Mobile Edge Computing, Blockchain and Reputation based Crowdsourcing Federated Learning: A Secure, Decentralized and Privacy-preserving System

Yang Zhao, *Student Member, IEEE,* Jun Zhao, *Member, IEEE,* Linshan Jiang, *Student Member, IEEE,* Rui Tan, *Senior Member, IEEE,* Dusit Niyato, *Fellow, IEEE,* Zengxiang Li, *Member, IEEE,* Lingjuan Lyu, *Member, IEEE,* and Yingbo Liu, *Member, IEEE*

***Abstract*—Home appliance manufacturers strive to obtain feedback from users to improve their products and services to build a smart home system. To help manufacturers develop a smart home system, we design a reputation-based federated learning (FL) system to facilitate home appliance manufacturers to train a machine learning model based on customers' data. Thus, manufacturers can use the machine learning model to predict customers' requirements and consumption behaviors in the future. The working flow of our system includes two stages: in the first stage, customers train the initial model provided by the manufacturer using both the mobile phone and the mobile edge computing (MEC) server. Customers collect data from various home appliances using phones, and then download and train the initial model with their local data. After deriving local models, customers sign on their models and send them to the blockchain. Then, we use the blockchain to replace the centralized aggregator in the traditional FL system. In the second stage, manufacturers will select customers or organizations to be miners for calculating the averaged model using received models from customers. By the end of the crowdsourcing job, one of the miners, who is selected as the temporary leader, uploads the model to the blockchain. To protect customers' privacy and enhance test accuracy, we enforce differential privacy on the extracted features and propose a new normalization technique. We experimentally demonstrate that our normalization technique outperforms batch normalization when features are under ϵ-differential privacy protection. In addition, to attract more customers to participate in the crowdsourcing FL task, we design an incentive mechanism to award customers with coins that can be used to purchase other services provided by manufacturers.**

***Index Terms*—Blockchain, Crowdsourcing, Differential privacy, Federated learning, IoT, Mobile edge computing.**

I. Introduction

IoT-enabled smart home system concept has attracted attention and gained great popularity in the last few years since they have an aim to increase the quality of life. A report by Statista [1] estimates that by 2022, the smart home market size around the world will be 53.3 billion. This smart home concept is mainly enabled by IoT devices, smart phone, modern wireless communications, cloud & edge computing, big data analytics, and AI. Particularly, these advanced technologies enable manufacturers to maintain a seamless connection between their smart home appliances. In the future, the IoT-enabled connected home appliances will be able to not only interact with each other but also with manufacturers to provide better services, and more devices will be added with smarter features. However, with the passage of time, the smart home systems are depending on the collection of data. To help home appliance manufacturers improve their products to be smarter, we design a crowdsourcing federated learning system. Our system considers home appliances of the same brand in a family as a unit, and one mobile phone is used to collect data from home appliances periodically and train the model locally [2]. Since mobile phones have limited computational power and battery life, we offload part of the training task to the edge computing sever. Then, the blockchain smart contract is leveraged to generate a global model by averaging the sum of locally trained models submitted by users. In this federated way, source data are supposed to maintain security and privacy.

However, federated learning may still have privacy issues. For example, Melis *et al.* [3] demonstrate that gradient updates may leak significant information about customers' training data. Besides, the federated approach for training the model is susceptible to model poisoning attacks [4]. In addition, information leakage risks exist in the third party's mobile edge computing (MEC) server [5].

To address aforementioned privacy and security issues, we adopt the untampered feature of blockchain and differential privacy to secure our system. Manufacturers upload a preliminary model with initialized parameters. The model is available on the blockchain for customers to download and train with their local data. The blockchain helps the crowdsourcing requester to audit whether there are malicious updates from customers. The traditional crowdsourcing system is hosted by a third party, which charges customers costly service fees, while our designed system uses blockchain to record crowdsourcing activities. Therefore, customers and the requester

Yang Zhao, Jun Zhao, Linshan Jiang, Rui Tan and Dusit Niyato are with School of Computer Science and Engineering, Nanyang Technological University, Singapore, 639798. (Emails: s180049@e.ntu.edu.sg, junzhao@ntu.edu.sg, linshan001@e.ntu.edu.sg, tanrui@ntu.edu.sg, dniyato@ntu.edu.sg).

