IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 18, 2020, accepted December 3, 2020, date of publication December 7, 2020,
date of current version December 17, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3043037

Blockchain-Enabled Federated Learning With Mechanism Design

KENTAROH TOYODA[1,2], (Member, IEEE), JUN ZHAO[3], (Member, IEEE), ALLAN NENG SHENG ZHANG[1], (Associate Member, IEEE), AND P. TAKIS MATHIOPOULOS[4], (Senior Member, IEEE)

[1]Singapore Institute of Manufacturing Technology, A*STAR, Singapore 138634
[2]Faculty of Science and Technology, Keio University, Tokyo 108-8345, Japan
[3]School of Computer Science and Engineering, Nanyang Technological University, Singapore 639669
[4]Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, 157 84 Athens, Greece

Corresponding author: Kentaroh Toyoda (kentaroh.toyoda@ieee.org)

This work was supported in part by A*STAR's through the Project P20-O1-068USV Blockchain-based Federated Learning Platform with Mechanism Design under Grant SC26/19-331190, and in part by KAKENHI from MEXT/JSPS, Japan, under Grant 18K18162.

ABSTRACT Federated learning (FL) is a promising decentralized deep learning technique that allows users to collaboratively update models without sharing their own data. However, due to its decentralized nature, no one can monitor workers' behavior, and they may thus deviate protocols (e.g., participating without updating any models). To solve this problem, many researchers have proposed blockchain-enabled FL to reward workers (or users) with cryptocurrencies to encourage workers to follow the protocols. However, there is a lack of theoretical discussions concerning how such rewards impact workers' behavior and how much should be given to workers. In this article, we propose a mechanism-design-oriented FL protocol on a public blockchain network. Mechanism design (MD) is often used to make a rule intended to achieve a specific goal. With MD in mind, we introduce the concept of competition into blockchain-based FL so that only workers who have contributed well can obtain rewards, which naturally prevents workers from deviating from the protocol. We then mathematically answer the following questions with contest theory, a novel field of study in economics: i) What behavior will workers take?; ii) how much effort should workers exert to maximize their profits?; iii) how many workers should be rewarded?; and iv) what is the best proportion for reward distribution?

INDEX TERMS Federated learning, decentralized deep learning, blockchain, mechanism design, contest theory.

I. INTRODUCTION

Artificial intelligence (AI) has been widely used to make decisions about our daily lives. However, the accuracy of AI is still not satisfactory in many sectors, such as autonomous vehicle systems. A straightforward approach to improving accuracy is to collect more training data from the wide range of users. That said, most people are not willing to share personal data such as images, audio, and location information. To overcome this problem, Google has proposed federated learning (FL), an iterative and decentralized approach that allows those who possess data to update a deep learning (DL) model without disclosing their data [1]. More specifically, users (or workers)[1] locally update models using their own data and submit it to a central server, in which models are merged together. This approach prevents users from disclosing their data, as only updated models are submitted.

Although the concept of FL is novel, many remaining issues prevent it from being practical. For example, users could be ''volunteers'' who work to update the model. In this situation, users can be free riders, meaning that they can obtain improved models without exerting any effort. To solve this problem, giving workers a monetary incentive is an effective approach.[2] Recently, researchers have proposed the

The associate editor coordinating the review of this manuscript and approving it for publication was Mingjun Dai.

[1]The terms of users and workers are interchangeably used in the following.
[2]Some similar applications, for example, Waze (www.waze.com), can provide an incentive for workers without offering any monetary incentive.

concept of blockchain-enabled FL to incentivize workers to contribute using cryptocurrencies (e.g., [2], [3]).

However, no existing work has theoretically clarified how rewarding impacts workers' behavior and what the optimal conditions are. Previously, we introduced a concept of competition to the model update process that allows only workers who have contributed to earn rewards [4]. However, the previous work was somewhat incomplete, and the following problems remain unsolved and questions unanswered:

1) The protocol is not fully described, and elaboration is needed.
2) How much effort should workers exert to maximize their profits?
3) How many workers should be rewarded?
4) What amounts should workers receive?

In this article, we first present a full-fledged mechanism-design-oriented FL protocol on a public blockchain network, which is inspired by mechanism design (MD), an economic rule-making approach to achieve a specific goal. We then present a theoretical analysis to address the remaining problems by leveraging *contest theory*. A contest is an auction where participants must expend costly and irreversible efforts, and prizes are determined based on the quality of their output [5]. Workers must update a model, which is an irreversible effort, and thus the model update process of FL can be seen as a contest.

The contributions of this article compared to our previous article [4] are as follows:

1) We provide a more complete survey of related work.
2) We provide a full-fledged protocol design.
3) We provide a complete theoretical analysis of the optimal reward policy based on the contest theory. In particular, we provide answers to the following questions:
   a) What is the condition for a worker to join a task?
   b) How much effort should workers exert to maximize their profits?
   c) How many workers should be rewarded?
   d) How much rewards should be given to the workers?

The rest of this article is organized as follows: Section II explains the fundamentals of FL, cryptocurrency, and blockchain. Section III summarizes related work. Section IV describes the proposed protocol, while Section V presents a theoretical analysis. Finally, Section VI concludes this article.

II. PRELIMINARIES

A. FEDERATED LEARNING

Federated learning enables users to collaboratively train a deep neural network (DNN) model while keeping all the training data on the user side [1]. FIGURE 1 depicts the basic flow chart of an FL system with a total of K workers. Let us assume that a central server delegates a DNN task, for example, teaching a DNN model that can identify objects in given images, to distributed workers. The central server first distributes an initialized model to $K' = \max(\lceil K \cdot C \rceil, 1)$ workers, where C denote the fraction of workers in a round.

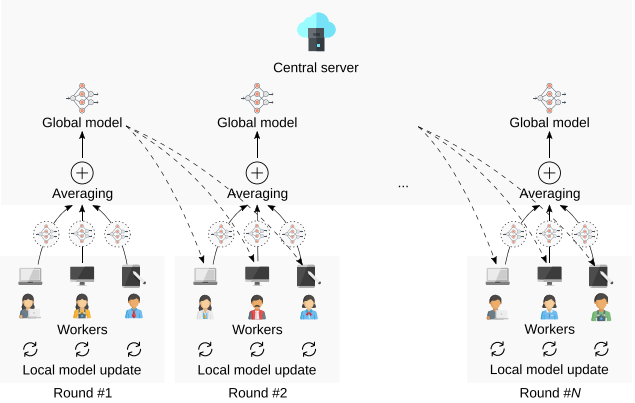


FIGURE 1. FL's fundamental procedures.

Each worker updates the received model with his or her own data and sends an updated model, that is, a set of weights and biases, to the central server. The central server distributes a new model, which can be obtained by averaging the models updated by the workers in the first round to those in the second round. These procedures are repeated until the model is converged or at specific round, say N rounds. As workers only submit the model, their original data is not shared with the central server, and their privacy thus can be somewhat preserved.

Although the concept of FL is promising, many issues, such as the following, must be solved:

1) Some workers may intentionally (or unintentionally) submit negatively affecting models.
2) Data used for training a model may be leaked from a set of weights and biases [6].
3) Workers may be less motivated to contribute to the model update process, as they do not obtain explicit rewards.

To solve these issues, some researchers, for example, [2], [3], [7]–[9] have proposed leveraging blockchain technologies to give monetary incentives to workers in the form of cryptocurrencies and to track the updating of a model in a trustworthy manner. To better understand their work, the fundamentals of blockchain technologies are explained in the next section.

B. BLOCKCHAIN TECHNOLOGIES

A cryptocurrency is a digital currency, some of which (e.g., Bitcoin and Ethereum) can be exchanged from/to fiat currencies. In general, an *address* is used to receive and send cryptocurrencies, and it is transferable among addresses via a *transaction*. Transactions are stored in a blockchain, which is a distributed append-only ledger. Blockchain is classified according to two types: public and private (or permissioned) blockchains. A public blockchain is a blockchain on which anyone can check the stored transactions. By contrast, a private blockchain is run by predefined users, and its content is only accessible by them. Hyperledger Fabric is an example of a private blockchain.

TABLE 1. Comparison of related work and our study. '–' and "NA" denote "not mentioned" and "not applicable," respectively.

