evaluation. A lot of work has been done to address these challenges, but another effective approach called personalized federated learning mentioned in the article is personalization at the device, data, and model level to mitigate heterogeneity and get high-quality personalized models for each device. Personalized federated learning may be the focus of future research on heterogeneity challenges. In addition, the exploration of providing standard communication protocols for heterogeneous devices and networks to support the communication quality of different types of edge devices and cloud servers is also an urgent challenge for distributed and decentralized learning.

F. FURTHER CHALLENGES AND OPPORTUNITIES FOR MANAGEMENT, NETWORKING, AND DATA SHARING IN AIOT

With the deployment of large-scale end devices and the explosive growth of data volume of AIoT, wireless communication has been widely used in distributed learning and decentralized learning technologies. Distributed learning, such as federated learning, can learn and infer models locally without sharing data, thus reducing data exchange and bandwidth usage in wireless communication networks. In fact, bandwidth limitations and network latency also remain major obstacles. Distributed learning requires frequent parameter transmission and model synchronization, and the existing network infrastructure may not be able to meet the demand for high bandwidth and low latency, resulting in inefficient data transmission and slower model training. Especially in edge computing environments, where the stability of network connections is critical, any network interruption can lead to data loss or model inconsistencies during training. In addition, distributed sensors and devices are often powered by batteries with limited computing and storage capacity, making it difficult to support highly dynamic service environments. So, efficiently allocating and scheduling communication and computing resources to optimize overall system performance is a challenge.

To address these challenges, various communication offloading, caching, and cloud, fog, and edge computing mechanisms allocate resources across heterogeneous networks, respectively, becoming solutions for low latency and on-demand services. Sun et al. [172] optimized joint caching and computation strategies to minimize transmission bandwidth under latency and local cache constraints. With the introduction of the 6th generation wireless networks (6G) concept, distributed and decentralized learning in AIoT will become more dependent on reliable, low-latency network management [173]. AI/ ML-based management and network resource allocation and scheduling technologies, such as DRL, will become more reliable solutions to support the networks of the future. Nguyen et al. [141] proposed a DQL algorithm for resource allocation in a mobile-aware federated learning network using a deep Q-Network (DQN), which can find optimal decisions about energy and channels without any prior network knowledge.

VIII. CONCLUSION

This paper has presented a comprehensive survey of paradigms for decentralized and distributed learning for artificial intelligent Internet of Things. This review aims to provide useful references and discuss how to apply these new paradigms to solve the collaborative learning of large-scale edge devices based on data privacy in AIoT. This survey covers the various patterns and architectures of decentralized and distributed learning and gives insights into the optimization algorithms of these paradigms, such as in terms of data, devices, and parameter transmission. The paper discusses the generality and empowerment provided by AI algorithms in AIoT over different distributed and decentralized learning frameworks, including machine learning, deep learning, and reinforcement learning. Use cases and applications for decentralized and distributed learning for AIoT have been given. The review has also uncovered open challenges that remain to be resolved and given recommendations for future work in the following areas: (1) Data privacy and model security for AIoT systems; (2) Real-time processing and collaborative learning based on real-time data streams; (3) Incentive mechanism for participants; (4) Decentralized and distributed learning for dealing with multimodal data; and (5) Statistical, systems and model heterogeneity challenges.

REFERENCES

[1] L. Shen, F. Wang, M. Zhang, J. Liu, G. Liu, and X. Fan, "AIoT-empowered smart grid energy management with distributed control and non-intrusive load monitoring," in *Proc. IEEE/ACM 31st Int. Symp. Quality Service (IWQoS)*, Jun. 2023, pp. 1–10, doi: 10.1109/IWQoS57198.2023.10188781.

[2] M. Derawi, Y. Dalveren, and F. A. Cheikh, "Internet-of-Things-based smart transportation systems for safer roads," in *Proc. IEEE 6th World Forum Internet Things (WF-IoT)*, Jun. 2020, pp. 1–4, doi: 10.1109/WF-IoT48130.2020.9221208.

[3] M. A. Rahman, M. S. Hossain, A. J. Showail, N. A. Alrajeh, and A. Ghoneim, "AI-enabled IIoT for live smart city event monitoring," *IEEE Internet Things J.*, vol. 10, no. 4, pp. 2872–2880, Feb. 2023, doi: 10.1109/JIOT.2021.3109435.

[4] Z. Wang, Y. Hu, S. Yan, Z. Wang, R. Hou, and C. Wu, "Efficient ring-topology decentralized federated learning with deep generative models for medical data in eHealthcare systems," *Electronics*, vol. 11, no. 10, p. 1548, May 2022, doi: 10.3390/electronics11101548.

[5] M. S. Al-Abiad and M. J. Hossain, "Coordinated scheduling and decentralized federated learning using conflict clustering graphs in fog-assisted IoD networks," *IEEE Trans. Veh. Technol.*, vol. 72, no. 3, pp. 3455–3472, Mar. 2023, doi: 10.1109/TVT.2022.3217963.

[6] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-IID data with reinforcement learning," in *Proc. IEEE Conf. Comput. Commun. (IEEE INFOCOM)*, Jul. 2020, pp. 1698–1707, doi: 10.1109/INFOCOM41043.2020.9155494.

[7] S. Abdulrahman, H. Tout, H. Ould-Slimane, A. Mourad, C. Talhi, and M. Guizani, "A survey on federated learning: The journey from centralized to distributed on-site learning and beyond," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5476–5497, Apr. 2021, doi: 10.1109/JIOT.2020.3030072.

[8] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for Internet of Things: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1622–1658, 3rd Quart., 2021, doi: 10.1109/COMST.2021.3075439.

[9] E. T. Martínez Beltrán, M. Q. Pérez, P. M. S. Sánchez, S. L. Bernal, G. Bovet, M. G. Pérez, G. M. Pérez, and A. H. Celdrán, "Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 25, no. 4, pp. 2983–3013, Sep. 2023, doi: 10.1109/COMST.2023.3315746.

[10] A. Z. Tan, H. Yu, L. Cui, and Q. Yang, "Towards personalized federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 12, pp. 9587–9603, Dec. 2023, doi: 10.1109/TNNLS.2022.3160699.

[11] E. Baccour, N. Mhaisen, A. A. Abdellatif, A. Erbad, A. Mohamed, M. Hamdi, and M. Guizani, "Pervasive AI for IoT applications: A survey on resource-efficient distributed artificial intelligence," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 4, pp. 2366–2418, 4th Quart., 2022, doi: 10.1109/COMST.2022.3200740.

[12] M. U. Afzal, A. A. Abdellatif, M. Zubair, M. Q. Mehmood, and Y. Massoud, "Privacy and security in distributed learning: A review of challenges, solutions, and open research issues," *IEEE Access*, vol. 11, pp. 114562–114581, 2023, doi: 10.1109/ACCESS.2023.3323932.

