ChainFL: A Simulation Platform for Joint Federated Learning and Blockchain in Edge/Cloud Computing Environments

Guanjin Qu, Naichuan Cui, Huaming Wu, *Member, IEEE*, Ruidong Li, *Senior Member, IEEE*, and Yuemin Ding, *Member, IEEE*

***Abstract*—As a distributed computing paradigm, edge computing has become a key technology for providing timely services to mobile devices by connecting Internet of Things (IoT), cloud centers and other facilities. By offloading compute-intensive tasks from IoT devices to edge/cloud servers, the communication and computation pressure caused by the massive data in industrial IoT can be effectively reduced. In the process of computation offloading in edge computing, it is critical to dynamically make optimal offloading decisions to minimize the delay and energy consumption spent on the devices. Although there are a large number of task offloading-decision models, how to measure and evaluate the quality of different models and configurations is crucial. In this paper, we propose a novel simulation platform named ChainFL, which can build an edge computing environment among IoT devices while being compatible with Federated Learning and Blockchain technologies to better support the embedding of security-focused offloading algorithms. ChainFL is lightweight and compatible, and it can quickly build complex network environments by connecting devices of different architectures. Moreover, due to its distributed nature, ChainFL can also be deployed as a federated learning platform across multiple devices to enable federated learning with high security due to its embedded blockchain. Finally, we validate the versatility and effectiveness of ChainFL by embedding a complex offloading decision model in the platform, as well as deploying it in an industrial IoT environment with security risks.**

***Index Terms*—Edge Computing, Computation Offloading, Federated Learning, Blockchain, Simulation Platform.**

I. Introduction

OVER the last few years, the rapid proliferation of various Internet of Things (IoT) devices has brought great convenience to people's daily lives. However, due to the limited computing resources of IoT devices, it is still difficult to directly deploy compute-intensive applications on them, such as face recognition and augmented reality [1]. In order to prevent resource-constrained devices from high computation latency and high energy consumption caused by running large amounts of computing, we generally rely on cloud servers closely to assist in computing and storing by offloading local tasks to cloud servers for remote processing, thereby reducing application response time and extending battery life. However, it still suffers from limited communication resources as well as high latency since the cloud center is usually far away from the users. In addition, in the field of Industrial IoTs (IIoT), extremely vast amounts of data from a huge number of sensors still pose communication and computational difficulties for traditional cloud centers. What's more, in the cases of limited heterogeneous network resources, it is of great challenge to guarantee the Quality of Service (QoS), protect IIoT applications from a variety of threats and attacks [2], as well as ensure the fairness of various IIoT devices [3].

As a supplement to traditional cloud computing, edge computing is a distributed computing paradigm that uses computing resources located at the proximity of IoT devices to provide efficient services in a timely manner [4]. Edge servers are generally connected to the IoT and the cloud center via Local Area Networks (LAN), and complex computing tasks of IoT devices may be offloaded to the edge or the cloud for computing, which can break through the resource limitations of mobile devices, reduce computing load, improve task processing efficiency and save energy consumption. Blockchain technology has emerged in edge computing environments due to its seamless network control, distributed services and security, which is reliable in providing edge services according to user requirements, and thus can improve the distributed resource scheduling and task offloading at ease [5].

Currently, a large number of methods and models for task offloading decision-making are mainly divided into traditional offloading techniques and intelligent offloading techniques. The former usually apply some heuristic algorithms, where a large amount of computation is required to make offloading decisions [6], while the latter are based on deep learning models, where the internal laws and representation levels of standard sample data are learned through deep neural networks so that computers can have analytical capabilities like humans. For instance, deep reinforcement learning-based methods can promote offloading decision-making, dynamic resource allocation, and content caching, which are conducive to coping with the explosive growth of communication and computing in emerging IoT applications. Intelligent offloading algorithms have gradually emerged in recent years and become ever-increasing popular [7]–[9]. By introducing neural networks and other methods, offloading decision-making can achieve

G. Qu and H. Wu are with the Center for Applied Mathematics, Tianjin University, Tianjin 300072, China (e-mail: {guanjinqu, whming}@tju.edu.cn).

N. Cui is with School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: CUIN0001@e.ntu.edu.sg).

R. Li is with the Institute of Science and Engineering, Kanazawa University, Kanazawa 920-1192, Japan (e-mail: liruidong@ieee.org).

Y. Ding is with the Department of Electrical and Electronic Engineering, University of Navarra, San Sebastian, Spain (email: yuemin.ding1986@gmail.com).

(Corresponding author: Huaming Wu)

good results, but there still exist several challenges, e.g., slow learning speed and long training time. Besides, most of the current intelligent offloading algorithms are based on neural networks, which are generally deployed on edge servers since strong computing capacities are required.

In order to solve the aforementioned challenging issues, more and more approaches have been introduced into the intelligent offloading model. Among them, distributed learning models can effectively help multiple institutions to perform data usage and machine learning modeling under the requirements of user privacy protection, data security, and government regulations. As a distributed machine learning paradigm, federated learning can effectively solve the problem of data islands, allowing participants to jointly model on the basis of not sharing data, thereby preventing privacy leakage as well as protecting users' privacy [10], while improving the training speed of the model [11]. In addition, as a new data structure, the data or information stored in blockchain has the characteristics of unforgeability, traceability, openness and transparency, and collective maintenance [12]. With these characteristics, blockchain technology has laid a solid foundation of trust and established a reliable cooperation mechanism. By introducing blockchain into edge computing, the privacy and security requirements for task offloading can be well improved.