REF.	TYPE OF BLOCKCHAIN	USED BLOCKCHAIN	INCENTIVES	THEORETICAL ANALYSIS
[3]	Public	Original	✓	
[11]	Private	Original	-	NA
[12]	Private	Multichain	-	NA
[13]	Public	Not specified	✓	✓
[14]	Public	Ethereum	-	NA
[15]	-	Original	-	NA
[2]	Private	Corda V3.0	-	NA
Our method	Public	Any blockchains with smart contract, such as Ethereum	✓	✓

```
struct User {
  bool isRequester;
  bool isWorker;
}

mapping (address => User) public users;

function regUser(
  address _addr,
  bool _isRequester,
  bool, _isWorker) public onlyAdmin {
    users[_addr].isRequester = _isRequester;
    users[_addr].isWorker = _isWorker;
}

modifier onlyAdmin {
  require(msg.sender == admin); _;
}
```

FIGURE 2. An example of `Ethereum`'s smart contract.

A smart contract is another important invention that enables us to execute a computer program on the blockchain [10]. The code of a smart contract itself is stored in the blockchain and can be executed by sending a transaction to the address of a smart contract with program arguments. There are two advantages to executing programs with a smart contract: The first is that cryptocurrencies can be transferred via computer programs. The second is transparency: the status of programs is stored in the blockchain, and the history of code execution can be tracked. These are the fundamentals that allow FL workers to submit their DNN model updates and to receive cryptocurrency via smart contracts. The procedures involved in an FL, namely, user registration, task creation, model updating and averaging, can be fully described with smart contracts and executed via transactions. FIGURE 2 shows a smart contract for user registration. Through the function `regUser()`, a user can be registered as a requester and/or worker. A functionality of access control can be implemented, for example, `onlyAdmin` in FIGURE 2, restricting the execution of a function to a specific user.

III. RELATED WORK

Our work is relevant to blockchain-based (i) decentralized machine learning (ML) and (ii) crowdsourcing. However, we only summarize the former, and the latter is presented in Appendix, as space is limited here.

TABLE 1 lists the state-of-the-art distributed ML work using a blockchain technology. Shae and Tsai proposed a concept that leverages the smart contract of blockchain for large medical data processing [7]. However, they only show the concept and do not discuss any details.

Lu *et al.* proposed an approach to crowdsource ML tasks with a public blockchain [13]. In their proposal, a commitment scheme is introduced to avoid the situation in which a malicious worker simply copies and submits other workers' outputs. In addition, a strategic game is leveraged to incentivize workers to behave appropriately.

Preuveneers *et al.* proposed a permissioned blockchain-based FL platform that enables the auditing of trained ML models [12]. Their platform is implemented on Multichain. The authors demonstrated its practicability with a network trace dataset for network anomaly detection.

Idé proposed a permissioned blockchain-enabled collaborative platform for anomaly detection [16]. The author gave an example of factories located in different places being able to collaborate with each other to build a common model with local sensor data. A permissioned blockchain is used to accumulate and update models from decentralized factories.

Chen *et al.* proposed a privacy-preserving decentralized ML platform, `LearningChain` [14]. Although the user's privacy is somewhat preserved in FL, the authors noted that users' inputs could be revealed by analyzing the weights of neural networks [6]. To address this problem, the authors applied differential privacy, allowing workers to update their own data with perturbation to preserve privacy.

Kim *et al.* analyzed the end-to-end latency of blockchain-enabled FL [15]. When mobile devices participate in the model training processes, network latency can heavily impact the performance of FL [17]. In [15], several metrics, such as latency, were evaluated on the authors' own blockchain.

Weng *et al.* proposed a blockchain-enabled privacy-preserving ML platform, `DeepChain` [2]. To prevent the leakage of user information, the additive homomorphic encryption of threshold Pailler encryption was introduced. An original coin, `DeepCoin`, is distributed to users as incentive, but misbehaviors, such as providing stale reports and incorrect execution, are punished.

Syayan *et al.* proposed a public blockchain-based privacy-preserving ML platform, `Biscotti` [3]. Users with data update a neural-network model by stochastic gradient descent and store the updated model on the blockchain. To preserve the privacy of the contributing users, perturbation,

a commitment scheme, and Shamir's secret sharing scheme are applied. Furthermore, to improve the quality of the model, negatively contributing updates are to be eliminated.

Zhu *et al.* proposed a blockchain-enabled decentralized ML platform to eliminate malfunctioning nodes that negatively influence the model update [11]. The global ML model is repeatedly updated. In each round, one of the participants broadcasts his or her local model update, and other participants sends back feedback based on how much their local models improve.

Since no one can monitor the behavior of workers, some measures must be introduced to prevent workers from deviating from protocols. An example of such a measure would be the use of zero knowledge proof and a hardware-based attestation, in which one can assure that programs have actually been executed in a secure environment (e.g., [2], [3]). However, considering that FL is mainly used in mobile devices [1], such functional requirements significantly impair usability. Hence, it is necessary to devise an approach that does not require special hardware or heavy computation for an audit. Given this requirement, we focus on MD, which is often used to make a rule intended to achieve a specific goal and has been well-studied in economics. The design of Bitcoin was inspired by the MD. In Bitcoin, a monetary incentive, that is, cryptocurrency, is given to basically untrusted miners who are supposed to verify transactions and seal them in a new block. Bitcoin is innovative in the sense that the combination of competition and incentive makes miners work well, and we adopt a similar approach for the blockchain-enabled FL. We explain the proposed protocol in detail in the next section.

IV. PROPOSED PROTOCOL

Before examining the procedures in detail, we explain how our protocol works in FIGURE 3. We assume that our protocol has been deployed on a public blockchain network, for example, `Ethereum`, as valuable cryptocurrencies provide a straightforward way to give workers incentives. Furthermore, as listed in TABLE 3, we assume that there are three entities, namely i) `Admin` (an administrator who deploys SCs on the blockchain), ii) $\mathcal{R}$ (requesters who post DNN tasks with cryptocurrencies), and iii) $\mathcal{W}$ (workers who train DNN models to earn rewards). A requester adds a task to a smart contract with an initial DNN model. Interested workers join such a task and are randomly assigned into N rounds. The workers in the first round fetch the initialized model, update it with their own data, and submit their local updates. The workers in the second round fetch all the previously submitted models, evaluate them with their own data and vote top-k models based on classification accuracy. Each worker trains the chosen top-k models with his or her own data and submits the trained model.[3] Cryptocurrencies are given to the workers in the first round based on the vote. These procedures are repeated until N.

[3]Chosen top-k models may differ by workers.

TABLE 2. List of notation.

SYMBOL	DESCRIPTION
$\mathcal{R}$	Entire set of requesters
$\mathcal{W}$	Entire set of workers
t	Task index
i	Worker index
e	Round index
k	Number of chosen models
N	Number of rounds
K	Number of workers joined to a task t
C	Fraction of workers in a round
K'	Number of workers in a round, that is, $K' = \max(\lceil K \cdot C \rceil, 1)$
T	Start time of a task t
T_{com}	Deadline of a commitment
T_{rev}	Deadline of a reveal
$\mathcal{W}_t$	Workers joined to task t
r	The total reward distributed to workers in a round
$r_{i,e}$	The reward distributed to worker i in round e
d	The deposit for a task t, $d = rN$
$\mathcal{D}_i$	Dataset that the worker i of round e possesses
B	Batch size in DNN
η	Learning rate in DNN
$\mathcal{H}(\cdot)$	Hash function, for example, SHA-256
s	Random value used with $\mathcal{H}(\cdot)$
$\mathsf{enc}(\cdot, \mathcal{K})$	Encryption function with a symmetric key $\mathcal{K}$
$\mathsf{dec}(\cdot, \mathcal{K})$	Decryption function with a symmetric key $\mathcal{K}$
$\mathcal{L}(\cdot)$	Loss function in DNN
$\mathsf{split}(\mathcal{D}, B)$	Function to randomly split a dataset $\mathcal{D}$ into B batches

A. PROCEDURES

We then explain the entire procedure, which is composed of six stages: i) user registration, ii) task solicitation, iii) task initiation, iv) model update, v) reward distribution, and vi) task closure. The list of notation is shown in TABLE 2.

