Blockchain and Federated Edge Learning for Privacy-Preserving Mobile Crowdsensing

Qin Hu, Zhilin Wang, Minghui Xu, and Xiuzhen Cheng, *Fellow, IEEE*

***Abstract*—Mobile crowdsensing (MCS) counting on the mobility of massive workers helps the requestor accomplish various sensing tasks with more flexibility and lower cost. However, for the conventional MCS, the large consumption of communication resources for raw data transmission and high requirements on data storage and computing capability hinder potential requestors with limited resources from using MCS. To facilitate the widespread application of MCS, we propose a novel *MCS learning framework* leveraging on blockchain technology and the new concept of edge intelligence based on federated learning (FL), which involves four major entities, including requestors, blockchain, edge servers and mobile devices as workers. Even though there exist several studies on blockchain-based MCS and blockchain-based FL, they cannot solve the essential challenges of MCS with respect to accommodating resource-constrained requestors or deal with the privacy concerns brought by the involvement of requestors and workers in the learning process. To fill the gaps, four main procedures, i.e., task publication, data sensing and submission, learning to return final results, and payment settlement and allocation, are designed to address major challenges brought by both internal and external threats, such as malicious edge servers and dishonest requestors. Specifically, a mechanism design based data submission rule is proposed to guarantee the data privacy of mobile devices being truthfully preserved at edge servers; consortium blockchain based FL is elaborated to secure the distributed learning process; and a cooperation-enforcing control strategy is devised to elicit full payment from the requestor. Extensive simulations are carried out to evaluate the performance of our designed schemes.**

***Index Terms*—Mobile crowdsensing, federated learning, data privacy, blockchain, game theory.**

I. Introduction

FACILITATED by a variety of embedded sensors on mobile devices and ubiquitous Internet access opportunities, mobile crowdsensing (MCS) evolves into a vigorous paradigm [1], benefiting the data collection of Internet of Things (IoT) and various practical applications, such as transportation monitoring [2], localization and navigation [3]. There are also an increasing number of mobile apps based on MCS, providing great convenience for our daily lives, such as Waze and Uber. In general, the success of MCS lies in the solicitation of distributed sensing capabilities of mobile devices across a large-scale area to serve for a major task in an environmentally friendly way, which can significantly reduce the cost of deploying traditional sensors in a large number of fixed locations.

Currently, mainstream research on MCS put efforts on incentive mechanisms design [4]–[6], quality control [7], and privacy preservation [8], lacking attention on huge communication resource consumption for raw data transmission. Moreover, the large chunk of data collected from mobile workers put certain requirements on the data storage, processing and analysis capabilities of MCS requestors, which can prevent resource-constrained users from exploring and experiencing this promising computing model, thus hindering the wider application of MCS.

To overcome these challenges, we propose a novel *MCS learning framework* for the first time, leveraging on blockchain technology and the new concept of edge intelligence based on federated learning (FL). Four parts are involved in this framework in a top-down order, namely requestors, blockchain system, edge servers and mobile devices as workers, where the blockchain is maintained by a group of consortium members specified at the initialization of the MCS learning system while all other entities access to the blockchain via a client application.

As a matter of fact, there exist plentiful studies about blockchain-based MCS [9]–[30] and blockchain-based FL [31]–[46]. Each can be further classified into tightly-coupled and loosely-coupled frameworks, where the tightly-coupled one employs blockchain to replace the originally centralized MCS platform or parameter aggregator in FL so as to achieve full decentralization and avoid the single point of failure, while the loosely-coupled framework implements the blockchain as an independent module to serve for some specific functions, such as the reputation management of MCS workers and FL participants. Our proposed framework compactly integrates blockchain into both MCS and FL, so the aforementioned existing schemes are not applicable to MCS learning due to the following two reasons. On the one hand, blockchain-based MCS still suffers from the huge cost of data transmission and storage, which becomes even worse by using the blockchain since every blockchain node is supposed to hold a copy of the complete data about the main chain; besides, the requirement on data processing capability of the requestor continues to be a difficulty for resource-limited users. On the other hand, blockchain-based FL cannot be directly employed here, either, since the additionally involved requestors and mobile workers can bring new challenges to privacy protection in FL.

To fill the gaps, we design main procedures for our proposed

This work is partly supported by the US NSF under grant CNS-2105004, the National Key RD Program of China under grant 2019YFB2102600, and the National Natural Science Foundation of China under grant 61832012.

Qin Hu and Zhilin Wang are with the Department of Computer and Information Science, Indiana University-Purdue University Indianapolis, USA. E-mail: {qinhu,wangzhil}@iu.edu

Minghui Xu (Corresponding author) and Xiuzhen Cheng are with the School of Computer Science and Technology, Shandong University, China. E-mail: {mhxu,xzcheng}@sdu.edu.cn

MCS learning system, including i) task publication, ii) data sensing and submission, iii) learning to return final results, and iv) payment settlement and allocation. The first procedure is initiated by the requestor sending an MCS learning request and then finished via a smart contract stored on the blockchain for worker recruitment. The second step is executed by the mobile devices and edge servers where devices undertake the sensing jobs according to the requirement of the requestor published on the blockchain and upload sensing data to the nearest edge servers for rewards. Edge servers can thus collect sensing data and establish local datasets for the next learning stage. FL is implemented in the third procedure where edge servers utilize their local data to collaboratively train the assigned machine learning (ML) model without leaking the raw data of mobile devices. And the final step is executed between the requestor and the blockchain once the task requirement is satisfied, where the blockchain system aims to successfully charge the payment from the requestor so that the fair distribution of rewards to all participants can be achieved for motivating their continuous contribution in the future.

However, there exist several challenges in the above-mentioned procedures. First, mobile devices may seriously concern whether the submitted sensing data to the edge servers can be protected from leakage; second, the conventional FL is vulnerable to malicious attacks on the centralized aggregator; third, the requestor might be reluctant to compensate the cost of all participants throughout the whole MCS learning process. Via solving these challenges, we make the following contributions:

- A novel and holistic MCS learning framework is proposed to conduct MCS and the subsequent data analysis in an integrated manner, which can not only lower the communication consumption and the requirement of computing and storage capabilities for requestors, but also preserve the privacy of workers at the local edge server level.
- To guarantee that the submitted data of mobile devices are preserved at edge servers without leaking to others, we resort to mechanism design theory for devising an incentive-compatible game rule to restrict the privacy disclosing behaviors of edge servers.
- For securing the learning process, consortium blockchain based FL is deployed, where the distributed ledger maintained by consortium nodes undertakes the coordination job and edge servers with local sensing data participate in the collaborative learning.
- To fully elicit compensation from the requestor, we employ the zero-determinant (ZD) game theory to work out a cooperation-enforcing control scheme, which is both theoretically and experimentally proved to be effective in driving requestors to pay fully, thus safeguarding the interests of all participants in the MCS learning system.

The remaining of this paper is organized as follows. We summarize the most related work in Section II and present the overall system model in Section III. Detailed designs of main procedures are elaborated in Section IV, followed by experimental evaluations on specifically designed schemes in Section V. Finally, we conclude the whole paper in Section VI.

II. Related Work

Since the proposed MCS learning framework is largely unexplored, here we mainly summarize the most related work about blockchain-based MCS [9]–[30] and blockchain-based FL [31]–[46].

A. Blockchain-based Mobile Crowdsensing

Most of the research on blockchain based MCS deeply embed the blockchain system into MCS framework via employing the blockchain to undertake jobs of the traditional centralized MCS platform, such as task allocation, incentive mechanism implementation, and sensing data verification. In [9]–[13], smart contracts were employed to handle the interactions between the requestors and workers in an automatic manner. Further, Liang *et al.* [14] used Trusted Execution Environments to avoid potential malice from requestors in blockchain-based MCS. To guarantee the data privacy of workers, Cai *et al.* [15] employed secret sharing technique to enable flexible submission aggregation among blockchain nodes and workers. TrustWorker, a trustworthy and privacy-preserving worker selection mechanism for blockchain-assisted crowdsensing, adopted the deterministic encryption method to facilitate worker selection and privacy protection [20]. In [21], URIM was introduced as an incentive mechanism to maximize the utilities of participants in blockchain-based crowdsensing. With more focus on specific performance improvement of MCS, An *et al.* [16], [17] utilized the blockchain nodes as verifiers to achieve *quality control*, and researchers in [18], [19], [22], [23] took advantage of the distributed blockchain nodes to realize k-anonymity for workers so as to protect their *location privacy* in MCS. Blockchain was also implemented in vehicular crowdsensing [24], [25] and radio frequency powered MCS [26].