[13] S. Hu, X. Chen, W. Ni, E. Hossain, and X. Wang, "Distributed machine learning for wireless communication networks: Techniques, architectures, and applications," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1458–1493, 3rd Quart., 2021, doi: 10.1109/COMST.2021.3086014.

[14] Z. Chang, S. Liu, X. Xiong, Z. Cai, and G. Tu, "A survey of recent advances in edge-computing-powered artificial intelligence of things," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 13849–13875, Sep. 2021, doi: 10.1109/JIOT.2021.3088875.

[15] Y. Sun, H. Ochiai, and H. Esaki, "Decentralized deep learning for multi-access edge computing: A survey on communication efficiency and trustworthiness," *IEEE Trans. Artif. Intell.*, vol. 3, no. 6, pp. 963–972, Dec. 2022, doi: 10.1109/TAI.2021.3133819.

[16] P. Joshi, M. Hasanuzzaman, C. Thapa, H. Afli, and T. Scully, "Enabling all in-edge deep learning: A literature review," *IEEE Access*, vol. 11, pp. 3431–3460, 2023, doi: 10.1109/ACCESS.2023.3234761.

[17] C. Shiranthika, P. Saeedi, and I. V. Bajic, "Decentralized learning in healthcare: A review of emerging techniques," *IEEE Access*, vol. 11, pp. 54188–54209, 2023, doi: 10.1109/ACCESS.2023.3281832.

[18] Q. Duan, S. Hu, R. Deng, and Z. Lu, "Combined federated and split learning in edge computing for ubiquitous intelligence in Internet of Things: State-of-the-art and future directions," *Sensors*, vol. 22, no. 16, p. 5983, Aug. 2022, doi: 10.3390/s22165983.

[19] Q. W. Khan, A. N. Khan, A. Rizwan, R. Ahmad, S. Khan, and D.-H. Kim, "Decentralized machine learning training: A survey on synchronization, consolidation, and topologies," *IEEE Access*, vol. 11, pp. 68031–68050, 2023, doi: 10.1109/ACCESS.2023.3284976.

[20] A. Grammenos, R. Mendoza-Smith, J. Crowcroft, and C. Mascolo, "Federated principal component analysis," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33. Red Hook, NY, USA: Curran Associates, 2020, pp. 6453–6464. Accessed: Dec. 19, 2023. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/47a658229eb2368a99f1d032c8848542-Abstract.html

[21] W. Briguglio, W. A. Yousef, I. Traore, and M. Mamun, "Federated supervised principal component analysis," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 646–660, 2024, doi: 10.1109/TIFS.2023.3326981.

[22] P. Narayanamurthy, N. Vaswani, and A. Ramamoorthy, "Federated over-air robust subspace tracking from missing data," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2022, pp. 5598–5602, doi: 10.1109/ICASSP43922.2022.9747220.

[23] H. Zhu and Y. Jin, "Real-time federated evolutionary neural architecture search," *IEEE Trans. Evol. Comput.*, vol. 26, no. 2, pp. 364–378, Apr. 2022, doi: 10.1109/TEVC.2021.3099448.

[24] A. Nandi and F. Xhafa, "A federated learning method for real-time emotion state classification from multi-modal streaming," *Methods*, vol. 204, pp. 340–347, Aug. 2022, doi: 10.1016/j.ymeth.2022.03.005.

[25] S. M. Rajagopal and R. Buyya, "FedSDM: Federated learning based smart decision making module for ECG data in IoT integrated edge–fog–cloud computing environments," *Internet Things*, vol. 22, Jul. 2023, Art. no. 100784, doi: 10.1016/j.iot.2023.100784.

[26] R. Saha, S. Misra, and P. K. Deb, "FogFL: Fog-assisted federated learning for resource-constrained IoT devices," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 8456–8463, May 2021, doi: 10.1109/JIOT.2020.3046509.

[27] I. B. Lahmar and K. Boukadi, "Resource allocation in fog computing: A systematic mapping study," in *Proc. 5th Int. Conf. Fog Mobile Edge Comput. (FMEC)*, Apr. 2020, pp. 86–93, doi: 10.1109/fmec49853.2020.9144705.

[28] V. Sethi and S. Pal, "FedDOVe: A federated deep Q-learning-based offloading for vehicular fog computing," *Future Gener. Comput. Syst.*, vol. 141, pp. 96–105, Apr. 2023, doi: 10.1016/j.future.2022.11.012.

[29] P. Dube, T. Suk, and C. Wang, "AI gauge: Runtime estimation for deep learning in the cloud," in *Proc. 31st Int. Symp. Comput. Archit. High Perform. Comput. (SBAC-PAD)*, Oct. 2019, pp. 160–167, doi: 10.1109/SBAC-PAD.2019.00035.

[30] Y. Xiao and M. Krunz, "AdaptiveFog: A modelling and optimization framework for fog computing in intelligent transportation systems," *IEEE Trans. Mobile Comput.*, vol. 21, no. 12, pp. 4187–4200, Dec. 2022, doi: 10.1109/TMC.2021.3080397.

[31] R. Beri, M. K. Dubey, A. Gehlot, R. Singh, M. Abd-Elnaby, and A. Singh, "A novel fog-computing-assisted architecture of E-healthcare system for pregnant women," *J. Supercomput.*, vol. 78, no. 6, pp. 7591–7615, Apr. 2022, doi: 10.1007/s11227-021-04176-7.

[32] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. 41st Annu. ACM Symp. Theory Comput.* New York, NY, USA: Association for Computing Machinery, May 2009, pp. 169–178, doi: 10.1145/1536414.1536440.

[33] Y. Jiang, J. Hamer, C. Wang, X. Jiang, M. Kim, Y. Song, Y. Xia, N. Mohammed, M. N. Sadat, and S. Wang, "SecureLR: Secure logistic regression model via a hybrid cryptographic protocol," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 16, no. 1, pp. 113–123, Jan. 2019, doi: 10.1109/TCBB.2018.2833463.

[34] N. Busom, R. Petrlic, F. Sebé, C. Sorge, and M. Valls, "Efficient smart metering based on homomorphic encryption," *Comput. Commun.*, vol. 82, pp. 95–101, May 2016, doi: 10.1016/j.comcom.2015.08.016.

[35] N. Dowlin, R. Gilad-Bachrach, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proc. 33rd Int. Conf. Mach. Learn.*, vol. 48, New York, NY, USA, Jun. 2016, pp. 201–210.