1) USER REGISTRATION

All the participating users, namely requesters and workers, must register themselves on the system via a smart contract. FIGURE 2 shows a code snippet of the smart contract for user registration. `Admin` is expected to register a user by giving (i) address and (ii) two Boolean values to indicate whether he or she is registered as requester and/or worker. Once registered, requesters can post tasks, while workers can join tasks.

2) TASK SOLICITATION

FIGURE 4 shows the flowchart of task solicitation. To post a task, a requester must give i) a description of the DNN task, for example, input data format, number of layers, number of units, loss function, learning rate, and activation function; ii) other parameters, for example, starting time T and total amount of reward; and iii) cryptocurrencies for rewards d. A posted task is solicited to registered workers, and they can determine whether to participate in the task. Interested workers must register themselves via a smart contract before the task has started.

3) TASK INITIATION

FIGURE 5 shows the procedures of task initiation. When starting time T arrives, the number of rounds N and the

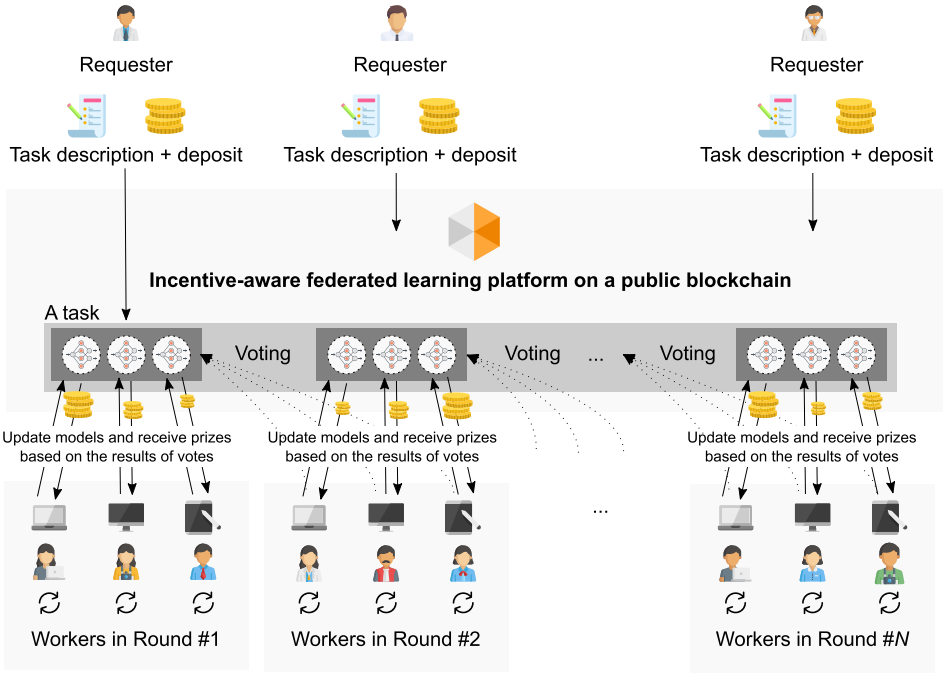


FIGURE 3. Overview of our FL platform with blockchain.

TABLE 3. Comparison of entities' roles.

ENTITY	DESCRIPTION	REMARKS
Administrator	Deploy SCs on the blockchain and register users (both workers and requesters)	-
Requesters	Request a DNN task	They may or may not have training data
Workers	Train a given DNN model with their own data to earn rewards	They have training data and machines to train a model

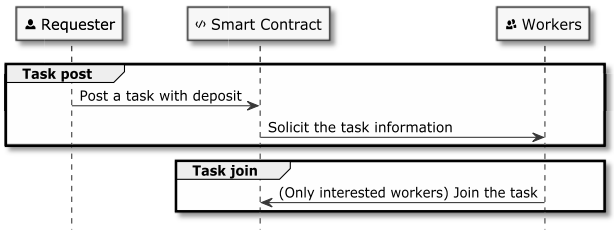


FIGURE 4. Task solicitation.

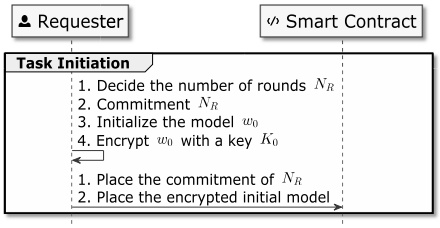


FIGURE 5. Task initiation.

number of workers in a round K' are calculated based on the number of participating workers $\mathcal{W}_t$. For each task, K' workers are randomly chosen from $\mathcal{W}_t$ every round. None of the participants will be chosen in the future rounds, meaning that every participant will join a round only once. Note that if few workers join, such a task is immediately closed, and the deposited cryptocurrencies are paid back to a requester.

As is explained in Section IV-A6, N must be hidden from workers until all the rounds are finished, and a requester must reveal N thereafter. For this purpose, a commitment scheme, a technique to conceal a value until a specific point [18], is applied to N. More specifically, a requester sends a hash value h, which is calculated by $\mathcal{H}(N, s)$, to a smart contract before a task starts, and reveals N with s that satisfies $h = \mathcal{H}(N, s)$ when N is revealed.

The requester initializes a DNN model w_0 with an existing algorithm (e.g., [19]), encrypts w_0 with a symmetric key $\mathcal{K}_0$, and places the encrypted model (i.e., $\mathsf{enc}(w_0, \mathcal{K}_0)$) in the data storage. The data storage can be either a blockchain or a database, and this must be determined by a system operator.

4) MODEL UPDATE

FIGURE 6 illustrates the procedures of model update. Every round K' workers are randomly chosen from the participants.

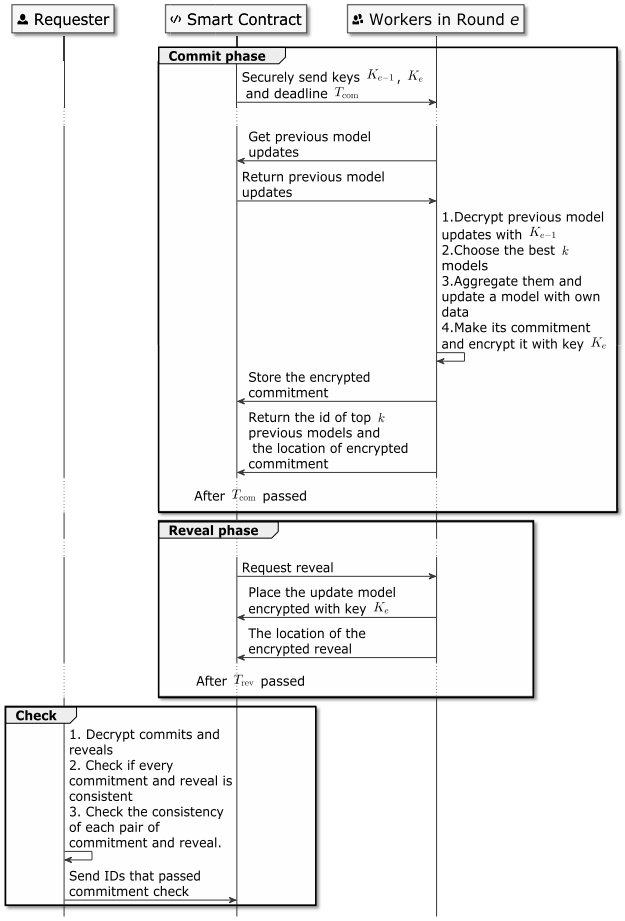


FIGURE 6. Model update.

The requester sends two symmetric keys, $\mathcal{K}_{e-1}$ and $\mathcal{K}_e$, a key to decrypt the model updates in the previous round (not applicable for the workers in the first round) and a key to encrypt model updates, to the workers in round e. The two keys must be securely distributed to the workers. Hence, they are encrypted with each worker's public key and distributed individually.

Algorithm 1 shows the procedures of model update. Except for the first round, after retrieving the models in the previous round, that is, $\{w_{i,e-1}\}$, by deciphering the encrypted models with $\mathcal{K}_{e-1}$, each worker chooses their top k models. To do this, the loss for each model is evaluated using his or her own data $\mathcal{D}_i$, and those k models whose loss is the lowest are chosen. Any loss function (e.g. mean absolute error) can be used as a criterion to measure the goodness of models. As each worker has a different dataset, chosen top k models may differ by workers. A base model, $w'_{i,e}$, is then calculated by averaging the chosen models (see line 10 in Algorithm 1). We assume that a stochastic gradient descent, an algorithm to iteratively optimize a loss function with batch data, is used to update $w'_{i,e}$ with $\mathcal{D}_i$, as in [1].