Other studies utilizing blockchain for MCS work in a loosely coupled fashion, where the blockchain network is used to facilitate some specific MCS procedure(s) as an independent function module rather than acting as the distributed MCS platform. In [27], [28], blockchain was adopted to evaluate or verify the submissions from workers so as to eliminate their malicious behaviors and resist information tampering. Jia *et al.* [29] designed a blockchain-powered confusion mechanism to hide the real locations of workers in MCS. While blockchain in [30] was mainly introduced to manage the reputation of workers at the network edge.

B. Blockchain-based Federated Learning

Blockchain applied in FL mainly aims at achieving fully distributed machine learning without a centralized aggregator. To coordinate learning procedures among all participated clients, blockchain stores all learning related information, such as initial model, local updates, and globally aggregated model. As blockchain can be classified into two general types, i.e., public and permissioned, researchers considered utilizing both

blockchains in FL. For public chain based FL, FLChain was studied in [31]. Majeed *et al.* [32] proposed another FLchain for mobile edge combing enabled FL. The work in [33] utilized the public blockchain to verify the data for FL in the healthcare industry. Kim *et al.* [35] designed BlockFL architecture and analyzed the end-to-end latency. Pokhrel *et al.* [36], [37] utilized the public blockchain with proof-of-work consensus algorithm to facilitate on-vehicle machine learning. And mechanism design was involved in [38] to guarantee the honesty of clients in public blockchain based FL platform. For permission chain based FL, Weng *et al.* [39] devised DeepChain to achieve auditability of the whole FL training process with the help of Algorand consensus protocol. In [34], a permissioned blockchained FL system with the differential privacy method was designed to predict the traffic flow. Lu *et al.* [40] integrated the FL in the consensus of permissioned blockchain and took advantage of it to realize privacy-preserving data sharing for industrial IoT. Dynamic weighting scheme to improve learning performance was investigated in [41] for the private blockchain powered FL. And Preuveneers *et al.* [42] applied the permissioned blockchain based FL on intrusion detection. To adapt to the Internet of Vehicles (IoV) scenario with increasing security and reliability of FL, a hybrid blockchain system consisted of the permissioned chain and the local Directed Acyclic Graph (DAG) was proposed in [43], where a deep reinforcement learning based client selection was also designed to further improve the FL efficiency. Desai *et al.* [46] designed a hybrid blockchain-based FL framework to automatically prevent the system from being attacked by malicious users via smart contracts.

The blockchain was also employed to operate independently to enhance FL. In [44], blockchain was employed to realize reputation management for FL clients in a reliable manner. And Awan *et al.* [45] utilized the blockchain technique to guarantee provenance of model updates in FL so as to avoid intentional malice from clients.

It is clear that the existing studies are not applicable to MCS learning. In detail, blockchain-based MCS still fails to address main challenges of serving resource-limited requestors, while blockchain-based FL cannot be directly adopted since the involvement of requestors and mobile workers may hamper data privacy protection. In this paper, we design specific schemes based on mechanism design and game theory to solve these challenges, thus facilitating the function of the MCS learning system.

III. System Model

With the pervasive wireless network access opportunity, there is an increasing demand for mobile crowdsensing, where mobile devices (e.g., smart phones with embedded sensors) can collect sensing data from various locations all over the world as required by the requestor. On the one hand, the huge amount of sensing data can make the data processing and analysis a big challenge for potential MCS requestors with limited computation resource, hindering the large-scale application of MCS. On the other hand, nowadays, workers in MCS become increasingly concerned about the privacy leakage during the sensing data submission process, discouraging their active participation. To solve these challenges, we propose an MCS learning system with privacy preservation based on federated learning and blockchain, by which any requestor can easily obtain the MCS data analysis results online without worrying about the sequel data processing cost while mobile devices as workers can be assured to participate in data sensing with a certain degree of privacy protection.

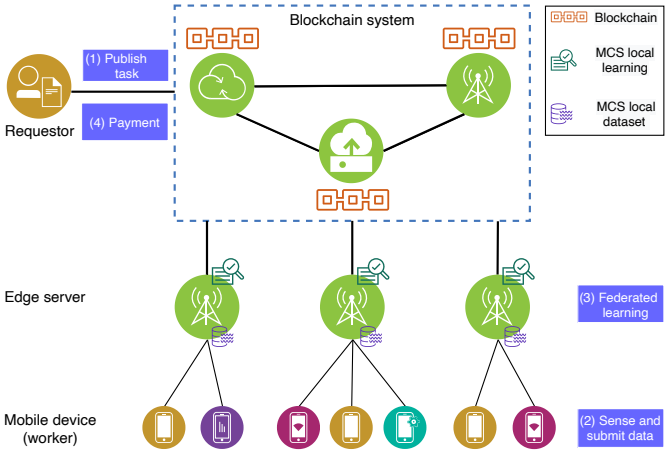


Fig. 1. The proposed MCS learning framework, consisting of requester, consortium blockchain, edge servers, and mobile devices. After the requester publishes the MCS task to the blockchain network, mobile devices will be recruited as workers to collect data and send them to the nearest edge servers, where the raw data stored at the edge servers will be proceeded via federated learning; appropriate rewards will be distributed to all participates once the learning process finishes.

As shown in Fig. 1, our proposed MCS learning system consists of four main parts, i.e., requestor, blockchain system, edge servers, and mobile devices as workers, where the blockchain is maintained by a predefined group of consortium members while all other entities connect to the blockchain system as clients via an interface. To accomplish an MCS learning request, we consider four general procedures as follows:

(1) **Task publication.** The requestor sends an MCS learning request to the blockchain network, which will be recorded on the main chain with necessary task information, such as the requestor's identity, task description, and performance requirement. This on-chain record will trigger a smart contract to recruit appropriate workers.

(2) **Data sensing and submission.** After workers receive the MCS task, they collect the required data in their convenience and submit sensing data to the nearest edge server so as to alleviate its own storage burden and accelerate the whole MCS process. All sensing data collected at the edge server constitutes a private dataset, which will be locally processed during the next stage. By this means, the private information of mobile devices (e.g., location) can be preserved to some extent as the raw data will only be visible to their local edge servers rather than the centralized platform in the conventional MCS. Besides, workers can receive sensing reward from edge servers in a timely manner.

(3) **Learning to return final results.** To avoid privacy

leakage, edge servers with local datasets will collaboratively conduct machine learning based on the federated learning (FL) framework to derive the required model for the requestor.

(4) **Payment settlement and allocation.** Once the FL achieves the performance requirement, the requestor can easily obtain the final learning results from the blockchain system. To motivate continuous contribution from all participants during the whole MCS learning process, an appropriate payment will be charged from the requestor to cover sensing rewards for mobile devices, learning rewards for edge servers, and block rewards for blockchain nodes.

However, within the above working process, there exist several problems threatening the implementation of the MCS learning system. First, whether the edge server can keep the promise of protecting the data privacy for mobile devices remains an uncertainty; second, the centralized parameter aggregator in FL is prone to be attacked by both inside computation clients and outsiders, leading to the risk of inefficient learning or model breaches; third, given potential malicious requestors, it is challenging to successfully elicit enough payment from the requestor so that all contributors can be rewarded properly. All these concerns need to be carefully considered and well addressed in the detailed design of the MCS learning system, which will be discussed in the next section.

IV. Design of Main Procedures

In this section, we detail the design of our proposed MCS learning system via specifying the aforementioned main steps and solving the aforementioned challenges. For reference, we summarize key notations used in this section in Table I.

TABLE I
KEY NOTATIONS.