[36] L. Cheng, S. Cheng, and F. Jiang, "ADKAM: A-diversity K-anonymity model via microaggregation," in *Information Security Practice and Experience* (Lecture Notes in Computer Science), J. Lopez and Y. Wu, Eds. Cham, Switzerland: Springer, 2015, pp. 533–547, doi: 10.1007/978-3-319-17533-1_36.

[37] A. Machanavajjhala, J. Gehrke, D. Kifer, and M. Venkitasubramaniam, "*L*-diversity: Privacy beyond *k*-anonymity," in *Proc. 22nd Int. Conf. Data Eng.*, Apr. 2006, p. 24, doi: 10.1109/ICDE.2006.1.

[38] K. A. Hashem, "T-proximity compatible with T-neighbourhood structure," *J. Egyptian Math. Soc.*, vol. 20, no. 2, pp. 108–115, Jul. 2012, doi: 10.1016/j.joems.2012.08.004.

[39] T. Wang, Y. Mei, W. Jia, X. Zheng, G. Wang, and M. Xie, "Edge-based differential privacy computing for sensor–cloud systems," *J. Parallel Distrib. Comput.*, vol. 136, pp. 75–85, Feb. 2020, doi: 10.1016/j.jpdc.2019.10.009.

[40] B. I. P. Rubinstein, P. L. Bartlett, L. Huang, and N. Taft, "Learning in a large function space: Privacy-preserving mechanisms for SVM learning," *J. Privacy Confidentiality*, vol. 4, no. 1, pp. 65–100, Jul. 2012.

[41] T. Zhang and Q. Zhu, "Dynamic differential privacy for ADMM-based distributed classification learning," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 1, pp. 172–187, Jan. 2017, doi: 10.1109/TIFS.2016.2607691.

[42] A. D. Sarwate and K. Chaudhuri, "Signal processing and machine learning with differential privacy: Algorithms and challenges for continuous data," *IEEE Signal Process. Mag.*, vol. 30, no. 5, pp. 86–94, Sep. 2013, doi: 10.1109/MSP.2013.2259911.

[43] Y. Cao, F. Yu, and Y. Tang, "A digital watermarking encryption technique based on FPGA cloud accelerator," *IEEE Access*, vol. 8, pp. 11800–11814, 2020, doi: 10.1109/ACCESS.2020.2966251.

[44] X. Dong, W. Zhang, M. Shah, B. Wang, and N. Yu, "Watermarking-based secure plaintext image protocols for storage, show, deletion and retrieval in the cloud," *IEEE Trans. Services Comput.*, vol. 15, no. 3, pp. 1678–1692, May 2022, doi: 10.1109/TSC.2020.3008957.

[45] H. Cheng, Q. Huang, F. Chen, M. Wang, and W. Yan, "Privacy-preserving image watermark embedding method based on edge computing," *IEEE Access*, vol. 10, pp. 18570–18582, 2022, doi: 10.1109/ACCESS.2022.3151115.

[46] L. Xie, I. M. Baytas, K. Lin, and J. Zhou, "Privacy-preserving distributed multi-task learning with asynchronous updates," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Aug. 2017, pp. 1195–1204, doi: 10.1145/3097983.3098152.

[47] A. Vulimiri, C. Curino, P. B. Godfrey, T. Jungblut, J. Padhye, and G. Varghese, "Global analytics in the face of bandwidth and regulatory constraints," in *Proc. 12th USENIX Conf. Netw. Syst. Design Implement. (NSDI)*. Berkeley, CA, USA: USENIX Association, May 2015, pp. 323–336.

[48] A. C. Zhou, Y. Xiao, Y. Gong, B. He, J. Zhai, and R. Mao, "Privacy regulation aware process mapping in geo-distributed cloud data centers," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 8, pp. 1872–1888, Aug. 2019, doi: 10.1109/TPDS.2019.2896894.

[49] Y. Zhang, D. Ramage, Z. Xu, Y. Zhang, S. Zhai, and P. Kairouz, "Private federated learning in Gboard," 2023, *arXiv:2306.14793*. Accessed: Dec. 14, 2023.

[50] H. Brendan McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," Jan. 2016, *arXiv:1602.05629*. Accessed: Dec. 14, 2023.

[51] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proc. 37th Int. Conf. Mach. Learn. (ICML)*, vol. 119, Jul. 2020, pp. 5132–5143.

[52] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," 2019, *arXiv:1903.03934*. Accessed: Dec. 14, 2023.

[53] T.-M. Harry Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," 2019, *arXiv:1909.06335*. Accessed: Dec. 14, 2023.

[54] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*. Accessed: Dec. 14, 2023.

[55] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," 2020, *arXiv:2007.07481*. Accessed: Dec. 14, 2023.

[56] J. Hamer, M. Mohri, and A. T. Suresh, "FedBoost: Communication-efficient algorithms for federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 3973–3983.

[57] Y. Guo, X. Tang, and T. Lin, "FedBR: Improving federated learning on heterogeneous data via local learning bias reduction," 2022, *arXiv:2205.13462*. Accessed: Dec. 14, 2023.

[58] Q. Wu, K. He, and X. Chen, "Personalized federated learning for intelligent IoT applications: A cloud-edge based framework," *IEEE Open J. Comput. Soc.*, vol. 1, pp. 35–44, 2020, doi: 10.1109/OJCS.2020.2993259.

[59] Y. Mansour, M. Mohri, J. Ro, and A. T. Suresh, "Three approaches for personalization with applications to federated learning," 2020, *arXiv:2002.10619*. Accessed: Dec. 15, 2023.

[60] J. Guo, I. W. Ho, Y. Hou, and Z. Li, "FedPos: A federated transfer learning framework for CSI-based Wi-Fi indoor positioning," *IEEE Syst. J.*, vol. 17, no. 3, pp. 4579–4590, Sep. 2023, doi: 10.1109/JSYST.2022.3230425.

[61] J. Mills, J. Hu, and G. Min, "Multi-task federated learning for personalised deep neural networks in edge computing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 3, pp. 630–641, Mar. 2022, doi: 10.1109/TPDS.2021.3098467.

[62] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning: A meta-learning approach," 2020, *arXiv:2002.07948*. Accessed: Dec. 15, 2023.

[63] Y. Xu and H. Fan, "FedDK: Improving cyclic knowledge distillation for personalized healthcare federated learning," *IEEE Access*, vol. 11, pp. 72409–72417, 2023, doi: 10.1109/ACCESS.2023.3294812.

[64] L. Collins, H. Hassani, A. Mokhtari, and S. Shakkottai, "Exploiting shared representations for personalized federated learning," Mar. 2021, *arXiv:2102.07078*. Accessed: Dec. 15, 2023.

[65] F. Hanzely and P. Richtárik, "Federated learning of a mixture of global and local models," 2020, *arXiv:2002.05516*. Accessed: Dec. 15, 2023.