When designing optimal offloading-decision algorithms, it is necessary to comprehensively consider the overall latency and energy consumption of all IoT devices, especially for delay-sensitive applications in large-scale industries [13], [14]. Therefore, it is urgent to have a simulation platform that measures the offloading effectiveness of different task offloading models. In addition, blockchain owns disadvantages such as huge data volume and slow embedding speed. How to rationally combine blockchain and federated learning to ensure that the efficiency of federated learning is less affected, while still meeting the security requirements, has become a research hotspot in related fields [15]. Although there are various simulation platforms for edge computing, they are often incompatible with some new models due to their early construction and the impact of technologies, e.g., distributed learning and blockchain. To address these challenges, we have built a new simulation platform called ChainFL, which owns the following advantages:

- ChainFL has strong compatibility and can be built between devices under different architectures. It can quickly build an environment that simulates multiple IoT devices, edges and clouds. In addition, ChainFL itself is very lightweight and can be easily extended to the existing task offloading decision model in the form of a toolkit. By introducing simulated data sets or collecting real-world task information, the process of task offloading can be simulated on the platform.
- The network structure of ChainFL is very suitable for blockchain embedding and supports distributed learning such as federated learning. In addition, it has better applicability to task offloading models involving blockchain, federated learning and other algorithms. As far as we know, this is the first simulation platform that is jointly optimized for blockchain and federated learning in edge/cloud computing environments.
- ChainFL can be deployed as a federated learning platform across multiple devices to enable federated learning in edge computing environments. Compared to traditional federated learning algorithms, ChainFL has an embedded blockchain structure, which can increase the security and reliability of federated learning in IIoTs.

The rest of this paper is organized as follows. Section II summarizes related work. Section III details the structure and application of the simulation platform. Section IV presents extensive experimental results to evaluate the performance of our proposed ChainFL. Section V discusses several potential application scenarios to be supported. Finally, we conclude the paper in Section VI.

II. Related Work

A large number of studies have been proposed to imitate the environment, including IoT, edge computing and cloud computing. Some of these systems have also been developed to allow embedded task offloading decision models. In this section, we review related work and compare it with our simulation platform. In addition, we compare the feature differences between existing federated learning platforms and the proposed ChainFL.

A. Edge/Cloud Simulation Platform

Cloudsim [16] is a popular cloud computing simulation platform, which is a function library developed on the discrete event simulation package SimJava. Cloudsim supports the research and development of cloud computing, and the modeling and simulation of a variety of cloud computing infrastructures. This has attracted widespread attention from both academia and industry, and has led to various follow-up projects based on CloudSIm, e.g., iFogSim [17], cloudsimSDN [18] and ContainerCloudSim [19].

IfogSim [17] is a toolkit proposed to simulate a fog computing environment, whose concept is similar to edge computing. Ifogsim uses not only a central server, but also an edge server closer to the user level. It can be used to create an integrated edge and cloud simulation environment to evaluate the resource management strategy of edge cloud. However, ifogsim focuses more on the management of computing resources such as CPU, memory and storage. On the contrary, in addition to these functions, our simulation platform can implement more complex network functions.

CLONE [20] is an edge-based collaborative learning framework, which is mainly used to deal with the contradiction between edge intelligence and privacy protection, and the limitation of insufficient bandwidth. In order to support the collaborative training and inference of models on edge devices, its application scenarios are classified into two categories, namely, CLONE training and CLONE inference. The core idea is that training/inference tasks are solved by a set of distributed edge nodes and coordinated by the edge server. The edge server is responsible for performing aggregation or other necessary operations on the uploaded parameters and sending the updated parameters back to the edge node. Each

edge node trains the neural network model locally based on its private training data, which will not be offloaded to the edge or the cloud, and pushes the corresponding parameters to the edge server during the training/inference process.

With the popularity of Kubernetes as a containerized application that manages multiple hosts on the cloud platform, it makes the deployment of containerized applications simple and efficient. However, Kubernetes is designed for cloud data centers after all. KubeEdge [21] is the world's first open edge computing platform based on Kubernetes and provides cloud-side collaboration capabilities. It relies on Kubernetes' container orchestration and scheduling capabilities to achieve cloud-edge collaboration, resource heterogeneity, and lightweight functions. K3S [22] is designed for R&D, operation and maintenance personnel who run Kubernetes in a resource-limited environment. The purpose is to run small Kubernetes clusters on edge nodes of x86, ARM64 and ARMv7D architectures. However, all components of K3S (including server and agent) run on the edge, so the cloud-edge collaboration is not involved.

Although these related systems provide examples of running their frameworks, building such a system requires a lot of equipment and time to build the environment and execute the corresponding model. Therefore, to perform large-scale and cost-effective simulation and evaluation for certain offloading decision or resource scheduling algorithms, it is still necessary to establish an efficient and lightweight toolbox. In order to meet this requirement, a fast simulation platform for evaluating task offloading decisions and strategies in edge-cloud environments is developed.

B. Federated Learning Platform

Federated learning platforms have developed rapidly in recent years. Table I summarizes the technical comparison of different federated learning platforms. Among them, TensorFlow-Federated (TFF)[1], the earliest federated learning platform proposed by Google, supports optimization algorithms, e.g., FedAvg and FedProx, but it cannot accommodate some of the newer algorithms as it can only perform centralized federated learning. LEAF [23] and PySyft [24] are similar to TFF in that there are more methods for training data than TFF. However, they only support algorithms with a central structure, and are not applicable for algorithms that require the exchange of complex auxiliary information and custom training procedures. In addition, federated learning platforms such as FATE [25] and PaddleFL [26] have been released by related industries, but the industry-dominated products often have cumbersome system designs, inflexible Application Programming Interfaces (APIs) and complex environment settings that are not friendly to edge computing and the embedding of decision algorithms for task offloading. FedML [27] is an open-source framework for federated learning, which facilitates various algorithm research through flexible and universal API design and reference benchmarks. However, since it does not support the embedding of blockchain algorithms, some of the latest blockchain structure-based algorithms will not be introduced for use, especially in the area of security.

The federated learning platforms mentioned above are rarely designed for platform security. This would lead to certain security risks in environments such as edge computing and IIoTs. In contrast to these platforms, ChainFL allows for customized encryption algorithms for information flow and supports embedding in blockchains to enhance the tamper-evident nature of information, thus improving the security of the global platform.

[1]https://medium.com/tensorflow/introducing-tensorflow-federated-a4147aa20041

III. Simulation Platform Structure and Functions

This section mainly provides the detailed structure and system model of ChainFL, as well as the related functions and information of the platform.