Notation	Meaning
θ	The data privacy leakage degree of the edge server
v	The self evaluation of device on the value regarding the sensing data
s	The amount of profit paid by the server to the device
U_s	The expected utility of the edge server with respect to the device's sensing data
R	The amount of reward obtained from the requestor
$r(\theta)$	The extra reward of leaking the sensing data
$g(v,s)$	The successful data collection probability between the device and the edge server
U_d	The expected utility of the mobile device
C	The overall cost of the device
W	The total amount of incentive each requestor needs to pay
a_r	The action of the requestor finalizing the ML in blockchain
a_b	The action of the leader finalizing the ML in blockchain
$\mathbf{p}$	The mixed strategies of the leader
$\mathbf{q}$	The mixed strategies of the requestor

A. Task Publication

Since there exists no centralized MCS platform in our proposed MCS learning system, the task publication step will be accomplished by the decentralized blockchain system. In detail, when the requestor needs to finish an MCS learning task, a request following the predefined format will be sent to the blockchain system as a transaction via the blockchain interface, which can uniquely characterize this request with task-related information, such as the requestor's identity, task description, and learning performance requirement[1]. Once entering the blockchain network, this request will be verified and widely forwarded to reach most of the consortium members so that it can be efficiently recorded on the main chain.

After the request is visible on the main chain to all blockchain nodes, the smart contract for worker recruitment will be triggered, which is a piece of program stored on the blockchain with a specific address. To invoke this smart contract, the requestor sets the recipient of the request as the address of the smart contract and then the information included in this request will be processed as input of this worker recruitment program. Note that the logic of this smart contract can directly follow the ideas of existing worker selection and assignment algorithms for MCS [5], [6], where the only difference is that via using the smart contract, the worker recruitment process is executed by peering blockchain nodes in a decentralized manner.

B. Mechanism Design for Data Submission

After the determination of worker recruitment in the previous stage, selected mobile devices can start collecting required sensing data at specific locations and then submit the sensed data to the nearest edge server for further processing. By doing so, devices can expect to achieve local privacy preservation at the level of edge servers as their submitted data will be processed and analyzed by the server locally, along with the collected sensing data from other connected mobile devices, instead of being directly submitted to a global platform. Nevertheless, even with this computation paradigm, devices may still have the concern about whether edge servers can really keep the local privacy protection expectation in practice without leaking the received sensing data to other parties, which is also a private behavior for the edge server and thus can never be explicitly known to devices. To solve this concern, we take advantage of the mechanism design theory [48] which empowers devices to utilize the market power to restrain potential data privacy leakage behavior of edge servers.

As indicated by *revelation principle* in mechanism design, any Nash equilibrium of a Bayesian game (i.e., incomplete-information game) can be identically achieved in another incentive-compatible direct mechanism where game players report their private information truthfully. In the sensing data submission scenario, as mentioned above, we define the data privacy leakage degree of the edge server as its private information and denote it as $\theta \in [0,1]$, while the objective of the mobile device is to design a mechanism, or a game rule in other words, which can push the rational server to

[1]Other information can also be defined to describe an MCS learning request but cannot be fully listed here. For example, the requestor may also submit a test dataset for accuracy evaluation, which can be stored on the blockchain with an address based on InterPlanetary File System (IPFS) [47].

behave based on its real private information θ for obtaining the optimum payoff.

To maintain the long-term participation enthusiasm of workers in MCS, mobile devices should be compensated with sensing rewards. Since mobile devices accomplish sensing tasks with highly random trajectories, making it impractical for some devices to return to the previously connected edge servers for obtaining sensing rewards, we consider that edge servers pay devices for the submitted sensing data in an immediate manner. Specifically, the device has a self-evaluation on the value of the sensing data, denoted by v, to cover at least the sensing cost; and the server will decide how much profit it likes to pay to the device, denoted by s, which leads to a total payment of $v + s$ for the sensing data.

During a period of time $[0, T]$, the expected utility of the edge server with respect to the device's sensing data can be defined as

$$U_s = \int_0^T (R + r(\theta) - v - s) \cdot g(v, s) dt, \tag{1}$$

where R is the amount of reward obtaining from processing it for the requestor, and $r(\theta)$ is the extra reward of leaking the sensing data, defined as $r(\theta) = \alpha_s \theta + \beta_s$ with coefficients $\alpha_s, \beta_s \geq 0$. Besides, $g(v, s)$ indicates the successful data collection probability between the device and the edge server, which is defined as

$$g(v, s) = \epsilon \frac{v}{\bar{v}} + (1 - \epsilon) \frac{s}{\bar{s}}. \tag{2}$$

In the above equation, $\epsilon \in [0, 1]$ is a scalar; $\bar{v}$ and $\bar{s}$ are the maximum values of v and s, respectively. In detail, the successful probability is positively related to both the server's decision on the profit it willing to offer to the device and the device's self-assessment of data value. It is worth noting that the above definition of $g(v, s)$ is known by both sides prior to the mechanism design process.

At the same time, the expected utility of the mobile device can be calculated by

$$U_d = \int_0^T (v + s - C) \cdot g(v, s) dt, \tag{3}$$

where $C = c_d + \eta c_s$ is the overall cost of the device with c_d denoting the sensing cost, c_s representing the expected lost caused by the privacy leakage behavior of the server, and η being a positive scalar. Considering that the privacy-leaking lost is related to the value of the device's sensing data, we define $c_s = \xi v$ where $\xi > 0$.

According to mechanism design theory, v and s are the strategies of the device and the server, respectively, where v is a function of s, acting as a game rule derived by the device for guaranteeing the privacy protection behavior of the server. In particular, the delicately designed game rule $v^*(s)$ can push the server to select the strategy s based on its real private information θ. The overall process is summarized in Algorithm 1 and can be described as follows:

1) The device proposes a game rule $v^*(s)$ to maximize the expected utility U_d and sends it to the server.
2) After receiving $v^*(s)$, the server calculates the best strategy s^* maximizing the expected utility U_s based on the hidden private information θ. With s^* and further derived $v^*(s)$, the server can determine whether it is profitable to accept the game rule or not. If the answer is yes, the server will send s^* back to the device.
3) If the device receives the returned s^* from the server within a certain time limit, v^* can be calculated accordingly; otherwise, the device will terminate the sensing data submission process.

Specifically, $v^*(s)$ and s^* can be calculated by maximizing U_d and U_s, respectively. To maximize U_d in (3), we denote the integrand as $F_d = (v + s - C) \cdot g(v, s)$. Employing the calculus of variations method, we can derive $v^*(s)$ via solving the associated Euler-Lagrange equation $\frac{\partial F_d}{\partial v} - \frac{d}{dt}\frac{\partial F_d}{\partial v'} = 0$ under the constraint $\frac{\partial^2 F_d}{\partial v^2} < 0$. Accordingly, we can obtain that under the condition of $\eta\xi > 1$, the optimal game rule of the device is

$$v^*(s) = \frac{(1 - \epsilon)(\eta\xi - 1)\bar{v}s - \epsilon\bar{s}(s - c_d)}{2\epsilon(1 - \eta\xi)\bar{s}}. \tag{4}$$

With the above $v^*(s)$, we can obtain s^* through maximizing U_s in (1) using the similar method. To avoid redundancy, we omit the detailed calculation process of s^* and report the result as follows. Let

$$A_0 = \frac{\epsilon}{\bar{v}} + \frac{(\eta\xi - 1)(\epsilon - 1)}{\bar{s}},$$
$$A_1 = 2(\eta\xi - 1),$$

then we can have the optimal strategy of the edge server as

$$s^* = \frac{R + \alpha\theta + \beta + \frac{c_d}{A_1}}{2(\frac{\bar{v}A_0}{\epsilon A_1} + 1)} + \frac{c_d\epsilon}{2\bar{v}A_1(\frac{A_0}{A_1} - \frac{\epsilon - 1}{\bar{s}})}.$$

In fact, our proposed mechanism satisfies the incentive compatibility constraint, which can be demonstrated by the following theorem.

Theorem IV.1. *The game rule $v^*(s)$ proposed by the device is incentive-compatible.*

Proof. Let $\hat{\theta}$ be the fake private information of the server. Then the optimal strategy of the server based on this $\hat{\theta}$, denoted as $\hat{s}^*$, will be sent to the device. As $\hat{\theta}$ is different from the real θ, we have $\hat{s}^* \neq s^*(\theta)$. On the other hand, since $s^*(\theta)$ is calculated via maximizing U_s, we have $U_s(\hat{s}^*) < U_s(s^*)$, which makes the server's behavior of submitting the optimal strategy based on its fake private information not beneficial. Thus, any profit-driven and reasonable server will choose to respond with the real private information, which demonstrates the incentive compatibility of our proposed mechanism. □

Since the payment of $v + s$ for each device is covered in advance by the edge server, the total amount of payment for all locally connected devices accomplishing a specific MCS learning task should be reported to the blockchain system during the following FL step so that the edge server can be appropriately compensated finally from the requestor's payment. The specific allocation of compensation will be elaborated in Section IV-D.