[66] O. Gupta and R. Raskar, "Distributed learning of deep neural network over multiple agents," *J. Netw. Comput. Appl.*, vol. 116, pp. 1–8, Aug. 2018, doi: 10.1016/j.jnca.2018.05.003.

[67] C. Thapa, M. A. P. Chamikara, and S. A. Camtepe, "Advancements of federated learning towards privacy preservation: From federated learning to split learning," in *Federated Learning Systems: Towards Next-Generation AI* (Studies in Computational Intelligence), M. H. U. Rehman and M. M. Gaber, Eds. Cham, Switzerland: Springer, 2021, pp. 79–109, doi: 10.1007/978-3-030-70604-3_4.

[68] S. Lyu, Z. Lin, G. Qu, X. Chen, X. Huang, and P. Li, "Optimal resource allocation for U-shaped parallel split learning," 2023, *arXiv:2308.08896*. Accessed: Dec. 15, 2023.

[69] I. Ceballos, V. Sharma, E. Mugica, A. Singh, A. Roman, P. Vepakomma, and R. Raskar, "SplitNN-driven vertical partitioning," 2020, *arXiv:2008.04137*. Accessed: Dec. 15, 2023.

[70] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar, "Split learning for health: Distributed deep learning without sharing raw patient data," 2018, *arXiv:1812.00564*. Accessed: Dec. 15, 2023.

[71] A. Singh, P. Vepakomma, O. Gupta, and R. Raskar, "Detailed comparison of communication efficiency of split learning and federated learning," 2019, *arXiv:1909.09145*. Accessed: Dec. 15, 2023.

[72] Y. Gao, M. Kim, S. Abuadbba, Y. Kim, C. Thapa, K. Kim, S. A. Camtep, H. Kim, and S. Nepal, "End-to-end evaluation of federated learning and split learning for Internet of Things," in *Proc. Int. Symp. Reliable Distrib. Syst. (SRDS)*, Sep. 2020, pp. 91–100, doi: 10.1109/SRDS51746.2020.00017.

[73] A. Chopra, S. K. Sahu, A. Singh, A. Java, P. Vepakomma, V. Sharma, and R. Raskar, "AdaSplit: Adaptive trade-offs for resource-constrained distributed deep learning," 2021, *arXiv:2112.01637*. Accessed: Dec. 15, 2023.

[74] A. Ayad, M. Renner, and A. Schmeink, "Improving the communication and computation efficiency of split learning for IoT applications," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2021, pp. 1–6, doi: 10.1109/GLOBECOM46510.2021.9685493.

[75] F. Zheng, C. Chen, L. Lyu, and B. Yao, "Reducing communication for split learning by randomized top-k sparsification," in *Proc. 32nd Int. Joint Conf. Artif. Intell.*, Macao, China, Aug. 2023, pp. 4665–4673, doi: 10.24963/ijcai.2023/519.

[76] C.-Y. Hsieh, Y.-C. Chuang, and A.-Y. Wu, "C3-SL: Circular convolution-based batch-wise compression for communication-efficient split learning," 2022, *arXiv:2207.12397*. Accessed: Dec. 15, 2023.

[77] X. Chen, J. Li, and C. Chakrabarti, "Communication and computation reduction for split learning using asynchronous training," in *Proc. IEEE Workshop Signal Process. Syst. (SiPS)*, Oct. 2021, pp. 76–81, doi: 10.1109/SiPS52927.2021.00022.

[78] D. Y. Zhang, Z. Kou, and D. Wang, "FedSens: A federated learning approach for smart health sensing with class imbalance in resource constrained edge computing," in *Proc. IEEE Conf. Comput. Commun.*, May 2021, pp. 1–10, doi: 10.1109/INFOCOM42981.2021.9488776.

[79] P. Joshi, C. Thapa, S. Camtepe, M. Hasanuzzaman, T. Scully, and H. Afli, "Performance and information leakage in splitfed learning and multi-head split learning in healthcare data and beyond," *Methods Protocols*, vol. 5, no. 4, p. 60, Jul. 2022, doi: 10.3390/mps5040060.

[80] C. Thapa, M. A. P. Chamikara, S. Camtepe, and L. Sun, "SplitFed: When federated learning meets split learning," 2020, *arXiv:2004.12088*. Accessed: Dec. 15, 2023.

[81] M. Gawali, C. S. Arvind, S. Suryavanshi, H. Madaan, A. Gaikwad, K. N. B. Prakash, V. Kulkarni, and A. Pant, "Comparison of privacy-preserving distributed deep learning methods in healthcare," 2020, *arXiv:2012.12591*. Accessed: Dec. 15, 2023.

[82] Y. Gao, M. Kim, C. Thapa, A. Abuadbba, Z. Zhang, S. Camtepe, H. Kim, and S. Nepal, "Evaluation and optimization of distributed machine learning techniques for Internet of Things," *IEEE Trans. Comput.*, vol. 71, no. 10, pp. 2538–2552, Oct. 2022, doi: 10.1109/TC.2021.3135752.

[83] Z. Zhang, A. Pinto, V. Turina, F. Esposito, and I. Matta, "Privacy and efficiency of communications in federated split learning," *IEEE Trans. Big Data*, vol. 9, no. 5, pp. 1380–1391, Oct. 2023, doi: 10.1109/TBDATA.2023.3280405.

[84] S. Park, G. Kim, J. Kim, B. Kim, and J. Chul Ye, "Federated split vision transformer for COVID-19 CXR diagnosis using task-agnostic training," 2021, *arXiv:2111.01338*. Accessed: Dec. 15, 2023.

[85] Y. Tian, Y. Wan, L. Lyu, D. Yao, H. Jin, and L. Sun, "FedBERT: When federated learning meets pre-training," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 4, pp. 66:1–66:26, Aug. 2022, doi: 10.1145/3510033.

[86] H. Jiang, M. Liu, S. Sun, Y. Wang, and X. Guo, "FedSyL: Computation-efficient federated synergy learning on heterogeneous IoT devices," in *Proc. IEEE/ACM 30th Int. Symp. Quality Service (IWQoS)*, Jun. 2022, pp. 1–10, doi: 10.1109/IWQoS54832.2022.9812907.

[87] R. Deng, X. Du, Z. Lu, Q. Duan, S.-C. Huang, and J. Wu, "HSFL: Efficient and privacy-preserving offloading for split and federated learning in IoT services," in *Proc. IEEE Int. Conf. Web Services (ICWS)*, Jul. 2023, pp. 658–668, doi: 10.1109/ICWS60048.2023.00084.

[88] E. Samikwa, A. D. Maio, and T. Braun, "ARES: Adaptive resource-aware split learning for Internet of Things," *Comput. Netw.*, vol. 218, Dec. 2022, Art. no. 109380, doi: 10.1016/j.comnet.2022.109380.