Zengxiang Li is with the Institute of High Performance Computing (IHPC), A*STAR, Singapore. (Email: liz@ihpc.a-star.edu.sg).

Lingjuan Lyu is with the Department of Computer Science, National University of Singapore. (Email: lingjuanlvsmile@gmail.com).

Yingbo Liu is with Big Data Research Institute of Yunnan Economy and Society, Yunnan University of Finance and Economics Kunming, China, 650201. (Email: liuyingbo@cnlab.net).

can save high service fees while keeping the crowdsourcing system functional. Due to the limitation of the block size, we propose to use the InterPlanetary File System (IPFS) [6] as the distributed storage solution when the model size is large.

More specifically, customers extract features in the mobile from their data using the deployed feature extractor and add noise with a formal privacy guarantee to perturb the extracted features in the first step. In the second step, customers train fully connected layers of the model with perturbed features in the MEC server. Moreover, we replace the traditional batch normalization with a novel normalization technique by removing constraints of mean value and variance, while constraining the bound within $[-\sqrt{N-1}, \sqrt{N-1}]$, where N denotes the batch size. After training, customers should sign on hashes of encrypted models with their private keys and transmit locally trained models to the blockchain. Miners verify identities of senders, download models and calculate the average of all model parameters to obtain the final model. One miner, selected as the temporary leader, encrypts and uploads the final model to the blockchain.

To further motivate more customers to participate in the crowdsourcing task and reduce malicious and poisoning updates, we design a reputation-based crowdsourcing incentive mechanism. In the beginning, all customers hold the same reputation. Then, customers will be rewarded with coins and reputation after they upload models successfully. However, when a customer is caught uploading malicious data, his or her reputation will be downgraded. The requester can choose reliable customers based on their reputation and previous performance, and customers with high reputations will receive more rewards.

Contributions. The major contributions of this paper are summarized as follows:

- First, a hierarchical crowdsourcing FL system is proposed to help build the machine learning model to predict home appliances' future usage to help home appliance manufacturers improve their service quality and optimize functionalities of home appliances.
- Second, we propose a new normalization technique which delivers a higher test accuracy than batch normalization, while preserving the privacy of the extracted features of each participant's data. By leveraging differential privacy, an adversary cannot exploit the learned model to infer customers' sensitive information.
- Third, our blockchain-based system prevents malicious model updates by ensuring that all model updates are held accountable.

Organization. The rest of the paper is organized as follows. In Section II, we explain the concepts of blockchain and differential privacy used in this paper. Section III presents related works and their deficiencies. We introduce our design of the system in Section IV. Section V shows the advantages and disadvantages of our designed system. Section VI presents the experimental results showing that our technique is working. Section VII discusses how we prevent information leakage using differential privacy technique in our designed system. Then, we conclude the paper and identify future directions in Section VIII.

Table I: Notation Table

Symbol	Definition
ϵ	differential privacy budget
N	batch size
μ	mean value of normalized features
σ	variance of normalized features
L	length of feature
W	width of feature
$X_{i,j,k}$	value at a position $\langle i,j \rangle$ for the feature of image k
$\widetilde{X}_{i,j,k}$	value at a position $\langle i,j \rangle$ for the feature of image k after batch normalized
$\hat{X}_{i,j,k}$	value at a position $\langle i,j \rangle$ for the feature of image k after our normalized technique

II. Preliminaries

We organize this section on preliminaries as follows. In Section II-A, we explain the concepts of blockchain and InterPlanetary File System. In Section II-B, we introduce the formal definition of differential privacy. Notations used in the rest of the paper are summarized in Table I.

A. Blockchain and InterPlanetary File System (IPFS)

The blockchain is a chain of blocks that contain the hash of the previous block, transaction information, and a timestamp. Blockchain originates from a bitcoin network as an append-only, distributed and decentralized ledger to record peer-to-peer transactions permanently and immutably. The IPFS is a peer-to-peer distributed file system that enables distributed computing devices to connect with the same file system. We implement off-chain storage by using IPFS, and store hashes of data locations on the blockchain instead of actual files. The hash can be used to locate the exact file across the system.