Algorithm 1 Mechanism Design based Data Submission

Raw data ← Mobile devices collect data
$v^*(s)$ ← The device determines the optimal game rule according to (4)
The device sends $v^*(s)$ to the edge server
U_s ← The edge server calculates U_s based on $v^*(s)$
s^* ← $\max(U_s)$
if Accepting $v^*(s)$ is profitable **then**
    The edge server sends s^* to the local device
    v^* ← $\max(U_d)$
    The device submits the raw data to the edge server
else
    The edge server keeps silent
    The device terminates the sensing data submission process
end if

C. Blockchain-based Federated Learning at Edge

With the received sensing data from mobile devices, edge servers can construct their local datasets so that they can further collaboratively conduct FL to derive the global learning results for requestors. However, there always exists an aggregator, such as a central server, to coordinate the FL process among all participated computing parties. Although some research suggest multiple servers to take the work of aggregation, the logical topology is still centralized, suffering from lots of weaknesses, such as single point of failure.

To fundamentally address these problems, the concept of fully decentralized FL is proposed where FL participants communicate with each other via the peer-to-peer channels. As an exemplary implementation of this concept, blockchain [32], [40] based FL is widely studied recently as discussed in Section II. Thus, in our proposed MCS learning framework, we also consider to employ the decentralized ledger to replace the centralized aggregator in FL, where a set of blockchain nodes undertake the coordination jobs. Specifically, the consortium blockchain is used here, which is maintained by a group of consortium members responsible for validating all content waiting to be included on the main chain as permanent records.

For the reason of using a consortium blockchain instead of a public or private one to achieve distributed FL in our proposed MCS learning scenario, it is quite straightforward. On the one hand, either the FL model updates or MCS task related records on the blockchain are not supposed to be unlimitedly available to anyone from the public; on the other hand, blockchain nodes are not authorized by one centralized entity but several predefined members, such as trusted edge servers, and they need to be general and scalable enough to achieve a certain level of decentralization of the blockchain system. It is worth mentioning that although the blockchain system is mainly introduced here for FL implementation, records on the blockchain are not limited to ML model parameters and updates but also including MCS task related information defined in Section IV-A.

Further, with the consortium members maintaining the distributed ledger, there are a lot of consensus protocols applicable to the consortium blockchain, such as PBFT and

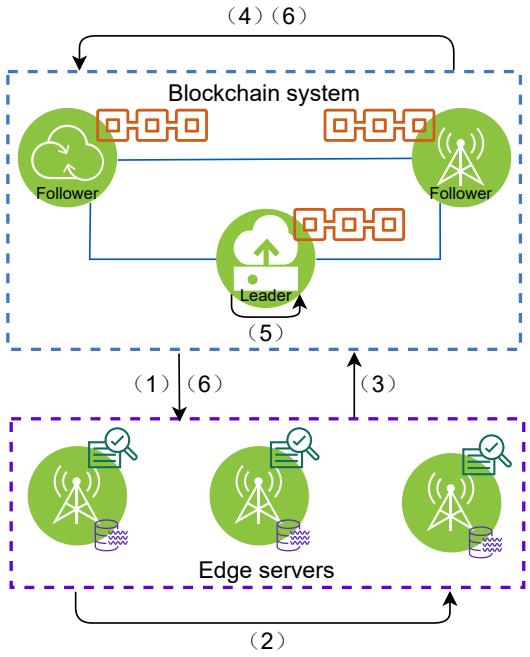


Fig. 2. The working process of blockchain-based FL, including six main steps: (1) initialization of ML model; (2) local training on edge servers; (3) submission of local model updates to the blockchain; (4) signature verification for the local model updates; (5) global model aggregation by the leader; (6) updating the aggregated model with edge servers.

HotStuff [49]. In our proposed system, we adopt PBFT as the consensus protocol in the consortium blockchain since it can improve the efficiency of reaching consensus and reduce time consumption. PBFT is one of the classical consensus protocols in blockchain, and its working process in our proposed blockchain can be described as below: at the beginning, the leader creates a block containing transactions, e.g., the local model updates, and then the leader broadcasts the block to other nodes in the blockchain, i.e., followers; next, all followers conduct the requested service (i.e., verifying the received block) and send back replies to the leader; in the end, if the leader receives $f + 1$ replies from followers with the same result, where f represents the maximum number of faulty nodes allowed, it means consensus has been reached in the blockchain system. For stability and sustainability of the blockchain system, we consider that all blockchain nodes will receive incentives based on the information recorded on the main chain. In particular, any node generating a valid block will be finally rewarded by the payment from the requestor. This can be achieved due to the following three-aspect reasons: 1) the identity of blockchain node who generates a valid block is clearly indicated on the block; 2) the information records included in the block body can be indexed by the requestor identity or a unique task number assigned when the request is recorded on the blockchain; and 3) the payment settlement with the requestor and the specific reward allocation will be guaranteed by a well-designed scheme which will be introduced in the next section.

In Algorithm 2, we present the process of blockchain-based FL at edge. To clearly illustrate the work flow of this process,

we present it in Fig. 2 and describe main steps as follows:

(1) **Initialization.** If the requestor has a clear sense or requirement on the ML model, the initial model can be sent to the blockchain system during the task publication process as introduced in Section IV-A, which will be included on the blockchain for reference (Lines 1-2). While if the requestor has no idea on initial model selection, the blockchain system, especially the leader packaging the MCS learning request, can decide one via comparing the received task with the historical tasks[2] (Lines 3-4).

(2) **Local learning.** Edge servers with local sensing datasets can obtain the ML model from the blockchain and then conduct the local training (Lines 6-8).

(3) **Updates submission.** Once edge servers accomplish the local learning process, they submit model updates signed with their private keys to the blockchain with the corresponding task index for the convenience of model aggregation (Line 9).

(4) **Signature verification.** After receiving local model updates from edge servers, blockchain nodes will verify the signatures to check whether the participants are legal or not (Line 11). If yes, these submissions will be broadcast to reach as many blockchain nodes as possible so as to be included in a valid block timely (Lines 12-13); otherwise, they will be discarded immediately (Lines 14-15).

(5) **Model aggregation.** Due to the function of the consensus protocol, there will be one blockchain node, i.e., leader, in charge of generating a valid block each time, who is generated by Leader Election process [50] (Line 18). In this case, we consider that the leader is also responsible to calculate the aggregated model according to all submitted local model updates in the current round of FL as received in the blockchain system, which can avoid redundant or conflicting calculation to save computing resources, and then generate a new block including the aggregated model (Lines 19-20).

(6) **Model updating.** If the global model is converged, the federated learning process can be terminated (Lines 21-22); otherwise, the above aggregated model recorded on the blockchain can be accessed by edge servers for updating their local models and begin the next round of FL (Lines 23-24).

The above process from (2) to (6) will be conducted repeatedly until the required model performance is realized, where the leader will explicitly associate the requestor identity with the finalized global model so as to be saved on the blockchain. Then the requestor can easily obtain the final result via the blockchain client.

Algorithm 2 Overall Process of the Proposed Blockchain-based FL at Edge

Require: Initial model $G(w)$, local model updates $f(w)$
Ensure: Final model $G^*(w)$
1: **if** Initial model $G(w)$ is specified by the requester **then**
2: $G(w)$ is recorded on the blockchain
3: **else**
4: $G(w)$ will be determined by comparing the received task with the historical tasks
5: **end if**
6: **while** Not reaching the required model performance **do**
7: Edge servers obtain $G(w)$ from the blockchain
8: Local model updates $f(w) \leftarrow$ Edge servers conduct local training to improve $G(w)$
9: Edge servers submit $f(w)$ with signature to the blockchain system
10: **while** Blockchain node **do**
11: Verify the signature of $f(w)$
12: **if** $f(w)$ is valid **then**
13: $f(w)$ will be broadcast to others
14: **else**
15: $f(w)$ will be discarded
16: **end if**
17: **end while**
18: Leader $i \leftarrow$ Leader Election
19: New global model $G'(w) \leftarrow$ Leader i conducts model aggregation
20: New block $\leftarrow$ Block generation
21: **if** $G'(w)$ is converged **then**
22: $G^*(w) \leftarrow G'(w)$
23: **else**
24: $G(w) \leftarrow G'(w)$
25: **end if**
26: **end while**
27: **return** $G^*(w)$

D. Payment Settlement and Allocation

With the joint effort of mobile devices, edge servers, and blockchain nodes, the whole MCS learning system can function to meet the demand of the requestor. However, this system cannot work in the long term unless the incentives for all participants are guaranteed and allocated appropriately. In this section, we design a scheme for payment settlement and allocation, dealing with how to enforce requestors to pay for the accomplished MCS learning tasks.