A. System Model

Fig. 1 illustrates the system model of the proposed simulation platform ChainFL. According to the characteristics of task offloading, the edge computing environment is composed of cloud, edge, user terminal and communication center. In order to be able to complete the most basic task offloading and communication transmission, it is also necessary to meet the requirements of compatible blockchain and federated learning. Therefore, the communication between various domains needs to be closer. For this reason, we set up a communication center node, which is not available in the conventional edge computing environment. The main role of the ChainFL platform is to connect individual devices and thus build an edge computing environment to provide communication support for federal learning. To be compatible with the embedding of blockchain algorithms, ChainFL allows the content of the communication to be passed on the blockchain. Specifically, local devices can upload parameters to the cloud server for parameter aggregation, and the updated parameters can be embedded in the blockchain and propagated to each device for parameter updates. As for task offloading models that do not involve blockchain and federated learning algorithms, the use of this simulation platform for experiments will not affect the effect evaluation of the original model. We will explain the structure and information of each level by domain as follows:

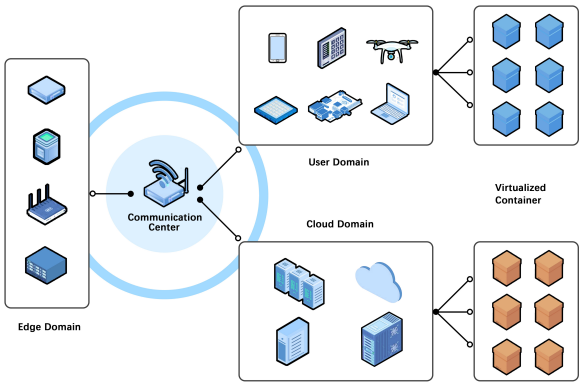


Fig. 1: System model of the simulation platform ChainFL

TABLE I: Comparison of features between different federated learning platforms

Aspects	Features	TFF	LEAF [23]	PaddleFL [26]	FATE [25]	PySyft [24]	FedML [27]	ChainFL (Ours)
Computing Paradigms	Standalone simulation	✓	✓	✓	✓	✓	✓	✓
	Distributed computing	✓	✗	✓	✓	✓	✓	✓
	Edge computing	✗	✓	✗	✗	✗	✓	✓
	IoT-device training	✗	✗	✗	✗	✗	✓	✓
Algorithm Customization	Information flow customization	✗	✗	✗	✗	✓	✓	✓
	Custom optimization algorithm	✗	✗	✓	✗	✓	✓	✓
	Allow distributed topology	✗	✗	✗	✗	✓	✓	✓
Security Measures	Customization of message encryption	✗	✗	✗	✗	✗	✓	✓
	Allow embedded blockchain	✗	✗	✗	✗	✗	✗	✓

- **Communication Center**: Since our simulation platform is compatible with blockchain and federated learning, we use a P2P network structure that is different from the conventional network structure. In order to ensure that each node can quickly connect to the entire network for offloading and training, we establish a communication center node. The communication center has a public IP address and is responsible for receiving IP address information from each node. When a new device joins the simulation environment, it first needs to access the communication center and obtain the IP addresses of other devices and then access the network. It should be noted that in order to protect information security and reduce the communication volume, the communication center is not responsible for the communication of specific information, so there is no risk of privacy leakage. In addition, when the node is connected to the network, it will not need to access the communication center frequently. The use of a communication center does not affect the efficiency of traditional task offloading and communication, nevertheless, it is only responsible for connecting the newly added devices to the system network.
- **Cloud Domain**: In addition to traditional cloud tasks such as task offloading and resource scheduling, the cloud nodes in this simulation platform also have other important functions, e.g., the propagation and embedding of blockchain, and the aggregation and transfer of parameters in the federated learning algorithm. By embedding related functions into the simulation platform, it provides great convenience for the implantation and effect evaluation of some complex offloading models.
- **Edge Domain**: As the most important part of the traditional edge computing environment, the edge end is responsible for task offloading, data transferring and communicating with the user terminal. In addition, most of the current offloading decision-making models put decision-making algorithms on the edge. For ChainFL, the edge end also takes the transfer function of blockchain and federated learning parameters into account.
- **User Terminal**: As the lowest layer in the edge computing environment, the user side is responsible for reasonably offloading tasks generated by itself or training the neural network in the federated learning algorithm.

B. Architecture of ChainFL

Fig. 2 shows the hierarchical layer structure of ChainFL for edge computing and task offloading. It consists of an infrastructure layer, a virtualization layer, a communication layer, a computing layer, and an application layer.

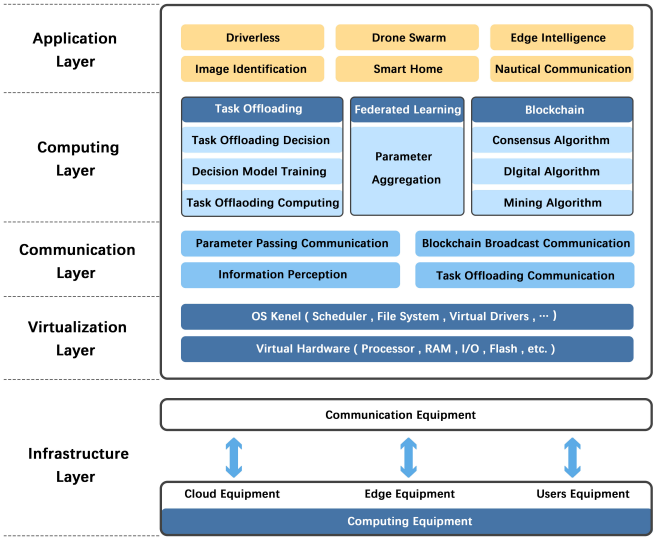


Fig. 2: ChainFL architecture for edge computing and task offloading

From bottom to top, the functions of each layer are defined as follows:

- **Infrastructure Layer**: This layer is mainly composed of computing equipment and communication equipment. Among them, computing devices include IoT devices that are responsible for simulating the user side, such as single-chip computers; devices that are responsible for simulating edge base stations, such as mid-performance computers; and devices that are responsible for simulating the cloud, such as large workstations. These computing devices are responsible for performing computing tasks such as task offloading, neural network learning, and parameter aggregation when performing edge computing. In addition, in order to ensure close communication between various computing devices, we need to connect the communication equipment of each computing device. The communication equipment is divided into a gateway and a communication center. For some small analog platforms, the gateway can be replaced by a router. Each

device will be connected to the gateway for communication, which also includes a communication center. The communication center can be formed by a single-chip microcomputer with low computing power because it does not perform computing tasks.