[89] L. Li, W. Chu, J. Langford, and R. E. Schapire, "A contextual-bandit approach to personalized news article recommendation," in *Proc. 19th Int. Conf. World Wide Web*, Apr. 2010, pp. 661–670, doi: 10.1145/1772690.1772758.

[90] J. Wang, H. Qi, A. S. Rawat, S. Reddi, S. Waghmare, F. X. Yu, and G. Joshi, "FedLite: A scalable approach for federated learning on resource-constrained clients," 2022, *arXiv:2201.11865*. Accessed: Dec. 15, 2023.

[91] S. Zhang, W. Wu, P. Hu, S. Li, and N. Zhang, "Split federated learning: Speed up model training in resource-limited wireless networks," in *Proc. IEEE 43rd Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2023, pp. 985–986, doi: 10.1109/icdcs57875.2023.00096.

[92] B. Yin, Z. Chen, and M. Tao, "Predictive GAN-powered multi-objective optimization for hybrid federated split learning," *IEEE Trans. Commun.*, vol. 71, no. 8, pp. 4544–4560, Aug. 2023, doi: 10.1109/TCOMM.2023.3277878.

[93] D.-J. Han, H. I. Bhatti, J. Lee, and J. Moon, "Accelerating federated learning with split learning on locally generated losses," in *Proc. ICML*, 2021, pp. 1–7.

[94] E. Belilovsky, M. Eickenberg, and E. Oyallon, "Decoupled greedy learning of CNNs," 2019, *arXiv:1901.08164*. Accessed: Dec. 15, 2023.

[95] Y. Chen, K. Yuan, Y. Zhang, P. Pan, Y. Xu, and W. Yin, "Accelerating gossip SGD with periodic global averaging," 2021, *arXiv:2105.09080*. Accessed: Dec. 15, 2023.

[96] A. Koloskova, S. U. Stich, and M. Jaggi, "Decentralized stochastic optimization and gossip algorithms with compressed communication," 2019, *arXiv:1902.00340*. Accessed: Dec. 15, 2023.

[97] A. Koloskova, N. Loizou, S. Boreiri, M. Jaggi, and S. U. Stich, "A unified theory of decentralized SGD with changing topology and local updates," 2020, *arXiv:2003.10422*. Accessed: Dec. 15, 2023.

[98] A. Guha Roy, S. Siddiqui, S. Pölsterl, N. Navab, and C. Wachinger, "BrainTorrent: A peer-to-peer environment for decentralized federated learning," 2019, *arXiv:1905.06731*. Accessed: Dec. 15, 2023.

[99] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10700–10714, Dec. 2019, doi: 10.1109/JIOT.2019.2940820.

[100] Y. Xu, Z. Lu, K. Gai, Q. Duan, J. Lin, J. Wu, and K. R. Choo, "BESIFL: Blockchain-empowered secure and incentive federated learning paradigm in IoT," *IEEE Internet Things J.*, vol. 10, no. 8, pp. 6561–6573, Apr. 2023, doi: 10.1109/JIOT.2021.3138693.

[101] M. Fan, K. Ji, Z. Zhang, H. Yu, and G. Sun, "Lightweight privacy and security computing for blockchained federated learning in IoT," *IEEE Internet Things J.*, vol. 10, no. 18, pp. 16048–16060, Sep. 2023, doi: 10.1109/JIOT.2023.3267112.

[102] T.-T. Kuo and L. Ohno-Machado, "ModelChain: Decentralized privacy-preserving healthcare predictive modeling framework on private blockchain networks," 2018, *arXiv:1802.01746*.

[103] X. Zhu, F. Zhang, and H. Li, "Swarm deep reinforcement learning for robotic manipulation," *Procedia Comput. Sci.*, vol. 198, pp. 472–479, Jan. 2022, doi: 10.1016/j.procs.2021.12.272.

[104] R. Ormándi, I. Hegedűs, and M. Jelasity, "Gossip learning with linear models on fully distributed data," *Concurrency Comput., Pract. Exper.*, vol. 25, no. 4, pp. 556–571, Feb. 2013, doi: 10.1002/cpe.2858.

[105] I. Hegedűs, G. Danner, and M. Jelasity, "Gossip learning as a decentralized alternative to federated learning," in *Distributed Applications and Interoperable Systems* (Lecture Notes in Computer Science), J. Pereira and L. Ricci, Eds. Cham, Switzerland: Springer, 2019, pp. 74–90, doi: 10.1007/978-3-030-22496-7_5.

[106] A. Lalitha, T. Javidi, S. Shekhar, and F. Koushanfar, "Fully decentralized federated learning," in *Proc. NeurIPS*, 2018, pp. 1–9.

[107] C. Hu, J. Jiang, and Z. Wang, "Decentralized federated learning: A segmented gossip approach," 2019, *arXiv:1908.07782*. Accessed: Dec. 15, 2023.

[108] S. Warnat-Herresthal et al., "Swarm learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, no. 7862, pp. 265–270, Jun. 2021, doi: 10.1038/s41586-021-03583-3.

[109] O. L. Saldanha et al., "Swarm learning for decentralized artificial intelligence in cancer histopathology," *Nature Med.*, vol. 28, no. 6, pp. 1232–1239, Jun. 2022, doi: 10.1038/s41591-022-01768-5.

[110] H. Gauttam, K. K. Pattanaik, S. Bhadauria, G. Nain, and P. B. Prakash, "An efficient DNN splitting scheme for edge-AI enabled smart manufacturing," *J. Ind. Inf. Integr.*, vol. 34, Aug. 2023, Art. no. 100481, doi: 10.1016/j.jii.2023.100481.

[111] A. Bemani and N. Björsell, "Aggregation strategy on federated machine learning algorithm for collaborative predictive maintenance," *Sensors*, vol. 22, no. 16, p. 6252, Aug. 2022, doi: 10.3390/s22166252.

[112] S. Bebortta, S. S. Tripathy, S. Basheer, and C. L. Chowdhary, "FedEHR: A federated learning approach towards the prediction of heart diseases in IoT-based electronic health records," *Diagnostics*, vol. 13, no. 20, p. 3166, Oct. 2023, doi: 10.3390/diagnostics13203166.

[113] İ. İlhan and G. Tezel, "A genetic algorithm–support vector machine method with parameter optimization for selecting the tag SNPs," *J. Biomed. Informat.*, vol. 46, no. 2, pp. 328–340, Apr. 2013, doi: 10.1016/j.jbi.2012.12.002.