B. Differential Privacy

Differential privacy provides a theoretical foundation with a provable privacy guarantee against adversaries with arbitrary prior knowledge and unlimited computational power. Intuitively, by incorporating some noise, the output of an algorithm under DP will not change significantly due to the presence or absence of one user's information in the database. By using the DP algorithm, analysts cannot derive confidential information when analyzing the algorithm's outputs. DP has received much interest in both the academia and the industry. Apple incorporated DP into its mobile operating system iOS. Google implemented a DP tool called RAPPOR in the Chrome browser to collect information about clients. A smaller privacy parameter ϵ means stronger privacy protection but less utility of Y as more randomness is introduced to Y. The quantity ϵ is also often referred to as the privacy cost since a smaller ϵ means stronger privacy protection, which can be understood as a lower privacy cost.

The **Laplace mechanism** of [7] can be used to ensure differential privacy by adding independent zero-mean Laplace noise with scale λ to each dimension of the output. Specifically, λ equals Δ/ϵ, where Δ is the ℓ_1-norm sensitivity of the query Q, which measures the maximum change of the true query output over neighboring databases.

The **post-processing** property [7] of differential privacy means that a data analyst, without additional knowledge about the private data, cannot compute a function of the output of a deferentially private algorithm and reduce its privacy guarantee. Hence, in our design, although noise is added to an intermediate layer of the neural network, the post-processing property ensures that the final trained model also satisfies differential privacy.

III. Related Work

Blockchain and federated learning (FL) techniques have been widely used in training a neural network with distributed data [8]–[15]. For example, Weng *et al.* [14] proposed a Deepchain system for the collaborative learning. They designed an incentive mechanism for attracting parties to participate in federated learning. However, they did not consider the malicious attack during transmitting gradients and models. Awan *et al.* [13] proposed a blockchain-based privacy-preserving FL framework, which secured the model update using blockchain's immutability and decentralized trust properties. Li *et al.* [16] designed a blockchain-based decentralized framework for crowdsourcing tasks, which enabled them to do crowdsourcing tasks without a centralized server. Lu *et al.* [9] proposed to leverage blockchain, FL, and differential privacy for data sharing. However, they directly added the differential privacy noise to the original data instead of the gradients, which may affect the accuracy seriously. Lyu [15] made the first-ever investigation on the federated fairness in a blockchain-assisted decentralized deep learning framework, and designed a local credibility mutual evaluation mechanism to enforce fairness. They also developed a three-layer onion-style encryption scheme to guarantee both accuracy and privacy.

On the other hand, FL has attracted substantial attention recently [17]–[21], and one of the most important issues in FL is privacy protection, which is explored in [22]–[26]. Li *et al.* [22] considered the privacy issue during sharing model updates in FL. They proposed to leverage the sketch algorithms to build the sketching-based FL, which provided privacy guarantees while maintaining the accuracy. Hao *et al.* [23] proposed a privacy-enhanced FL scheme to solve the privacy issue in FL. Their scheme helped to achieve efficient and privacy-enhanced FL for IAI. Dolui *et al.* [24] applied the FL paradigms in recommender systems and matrix factorization, which guaranteed recommender systems' functionality and privacy. Nasr *et al.* [25] performed a comprehensive privacy analysis with white-box inference attacks. Wang *et al.* [26] proposed a framework incorporating GAN (Generative Adversarial Network) with a multitask discriminator to solve the user-level privacy leakage in the FL against attacks from a malicious server.

Furthermore, the edge computing has gained popularity as IoT becomes increasingly ubiquitous [27]–[29], and the edge computing server's high computational power can assist customers' training locally. However, only a few works combined deep learning or FL with edge computing [30,31]. Lyu *et al.* [30] proposed a fog-embedded privacy-preserving deep learning framework, where both training data and test data were crowdsourced from end devices. Different techniques, i.e. *Random Projection* (RP) and *Differentially Private SGD* (DPSGD), were considered in a two-level protection mechanism to protect the privacy of both training data and test data. Jiang *et al.* [31] designed a collaborative training method to protect features' privacy. In detail, the feature extraction was performed locally in the edge devices while the classification was executed in the cloud service. In this paper, we proposed a hierarchical federated learning system in which partitioned deep model training was performed between the end devices and the edge computing server, whereas models built by the edge servers were sent to the blockchain for FL. The FL in the blockchain well addressed the training data imbalance issue. Thus, the partitioned deep model training approach was used as a building block of our system in [31].