Updated models are encrypted with $\mathcal{K}_e$ and stored in data storage. However, to prevent an updated model from being exposed to other workers, a commitment scheme is

Algorithm 1: Procedures of Model Update

Input : Models submitted by the workers in the round $e-1$, $\{w_{i,e-1}\}_{i\in K'}$ (when $e \geq 2$) or w_0 (when $e = 1$) Dataset $\mathcal{D}_i = \{\mathbf{x}_j, y_j\}$

Output: w_i

`/* 1. Choose the top k models by evaluating them with its own dataset: */`

1 **if** $e \geq 2$ **then**
2 **for** $m \in K'$ **do**
3 $l_m = \frac{1}{|\mathcal{D}_i|} \sum_{j\in|\mathcal{D}_i|} \mathcal{L}(\mathbf{x}_j, y_j; w_{m,e-1})$
4 **end**
5 $\mathcal{M}_{i,e} \leftarrow$ Choose k models' indices of which l_m is in the k lowest values.
6 **end**

`/* 2. Average the chosen models */`

7 **if** $e == 1$ **then**
8 $w'_{i,e} \leftarrow w_0$
9 **else**
10 $w'_{i,e} \leftarrow \frac{1}{k} \sum_{m\in\mathcal{M}_{i,e}} w_{m,e-1}$
11 **end**

`/* 3. Model update with its own data */`

12 $\mathcal{B} \leftarrow \mathsf{split}(\mathcal{D}_i, B)$
13 $w_{i,e} \leftarrow w'_{i,e}$
14 **for** *each local epochs* E **do**
15 **for** *each batch* b *in* $\mathcal{B}$ **do**
16 $w_{i,e} \leftarrow w_{i,e} - \eta\nabla\mathcal{L}(w_{i,e}, b)$
17 **end**
18 **end**
19 **return** $w_{i,e}$

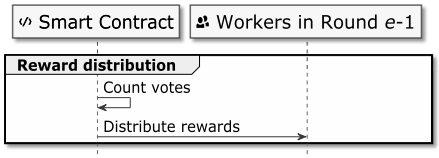


FIGURE 7. Reward distribution.

applied, as in [13]. Specifically, the procedure of storing a new model is divided into two phases, namely, i) the commit phase and ii) the reveal phase. In the commitment phase, the commitment of $w_{i,e}$ is calculated as $\texttt{commit}(w_{i,e}) = \mathcal{H}(w_{i,e}, s_{i,e})$, and $\texttt{commit}(w_{i,e})$ is encrypted with $\mathcal{K}_e$ as $\mathsf{enc}(\texttt{commit}(w_{i,e}), \mathcal{K}_e)$ and is submitted to data storage. In the reveal phase, the used salt should be revealed by T_{rev}. The requester checks whether the commitments are valid on an individual basis.

5) REWARD DISTRIBUTION

FIGURE 7 illustrates the procedures of reward distribution. At the end of round, k models are voted by K' workers, and

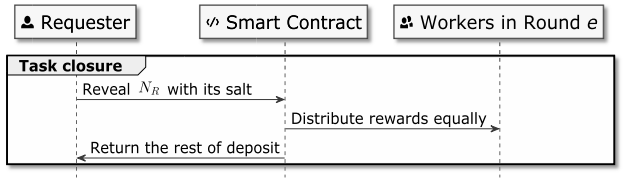


FIGURE 8. Task closure.

the ranking is determined by tallying kK' votes. This process can be automated in the smart contract. At the end of round, a reward is distributed to the workers in the previous round as $r_1 \geq r_2 \geq \cdots \geq r_{K'} \geq 0$ ($\sum_{j\in[1,K']} r_j = r$). Hence, the most-voted worker receives r_1 and the first runner-up receives r_2 and so on. Such an order is determined by the number of models chosen by the workers in the current round. Obviously, the more a worker's model is chosen, the more reward should be given to him or her. Hence, the smart contract counts votes for each worker in the previous round and determines their rankings. The optimal distribution of r_j is discussed in Section V.

6) TASK CLOSURE

The procedures of model update and reward distribution are repeated for N rounds, with the exception of the last one. The workers in the final round will not be voted, since no subsequent workers exist. Hence, as shown in FIGURE 8, a reward is equally distributed as $r_1 = r_2 = \cdots = r_{K'} = r/K'$. However, if the workers knew that this is the final round, such distribution would demotivate them, as they would be able to receive rewards without having updated the model. This is why the requester cannot reveal N before N rounds have been finished and a commitment scheme is used to N in Section IV-A3. Clever workers may guess N from the remaining reward. To avoid this, the requester must put total rewards d larger than what are actually needed, that is, $d > N \cdot r$. The requester will not lose excessive rewards, since the excess will be returned by the smart contract.

B. DISCUSSION

1) ADVANTAGES

The proposed design has some noticeable advantages over the existing ones: First, implementation will be feasible, as our design can be implemented with an existing public blockchain, for example, Ethereum, and it only requires symmetric key cryptography and a commitment scheme. Hence, neither special hardware nor a custom-built blockchain is required for implementation. Second, as is proven later, if workers are rational, they will not be lazy and will expend effort, and this is mathematically guaranteed. In particular, the latter is important, as it will allow requesters to feel certain that they can obtain accurate models and that their rewards will not be wasted.

2) CAVEAT OF TIME-SENSITIVE PROCEDURES

Our design contains time-sensitive procedures, for example, the deadlines for (i) joining a task and (ii) the submission of commit and reveal in a commitment scheme. Major blockchains provide a time function, which should be used to realize the time-sensitive procedures. For example, Ethereum has `block.timestamp` to obtain the current time based on the mined time written in a block. However, participants should be mindful that this value may vary up to 900 seconds, as miners can specify any timestamp as long as the block timestamp is greater than that of the previous block and is within 900 seconds of their local computer [20].

3) COMPLEXITY OF KEY DISTRIBUTION

In Section IV-A4, two keys, $\mathcal{K}_{e-1}$ and $\mathcal{K}_e$, are encrypted with each user's public key and are distributed to workers. Hence, the complexity of calculation and communications is $\mathcal{O}(K')$. Although this would not be a problem in a real-world scenario, a group key distribution method, for example, [21], may reduce computation.

4) ASSUMPTION OF DATA DISTRIBUTION

We assume that datasets that workers use for specific tasks are independent and identically distributed. This assumption is natural, as, after a requester posts a model, only the workers who have data relevant to the task will join. For example, if a task is to train a model that recognizes cats in images, only those workers who possess cat images will join this task.

5) POSSIBILITY OF COLLUSION

Possible attacks should be kept in mind for secure protocol design. In our design, collusion might be possible if an adversary could register a group of decoy workers so as to increase the chance of winning rewards or even to contaminate a model by submitting bogus model updates. Our design is robust against this attack, as (i) users must be approved by an administrator to register themselves and (ii) workers are randomly chosen every round. Implementing such a function is possible even though the underlying blockchain is public. For instance, Ethereum has a logic to restrict access [22], and the user management function can be implemented as shown in FIGURE 2. Another effective mitigation approach to reduce the number of decoy workers would be to impose a registration fee for workers to register themselves. Such fee can be paid by cryptocurrency, as our design is deployed on a public blockchain.

6) PUNISHING DISHONEST WORKERS

Some existing methods penalize dishonest workers, those who deteriorate the quality of models (e.g. [2]). However, we do not do so because of the following two reasons. First, it is infeasible to differentiate honest and dishonest workers. Even if workers are "honest" they may not contribute much (or even negatively contribute). This could happen, for example, when a model is already over-trained, and obviously, they should not be punished in this case. Second, dishonest workers' payoffs will be likely negative without a punishing mechanism, as their updates will hardly be chosen in top-k models despite their "efforts" to deteriorate models. Hence,

the competition mechanism will demotivate such dishonest workers.

7) CLIENT SELECTION METHOD

We follow the random client selection method proposed in the original FL article [1]. However, there may be better client selection methods in terms of faster convergence. We may work on the better worker selection method in the future.