[114] R.-H. Hsu, Y.-C. Wang, C.-I. Fan, B. Sun, T. Ban, T. Takahashi, T.-W. Wu, and S.-W. Kao, "A privacy-preserving federated learning system for Android malware detection based on edge computing," in *Proc. 15th Asia Joint Conf. Inf. Secur. (AsiaJCIS)*, Aug. 2020, pp. 128–136, doi: 10.1109/AsiaJCIS50894.2020.00031.

[115] Y. Liu, Z. Ma, X. Liu, S. Ma, S. Nepal, and R. Deng, "Boosting privately: Privacy-preserving federated extreme boosting for mobile crowdsensing," 2019, *arXiv:1907.10218*. Accessed: Dec. 16, 2023.

[116] Q. Li, Z. Wen, and B. He, "Practical federated gradient boosting decision trees," 2019, *arXiv:1911.04206*. Accessed: Dec. 16, 2023.

[117] L. Zhao, L. Ni, S. Hu, Y. Chen, P. Zhou, F. Xiao, and L. Wu, "InPrivate digging: Enabling tree-based distributed data mining with differential privacy," in *Proc. IEEE Conf. Comput. Commun.*, Apr. 2018, pp. 2087–2095, doi: 10.1109/INFOCOM.2018.8486352.

[118] K. Cheng, T. Fan, Y. Jin, Y. Liu, T. Chen, D. Papadopoulos, and Q. Yang, "SecureBoost: A lossless federated learning framework," 2019, *arXiv:1901.08755*. Accessed: Dec. 16, 2023.

[119] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Aug. 2016, pp. 785–794, doi: 10.1145/2939672.2939785.

[120] F. Yamamoto, S. Ozawa, and L. Wang, "eFL-boost: Efficient federated learning for gradient boosting decision trees," *IEEE Access*, vol. 10, pp. 43954–43963, 2022, doi: 10.1109/ACCESS.2022.3169502.

[121] F. Yamamoto, L. Wang, and S. Ozawa, "New approaches to federated XGBoost learning for privacy-preserving data analysis," in *Neural Information Processing* (Lecture Notes in Computer Science), H. Yang, K. Pasupa, A. C.-S. Leung, J. T. Kwok, J. H. Chan, and I. King, Eds. Cham, Switzerland: Springer, 2020, pp. 558–569, doi: 10.1007/978-3-030-63833-7_47.

[122] D. Kurian Dennis, T. Li, and V. Smith, "Heterogeneity for the win: One-shot federated clustering," 2021, *arXiv:2103.00697*. Accessed: Dec. 16, 2023.

[123] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečný, H. B. McMahan, V. Smith, and A. Talwalkar, "LEAF: A benchmark for federated settings," 2018, *arXiv:1812.01097*. Accessed: Dec. 16, 2023.

[124] E. S. Lubana, C. I. Tang, F. Kawsar, R. P. Dick, and A. Mathur, "Orchestra: Unsupervised federated learning via globally consistent clustering," 2022, *arXiv:2205.11506*. Accessed: Dec. 15, 2023.

[125] F. Sattler, K.-R. Müller, and W. Samek, "Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3710–3722, Aug. 2021, doi: 10.1109/TNNLS.2020.3015958.

[126] A. Abedi and S. S. Khan, "FedSL: Federated split learning on distributed sequential data in recurrent neural networks," *Multimedia Tools Appl.*, vol. 83, no. 10, pp. 28891–28911, Sep. 2023, doi: 10.1007/s11042-023-15184-5.

[127] L. Jiang, Y. Wang, W. Zheng, C. Jin, Z. Li, and S. G. Teo, "LSTMSPLIT: Effective SPLIT learning based LSTM on sequential time-series data," 2022, *arXiv:2203.04305*. Accessed: Dec. 17, 2023.

[128] A. Ayad, M. Barhoush, M. Frei, B. Völker, and A. Schmeink, "An efficient and private ECG classification system using split and semi-supervised learning," *IEEE J. Biomed. Health Informat.*, vol. 27, no. 9, pp. 4261–4272, Sep. 2023, doi: 10.1109/JBHI.2023.3281977.

[129] C. He, M. Annavaram, and S. Avestimehr, "Group knowledge transfer: Federated learning of large CNNs at the edge," 2020, *arXiv:2007.14513*. Accessed: Dec. 17, 2023.

[130] X. Cao, G. Sun, H. Yu, and M. Guizani, "PerFED-GAN: Personalized federated learning via generative adversarial networks," *IEEE Internet Things J.*, vol. 10, no. 5, pp. 3749–3762, Mar. 2023, doi: 10.1109/JIOT.2022.3172114.

[131] Y. Wu, Y. Kang, J. Luo, Y. He, L. Fan, R. Pan, and Q. Yang, "FedCG: Leverage conditional GAN for protecting privacy and maintaining competitive performance in federated learning," in *Proc. 31st Int. Joint Conf. Artif. Intell.*, Jul. 2022, pp. 2334–2340, doi: 10.24963/ijcai.2022/324.

[132] C. He, K. Balasubramanian, E. Ceyani, C. Yang, H. Xie, L. Sun, L. He, L. Yang, P. S. Yu, Y. Rong, P. Zhao, J. Huang, M. Annavaram, and S. Avestimehr, "FedGraphNN: A federated learning system and benchmark for graph neural networks," 2021, *arXiv:2104.07145*. Accessed: Dec. 17, 2023.

[133] H. Xie, J. Ma, L. Xiong, and C. Yang, "Federated graph classification over non-IID graphs," 2021, *arXiv:2106.13423*. Accessed: Dec. 17, 2023.

[134] K. Zhang, C. Yang, X. Li, L. Sun, and S. M. Yiu, "Subgraph federated learning with missing neighbor generation," 2021, *arXiv:2106.13430*. Accessed: Dec. 17, 2023.

[135] J. Baek, W. Jeong, J. Jin, J. Yoon, and S. J. Hwang, "Personalized subgraph federated learning," 2022, *arXiv:2206.10206*. Accessed: Dec. 17, 2023.

[136] C. Wu, F. Wu, L. Lyu, T. Qi, Y. Huang, and X. Xie, "A federated graph neural network framework for privacy-preserving personalization," *Nature Commun.*, vol. 13, no. 1, Jun. 2022, Art. no. 1, doi: 10.1038/s41467-022-30714-9.

[137] C. Meng, S. Rambhatla, and Y. Liu, "Cross-node federated graph neural network for spatio-temporal data modeling," 2021, *arXiv:2106.05223*. Accessed: Dec. 17, 2023.

[138] Y. Pei, R. Mao, Y. Liu, C. Chen, S. Xu, and F. Qiang, "Decentralized federated graph neural networks," in *Proc. IJCAI*, 2021, pp. 1–7.

[139] X. Meng, Y. Li, J. Lu, and X. Ren, "An optimization method for non-IID federated learning based on deep reinforcement learning," *Sensors*, vol. 23, no. 22, p. 9226, Nov. 2023, doi: 10.3390/s23229226.