IV. System Design

We design a system for smart home appliance manufacturers who are interested in building a machine learning model using data from clients' home appliances to analyze customers' habit and improve their service and products. Figure 1 shows an overview of our system architecture. The system consists of three main components: manufacturers, customers and blockchain. Specifically, manufacturers raise a request for a crowdsourcing FL task. Then, customers who are interested in the crowdsourcing FL tasks submit their trained models to the blockchain. Finally, the blockchain gathers customers' models and a selected miner calculates and generates the global FL model for home appliance manufacturers. In the following, We will introduce each component in detail.

A. Manufacturers

We consider building a machine learning model to help manufacturers to predict customers' consumption behaviours and improve home appliances. Manufacturers raise a request to build a machine learning model, which is a crowdsourcing FL task. Customers who have home appliances can participate in the task. The crowdsourcing system implemented with the blockchain records the progress of the crowdsourcing task. To attract more customers, we design an incentive mechanism as follows.

1) Incentive mechanism: Data in home appliances contain customers' confidential information, thus many people are unwilling to contribute data to building the machine learning model. With the incentive mechanism, customers can receive rewards or penalties based on their contributions. After purchasing home appliances, customers may concern about the quality and warranty of their appliances. Manufacturers have enough resources to provide after-sales services. Therefore, after customers contribute to the crowdsourcing FL task, they will be rewarded with coins. Manufacturers distribute these coins so that customers can use coins to trade for services provided by manufacturers, including maintenance and upgrade of appliances. In addition, we design a reputation-based crowdsourcing mechanism. In the beginning, every customer's family will be assigned with an equally initial

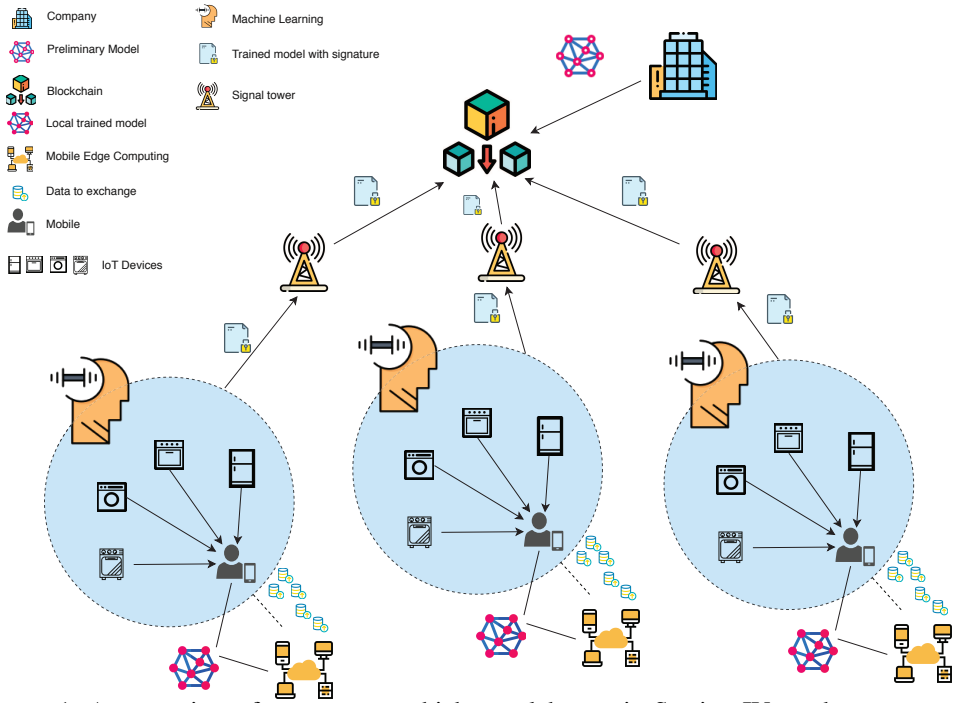

Figure 1: An overview of our system, which we elaborate in Section IV on the next page.

reputation value 10. When customers contribute correct and useful model parameters, their reputation will increase 1. In contrast, if they upload malicious models, their reputation will decrease 1. Customers with a higher reputation will have more opportunities to participate in the next crowdsourcing task and gain more rewards. The reputation status is recorded by the blockchain.