All the payment from the requestor is used to cover the incentives for data sensing of mobile devices, local learning of edge servers, FL model aggregation and block generation of blockchain nodes. In particular, the incentive for any mobile device has been immediately covered as $v + s$ by the edge server as introduced in Section IV-B; while the incentives for FL local training, model aggregation, and block generation can be designed as flat rates for fairness consideration, denoted by w_l, w_a, and w_b, respectively. Note that all the above incentives can be calculated by tracking the recorded information on the blockchain, and thus the total amount of incentive each requestor needs to pay can be easily calculated out by the leader who aggregates the final model, denoted by as W.

Here we consider that the requestor will use the MCS learn-

[2]Since the size of model parameters can be too large to fit in a block, we may employ the IPFS to store the model data, by which the records on the blockchain are referring to the address of the data.

ing service repeatedly via interacting with the blockchain system. To provide better user experience to requestors, our proposed system updates the MCS learning result on blockchain first and then conducts the payment settlement and allocation. This sequential interaction process makes room for malicious requestors rejecting to fully pay the total amount of incentive W. To alleviate this problem, we propose to take advantage of zero-determinant (ZD) game theory [7] to safeguard the interests of all participants in the MCS learning system. Since the leader finalizing the global model works as the last step determining whether the requestor can successfully obtain the final MCS learning result, we consider to deploy the payment enforcement scheme there with a smart contract implementing the detailed strategy. Note that the smart contract is stored on the blockchain as a record so that any blockchian node being the leader can run this program if the finalized global model satisfies the requested model performance.

With a_r and a_b respectively denoting the actions of the requestor and the leader finalizing the ML model in blockchain, we define the requestor not paying the full amount of incentive W as defection, denoted by $a_r = d$, and the action of paying W as cooperation with $a_r = c$; while as the key point serving the requestor, the leader can opportunistically choose to update the final learning results to the main chain or not, denoted as $a_b = c$ and d, respectively. With each having two actions, there exist four combinations of game states, i.e., $a_b a_r = \{cc, cd, dc, dd\}$. And their payoffs under all cases can be expressed as $\mathbf{x} = (x_1, x_2, x_3, x_4)$ for the leader and $\mathbf{y} = (y_1, y_2, y_3, y_4)$ for the requestor. Referring to the practical outcomes of these four cases, we can observe that 1) the defector in cd and dc states can gain the highest payoff while the cooperator receives the lowest payoff; 2) cc leads to the second highest payoffs for both since they cost and gain normally; and 3) dd brings the second lowest payoffs for both as nobody in this case loses anything. Thus, there exist the relationships $x_3 > x_1 > x_4 > x_2$ and $y_2 > y_1 > y_4 > y_3$.

For the game being repeated, we define the mixed strategies of the leader and the requestor as $\mathbf{p} = (p_1, p_2, p_3, p_4)$ and $\mathbf{q} = (q_1, q_2)$, respectively, where $p_i,\ i \in \{1, 2, 3, 4\}$ is the probability of $a_b = c$ given each game result in the last round, while $q_i,\ i \in \{1, 2\}$ is the probability of $a_r = c$ when $a_b = c$ or $a_b = d$ in this round. Note that the difference between the definitions of $\mathbf{p}$ and $\mathbf{q}$ lies in their action order, where the leader performs according to the action result in the last round while the requestor behaves based on the leader's action in the current round.

According to the extended version of ZD strategy we previously proposed in [7], we can see that it becomes beneficial for the leader to be the first mover if we empower the leader to use the ZD strategy. By this means, the leader can unilaterally control the relationship between the expected payoffs of both sides and thus enforce the desired game result. To be specific, when the strategy of the leader $\mathbf{p}$ satisfies $\tilde{\mathbf{p}} = (p_1 - 1, p_2 - 1, p_3, p_4) = \alpha\mathbf{x} + \beta\mathbf{y} + \gamma\mathbf{1}$ with $\mathbf{1}$ denoting a vector of four ones, a linear relationship between their expected payoffs can be enforced as

$$\alpha E_b + \beta E_r + \gamma = 0,$$

where E_b and E_r are respectively the expected payoffs of the leader and the requestor in the long term.

In our case, as the leader desires to enforce the long-term cooperation from the requestor, mutual cooperation becomes a feasible and stable solution, where their expected payoffs are x_1 and y_1. Thus, inspired by the *cooperation-enforcing control* in [51], the leader can set a specific linear relationship as $E_b - x_1 = \chi(E_r - y_1)$ with $\chi \geq 1$ to drive the full cooperation of the requestor, which yields the strategy of the leader $\mathbf{p}$ satisfying $\tilde{\mathbf{p}} = (p_1 - 1, p_2 - 1, p_3, p_4) = \gamma((\mathbf{x} - x_1\mathbf{1}) - \chi(\mathbf{y} - y_1\mathbf{1}))$ ($\gamma \neq 0$). The effectiveness of this scheme can be demonstrated in the following theorem.

Theorem IV.2. *When the leader sets the strategy* $\mathbf{p}$ *to meet* $\tilde{\mathbf{p}} = \gamma((\mathbf{x} - x_1\mathbf{1}) - \chi(\mathbf{y} - y_1\mathbf{1}))$ *where* $\gamma \neq 0$*, the stable state is mutual cooperation.*

Proof. As a utility-driven player, the requestor usually hopes to maximize the payoff. With the leader setting the strategy $\mathbf{p}$ to meet $\tilde{\mathbf{p}} = \gamma((\mathbf{x} - x_1\mathbf{1}) - \chi(\mathbf{y} - y_1\mathbf{1}))$, there exists $E_b - x_1 = \chi(E_r - y_1)$. In this case, if the requestor gains E_r higher than y_1, leading to $E_b - x_1$ with a value of χ times $E_r - y_1$, both can gain payoffs higher than those at the mutual-cooperation state, i.e., y_1 and x_1, at the same time. This cannot be true since the requestor can obtain the payoff larger than y_1 only if the leader gets the payoff less than x_1, while the leader with the payoff larger than x_1 leads to the requestor obtaining the payoff less than y_1. Thus, the highest expected payoff that the requestor can obtain is $E_r = y_1$, which leads to $E_b = x_1$, corresponding to the mutual-cooperation state in the long term. □

With the function of the above ZD-based payment enforcement scheme, the total amount of incentives W will be sent to the MCS learning system and recorded on the blockchain. All blockchain nodes and edge servers can fetch their respective rewards in a transparent and honest way according to the task accomplishing records on the blockchain, which will also be included on the main chain for potential error tracking.

V. Experimental Evaluation

In this section, we conduct extensive experiments to evaluate our proposed MCS learning framework via investigating the performance of specific schemes designed for main procedures. Since the first procedure in Section IV-A mainly introduces the input of the MCS learning system without algorithm design, we only present the experimental results of the procedures from Section IV-B to Section IV-D.

A. Data Submission Mechanism Design

As proved in Theorem IV.1, given the mechanism design for data submission process where the mobile device releases the game rule $v^*(s)$, the edge server can obtain the maximized utility only when the real data leakage degree θ^* is indicated to form the best response strategy s^*. Since the private θ^* of the server is not available for the device to observe, we cannot evaluate its impact. Instead, we study the impacts of three observable parameters, i.e., η, ξ and ϵ, on the maximized utilities of both sides. In detail, we set $\eta, \xi \in [0, 15]$, $\epsilon \in [0, 1]$, and

$R = 10, \alpha_s = 3, \beta_s = 2, \bar{v} = 50, \bar{s} = 500, c_d = 2, \theta = 0.5$. Note that other parameter settings are also evaluated, which present similar trends and thus are omitted here.