8) MULTI-ROUND VERSUS SINGLE-ROUND

We follow the iterated training process proposed in [1]. Some may feel that this is inefficient in terms of training speed. This is true, but there are two reasons why we adopt the multi-round style. First, low quality models can be naturally eliminated (including negatively-affecting models by adversaries) through the iterated competition process. Second, our method will not work in the single-round style. If this were a single shot contest, a task requester would obtain a total of NK' models and must choose the best models and merge them. As a task requester may not have a dataset to evaluate the quality of models, he or she will not be able to determine the ranking of workers, and rewards thus cannot be determined either.

V. THEORETICAL ANALYSIS

For our design to work, it would be necessary to mathematically prove that workers' contributions pay off and to determine an optimal reward policy. We use contest theory for our analysis [23]–[25]. Contest theory has been employed to analyze the efforts and payoff of workers. A contest is defined as a game in which costly and irreversible efforts must be exerted by workers, and rewards are determined based on the quality of their output [5]. The series of procedures can be seen as a contest, since workers must update a model, which is an irreversible effort, and rewards are distributed based on their quality (determined by votes). The following analysis may confuse the readers who are familiar with auction theory but contest theory. We admit that this may not be exactly same, but we call the following first and second characteristics as individual rationality and Bayesian-Nash incentive compatibility, respectively.

1) Individual rationality: We find a condition in which all rational workers join model updates.
2) Bayesian-Nash incentive compatibility: We find an optimal strategy that workers can maximize their payoffs so that there exist no other advantageous strategies.
3) Optimal reward distribution: We find an optimal reward distribution policy.

We analyze the above characteristics and rule in order. As the same amount of rewards r are distributed every round, all rounds are identical in terms of theoretical analysis. Hence, the worker's ordering sequence does not affect the analysis, and we do not differentiate rounds in the following analysis.

TABLE 4. Comparison of information that requester and worker know.

	Requester	Worker
Given rewards	Yes	Yes
Reward policy (r_j and $\bar{j}$)	Yes	Yes
Competitors' costs	No	No (Only knows his or her own cost)
Distribution of c	Yes	Yes

A. SETUP

We explain the mathematical notations used for theoretical analysis. We first define prizes for workers. The j-th ranked worker's prize is denoted as r_j, where $r_1 \geq r_2 \geq \cdots \geq r_{K'} \geq 0$, meaning that the top worker will be given r_1 and the first runner-up will be given r_2, and so on. As the total rewards in a round is r, $\sum_{j\in[1,K']} r_j = r$ holds. To quantify cost and effort, the number of data and processing cost per data are denoted as x and c, respectively. Every worker is assumed to not know the other workers' c, making it a so-called imperfect (or incomplete information) contest. TABLE 4 shows the comparison of information that a requester and worker know. For the sake of simplicity, each datum x equally contributes to the improvement of the model update. We assume that the cost of the model update, denoted as $\mathcal{C}(\cdot)$, linearly increases with the number of data, x. Hence, $\mathcal{C}(\cdot)$ is expressed as follows:

$$\mathcal{C}(c, x) = cx. \tag{1}$$

c is determined by workers' competence. More specifically, the abler the worker, the lower c. The highest c, meaning the c of the least skilled worker, is denoted as $\bar{c}$. For example, a worker with a small c can process more data at the same cost. c is sampled from a uniform distribution of [0, 1]. Hence, its cumulative distribution function (CDF), $F(c)$, is simply expressed as follows.

$$F(c) = c. \tag{2}$$

Similarly, let $G(x)$ denote a CDF of x for $x \geq 0$. $G(x)$ is continuous and strictly increasing in $(0, x_{\max})$, where $x_{\max}$ is the maximum x.

B. INDIVIDUAL RATIONALITY

In order for our mechanism to be individually rational, the expected payoff must not be negative for all participants. If they were always positive, all workers would have been guaranteed to obtain rewards even though they do not update models or even by deteriorating models. We formulate the expectation of worker's payoff, and the condition of individual rationality can be derived by showing that the expectation is larger than or equal to the expected minimum reward.

A worker's payoff $\pi(c, x)$ is calculated by subtracting cost, $\mathcal{C}(c, x)$, from his or her prize, r_j, when he or she is ranked in j-th; otherwise, it is simply sunk cost, $-\mathcal{C}(c, x)$. Hence,

$\pi(c, x)$ can be expressed as follows.

$$\pi(c,x)=\begin{cases} u(r_j)-\mathcal{C}(c,x), & \text{if a worker is ranked in } j-\text{th} \\ -\mathcal{C}(c,x), & \text{otherwise.} \end{cases} \tag{3}$$

where $u(r_j)$ is a von Neumann-Morgenstern utility function, which is used to reflect the worker's risk for a reward [26].[4] Since rewards are not necessarily guaranteed for a contribution, we assume that workers are risk-averse. This means that the expected payoff is not linear, and $u(\cdot)$ is monotonously increasing and concave.

A worker will maximize the expectation of payoff, $E[\pi(c, x)]$.

$$E[\pi(c,x)] = U(x) - \mathcal{C}(c,x) = U(x) - cx, \tag{4}$$

where $U(x)$ denotes the expected utility of worker with x. When a worker is ranked in j-th place, $j-1$ workers exerted greater efforts than him or her while $K'-j$ workers exerted less effort than him or her. Hence, $U(x)$ is represented as

$$U(x) = \sum_{j=1}^{K'} \binom{K'-1}{j-1} G(x)^{K'-j} \left(1-G(x)\right)^{j-1} u(r_j). \tag{5}$$

As can be seen from Eq. (4), rational workers will not update a model if their expected profits are negative. We identify a condition in which all workers contribute model updates for rewards.

Lemma 1 (Individual Rationality): The following condition must be satisfied if all the workers exert their efforts.

$$u(r_{K'}) \geq E[\pi(\bar{c}, x(\bar{c}))]. \tag{6}$$

Otherwise, only the workers with $c < \bar{c}$ will participate, and the others will choose not to join model updates (i.e., $x(c) = 0$).

Proof: All workers must update models with their own data, which is irreversible and costs some resources, for example, time and energy. Hence, the workers will gain a profit as long as the smallest prize, $r_{K'}$, is larger than their cost c. By contrast, the other, less skilled workers will not join a task, as they have little chance of making a profit. ■

Individual rationality is achievable as every worker can judge if it is better to join a task by calculating the expected payoff after a requester advertises it. The rational workers of which c for the task is larger than $\bar{c}$ will not join the competition in the first place.

C. INCENTIVE COMPATIBILITY

We prove that workers have an optimal strategy to maximize their profits. For this, we first prove Lemma 2 that the abler worker should contribute more, which is necessary to derive the optimal amount of effort that workers should exert as shown in Proposition 1.

[4]Note that the payoffs can be negative in Equation (3). Instead, the "expectation" of payoffs should not be negative.

Lemma 2: The abler worker contributes more, meaning that the ablest worker exerts the greatest effort among the participants.

Proof: This lemma can be proven by showing that $x(c)$ is non-increasing for $(0, \bar{c})$. For this, we first show that the first derivative of $x(c)$, that is, $dx(c)/dc$, is decreasing. The workers with $c < \bar{c}$ choose the optimal x to maximize their expected profits $E[\pi(c, x)]$. In other words, x is chosen so that its first and second order conditions are satisfied with respect to x.

$$\frac{dE[\pi(c,x)]}{dx} = 0 \Leftrightarrow \frac{dU(c,x)}{dx} = \frac{\partial}{\partial x}\mathcal{C}(c,x) = c, \tag{7}$$

and

$$\begin{aligned} \frac{d^2E[\pi(c,x)]}{dx^2} &< 0 \\ \frac{d^2U(c,x)}{dx^2} - \frac{\partial^2}{\partial x^2}\mathcal{C}(c,x) &< 0 \Leftrightarrow \frac{d^2U(c,x)}{dx^2} < 0. \end{aligned} \tag{8}$$

In Eq. (8), $\frac{\partial^2}{\partial x^2}\mathcal{C}(c,x) = \frac{\partial^2}{\partial x^2}cx = 0$ is used.