- **Virtualization Layer**: With the popularity of virtualization technology, more and more virtual services are being used in simulation platforms. ChainFL serves as a highly compatible platform that allows programs to run on virtual machines. The virtual machine is capable of customizing parameters such as bandwidth performance for better performance evaluation. ChainFL's strong support for virtualization allows users to simulate complex network environments with multiple devices on a single computer.
- **Communication Layer**: This layer is responsible for communication between computing devices, which includes not only the information communication for task offloading in traditional edge computing and the most basic necessary communication, but also the communication process of federated learning and blockchain. Since the communication layer is built on a gateway-based communication device, we can simulate the environment under different bandwidths by adjusting the bandwidth value in the virtualization layer. The specific information to be communicated is as follows:

  1) *Information related to task offloading*: It is mainly composed of the communication required for the task offloading decision process when the decision algorithm is at the edge, task information and related decision instructions required to transfer for each offloading decision, and the dynamic perception between devices, e.g., real-time bandwidth and CPU occupancy rate, as well as the task upload and result download in the most critical task offloading process, where the task may be in the form of text or pictures.
  2) *Communication transmission required for federated learning*: It mainly includes the upload of neural network parameters on the user side after a fixed number of rounds, and the decentralization of parameters after aggregation. Unlike conventional federated learning, because this network adopts a P2P structure, there is no need to specify an additional receiving node for the parameters uploaded by the client. The network will automatically aggregate them to the cloud node, and the client only needs to upload the parameters once per round. This reduces the amount of communication required for federated learning. In addition, ChainFL supports both elastic computing power and edge-cloud collaboration. For example, when the edge server on which the parameters reside is unable to access the cloud, it will be automatically passed to other edge servers to ensure the propagation of the parameters.
  3) *Communication transmission required by blockchain*: It includes the transmission of transactions and the spread of blockchains. Similar to federated learning, information related to the blockchain will be broadcast to all nodes in the entire network at the communication layer for consensus algorithm authentication, blockchain competitive leadership and embedded blocks.
- **Computing Layer**: This layer is responsible for the parts that need to be computed in edge computing, including calculations in task offloading, calculations required for task decision algorithm training, as well as parameter aggregation in federated learning and consensus algorithms and workload proofs in the blockchain. The computing layer usually consists of offloading decision-making algorithms, bandwidth allocation algorithms, task offloading and other parts.
- **Application Layer**: This layer is mainly responsible for IoT applications deployed in edge computing, including various applications of IoT devices, drone scheduling, and unmanned vehicles. It should be noted that ChainFL does not include specific algorithms in the computing layer and the application layer. It only provides a toolkit to embed the existing offloading models to simulate a complex network environment and evaluate the offloading effect of these models.

IV. Performance Evaluation and Case Study

In this section, we introduce a complex intelligent task offloading decision algorithm into ChainFL and simulate a multi-terminated edge computing environment in an IoT scenario. We first verify the utility of the simulation platform and the effectiveness of the federation algorithm by testing the utility of the offloading decision algorithm. After that, we simulate an environment where security attacks exist and verify that this platform can be effective against security attacks.

A. Experimental Setup

TABLE II: Hardware tools for experimental verification.

Tools	Description
Router	Responsible for building the network
Dell Workstation	Simulate cloud server
NVIDIA Jetson nano	Simulate edge server
Raspberry pi	Simulate IoT devices
Nanopi	Communication center

The ChainFL platform is written in python, the communication is based on the HTTP protocol and Flask is applied as the HTTP service framework. ChainFL contains communication procedures for each node and packages that reference other offload algorithms. The devices and tools used in the experiments are summarized in Table II. We set this environment to have a cloud server, an edge server, a communication center, and multiple clients. The whole structure of the experimental platform is shown in Fig. 3.

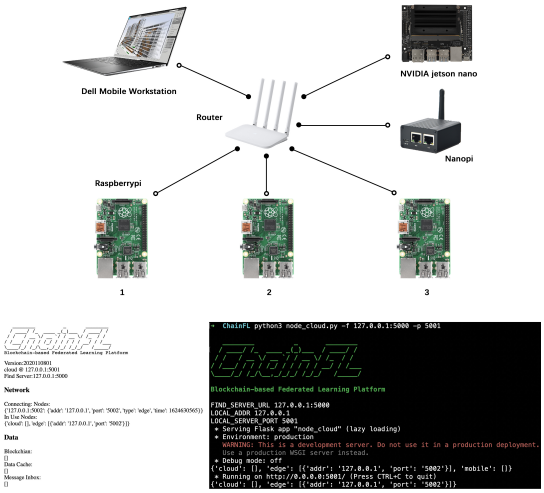


Fig. 3: The equipment structure with ChainFL

We first run ChainFL's communication program on each device, and the results after running are shown in the bottom right of the figure, where the terminal window shows the other nodes currently connected to that device in real time. In addition, as shown in the bottom left corner of the figure, we can access the backend management page by typing "local IP address: port number/observe" into the browser to view the current connectivity and blockchain information. Once the devices are running their respective communication programs, they have formed an IoT environment with edge computing, and then by referencing the package to the original task offload decision algorithm, they can use the full functionality of the platform, such as uploading parameters, performing federation aggregation, and embedding blockchain.

To validate the effectiveness of ChainFL on the task offloading decision model, we introduce the Deep Reinforcement Learning (DRL) algorithm as an offloading decision algorithm into the platform, which is capable of effectively solving semi-supervised problems such as task offloading decision by combining deep learning with reinforcement learning [28]. A number of studies have been conducted to introduce DRL algorithms into edge computing [29]–[31], but few methods combine DRL with federated learning for task offloading. In our experiments, we introduce Deep Q-network (DQN) Algorithm [32] into the platform [9]. We set the state space as task information and the action space as the decision to offload to a particular server. Our overall optimization objective is the total utility, which is a negatively correlated linear weighted sum of the overall energy consumption and the overall delay, where the former is the total energy incurred due to offloading the task, and the latter is the total delay generated by offloading the task, including computation latency and communication latency. A reasonable task offloading strategy will generate less energy and latency, and thus have higher total utility.