2) Manufacturers put a preliminary model in the blockchain: To facilitate the progress of FL, we use the blockchain to store the initial model. Otherwise, manufacturers need to send the model to everyone or save it in a third party's cloud storage. The initial model's parameters are randomly selected. If enough customers participate in the crowdsourcing FL task, manufacturers can learn a satisfactory model eventually.

B. Customers

Customers who have home appliances satisfying crowdsourcing requirements can apply for participating in the crowdsourcing task. Based on the incentive mechanism, customers can obtain coins and their respective reputation based on their contributions. Large home appliance manufacturers always produce a variety of home appliances. Different home appliances contain different storage and computational powers. Therefore, it is difficult to enable every IoT device to train the deep model. To address this, we adopt the proposed partitioned deep model training approach in [31]. Specifically, we use an mobile phone to collect data from all the home appliances, and extract features on the mobile phone first. To preserve privacy, we add ϵ-DP noise to the features. Then, customers can continue to train fully connected layers in the MEC server. We clarify the customers' responsibilities in four detailed steps as follows:

1) Customers download the initial model from the blockchain: Participants in crowdsourcing FL should check and download the initial model that is available in the blockchain.

2) Customers extract features on the mobile: The mobile phone collects all participating home appliances' data periodically. Then, customers can start training the model using collected data. Since the MEC server is provided by a third-party, there is a potential security issue if the feature extraction based on the raw data is performed on the edge computing server. Therefore, we partition the local training system into two phrases: the mobile training and the MEC server training. Because perturbing original data directly may compromise model accuracy, we take the previous CNN layers as feature extractor to extract features from the original data in the mobile. Then, we add ϵ-DP noise to features before they are fed into the fully connected layers.

3) Customers train fully connected layers in the mobile edge computing server: Mobile sends privacy-preserving features and original labels to the mobile edge computing server so that the edge computing server can help train the fully connected layers and compute the training loss. The perturbed features serve as the input of fully connected layers. The training loss is finally fed back to the mobile phone to update the front layers.

4) Customers upload models to the blockchain: After training the model, customers should sign on the hash of the model with their private keys, and then upload models to the blockchain. However, if miners determine that the signature is invalid, the transaction fails because it is possible that an adversary is attempting to attack the learning process using faked data. After miners confirm the transaction, customers can use the transaction history as an invoice to claim reward and reputation. By using the immutable property of the blockchain,

both manufacturers and customers cannot deny the transaction. Manufacturers can download and check customers' updates to verify whether there are malicious updates. Malicious customers' reputation should deduct if they are caught by manufacturers. Customers with a lower reputation will have a lower chance to participate in the crowdsourcing job in the future.

Now, we present the improvement for the normalization technique proposed in [31]. Although the CNN has many channels, our analysis below focuses on one channel only for simplicity. For this channel, suppose the output of the convolutional layers has dimension $L \times W$. Let the value at a position $\langle i, j \rangle$ for the feature of image k be $X_{i,j,k}$. Given i and j, Jiang *et al.* [31] adopts batch normalization, which transforms $X_{i,j,k}$ to $\widetilde{X}_{i,j,k}$, so that for each batch B, the values $\widetilde{X}_{i,j,k}$ for $k \in B$ have a zero mean and a variance of 1; i.e.,

$$\frac{1}{|B|}\sum_{k\in B}\widetilde{X}_{i,j,k} = 0,$$

while

$$\frac{1}{|B|}\sum_{k\in B}(\widetilde{X}_{i,j,k})^2 = 1.$$

From the Cauchy–Schwarz inequality, [31] bounds

$$\widetilde{X}_{i,j,k} \in [-\sqrt{N-1}, \sqrt{N-1}]$$

for any i, j, k, so that if one value in the feature

$$\left\{X_{i,j,k} \mid i \in \{1,2,\ldots,L\} \text{ and } j \in \{1,2,\ldots,W\}\right\}$$

of image k varies, the sensitivity of

$$\left\{\widetilde{X}_{i,j,k} \mid i \in \{1,2,\ldots,L\} \text{ and } j \in \{1,2,\ldots,W\}\right\}$$

is at most $2\sqrt{N-1}$.