We first investigate the impacts of the device's cost parameters η and ξ when ϵ is set as 0.9. As shown in Fig. 3, with the increase of η and ξ, the maximized utility of the device will decrease sharply to a stable value while that of the server presents different trends. With the increasing η and ξ, the server's maximized utility increases from zero to the largest point firstly and then gradually decreases to a stable value. This is because the increase of cost parameters directly affect the utility of the device, where the higher the cost, the lower the utility; while for the server, the maximized utility U_s is impacted indirectly through the optimal strategies s^* and $v^*(s)$.

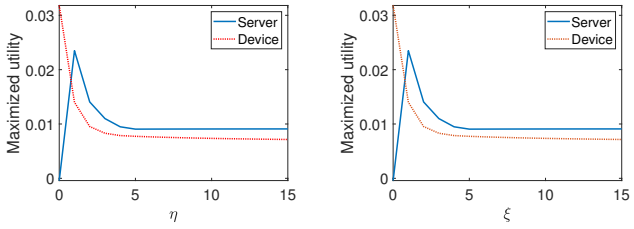


Fig. 3. Impacts of η and ξ on the maximized utilities.

Then we evaluate the impacts of ϵ which is the main parameter determining the successful data collection probability. As presented in Fig. 4, the impact of ϵ on the server differs from that on the device. For the server, the maximized utility increases first and then decreases to almost zero; while the device's maximized utility decreases all the way to zero. According to (2), the increase of ϵ implies that the optimal strategy of the device v^* matters more for the successful probability. While referring to the expression of v^* in (4), one can see that the larger the ϵ, the lower the v^*, which leads to the decrease of U_s according to (3). But for the server, the increasing ϵ makes the decreasing v^* result in a larger U_d according to (1) at the initial stage; when v^* decreases to a certain degree, the decreased successful probability g gradually functions to decrease U_d.

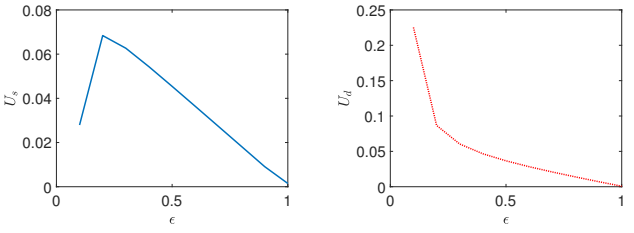


Fig. 4. Impacts of ϵ on the maximized utilities.

B. Blockchain-based Federated Learning at Edge

Considering that the consortium blockchain system for model parameter coordination has limited impact on the FL performance, we conduct the simulation of FL first and focus on investigating the influences of available data size and edge servers' participation on learning results during the FL process. To simulate the FL process conducted among edge servers, we utilize a benchmark named LEAF [52] to execute federated learning using a 2-layer convolutional neural network (CNN) classifier for the FEMNIST dataset with 805,263 samples in total and 3,550 available participants for local computing. In our experiments, we first change the total data size used for FL indicated by the varying dataset fraction as 0.001, 0.005, and 0.05, with the fixed ratio between training and test dataset sizes as 9:1 and 35 FL participants; and then the number of participates is changed when the total amount of data used for FL keeps constant, where the participant fraction is set as 0.01, 0.03, and 0.05 to approximately simulate the number of participated edge servers as 35, 105, and 175, respectively.

The learning results under two parameter settings are reported in Fig. 5. It can be seen from the left figure that given the same number of FL participants, the higher the dataset fraction used for FL, the better the learning performance with respect to the testing accuracy. With too few data samples (blue), the global model cannot even converge after 2,000 rounds of FL. While from the right figure, one can see with the same amount of data used for FL, the more the participants, the slower to reach convergence. This is reasonable since more participants splitting the total dataset will result in fewer data samples available for each one, leading to the longer training time. Corresponding to the MCS learning scenario, given the fixed amount of mobile devices to collect sensing data, the more edge servers involved, the smaller the local dataset of each edge server, and thus the slower the learning process.

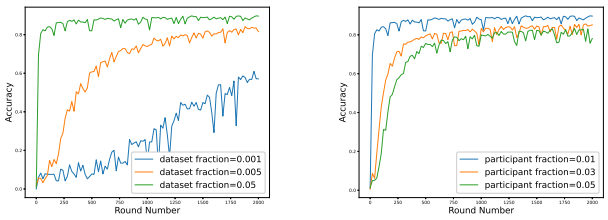


Fig. 5. Impacts of data size and the number of edge servers on the learning accuracy.

After the edge servers submit the local model updates to the blockchain, blockchain nodes generate a new block to include all updates and the aggregated model, which has to be supported by the predefined consensus protocol. To explore how the blockchain works in our framework, we design experiments to simulate the PBFT consensus process based on the framework in [53] as an example. For convenience, we assume that each edge server submit the local model updates with a fixed data size of 300 KB. We implement this set of simulations using Python 3.8.5 in macOS 11.0.1 running on Intel i7 processor with 32 GB RAM and 1 TB SSD.

First, we explore the relationship between the consensus efficiency in the blockchain system and the number of edge servers as FL clients given 10 nodes in the consortium blockchain. From Fig. 6, we can see that the time cost to reach consensus is linearly correlated to the number of edge

servers. This is because increasing the number of edge servers will lead a larger amount of data that need to be packed into blocks, resulting in more time to synchronizing the data on blockchain.

Then we examine how the number of blockchain nodes affects the consensus efficiency when the number of FL participants (i.e., edge servers) is 10. The experimental results are shown in Fig. 6. It is clear that the time consumption increases as the number of blockchain nodes grows. The underline reason is that in the blockchain system running on a peer-to-peer networking structure, more nodes will lead to a larger increase in the communication times among them, and thus making the time consumed for reaching consensus increase near-exponentially.

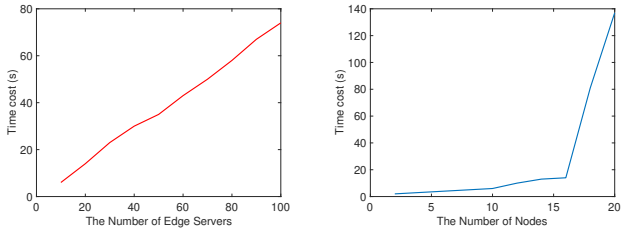


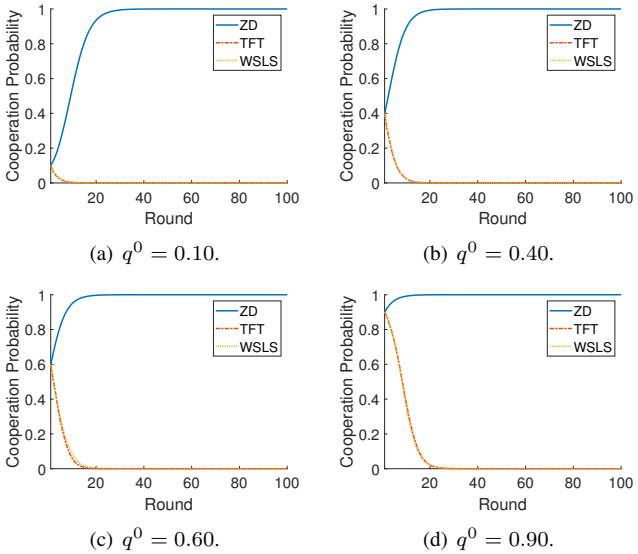
Fig. 6. Impacts of the numbers of edge servers and blockchain nodes on the efficiency of the blockchain consensus.

C. ZD-based Payment Settlement

To figure out whether the proposed ZD-based payment settlement strategy can function effectively, we compare it with other two classical strategies, named tit-for-tat (TFT) and win-stay-lose-shift (WSLS). With the TFT strategy, the leader will choose the action that the requestor adopted in the last round; with the WSLS strategy, the leader keeps performing an action if it brings a high payoff while changes to the other one if it results in a low payoff in the last round. Besides, we set the initial cooperation probability of the requestor as different values to further indicate the robustness of the proposed scheme. Specifically, we denote the requestor's initial cooperation probability as q^0 and study the evolution of the requestor's cooperation probability with respect to q_1 or q_2 according to the action of the leader in the current round. Main parameters in this scheme are payoff vectors of two player, which are set as $\mathbf{x} = (6, 3, 8, 5)$ and $\mathbf{y} = (6, 8, 3, 5)$ in the simulations.