B. Experimental Analysis

1) Impact of Federated Learning: Fig. 4 shows the comparison of total utility with and without federated learning, here the horizontal coordinate is the number of training rounds and the vertical coordinate is the total utility, where a higher total utility means a better offloading effect. It can be seen that with the increase in the number of training rounds, both schemes are able to converge to a high level, demonstrating that the DRL algorithm can achieve good results in task offloading decision-making. It should be emphasized that the DRL algorithm with federated learning not only achieves a faster convergence rate, but also has better stability after convergence. At 500 rounds of training, the model with federated learning improved its effectiveness by 11.7% over the traditional DRL algorithm. In short, the model with federated learning achieves much better results than the original model.

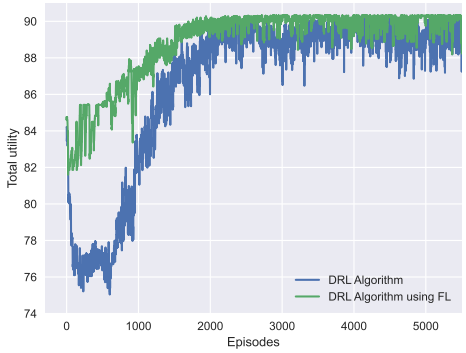


Fig. 4: Effect of federated learning for task offloading

2) Impact of Number of Layers: As shown in Fig. 5, we compare the effect of different numbers of neural network layers. With the increase in the number of layers, we can obtain a much better utility for task offloading. In addition, it can be seen that even if the number of neural network layers has changed, the model using federated learning is still superior to the one without federated learning.

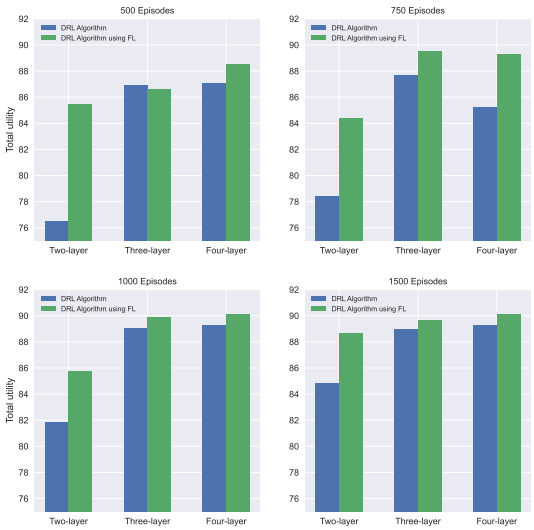


Fig. 5: Effect of different numbers of neural network layers for task offloading

3) Number of Rounds of Local training in Federated Learning: As shown in Fig. 6, we compare the different numbers of local training rounds in federation learning. It can be seen

that both too high and too few local training rounds will cause fluctuations in parameters, while 250 and 500 rounds possess a smoother trend. Therefore, the number of local training rounds for federal learning in this section is uniformly set to be within the 250-500 interval.

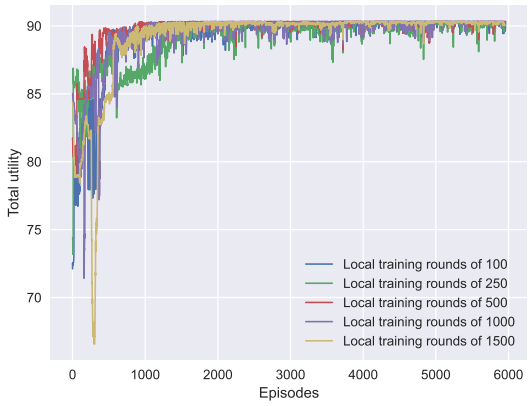


Fig. 6: Effect of different numbers of local training rounds in federation learning

4) Impact of Number of Devices: Fig. 7 illustrates the effect of the number of devices for task offloading. It can be clearly seen that as the number of devices involved in federated learning increases, the effect of federated learning is getting better. Especially in the first 1,000 rounds, the federated learning model with more devices can converge to the desired effect more quickly.

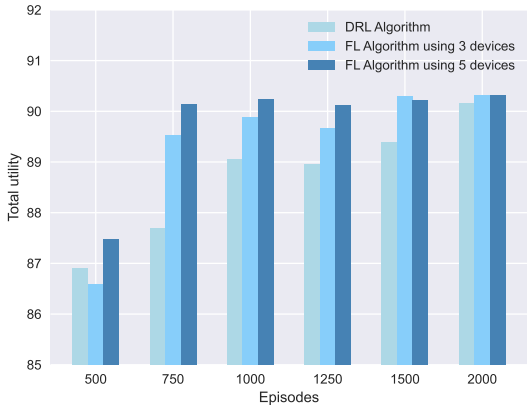


Fig. 7: Effect of different numbers of participating devices for task offloading

5) Impact of ChainFL on Security: Federated learning, as a newly emerged distributed learning framework, may have multiple security risks [33]. In this paper, we focus on two major security risks: one is the malicious modification of uploaded parameters due to a system vulnerability (upload attack), and the other is that the updated parameters are maliciously tampered with by poisoned devices during propagation (download attack). To cope with the former, our platform allows custom encryption of the uploaded parameters, while for the latter, ChainFL uses blockchain algorithms to prevent the modification of updated parameters. Here, we use ECDSA [34] as the encryption algorithm and use blockchain to record the updated parameters.