Then, according to Laplace mechanism [7], the independent zero-mean Laplace noise with scale $2\sqrt{N-1}/\epsilon$ is added to each $\widetilde{X}_{i,j,k}$ for $i \in \{1,2,\ldots,L\}$ and $j \in \{1,2,\ldots,W\}$ to protect $X_{i,j,k}$ under ϵ-differential privacy. In our approach, we normalize $X_{i,j,k}$ for $i \in \{1,2,\ldots,L\}$ and $j \in \{1,2,\ldots,W\}$ as

$$\hat{X}_{i,j,k} \in [-\sqrt{N-1}, \sqrt{N-1}],$$

so that if one value in the feature

$$\left\{X_{i,j,k} \mid i \in \{1,2,\ldots,L\} \text{ and } j \in \{1,2,\ldots,W\}\right\}$$

of image k varies, the sensitivity of

$$\left\{\hat{X}_{i,j,k} \mid i \in \{1,2,\ldots,L\} \text{ and } j \in \{1,2,\ldots,W\}\right\}$$

is $2\sqrt{N-1}$. Then, based on Laplace mechanism [7], the independent zero-mean Laplace noise with scale $2\sqrt{N-1}/\epsilon$ is added to each $\hat{X}_{i,j,k}$ for $i \in \{1,2,\ldots,L\}$ and $j \in \{1,2,\ldots,W\}$ to protect $X_{i,j,k}$ under ϵ-differential privacy. From the above discussions, batch normalization of [31] enforces not only

$$\widetilde{X}_{i,j,k} \in [-\sqrt{N-1}, \sqrt{N-1}]$$

but also the mean is

$$\frac{1}{|B|}\sum_{k\in B}\widetilde{X}_{i,j,k} = 0$$

and the variance is

$$\frac{1}{|B|}\sum_{k\in B}(\widetilde{X}_{i,j,k})^2 = 1,$$

while our normalization technique requires only

$$\hat{X}_{i,j,k} \in [-\sqrt{N-1}, \sqrt{N-1}]$$

without any constraints on the mean and variance. Experiments to be presented in Section VI show that our normalization technique significantly improves the learning accuracy over that of [31].

Next, we explain why our normalization technique outperforms batch normalization. Both Jiang *et al.*'s [31] solution and our solution add the same zero-mean Laplace noise to normalized layer inputs. When using batch normalization, the mean of features $\mu = 0$ and the variance $\sigma = 1$. For ease of explanation, below we use a Gaussian distribution as an example for the distribution of the features since Gaussian distributions appear in many real-world applications. Note that the actual distribution of the features may not follow Gaussian. According to the three-sigma rule of Gaussian distribution [32], about 99.73% values lie within three standard deviations of the mean. Similarly, most feature values after batch normalization lie in $[-3\sigma, 3\sigma]$ which is $[-3, 3]$ instead of $[-\sqrt{N-1}, \sqrt{N-1}]$. However, feature values lie more evenly in $[-\sqrt{N-1}, \sqrt{N-1}]$ when using our normalization technique. Thus, features are smaller when using batch normalization than using our normalization technique. Hence, when the same amount of Laplace noise is added, feature values using batch normalization will be perturbed more easily than using our normalization technique. For example, when the batch size $N = 64$ and scale of Laplace distribution is $\frac{2\sqrt{N-1}}{\epsilon}$, we calculate privacy parameter thresholds for feature values after batch normalization and our normalization technique respectively as follows:

$$\frac{2\sqrt{N-1}}{\epsilon} \gg 3\sigma,$$
$$\frac{2\sqrt{N-1}}{\epsilon} \gg 3,$$
$$\epsilon \ll \frac{16}{3} \approx 5.33.$$