From Fig. 7, one can see that the proposed ZD strategy can enable the cooperation probability of the requestor to gradually approach 1 and stay cooperative. While the other two classical strategies fail to achieve this job. Further, as presented in four subfigures, no matter how the initial cooperation probability of the requestor changes, the ZD-based scheme can always function successfully with respect to enforcing its full cooperation behavior, i.e., paying the full amount of incentive to the blockchain system. Thus, the effectiveness and robustness of our proposed scheme are validated.

Next, we explore the utilities of both the leader and requestor under the function of the proposed ZD-based scheme. As shown in Fig. 8, we present the utilities at the stable state in



(a) $q^0 = 0.10$. (b) $q^0 = 0.40$.

(c) $q^0 = 0.60$. (d) $q^0 = 0.90$.

Fig. 7. Cooperation probability of the requestor given different strategies of the leader.

the left bar graph and the evolution of utilities in the right-side line graphs. Similar to the previous experiment, we examine the results with different initial cooperation probabilities of the requestor. As can been seen from the left subfigure of Fig. 8, at the final state with stable strategies of the requestor and the leader, both can obtain similar utility around 6 no matter how cooperative the requestor is at the beginning, which is exactly the utility at the mutual cooperation state. This implies that the proposed scheme is fair with respect to the final utility. From four line figures, one can observe that with different initial cooperation probabilities of the requestor, the dynamic evolution paths of both the leader and the requestor are generally similar with slight difference, where the higher the initial cooperation probability, the faster both sides to achieve the maximized utilities at the stable state. This is consistent with the fact that the more cooperative the requestor, the easier to drive its full cooperation at the stable state.

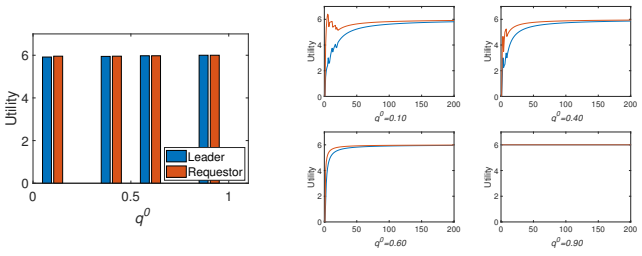


Fig. 8. Utilities of the leader and the requestor at the stable and dynamic states with the leader adopting the ZD strategy.

VI. Conclusion and Future Work

Although MCS has been applied to various fields and brings significant benefits to the whole society, the current MCS paradigm has high requirements on the communication, storage and computing capabilities of requestors, limiting the widespread application of MCS to go a step further. To

overcome this challenge, we propose a new MCS learning framework based on the consortium blockchain and the edge-participated FL, functioning among four major entities, i.e., requestors, blockchain, edge servers and mobile devices. Despite existing studies on blockchain for MCS and blockchain-based FL, they fail to lower the requirements on requestors' capabilities or cannot be directly applied to MCS. Thus, our proposed MCS learning framework fills the gaps with four main procedures, i.e., task publication, data sensing and submission, learning to return final results, and payment settlement and allocation. We design specific schemes in main steps to address challenges resulted from malicious edge servers, dishonest requestors, and even outside attacks. Experimental results demonstrate the effectiveness of our designed schemes.

In our future work, we will make efforts to extensively study the employed blockchain in our proposed MCS learning framework to further improve the efficiency, scalability and reliability; besides, we will also investigate security challenges in blockchain-based distributed learning during the model training process, which may benefit not only our proposed MCS learning system but also other applications involving extensive data collection and analysis.

REFERENCES

[1] D. C. Nguyen, M. Ding, Q.-V. Pham, P. N. Pathirana, L. B. Le, A. Seneviratne, J. Li, D. Niyato, and H. V. Poor, "Federated learning meets blockchain in edge computing: Opportunities and challenges," *IEEE Internet of Things Journal*, 2021.

[2] K. Farkas, G. Feher, A. Benczur, and C. Sidlo, "Crowdsending based public transport information service in smart cities," *IEEE Communications Magazine*, vol. 53, no. 8, pp. 158–165, 2015.

[3] Q. Wang, B. Guo, Y. Liu, Q. Han, T. Xin, and Z. Yu, "Crowdnavi: Last-mile outdoor navigation for pedestrians using mobile crowdsensing," *Proceedings of the ACM on Human-Computer Interaction*, vol. 2, no. CSCW, pp. 1–23, 2018.

[4] Q. Hu, S. Wang, X. Cheng, L. Ma, and R. Bie, "Solving the crowdsourcing dilemma using the zero-determinant strategies," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1778–1789, 2019.

[5] E. Wang, Y. Yang, J. Wu, W. Liu, and X. Wang, "An efficient prediction-based user recruitment for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 17, no. 1, pp. 16–28, 2017.

[6] Z. Wang, J. Zhao, J. Hu, T. Zhu, Q. Wang, J. Ren, and C. Li, "Towards personalized task-oriented worker recruitment in mobile crowdsensing," *IEEE Transactions on Mobile Computing*, 2020.

[7] Q. Hu, S. Wang, P. Ma, X. Cheng, W. Lv, and R. Bie, "Quality control in crowdsourcing using sequential zero-determinant strategies," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 5, pp. 998–1009, 2019.

[8] B. Zhao, S. Tang, X. Liu, and X. Zhang, "Pace: privacy-preserving and quality-aware incentive mechanism for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, 2020.

[9] X. Wei, Y. Yan, W. Jiang, J. Shen, and X. Qiu, "A blockchain based mobile crowdsensing market," *China Communications*, vol. 16, no. 6, pp. 31–41, 2019.

[10] J. Huang, L. Kong, H.-N. Dai, W. Ding, L. Cheng, G. Chen, X. Jin, and P. Zeng, "Blockchain based mobile crowd sensing in industrial systems," *IEEE Transactions on Industrial Informatics*, 2020.

[11] J. Hu, K. Yang, K. Wang, and K. Zhang, "A blockchain-based reward mechanism for mobile crowdsensing," *IEEE Transactions on Computational Social Systems*, vol. 7, no. 1, pp. 178–191, 2020.

[12] L. Wei, J. Wu, and C. Long, "A blockchain-based hybrid incentive model for crowdsensing," *Electronics*, vol. 9, no. 2, p. 215, 2020.

[13] M. Kadadha, H. Otrok, R. Mizouni, S. Singh, and A. Ouali, "Sensechain: A blockchain-based crowdsensing framework for multiple requesters and multiple workers," *Future Generation Computer Systems*, vol. 105, pp. 650–664, 2020.

[14] Y. Liang, Y. Li, and B.-S. Shin, "Faircs—blockchain-based fair crowdsensing scheme using trusted execution environment," *Sensors*, vol. 20, no. 11, p. 3172, 2020.

[15] C. Cai, Y. Zheng, Y. Du, Z. Qin, and C. Wang, "Towards private, robust, and verifiable crowdsensing systems via public blockchains," *IEEE Transactions on Dependable and Secure Computing*, 2019.

[16] J. An, D. Liang, X. Gui, H. Yang, R. Gui, and X. He, "Crowdsensing quality control and grading evaluation based on a two-consensus blockchain," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4711–4718, 2018.

[17] J. An, J. Cheng, X. Gui, W. Zhang, D. Liang, R. Gui, L. Jiang, and D. Liao, "A lightweight blockchain-based model for data quality assessment in crowdsensing," *IEEE Transactions on Computational Social Systems*, vol. 7, no. 1, pp. 84–97, 2020.

[18] D. Chatzopoulos, S. Gujar, B. Faltings, and P. Hui, "Privacy preserving and cost optimal mobile crowdsensing using smart contracts on blockchain," in *2018 IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. IEEE, 2018, pp. 442–450.

[19] J. Wang, M. Li, Y. He, H. Li, K. Xiao, and C. Wang, "A blockchain based privacy-preserving incentive mechanism in crowdsensing applications," *IEEE Access*, vol. 6, pp. 17 545–17 556, 2018.

[20] S. Gao, X. Chen, J. Zhu, X. Dong, and J. Ma, "Trustworker: A trustworthy and privacy-preserving worker selection scheme for blockchain-based crowdsensing," *IEEE Transactions on Services Computing*, 2021.