We differentiate Eq. (7) with c and obtain the following equation:

$$\begin{aligned} \frac{d^2U(c,x)}{dxdc}\cdot\frac{dx(c)}{dc} &= \frac{\partial^2\mathcal{C}(c,x)}{\partial x\partial c} \\ \frac{dx(c)}{dc} &= \frac{\partial^2\mathcal{C}(c,x)}{\partial x\partial c}\Big/\frac{d^2U(c,x)}{dxdc} \end{aligned} \tag{9}$$

As $\mathcal{C}(c,x) = cx$, $\partial^2\mathcal{C}(c,x)/\partial x\partial c = 1$. Using the result of Eq. (8), we can obtain $dx(c)/dc < 0$. ■

We then answer the question of how much effort x workers should exert. As x is dependent on c, we represent x as $x(c)$ and find the optimal x with it.

Proposition 1: Bayesian Nash Incentive Compatibility: A symmetric Bayesian equilibrium function $x(c)$ exists for the workers with c $(< \bar{c})$.

$$\begin{aligned} x(c) = -\int_c^1 \frac{(K'-1)}{z} \sum_{l=1}^{K'} \binom{K'-2}{l-1} \\ \cdot F(z)^{l-1}(1-F(z))^{K'-l-1} f(z)\Delta r_l \, dz, \end{aligned} \tag{10}$$

where $\Delta r_j = u(r_j) - u(r_{j+1})$.

Proof: From Lemma 2, the workers with $c < \bar{c}$ can be aligned by c. With this in mind, the expected utility of prize with cost c can be expressed as the summation of probabilities that a worker is ranked in j-th, $j-1$ workers have costs lower than c, and $K'-j$ workers have costs greater than c. As j can be $[1, K']$, the following equation holds:

$$U(c) = \sum_{j=l}^{K'} \binom{K'-1}{j-1} F(c)^{j-1} \left(1-F(c)\right)^{K'-j} r_j \tag{11}$$

$$= \sum_{l=1}^{K'} \sum_{j=l}^{K'} \binom{K'-1}{j-1} F(c)^{j-1} \left(1-F(c)\right)^{K'-j} \Delta r_j. \tag{12}$$

We calculate the derivative of Eq. (12) and obtain Eq. (13).

$$\begin{aligned}\frac{dU}{dc} &= \sum_{l=1}^{K'}\sum_{j=l}^{K'}\binom{K'-1}{j-1}\Big\{(j-1)F(c)^{j-2}(1-F(c))^{K'-j} \\ &\quad -(K'-j)F(c)^{j-1}(1-F(c))^{K'-j-1}\Big\}f(c)\Delta r_j. \\ &= \sum_{l=1}^{K'}\sum_{j=l}^{K'}(K'-1)\left\{\binom{K'-2}{j-2}F(c)^{j-2}(1-F(c))^{K'-j}\right. \\ &\quad \left.-\binom{K'-2}{j-1}F(c)^{j-1}(1-F(c))^{K'-j-1}\right\}f(c)\Delta r_j \\ &= -(K'-1)\sum_{l=1}^{K'}\binom{K'-2}{l-1}F(c)^{l-1}(1-F(c))^{K'-l-1} \\ &\quad \cdot f(c)\Delta r_j. \end{aligned} \tag{13}$$

From Eq. (7), we obtain the following equation.

$$\frac{dU(c)}{dc} = \frac{U(x(c))}{dc}\cdot\frac{dx}{dc} = \frac{\partial\mathcal{C}(c,x)}{\partial x}\cdot\frac{dx}{dc} = c\frac{dx}{dc}. \tag{14}$$

By substituting Eq. (13) into Eq. (14) and calculating its integral, we obtain Eq. (10). ■

D. OPTIMAL REWARD DISTRIBUTION

We finally derive the optimal reward distribution policy. This is required as the requester wants to maximize the quality of model with a given budget. We find the optimal number of rewards $\bar{j}$ and its distribution $r_1, \cdots, r_{\bar{j}}$. The basic idea is to find the optimal number of rewards and distribution that maximize the expected effort that a requester will obtain.

Proposition 2: The following conditions should be met for the optimized reward policy.

- *The total reward should be distributed to the top $\bar{j}$ workers, where $\bar{j}$ is the less than $(K'+1)/2$.*
- *When $u(\cdot)$ is $\ln(\cdot)$, which is a common concave function for risk-averse workers, $r_j = \frac{K'-2j+1}{K'-1}r_1$.*
- *When $u(\cdot)$ is linear (i.e., when workers are risk-neutral) all prizes are given to the best worker, as $r_1 = r$, and other workers are given no prize (i.e., $r_1 = \cdots = r'_K = 0$).*
- *The total reward sums up to the total reward r, that is, $\sum_{j=1}^{\bar{j}} r_j = r$.*

Proof: From a requester's perspective, he or she wants to maximize the workers' efforts, x. Hence, we find the optimal number of rewards and its proportion by maximizing x. We integrate Eq. (7) with respect to c and obtain Eq. (15):

$$x(c) = cU(c) - \int_0^c U(z)\,dz. \tag{15}$$

The expectation of $x(c)$ is obtained by integrating workers' c.

$$E[x(c)] = \int_0^1 cU(c)\,dc - \int_0^1 (1-c)U(c)\,dc. \tag{16}$$

By substituting Eq. (11) and Eq. (2), we obtain Eq. (17).

$$\begin{aligned}E[x(c)] = \sum_{j=1}^{K'}\binom{K'-1}{j-1}\Bigg[&\int_0^1 c^{j-1}(1-c)^{K'-j+1} \\ &-\int_0^1 c^j(1-c)^{K'-j}\Bigg]u(r_j).\end{aligned} \tag{17}$$

Since $\int_0^1 c^a(1-c)^b\,dc = \frac{a!b!}{(a+b+1)!}$, $E[x(c)]$ can be simplified as follows:

$$E[x(c)] = \sum_{j=1}^{K'}\frac{K'-2j+1}{K'(K'+1)}u(r_j). \tag{18}$$

To maximize $E[x(c)]$, no element in Eq. (18) should be negative. Hence, $K'-2j+1 > 0$, and the maximum number of rewards $\bar{j}$ should be less than $\frac{K'+1}{2}$.

We finally find the optimal proportion of r_j by leveraging Lagrange multipliers. Since the total reward should sum up to r, we have a constraint $\sum_{j=1}^{\bar{j}} r_j = r$. With λ,

$$\begin{aligned}E[x(r_1, r_2, \cdots, r_{\bar{j}}, \lambda)] = &\sum_{j=1}^{\bar{j}}\frac{K'-2j+1}{K'(K'+1)}u(r_j) \\ &-\lambda\left(\sum_{j=1}^{\bar{j}} r_j - r\right).\end{aligned} \tag{19}$$

For each r_j, the following derivative is calculated.

$$\frac{\partial E[x(r_1, r_2, \cdots, r_{\bar{j}}, \lambda)]}{\partial r_j} = \frac{K'-2j+1}{K'(K'+1)}u'(r_j) - \lambda = 0. \tag{20}$$

By solving this equation, we obtain the following equation.

$$u'(r_j) = \frac{K'(K'+1)}{K'-2j+1}\lambda. \tag{21}$$

Since $u'(r_1) = \frac{K'(K'+1)}{K'-1}\lambda$ is satisfied, we can obtain $u'(r_j) = \frac{K'-1}{K'-2j+1}u'(r_1)$. If workers are risk-averse, it is reasonable to assume that $u(\cdot) = \ln(\cdot)$.

$$u'(r_j) = \frac{1}{r_j}, u'(r_1) = \frac{1}{r_1}. \tag{22}$$

Hence, we can obtain the following r_j's condition:

$$r_j = \frac{K'-2j+1}{K'-1}r_1. \tag{23}$$

By contrast, if workers are risk-neutral, that is, $u(r_j) = r_j$, $u'(r_j) = u'(r_1) = 1$.

$$1 = \frac{K'-2j+1}{K'-1} \Rightarrow j = 1. \tag{24}$$

Eq. (24) indicates that all rewards should be given to r_1. ■

We show two real numerical examples when the workers are risk-averse. First, when $K' = 5$, from Preposition 2, the optimal number of prizes is determined to $\bar{j} = 2$, and the optimal reward proportion is given as $r_1 = \frac{2}{3}$, $r_2 = \frac{1}{3}$,

and $r_j = 0$ for $3 \leq j \leq 5$. Similarly, when $K' = 9$, $\bar{j} = 4$, $\{r_1, \cdots, r_4\} = \{\frac{4}{10}, \frac{3}{10}, \frac{2}{10}, \frac{1}{10}\}$, and $r_j = 0$ for $5 \leq j \leq 9$. These results do not violate the reward policy, that is, $r_1 \leq r_2 \leq \cdots \leq r_{K'} \leq 0$, and might be quite intuitive.