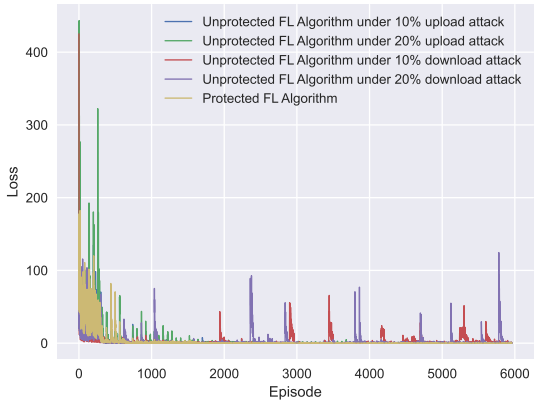


Fig. 8: Convergence performance under different security attacks

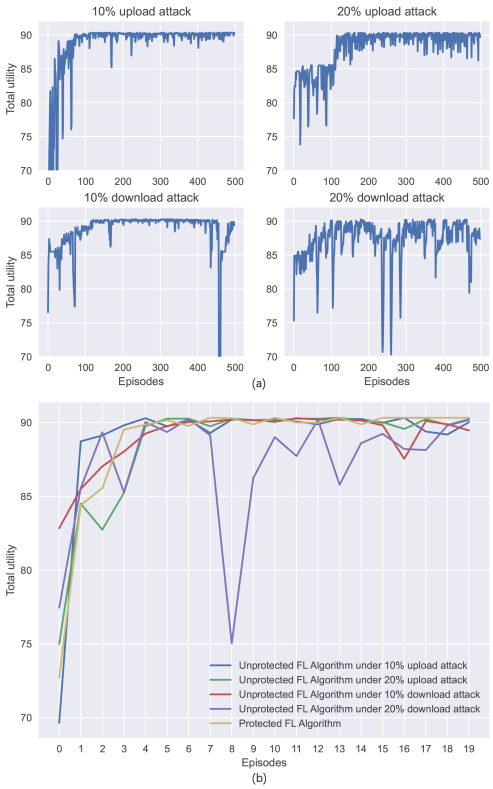


Fig. 9: Model performance under different security attacks

Fig. 8 shows the convergence of the model in the presence of security attacks, where the horizontal coordinate is the number of training steps and the vertical coordinate is the loss function of the neural network in the DRL algorithm. We can see that with the intensification of security attacks, the convergence of the model suffers more damage, while the download attack is even more damaging to the convergence. In contrast, the federated learning algorithm with protective measures is unaffected.

Fig. 9 shows the offloading effect of the model under the security attack, where the abscissa is the number of training

steps and the ordinate is the total effectiveness of offloading. In particular, Fig. 9(a) shows the federated learning effectiveness curves under normal conditions and the effect of different levels of security attacks on the model under unprotected conditions. It can be seen that at 10% of the upload attacks, the attacks have less impact on the unprotected model, but when the upload attack rate reaches 20%, the unprotected model starts to fluctuate. In addition, the upload attack is more destructive to the global model, making the global model appear less convergent. The download attack focuses more on the attack on the individual user side, and the user side under the download attack will occasionally fluctuate greatly, which leads to a large fluctuation in the effect of the model. Fig. 9(b) shows a line graph of the model effect under different attacks. It can be seen that the federated learning algorithm protected by the security measures of this platform can effectively avoid the effect fluctuations caused by security attacks, with faster convergence speed and better stability. It should be emphasized that ChainFL is mainly adopted as a platform to embed secure encryption algorithms and blockchain algorithms to enhance security. Specific security protection capabilities are influenced by the embedded security algorithms and blockchain settings.

V. Application Scenarios and Discussions

In this section, we provide some application scenarios that can be applied to the ChainFL simulation platform.

A. Case Study 1: Collaborate Edge and Cloud Computing

ChainFL can be used both as a simulation platform and as a deployment platform for federated learning. On the one hand, ChainFL can simulate a complex IoT environment supporting edge computing with a small number of devices, which in turn provides an experimental platform for task offloading models to evaluate their effectiveness in a complex environment. Furthermore, by embedding a task offloading decision model into ChainFL, ChainFL can serve as a complete task offloading model for edge computing. The model is embedded with a federated learning mechanism that can be deployed on devices under different systems to build a complex network including multiple clients, edges and clouds, and then implement complex task offloading at the clients. The ChainFL platform is very easy to deploy, users only need to know the IP of the communication center and run the specified communication code on the device to access the system and be recognized by other devices. By allowing the customization of transport encryption algorithms with blockchain algorithms, ChainFL has better security compared to currently available federated learning platforms and can be applied to complex scenarios, e.g., industrial cognitive Internet of Vehicles [35].

As illustrated in Fig. 10, the embedded intelligent offloading decision algorithm is located on the user side, which generates task offloading decisions by receiving task information, and then offloads the task either to the edge or the cloud. After each training period, the client will automatically upload the parameters of the neural network for federated learning, and update the learned parameters to the network. The ChainFL platform provides a way for many offloading decision-making

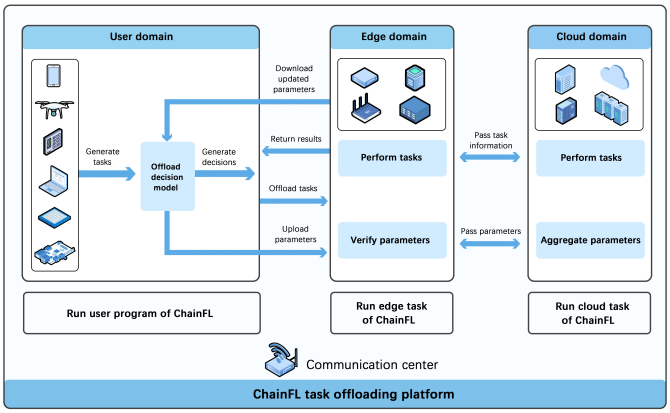


Fig. 10: The task offloading process with ChainFL in collaborate edge and cloud computing environments

algorithms that are in the research stage, which can be deployed in a real-world IoT environment and perform a real task offloading process.

B. Case Study 2: Joint Federated Learning and Blockchain Optimization

Compared with existing simulation platforms, ChainFL has made more optimizations for federated learning and blockchain algorithms, so it can train and evaluate the effects of traditional federated learning algorithms. Fig. 11 shows the network structure of the ChainFL platform. For the blockchain algorithm, the cloud is a full node, while the client and edge are lightweight nodes. Due to the unique network structure of ChainFL, it is more conducive to the generation and propagation of blockchain, which can promote the application of blockchain in edge computing environments. In practical deployments, ChainFL allows blockchain algorithms to customize their consensus algorithms and mining algorithms, thus increasing the compatibility of the platform. Specifically, ChainFL does not embed specific blockchain algorithms, and users can use it by introducing the required packages to complete the corresponding functions.