[21] Z. Xu, C. Liu, P. Zhang, T. Lu, and N. Gu, "Urim: Utility-oriented role-centric incentive mechanism design for blockchain-based crowdsensing," in *International Conference on Database Systems for Advanced Applications*. Springer, 2021, pp. 358–374.

[22] S. Zou, J. Xi, H. Wang, and G. Xu, "Crowdblps: A blockchain-based location-privacy-preserving mobile crowdsensing system," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4206–4218, 2019.

[23] M. Yang, T. Zhu, K. Liang, W. Zhou, and R. H. Deng, "A blockchain-based location privacy-preserving crowdsensing system," *Future Generation Computer Systems*, vol. 94, pp. 408–418, 2019.

[24] J. Zhang, X. Huang, W. Ni, M. Wu, and R. Yu, "Vesenchain: Leveraging consortium blockchain for secure and efficient vehicular crowdsensing," in *2019 Chinese Control Conference (CCC)*. IEEE, 2019, pp. 6339–6344.

[25] J. Wang, X. Feng, T. Xu, H. Ning, and T. Qiu, "Blockchain based model for nondeterministic crowdsensing strategy with vehicular team-cooperation," *IEEE Internet of Things Journal*, 2020.

[26] S. Feng, W. Wang, D. Niyato, D. I. Kim, and P. Wang, "Dynamic sensor renting in rf-powered crowdsensing service market with blockchain," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2019, pp. 1–7.

[27] Z. Noshad, A. Javaid, M. Zahid, I. Ali, N. Javaid *et al.*, "A blockchain based incentive mechanism for crowd sensing network," in *International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*. Springer, 2019, pp. 568–578.

[28] X. Gu, J. Peng, W. Yu, Y. Cheng, F. Jiang, X. Zhang, Z. Huang, and L. Cai, "Using blockchain to enhance the security of fog-assisted crowdsensing systems," in *2019 IEEE 28th International Symposium on Industrial Electronics (ISIE)*. IEEE, 2019, pp. 1859–1864.

[29] B. Jia, T. Zhou, W. Li, Z. Liu, and J. Zhang, "A blockchain-based location privacy protection incentive mechanism in crowd sensing networks," *Sensors*, vol. 18, no. 11, p. 3894, 2018.

[30] K. Zhao, S. Tang, B. Zhao, and Y. Wu, "Dynamic and privacy-preserving reputation management for blockchain-based mobile crowdsensing," *IEEE Access*, vol. 7, pp. 74 694–74 710, 2019.

[31] X. Bao, C. Su, Y. Xiong, W. Huang, and Y. Hu, "Flchain: A blockchain for auditable federated learning with trust and incentive," in *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*. IEEE, 2019, pp. 151–159.

[32] U. Majeed and C. S. Hong, "Flchain: Federated learning via mec-enabled blockchain network," in *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, 2019, pp. 1–4.

[33] R. Kumar, A. A. Khan, J. Kumar, A. Zakria, N. A. Golilarz, S. Zhang, Y. Ting, C. Zheng, and W. Wang, "Blockchain-federated-learning and deep learning models for covid-19 detection using ct imaging," *IEEE Sensors Journal*, 2021.

[34] Y. Qi, M. S. Hossain, J. Nie, and X. Li, "Privacy-preserving blockchain-based federated learning for traffic flow prediction," *Future Generation Computer Systems*, vol. 117, pp. 328–337, 2021.

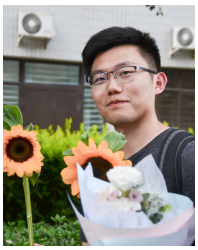
[35] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1279–1283, 2019.

[36] S. R. Pokhrel and J. Choi, "A decentralized federated learning approach for connected autonomous vehicles," in *2020 IEEE Wireless Commu-*

nications and Networking Conference Workshops (WCNCW). IEEE, 2020, pp. 1–6.
[37] ——, "Federated learning with blockchain for autonomous vehicles: Analysis and design challenges," *IEEE Transactions on Communications*, 2020.
[38] K. Toyoda and A. N. Zhang, "Mechanism design for an incentive-aware blockchain-enabled federated learning platform," in *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 395–403.
[39] J. Weng, J. Weng, J. Zhang, M. Li, Y. Zhang, and W. Luo, "Deepchain: Auditable and privacy-preserving deep learning with blockchain-based incentive," *IEEE Transactions on Dependable and Secure Computing*, 2019.
[40] Y. Lu, X. Huang, Y. Dai, S. Maharjan, and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4177–4186, 2019.
[41] Y. J. Kim and C. S. Hong, "Blockchain-based node-aware dynamic weighting methods for improving federated learning performance," in *2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS)*. IEEE, 2019, pp. 1–4.
[42] D. Preuveneers, V. Rimmer, I. Tsingenopoulos, J. Spooren, W. Joosen, and E. Ilie-Zudor, "Chained anomaly detection models for federated learning: An intrusion detection case study," *Applied Sciences*, vol. 8, no. 12, p. 2663, 2018.
[43] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in internet of vehicles," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4298–4311, 2020.
[44] 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 of Things Journal*, vol. 6, no. 6, pp. 10 700–10 714, 2019.
[45] S. Awan, F. Li, B. Luo, and M. Liu, "Poster: A reliable and accountable privacy-preserving federated learning framework using the blockchain," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 2561–2563.
[46] H. B. Desai, M. S. Ozdayi, and M. Kantarcioglu, "Blockfla: Accountable federated learning via hybrid blockchain architecture," in *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy*, 2021, pp. 101–112.
[47] J. Benet, "Ipfs-content addressed, versioned, p2p file system," *arXiv preprint arXiv:1407.3561*, 2014.
[48] L. Hurwicz and S. Reiter, *Designing economic mechanisms*. Cambridge University Press, 2006.
[49] M. Yin, D. Malkhi, M. K. Reiter, G. G. Gueta, and I. Abraham, "Hotstuff: Bft consensus with linearity and responsiveness," in *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, 2019, pp. 347–356.
[50] M. Castro, B. Liskov *et al.*, "Practical byzantine fault tolerance," in *OSDI*, vol. 99, no. 1999, 1999, pp. 173–186.
[51] D. Hao, K. Li, and T. Zhou, "Payoff control in the iterated prisoner's dilemma," in *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018, pp. 296–302.
[52] S. Caldas, 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.
[53] CyHsiung, "Pratical byzatine fault tolerance," https://github.com/CyHsiung/Practical-Byzantine-Fault-Tolerance-PBFT-.

Qin Hu received her Ph.D. degree in Computer Science from the George Washington University in 2019. She is currently an Assistant Professor with the Department of Computer and Information Science, Indiana University-Purdue University Indianapolis (IUPUI). She has served on the Editorial Board of two journals, the Guest Editor for two journals, the TPC/Publicity Co-chair for several workshops, and the TPC Member for several international conferences. Her research interests include wireless and mobile security, edge computing, blockchain, and crowdsensing.

Zhilin Wang received his B.S. from Nanchang University in 2020. He is currently pursuing his Ph.D. degree of Computer and Information Science In Indiana University-Purdue University Indianapolis (IUPUI). His research interests include blockchain, federated learning, and Internet of Things (IoT).



Minghui Xu received the BS degree in Physics from the Beijing Normal University, Beijing, China, in 2018, and the PhD degree in Computer Science from The George Washington University, Washington DC, USA, in 2021. He is currently an Assistant Professor in the School of Computer Science and Technology, Shandong University, China. His current research focuses on blockchain, distributed computing, and quantum computing.

Xiuzhen Cheng received her M.S. and Ph.D. degrees in computer science from the University of Minnesota Twin Cities in 2000 and 2002. She is currently a professor in the School of Computer Science and Technology, Shandong University, China. Her current research interests focus on privacy-aware computing, wireless and mobile security, dynamic spectrum access, mobile handset networking systems (mobile health and safety), cognitive radio networks, and algorithm design and analysis. She has served on the Editorial Boards of several technical publications and the Technical Program Committees of various professional conferences/workshops. She has also chaired several international conferences. She worked as a program director for the U.S. National Science Foundation (NSF) from April to October 2006 (full time), and from April 2008 to May 2010 (part time). She published more than 300 peer-reviewed papers.