VI. CONCLUSION AND FUTURE WORK

We have proposed a blockchain-enabled incentive-aware FL with an MD. There are two key contributions of this article: First, we introduce the use of competition to motivate workers to maximize their rewards, and our design thus naturally makes workers follow the protocol without using any cryptographic approaches, such as homomorphic encryption and zero knowledge proof. Second, we have mathematically identified some conditions for the protocol design to work by leveraging contest theory. These theoretical proof results are promising in terms of realizing the incentive-aware blockchain-enabled FL.

In the future, we will implement our design on top of a public blockchain network, such as Ethereum, and identify operational costs and system performances, such as latency.

APPENDIX
RELATED WORK ON CROWDSOURCING

Many studies on crowdsourcing have been conducted in which requesters can issue demands to users, for example, answers to questions or services, in return for rewards. For instance, Topcoder is a successful crowdsourcing competitive programming platform on which contestants submit programming codes on given subjects by deadlines and by which top contestant(s) are awarded. In that sense, the incentive-aware FL is part of crowdsourcing, and it is thus relevant to our work. In this section, we summarize existing studies that mainly analyze rewards in crowdsourcing.

Vojnović presented how contest theory is applied in online services — in particular, crowdsourcing services [27]. Contest theory provides insights into the users' possible behavior and reward distribution policy and provides algorithms to estimate users' abilities based on observed contest outcomes.

Archak and Sundararajan analyzed the optimal reward distribution for crowdsourcing [25]. They showed that if requesters are risk-averse, multiple prizes should be rewarded to workers. By contrast, requesters are risk-neutral, the reward should be given to the top contributor. This result does not contradict our result shown in Proposition 2.

Ghosh and McAfee analyzed equilibrium and optimal reward distribution for two real use cases in crowdsourcing, namely, (i) online Q&A forums (e.g., Quora and StackOverflow) and (ii) competitions (e.g., Topcoder) [28].

Singla and Karuse proposed an optimal price setting for crowdsourcing with the regret minimization in the online learning [29]. A mathematical analysis was conducted by combining procurement auctions and multi-armed bandits.

Luo *et al.* proposed an optimal reward design with all-pay auction [30] and Tullock contest [31]. In the all-pay auction-based approach, they demonstrated that requesters can maximize the contribution by rewarding only the top contributor [30]. However, they also demonstrated that participants who do not have high ability becomes risk-averse and are not willing to join contests. The subsequent article, which uses Tullock contest, mitigates this problem by introducing a lottery mechanism to give every player a strictly positive chance of winning as long as he or she participates [31].

Radanović *et al.* studied a mechanism in which a task requester cannot evaluate the correctness of outputs by workers [32]. In this case, the best strategy for workers is to simply submit random answers without solving the given tasks. To solve this problem, the authors proposed a monitoring approach in which participants compare their answers with those of others.

Sarne and Lepioshkin analyzed the participants' behavior in a multi-prize contest when the quality of workers' contributions is unknown at the time of participation [33]. The authors of presented several interesting results; for example, in some cases, a worker's expected profit is maximized when a second prize greater than the first is offered.

Chen *et al.* studied the design of optimal payment schemes for requesters to maximize their profits and analyzed an optimal method by which to select qualified workers for given tasks [34]. They proposed a reward policy based on a base salary and additional bonus and a worker selection scheme based on workers' workload demands and past performance.

Rokicki *et al.* conducted a large-scale real experiments to understand how different reward policies affect the cost and time efficiency of crowdsourcing [35]. They tested three reward assignment policies: (i) linear, (ii) ranking, and (iii) lottery. As a result of the experiment with the Amazon Mechanical Turk (AMT) platform, the ranking-based policy with an exponential strategy, in which, for instance, 25, 10, 5, 5, 1, 1, 1, 1, 0.5, and 0.5 USD were payed out to the top-10 users from a total budget of 50 USD, outperformed the others.

Levy and Sarne conducted a series of experiments using contests to understand the effect of the workers' strategy and determine whether they should participate in contests [36]. They tested 4,000 subjects in AMT using different settings, for example, whether contests were sequential or parallel and when the number of participants was different, say three or five.

REFERENCES

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016, *arXiv:1602.05629*. [Online]. Available: http://arxiv.org/abs/1602.05629

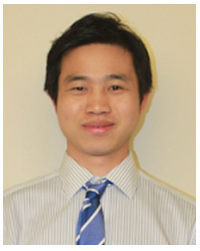
[2] J. Weng, J. Weng, J. Zhang, M. Li, Y. Zhang, and W. Luo, "DeepChain: Auditable and privacy-preserving deep learning with blockchain-based incentive," *IEEE Trans. Dependable Secure Comput.*, early access, Nov. 8, 2019, doi: 10.1109/TDSC.2019.2952332.

[3] M. Shayan, C. Fung, C. J. M. Yoon, and I. Beschastnikh, "Biscotti: A ledger for private and secure peer-to-peer machine learning," 2018, *arXiv:1811.09904*. [Online]. Available: http://arxiv.org/abs/1811.09904

[4] K. Toyoda and A. N. Zhang, "Mechanism design for an incentive-aware blockchain-enabled federated learning platform," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2019, pp. 1–9.