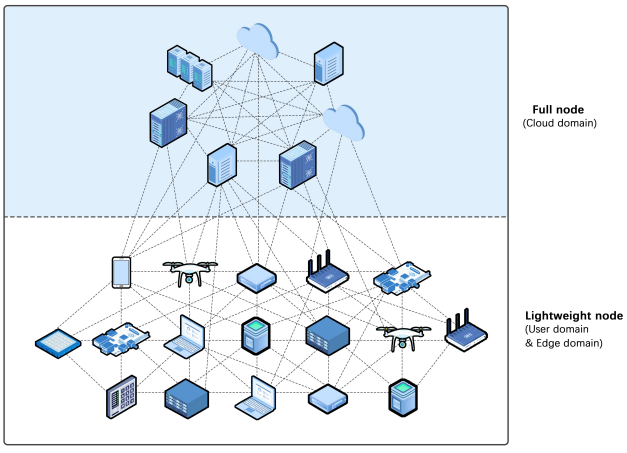


Fig. 11: The network structure of the ChainFL platform

VI. Conclusion and Future Work

In response to the complex conditions in the field of IIoTs, we have developed ChainFL, a novel simulation platform that provides a more realistic environment for offloading decision models for tasks using different algorithms. In order to fully improve the compatibility of the simulation platform, we have simplified ChainFL into a lightweight toolkit that is compatible with devices of different architectures, thus increasing the feasibility of ChainFL. In addition, ChainFL can be deployed as a federated learning platform with enhanced security to address the challenges posed by big data and potential attacks in the IIoTs. We have further validated its efficiency, lightweight and security by embedding a complex task offloading framework in an environment with potential attacks.

As a part of future work, we will utilize ChainFL in serverless edge computing to provide more scalability and reliability, as well as reduce costs for delay-sensitive tasks. We believe that ChainFL can still perform parameter aggregation via the edge server to complete the normal federation learning process in serverless edge computing environments. In addition, we will try to combine quantum computing to explore better encryption algorithms, in order to provide models with better security and high computational capacity.

Acknowledgment

This work was partly supported by the National Natural Science Foundation of China under Grant No. 62071327 and 61801325, JSPS KAKENHI under Grant No. JP19H04105, Tianjin Research Innovation Project for Postgraduate, and CCF-Tencent Open Research Fund.

References

[1] M. Xiong, Y. Li, L. Gu, S. Pan, D. Zeng, and P. Li, "Reinforcement learning empowered idps for vehicular networks in edge computing," *IEEE Network*, vol. 34, no. 3, pp. 57–63, 2020.

[2] 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 Vehicular Technology Magazine*, vol. 15, no. 4, pp. 122–134, 2020.

[3] J. Wang, C. Jiang, K. Zhang, X. Hou, Y. Ren, and Y. Qian, "Distributed q-learning aided heterogeneous network association for energy-efficient iiot," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2756–2764, 2020.

[4] D. Xu, T. Li, Y. Li, X. Su, S. Tarkoma, and P. Hui, "Edge intelligence: Architectures, challenges, and applications," *ArXiv*, vol. abs/2003.12172, 2020.

[5] G. Manogaran, S. Mumtaz, C. X. Mavromoustakis, E. Pallis, and G. Mastorakis, "Artificial intelligence and blockchain-assisted offloading approach for data availability maximization in edge nodes," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 3, pp. 2404–2412, 2021.

[6] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 4, pp. 2322–2358, 2017.

[7] Z. Zhou, H. Liao, X. Wang, S. Mumtaz, and J. Rodriguez, "When vehicular fog computing meets autonomous driving: Computational resource management and task offloading," *IEEE Network*, vol. 34, no. 6, pp. 70–76, 2020.

[8] S. Yu, X. Chen, L. Yang, D. Wu, M. Bennis, and J. Zhang, "Intelligent edge: Leveraging deep imitation learning for mobile edge computation offloading," *IEEE Wireless Communications*, vol. 27, no. 1, pp. 92–99, Feb. 2020.

[9] G. Qu, H. Wu, R. Li, and P. Jiao, "DMRO: A deep meta reinforcement learning-based task offloading framework for edge-cloud computing," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3448–3459, 2021.

[10] C. Jiang, N. Ge, and L. Kuang, "AI-enabled next-generation communication networks: Intelligent agent and AI router," *IEEE Wireless Communications*, vol. 27, no. 6, pp. 129–133, 2020.

[11] Y. Zhan, P. Li, Z. Qu, D. Zeng, and S. Guo, "A learning-based incentive mechanism for federated learning," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6360–6368, 2020.

[12] Y. Li, C. Chen, N. Liu, H. Huang, Z. Zheng, and Q. Yan, "A blockchain-based decentralized federated learning framework with committee consensus," *IEEE Network*, vol. 35, no. 1, pp. 234–241, 2021.

[13] M. Mukherjee, S. Kumar, C. X. Mavromoustakis, G. Mastorakis, R. Matam, V. Kumar, and Q. Zhang, "Latency-driven parallel task data offloading in fog computing networks for industrial applications," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6050–6058, 2020.

[14] M. Mukherjee, V. Kumar, S. Kumar, R. Matamy, C. X. Mavromoustakis, Q. Zhang, M. Shojafar, and G. Mastorakis, "Computation offloading strategy in heterogeneous fog computing with energy and delay constraints," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–5.

[15] H. Wu, K. Wolter, P. Jiao, Y. Deng, Y. Zhao, and M. Xu, "Eedto: An energy-efficient dynamic task offloading algorithm for blockchain-enabled iot-edge-cloud orchestrated computing," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2163–2176, 2021.

[16] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, "Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and experience*, vol. 41, no. 1, pp. 23–50, 2011.

[17] H. Gupta, A. Vahid Dastjerdi, S. K. Ghosh, and R. Buyya, "iFogSim: A toolkit for modeling and simulation of resource management techniques in the internet of things, edge and fog computing environments," *Software: Practice and Experience*, vol. 47, no. 9, pp. 1275–1296, 2017.