[140] Z. Tianqing, W. Zhou, D. Ye, Z. Cheng, and J. Li, "Resource allocation in IoT edge computing via concurrent federated reinforcement learning," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1414–1426, Jan. 2022, doi: 10.1109/JIOT.2021.3086910.

[141] H. T. Nguyen, N. C. Luong, J. Zhao, C. Yuen, and D. Niyato, "Resource allocation in mobility-aware federated learning networks: A deep reinforcement learning approach," in *Proc. IEEE 6th World Forum Internet Things (WF-IoT)*, Jun. 2020, pp. 1–6, doi: 10.1109/WF-IoT48130.2020.9221089.

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

[143] Q. Miao, H. Lin, X. Wang, and M. M. Hassan, "Federated deep reinforcement learning based secure data sharing for Internet of Things," *Comput. Netw.*, vol. 197, Oct. 2021, Art. no. 108327, doi: 10.1016/j.comnet.2021.108327.

[144] D. Wu, R. Ullah, P. Harvey, P. Kilpatrick, I. Spence, and B. Varghese, "FedAdapt: Adaptive offloading for IoT devices in federated learning," *IEEE Internet Things J.*, vol. 9, no. 21, pp. 20889–20901, Nov. 2022, doi: 10.1109/JIOT.2022.3176469.

[145] W. Bai and C. Qian, "Deep reinforcement learning for joint offloading and resource allocation in fog computing," in *Proc. IEEE 12th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, Aug. 2021, pp. 131–134, doi: 10.1109/ICSESS52187.2021.9522334.

[146] W. Zhang, D. Yang, W. Wu, H. Peng, N. Zhang, H. Zhang, and X. Shen, "Optimizing federated learning in distributed industrial IoT: A multi-agent approach," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 12, pp. 3688–3703, Dec. 2021, doi: 10.1109/JSAC.2021.3118352.

[147] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Trans. Ind. Informat.*, vol. 16, no. 10, pp. 6532–6542, Oct. 2020, doi: 10.1109/TII.2019.2945367.

[148] N. Majcherczyk, N. Srishankar, and C. Pinciroli, "Flow-FL: Data-driven federated learning for spatio-temporal predictions in multi-robot systems," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 8836–8842, doi: 10.1109/ICRA48506.2021.9560791.

[149] T. M. Ho, K.-K. Nguyen, and M. Cheriet, "Federated deep reinforcement learning for task scheduling in heterogeneous autonomous robotic system," in *Proc. IEEE Global Commun. Conf.*, Dec. 2022, pp. 1134–1139, doi: 10.1109/GLOBECOM48099.2022.10000980.

[150] T. Manoj, K. Makkithaya, and V. G. Narendra, "A federated learning-based crop yield prediction for agricultural production risk management," in *Proc. IEEE Delhi Sect. Conf. (DELCON)*, Feb. 2022, pp. 1–7, doi: 10.1109/DELCON54057.2022.9752836.

[151] G. Idoje, T. Dagiuklas, and M. Iqbal, "Federated learning: Crop classification in a smart farm decentralised network," *Smart Agricult. Technol.*, vol. 5, Oct. 2023, Art. no. 100277, doi: 10.1016/j.atech.2023.100277.

[152] F. Deng, W. Mao, Z. Zeng, H. Zeng, and B. Wei, "Multiple diseases and pests detection based on federated learning and improved faster R-CNN," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–11, 2022, doi: 10.1109/TIM.2022.3201937.

[153] C. Xu, Y. Qu, T. H. Luan, P. W. Eklund, Y. Xiang, and L. Gao, "An efficient and reliable asynchronous federated learning scheme for smart public transportation," *IEEE Trans. Veh. Technol.*, vol. 72, no. 5, pp. 6584–6598, May 2023, doi: 10.1109/TVT.2022.3232603.

[154] K. Xie, Z. Zhang, B. Li, J. Kang, D. Niyato, S. Xie, and Y. Wu, "Efficient federated learning with spike neural networks for traffic sign recognition," *IEEE Trans. Veh. Technol.*, vol. 71, no. 9, pp. 9980–9992, Sep. 2022, doi: 10.1109/TVT.2022.3178808.

[155] Y. Liu, S. Zhang, C. Zhang, and J. J. Q. Yu, "FedGRU: Privacy-preserving traffic flow prediction via federated learning," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2020, pp. 1–6, doi: 10.1109/ITSC45102.2020.9294453.

[156] Z. Su, Y. Wang, T. H. Luan, N. Zhang, F. Li, T. Chen, and H. Cao, "Secure and efficient federated learning for smart grid with edge-cloud collaboration," *IEEE Trans. Ind. Informat.*, vol. 18, no. 2, pp. 1333–1344, Feb. 2022, doi: 10.1109/TII.2021.3095506.

[157] M. Ashraf, M. Waqas, G. Abbas, T. Baker, Z. Abbas, and H. Alasmary, "FedDP: A privacy-protecting theft detection scheme in smart grids using federated learning," *Energies*, vol. 15, no. 17, p. 6241, Aug. 2022, doi: 10.3390/en15176241.

[158] I. Donevski, N. Babu, J. J. Nielsen, P. Popovski, and W. Saad, "Federated learning with a drone orchestrator: Path planning for minimized staleness," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 1000–1014, 2021, doi: 10.1109/OJCOMS.2021.3072003.

[159] G. Gad, A. Farrag, Z. M. Fadlullah, and M. M. Fouda, "Communication-efficient federated learning in drone-assisted IoT networks: Path planning and enhanced knowledge distillation techniques," in *Proc. IEEE 34th Annu. Int. Symp. Pers., Indoor Mobile Radio Commun. (PIMRC)*, Sep. 2023, pp. 1–7, doi: 10.1109/pimrc56721.2023.10294036.

[160] T. Yu, T. Li, Y. Sun, S. Nanda, V. Smith, V. Sekar, and S. Seshan, "Learning context-aware policies from multiple smart homes via federated multi-task learning," in *Proc. IEEE/ACM 5th Int. Conf. Internet-of-Things Design Implement. (IoTDI)*, Apr. 2020, pp. 104–115, doi: 10.1109/IoTDI49375.2020.00017.

[161] S. Lee and D.-H. Choi, "Federated reinforcement learning for energy management of multiple smart homes with distributed energy resources," *IEEE Trans. Ind. Informat.*, vol. 18, no. 1, pp. 488–497, Jan. 2022, doi: 10.1109/TII.2020.3035451.