[5] L. C. Corchón and M. Serena, "Contest theory," in *Handbook of Game Theory and Industrial Organization*, vol. 2. Cheltenham, U.K.: Edward Elgar Publishing, 2018.
[6] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 5, pp. 1333–1345, May 2018.
[7] Z. Shae and J. Tsai, "Transform blockchain into distributed parallel computing architecture for precision medicine," in *Proc. IEEE 38th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Jul. 2018, pp. 1290–1299.
[8] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 12:1–12:19, 2019.
[9] K. Salah, M. H. U. Rehman, N. Nizamuddin, and A. Al-Fuqaha, "Blockchain for AI: Review and open research challenges," *IEEE Access*, vol. 7, pp. 10127–10149, 2019.
[10] G. Wood, "Ethereum: A secure decentralised generalised transaction ledger," Ethereum Project Yellow Paper, 2014, pp. 1–32. [Online]. Available: https://files.gitter.im/ethereum/yellowpaper/VIyt/Paper.pdf
[11] X. Zhu, H. Li, and Y. Yu, "Blockchain-based privacy preserving deep learning," in *Proc. Int. Conf. Inf. Secur. Cryptol.* Cham, Switzerland: Springer, 2018, pp. 370–383.
[12] 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," *Appl. Sci.*, vol. 8, no. 12, p. 2663, Dec. 2018.
[13] Y. Lu, Q. Tang, and G. Wang, "On enabling machine learning tasks atop public blockchains: A crowdsourcing approach," in *Proc. IEEE Int. Conf. Data Mining Workshops (ICDMW)*, Nov. 2018, pp. 81–88.
[14] X. Chen, J. Ji, C. Luo, W. Liao, and P. Li, "When machine learning meets blockchain: A decentralized, privacy-preserving and secure design," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Dec. 2018, pp. 1178–1187.
[15] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," 2018, *arXiv:1808.03949*. [Online]. Available: http://arxiv.org/abs/1808.03949
[16] T. Ide, "Collaborative anomaly detection on blockchain from noisy sensor data," in *Proc. IEEE Int. Conf. Data Mining Workshops (ICDMW)*, Nov. 2018, pp. 120–127.
[17] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–7.
[18] G. Brassard, D. Chaum, and C. Crépeau, "Minimum disclosure proofs of knowledge," *J. Comput. Syst. Sci.*, vol. 37, no. 2, pp. 156–189, Oct. 1988.
[19] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1026–1034.
[20] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, 2016, pp. 254–269.
[21] A. Penrig, D. Song, and D. Tygar, "ELK, a new protocol for efficient large-group key distribution," in *Proc. IEEE Symp. Secur. Privacy. (S&P)*, May 2001, pp. 247–262.
[22] M. Wohrer and U. Zdun, "Design patterns for smart contracts in the ethereum ecosystem," in *Proc. IEEE Int. Conf. Internet Things (iThings), IEEE Green Comput. Commun. (GreenCom), IEEE Cyber, Phys. Social Comput. (CPSCom), IEEE Smart Data (SmartData)*, Jul. 2018, pp. 1513–1520.
[23] A. Glazer and R. Hassin, "Optimal contests," *Econ. Inquiry*, vol. 26, no. 1, pp. 133–143, 1988.
[24] B. Moldovanu and A. Sela, "The optimal allocation of prizes in contests," *Amer. Econ. Rev.*, vol. 91, no. 3, pp. 542–558, Jun. 2001.
[25] N. Archak and A. Sundararajan, "Optimal design of crowdsourcing contests," in *Proc. Int. Conf. Inf. Syst. (ICIS)*, 2009, pp. 1–16.
[26] J. Von Neumann and O. Morgenstern, *Theory of Games and Economic Behavior*. Princeton, NJ, USA: Princeton Univ. Press, 2007.
[27] M. Vojnović, "Contest theory," *Commun. ACM*, vol. 60, no. 5, pp. 70–80, Apr. 2017.
[28] A. Ghosh and P. McAfee, "Crowdsourcing with endogenous entry," in *Proc. 21st Int. Conf. World Wide Web (WWW)*, 2012, pp. 999–1008.
[29] A. Singla and A. Krause, "Truthful incentives in crowdsourcing tasks using regret minimization mechanisms," in *Proc. 22nd Int. Conf. World Wide Web (WWW)*, 2013, pp. 1167–1178.
[30] T. Luo, S. S. Kanhere, S. K. Das, and H.-P. Tan, "Incentive mechanism design for heterogeneous crowdsourcing using all-pay contests," *IEEE Trans. Mobile Comput.*, vol. 15, no. 9, pp. 2234–2246, Sep. 2016.
[31] T. Luo, S. S. Kanhere, H.-P. Tan, F. Wu, and H. Wu, "Crowdsourcing with tullock contests: A new perspective," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2015, pp. 2515–2523.
[32] G. Radanovic, B. Faltings, and R. Jurca, "Incentives for effort in crowdsourcing using the peer truth serum," *ACM Trans. Intell. Syst. Technol.*, vol. 7, no. 4, pp. 48:1–48:28, Mar. 2016.
[33] D. Sarne and M. Lepioshkin, "Effective prize structure for simple crowdsourcing contests with participation costs," in *Proc. AAAI Conf. Hum. Comput. Crowdsourcing (HCOMP)*, 2017, pp. 167–176.
[34] X. Chen and K. Xiong, "A payment scheme in crowdsourcing," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[35] M. Rokicki, S. Chelaru, S. Zerr, and S. Siersdorfer, "Competitive game designs for improving the cost effectiveness of crowdsourcing," in *Proc. 23rd ACM Int. Conf. Conf. Inf. Knowl. Manage. (CIKM)*, 2014, pp. 1469–1478.
[36] P. Levy and D. Sarne, "Understanding over participation in simple contests," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 1571–1578.

KENTAROH TOYODA (Member, IEEE) was born in Tokyo, Japan, in 1988. He received the B.E., M.E. and Ph.D. (engineering) degrees from the Department of Information and Computer Science, Keio University, Yokohama, Japan, in 2011, 2013, and 2016, respectively. He was an Assistant Professor with Keio University from 2016 to 2019. He is currently a Scientist with the Singapore Institute of Manufacturing Technology, A*STAR, Singapore. His research interests include blockchain analysis and application, security and privacy for emerging systems and services, data analysis, and eHealth. He is a member of the IEICE. He received the IEICE Communication Society Encouragement Award in 2012 and 2015, respectively, the Telecom System Technology Encouragement Award in 2015, and the Fujiwara Foundation Award in 2016.



JUN ZHAO (Member, IEEE) received the bachelor's degree from Shanghai Jiao Tong University, China, in 2010, and the Ph.D. degree in electrical and computer engineering from the CyLab Security and Privacy Institute, Carnegie Mellon University (CMU), USA, in 2015, under the supervision of Virgil Gligor and Osman Yagan, and collaborated with Adrian Perrig. He joined Nanyang Technological University (NTU), Singapore, as a Postdoctoral Researcher with Xiaokui Xiao and then as a Faculty Member. He was a Postdoctoral Researcher with Arizona State University as an Arizona Computing Postdoctoral Best Practices Fellow under the supervision of Junshan Zhang and Vincent Poor. He is currently an Assistant Professor with the School of Computer Science and Engineering, NTU. His research interests include security/privacy, communication networks, and AI. His co-authored articles received the Best Paper Award (IEEE Transaction article) from the IEEE Vehicular Society Singapore Chapter in 2019 and the Best Paper Award from the EAI International Conference on 6G for Future Wireless Networks 2020.

ALLAN NENG SHENG ZHANG (Associate Member, IEEE) is a currently a Senior Scientist with the Singapore Institute of Manufacturing Technology, A*STAR, Singapore. He has more than 25 years of experiences in the development of operations technologies using AI, such as knowledge-based systems and enterprise information systems development. He and his group are currently working toward research in cyber-physical system modeling, manufacturing system analyses, including data mining, supply chain information management, supply chain risk management using a complex systems approach, multi-objective vehicle routing problems, and urban last-mile logistics. His research interests include knowledge management, data mining, machine learning, artificial intelligence, computer security, software engineering, software development methodology and standards, and enterprise information systems.

P. TAKIS MATHIOPOULOS (Senior Member, IEEE) received the Ph.D. degree in digital communications from the University of Ottawa, Ottawa, ON, Canada, in 1989. From 1982 to 1986, he was with Raytheon Canada, Ltd., where he was involved in the areas of air navigational and satellite communications. In 1989, he joined the Department of Electrical and Computer Engineering, The University of British Columbia (UBC), Vancouver, BC, Canada, as an Assistant Professor, where he was a Faculty Member until 2003 and a Professor from 2000 to 2003. From 2000 to 2014, he was the Director and then the Director of research with the Institute for Space Applications and Remote Sensing (ISARS), National Observatory of Athens, where he established the Wireless Communications Research Group. He was the ISARS Director from 2000 to 2004, where he led the Institute to a significant expansion Research and Development growth and a growth in international scientific recognition. For these achievements, ISARS was selected as a National Centre of Excellence for the years 2005–2008. From 2008 to 2013, he was appointed as Guest Professor by Southwest Jiaotong University, Chengdu, China. Since 2014, he has been a professor of telecommunications with the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Athens, Greece. He was also appointed by the Government of China as a Senior Foreign Expert with the School of Information Engineering, Yangzhou University, Yangzhou, China, from 2014 to 2017, and by Keio University as a Guest Professor (Global) with the Graduate School of Science and Technology, from 2015 to 2016 and from 2017 to 2018, under the Top Global University Project of the Ministry of Education, Culture, Sports, Science and Technology, Government of Japan. For the last 25 years, he has been conducting research mainly on the physical layer of digital communication systems for terrestrial and satellite applications, and in the fields of remote sensing and the Internet of Things. In these areas, he has coauthored more than 120 journal articles published mainly in various IEEE journals, one book (edited), five book chapters, and more than 140 conference papers. He has been or currently serving on the Editorial Board of several archival journals, including the *IET Communications* and the IEEE TRANSACTIONS ON COMMUNICATIONS from 1993 to 2005.

He has regularly served as a consultant for various governmental and private organizations. Since 1993, he has been serving on a regular basis as a scientific advisor and a technical expert for the European Commission (EC). From 2001 to 2014, he served as a Greek Representative to high-level committees in the EC and the European Space Agency. He has been a member of the Technical Program Committees (TPC) of more than 70 international IEEE conferences. He received an Advanced Systems Institute (ASI) Fellowship as a Faculty Member at UBC and a Killam Research Fellowship. He was also a co-recipient of two best conference paper awards. In 2017, he received the 2017 award for outstanding contributions in the field of satellite and space communications from the Satellite and Space Communication Technical Committee of the IEEE Communications Society. He was the TPC Vice Chair of the 2006-S IEEE VTC and the 2008-F IEEE Vehicular Technology Conference, the Co-Chair of the International Conference on Future Information Technology in 2011 and AUTOMOTIVE17. He has delivered numerous invited presentations, including plenary and keynote lectures, and taught many short courses all over the world.