[18] J. Son, A. V. Dastjerdi, R. N. Calheiros, X. Ji, Y. Yoon, and R. Buyya, "Cloudsimsdn: Modeling and simulation of software-defined cloud data centers," in *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE, 2015, pp. 475–484.

[19] S. F. Piraghaj, A. V. Dastjerdi, R. N. Calheiros, and R. Buyya, "Containercloudsim: An environment for modeling and simulation of containers in cloud data centers," *Software: Practice and Experience*, vol. 47, no. 4, pp. 505–521, 2017.

[20] S. Lu, Y. Yao, and W. Shi, "CLONE: Collaborative learning on the edges," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10 222–10 236, 2021.

[21] S. Wang, Y. Hu, and J. Wu, "KubeEdge. AI: AI platform for edge devices," *arXiv e-prints*, pp. arXiv–2007, 2020.

[22] Y. Zhang, M. Saberi, M. Wang, and E. Chang, "K3S: Knowledge-driven solution support system," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 9873–9874.

[23] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar, "Leaf: A benchmark for federated settings," *arXiv preprint arXiv:1812.01097*, 2018.

[24] T. Ryffel, A. Trask, M. Dahl, B. Wagner, J. Mancuso, D. Rueckert, and J. Passerat-Palmbach, "A generic framework for privacy preserving deep learning," *arXiv preprint arXiv:1811.04017*, 2018.

[25] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, "Federated learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 13, no. 3, pp. 1–207, 2019.

[26] Y. Ma, D. Yu, T. Wu, and H. Wang, "Paddlepaddle: An open-source deep learning platform from industrial practice," *Frontiers of Data and Domputing*, vol. 1, no. 1, pp. 105–115, 2019.

[27] C. He, S. Li, J. So, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu, L. Shen, P. Zhao, Y. Kang, Y. Liu, R. Raskar, Q. Yang, M. Annavaram, and S. Avestimehr, "FedML: A research library and benchmark for federated machine learning," *arXiv preprint arXiv:2007.13518*, 2020.

[28] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, "Edge intelligence: Paving the last mile of artificial intelligence with edge computing," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1738–1762, 2019.

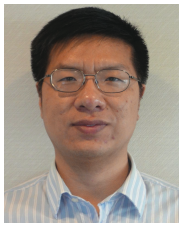
[29] L. Wang, K. Wang, C. Pan, W. Xu, N. Aslam, and A. Nallanathan, "Deep reinforcement learning based dynamic trajectory control for UAV-assisted mobile edge computing," *IEEE Transactions on Mobile Computing*, 2021.

[30] L. Huang, X. Feng, L. Qian, and Y. Wu, "Deep reinforcement learning-based task offloading and resource allocation for mobile edge computing," in *International Conference on Machine Learning and Intelligent Communications*. Springer, 2018, pp. 33–42.
[31] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, "Learning-based computation offloading for iot devices with energy harvesting," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 2, pp. 1930–1941, 2019.
[32] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.
[33] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Generation Computer Systems*, vol. 115, pp. 619–640, 2021.
[34] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ecdsa)," *International journal of information security*, vol. 1, no. 1, pp. 36–63, 2001.
[35] X. Xu, B. Shen, X. Yin, M. R. Khosravi, H. Wu, L. Qi, and S. Wan, "Edge server quantification and placement for offloading social media services in industrial cognitive IoV," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2910–2918, 2021.

Guanjin Qu received the bachelor's degree from Taiyuan University of Technology, China in 2019. He is currently working towards the Master's degree at the Center for Applied Mathematics, Tianjin University, China. His research interests include distributed deep learning and edge computing.

Naichuan Cui is currently working toward M.Sc of Signal Processing at School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His current research interests include image and video processing, content-based multimedia analysis, machine learning and signal processing.

Huaming Wu received the B.E. and M.S. degrees from Harbin Institute of Technology, China in 2009 and 2011, respectively, both in electrical engineering. He received the Ph.D. degree with the highest honor in computer science at Freie Universität Berlin, Germany in 2015. He is currently an Associate Professor in the Center for Applied Mathematics, Tianjin University, China. His research interests include wireless networks, mobile edge computing, internet of things and deep learning.



Ruidong Li is an associate professor at Kanazawa University, Japan. Before joining this university, he was a senior researcher at the National Institute of Information and Communications Technology (NICT), Japan. He received the M.Sc. degree and Ph.D. degree in computer science from the University of Tsukuba in 2005 and 2008, respectively. He serves as the secretary of IEEE ComSoc Internet Technical Committee (ITC), and are the founders and chairs of IEEE SIG on Big Data Intelligent Networking and IEEE SIG on Intelligent Internet Edge. He is the associate editor of IEEE Internet of Things Journal, and also served as the guest editors for a set of prestigious magazines, transactions, and journals, such as IEEE communications magazine, IEEE network, IEEE TNSE. He also served as chairs for several conferences and workshops, such as the general co-chair for IEEE MSN 2021, AIVR2019, IEEE INFOCOM 2019/2020/2021 ICCN workshop, TPC co-chair for IWQoS 2021, IEEE MSN 2020, BRAINS 2020, IEEE ICDCS 2019/2020 NMIC workshop, and ICCSSE 2019. His research interests include future networks, big data, intelligent Internet edge, Internet of things, network security, information-centric network, artificial intelligence, quantum Internet, cyber-physical system, and wireless networks. He is a senior member of IEEE and a member of IEICE.

Yuemin Ding received the Ph.D. degree in electronic systems engineering from Hanyang University, Ansan, South Korea, in 2014. He is currently an Associate Professor with the Department of Electrical and Electronic Engineering, University of Navarra, since 2021. From 2019 to 2021, he was a Postdoctoral Fellow at the Department of Energy and Process Engineering, Norwegian University of Science and Technology, Trondheim, Norway. From 2015 to 2019, he was an Associate Professor at the School of Computer Science and Engineering, Tianjin University of Technology, China. From 2017 to 2018, he was a Visiting Fellow with the Queensland University of Technology, Brisbane, QLD, Australia. His research interests include Internet of Things, communication networks in smart grid, smart homes/buildings, and smart manufacturing.