[162] A. Rasti-Meymandi, S. M. Sheikholeslami, J. Abouei, and K. N. Plataniotis, "Graph federated learning for CIoT devices in smart home applications," *IEEE Internet Things J.*, vol. 10, no. 8, pp. 7062–7079, Apr. 2023, doi: 10.1109/JIOT.2022.3228727.

[163] Z. Lian, Q. Yang, W. Wang, Q. Zeng, M. Alazab, H. Zhao, and C. Su, "DEEP-FEL: Decentralized, efficient and privacy-enhanced federated edge learning for healthcare cyber physical systems," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 5, pp. 3558–3569, Sep. 2022, doi: 10.1109/TNSE.2022.3175945.

[164] B. Camajori Tedeschini, S. Savazzi, R. Stoklasa, L. Barbieri, I. Stathopoulos, M. Nicoli, and L. Serio, "Decentralized federated learning for healthcare networks: A case study on tumor segmentation," *IEEE Access*, vol. 10, pp. 8693–8708, 2022, doi: 10.1109/ACCESS.2022.3141913.

[165] Y. J. Ha, G. Lee, M. Yoo, S. Jung, S. Yoo, and J. Kim, "Feasibility study of multi-site split learning for privacy-preserving medical systems under data imbalance constraints in COVID-19, X-ray, and cholesterol dataset," *Sci. Rep.*, vol. 12, no. 1, Jan. 2022, Art. no. 1, doi: 10.1038/s41598-022-05615-y.

[166] L. Benos, A. C. Tagarakis, G. Dolias, R. Berruto, D. Kateris, and D. Bochtis, "Machine learning in agriculture: A comprehensive updated review," *Sensors*, vol. 21, no. 11, p. 3758, May 2021, doi: 10.3390/s21113758.

[167] Z. Wu, Q. Ling, T. Chen, and G. B. Giannakis, "Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks," *IEEE Trans. Signal Process.*, vol. 68, pp. 4583–4596, 2020, doi: 10.1109/TSP.2020.3012952.

[168] L. Li, H. Xiong, Z. Guo, J. Wang, and C.-Z. Xu, "SmartPC: Hierarchical pace control in real-time federated learning system," in *Proc. IEEE Real-Time Syst. Symp. (RTSS)*, Dec. 2019, pp. 406–418, doi: 10.1109/RTSS46320.2019.00043.

[169] G. He, C. Li, M. Song, Y. Shu, C. Lu, and Y. Luo, "A hierarchical federated learning incentive mechanism in UAV-assisted edge computing environment," *Ad Hoc Netw.*, vol. 149, Oct. 2023, Art. no. 103249, doi: 10.1016/j.adhoc.2023.103249.

[170] T. Feng, D. Bose, T. Zhang, R. Hebbar, A. Ramakrishna, R. Gupta, M. Zhang, S. Avestimehr, and S. Narayanan, "FedMultimodal: A benchmark for multimodal federated learning," in *Proc. 29th ACM SIGKDD Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Aug. 2023, pp. 4035–4045, doi: 10.1145/3580305.3599825.

[171] F. Liu, X. Wu, S. Ge, W. Fan, and Y. Zou, "Federated learning for vision-and-language grounding problems," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2020, vol. 34, no. 7, pp. 11572–11579, doi: 10.1609/aaai.v34i07.6824.

[172] Y. Sun, Z. Chen, M. Tao, and H. Liu, "Bandwidth gain from mobile edge computing and caching in wireless multicast systems," *IEEE Trans. Wireless Commun.*, vol. 19, no. 6, pp. 3992–4007, Jun. 2020, doi: 10.1109/TWC.2020.2979147.

[173] J. Du, C. Jiang, J. Wang, Y. Ren, and M. Debbah, "Machine learning for 6G wireless networks: Carrying forward enhanced bandwidth, massive access, and ultrareliable/low-latency service," *IEEE Veh. Technol. Mag.*, vol. 15, no. 4, pp. 122–134, Dec. 2020, doi: 10.1109/MVT.2020.3019650.

HANYUE XU received the B.Eng. degree in data science and big data technology from Xi'an Jiaotong–Liverpool University, China. She is currently pursuing the Ph.D. degree with Xi'an Jiaotong–Liverpool University and the University of Liverpool. Her research interests include artificial intelligence, the Internet of Things, data analytics, machine learning, edge computing, and multimodal information processing.

KAH PHOOI SENG (Senior Member, IEEE) received the B.Eng. and Ph.D. degrees from the University of Tasmania, Australia. She was a Professor and the Department Head of computer science and networked system with Sunway University. Before joining Sunway University, she was an Associate Professor with the School of Electrical and Electronic Engineering, Nottingham University. She has worked or attached to Australian-based and U.K.-based universities, including Monash University, Griffith University, the University of Tasmania, the University of Nottingham, Sunway University, Edith Cowan University, and Charles Sturt University. She is currently a Professor of AI and the IoT with Xi'an Jiaotong–Liverpool University, and an Adjunct Professor with UniSC, Australia. She has participated in more than AUD $1.8 million research grant projects from the government and industry in Australia and overseas. She has supervised or co-supervised 16 Ph.D. students to completion and more than 25 higher-degree research students. She has a strong record of publications and has published more than 250 papers in journals and internationally refereed conferences. Her research interests include computer science and engineering, including AI, data science and machine learning, big data, multimodal information processing, intelligent systems, the IoT, embedded systems, mobile software development, affective computing, computer vision, and the development of innovative technologies for real-world applications. She is an Associate Editor of IEEE ACCESS. She also serves on the editorial board or committees of several journals and international conferences.

LI MINN ANG (Senior Member, IEEE) received the B.Eng. (Hons.) and Ph.D. degrees from Edith Cowan University (ECU), Australia. He was an Associate Professor of networked and computer systems with the School of Information and Communication Technology (ICT), Griffith University. He was with the Australian and U.K. universities, including Monash University, the University of Nottingham, ECU, Charles Sturt University, and Griffith University. He is currently a Professor of electrical and computer engineering with the School of Science, Technology, and Engineering, University of the Sunshine Coast (UniSC), Australia. His research interests include computer, electrical, and systems engineering, including the Internet of Things; intelligent systems and data analytics; machine learning; visual information processing; embedded systems; wireless multimedia sensor systems; reconfigurable computing (FPGA); and the development of innovative technologies for real-world systems, including smart cities, engineering, agriculture, environment, health, and defense. He is a Senior Fellow of the Higher Education Academy, U.K.

JEREMY SMITH received the degree in engineering science from the University of Liverpool, in 1984. Then, he undertook the Ph.D. research with the Automated Welding Group under the leadership of Prof. Lucas. He is a Professor with the Department of Electrical and Electronics, UoL. His research interests include vision-based sensors and control systems, advanced digital and parallel processing systems, embedded systems, neural network and fuzzy logic, and robotic and navigation.