Thus, true feature values will be seriously perturbed by noise when the privacy parameter $\epsilon \ll 5.33$ using batch normalization. However, when we use our normalization technique, we obtain that

$$\frac{2\sqrt{N-1}}{\epsilon} \gg \sqrt{N-1},$$
$$\epsilon \ll 2.$$

Hence, when the privacy parameter $\epsilon \ll 2$, the true value will be overwhelmed by the noise. The larger privacy parameter means the less noise, so that feature values using batch normalization are more vulnerable. Thus, our above example

implies that features will be perturbed more seriously when using batch normalization than our normalization technique. Therefore, the trained model with our normalization technique will achieve a higher test accuracy than trained using the batch normalization.

C. Blockchain

We use a consortium blockchain for our crowdsourcing system to store machine learning models permanently. For a consortium blockchain, miners are pre-defined nodes. In the beginning, a manufacturers uploads an initial model to the blockchain. Customers can upload their locally trained models to the blockchain. Because of the limitation of the block size, we propose to use IPFS as the off-chain storage. Then, customers just need to upload the hashes of files to the blockchain, while storing the actual data locally. When accessing the blockchain, customers can obtain the hash of the file location. Then, they can use the hash to obtain the actual file peer-to-peer. Miners are responsible for confirming transactions and calculating the averaged model parameters. After customers upload their locally trained models to the blockchain, miners verify that uploaded models' signatures are valid and confirm the transaction. After all customers upload their trained models, workers download them and start calculating the averaged model parameters. Then, one of workers is selected as the leader to upload the final model to the blockchain. We explain miners' work in detail as follows.

1) Miners verify the uploaded file: When a participant tries to upload his model to the blockchain, a miner checks the signature of the uploaded file. If the signature is valid, the miner confirms that the file is from the legal participant and puts the transaction in the transaction pool. Then one of the miners will generate the block and approve the transaction. If the signature is invalid, the miner should reject the transaction, because it is possible that an adversary uses the public key encrypting the faked model and uploads it to the blockchain to attack the FL model.

2) A selected miner updates the model: Miners are also responsible for calculating the average of uploaded models' parameters to get the FL model. We assume there is a program for miners to get correct averaged parameters. Miners compete for updating parameters to get the reward. Motivated by Algorand [33], we use the Verifiable Random Functions (VRF) as a local and non-interactive way to select a subset of users as the leader candidates (weighed by their coins) and determine their priorities. A leader candidate with the highest priority will become the leader to update the model parameters. As each user is weighted by their coins, one unit of coin can be regarded as a sub-user. A user with w coins has w "sub-users". Let τ be the expected number of sub-users that the system desires to choose, and W be the total mount of coins of all users. Then the probability p of any coin being chosen can be set as τ/W. For a user u with w units of currency, it will first use its secret key to generate a hash and proof via VRF. The interval $[0, 1]$ is divided into $w + 1$ sub-intervals, so that the number j of selected sub-users for this user is determined by which sub-interval $hash/2^{hashlen}$ falls in ($hashlen$ denotes the length of $hash$); i.e., j satisfies $hash/2^{hashlen} \in [\sum_{k=0}^{j} \binom{w}{k} p^k (1-p)^{w-k}, \sum_{k=0}^{j+1} \binom{w}{k} p^k (1-p)^{w-k})$ (if $hash/2^{hashlen} = 1$, then $j = w$). Other users can use the proof to check that user u indeed has j sub-users selected. The number of selected sub-users is each user's priority. The user with the highest priority will become the leader. The selected leader is responsible for updating the model parameters, and uploading the final model to the blockchain.

V. Pros and Cons of our framework

A. Privacy and Security

Our system employs differential privacy technique to protect the privacy of the extracted features. Thus, to some extent, the system keeps the participating customers' data confidential. Furthermore, the trained model is encrypted and signed by the sender to prevent the attackers and imposters from stealing the model or deriving original data through reverse-engineering.

B. Delay in Crowdsourcing

Assume there is a large number of customers, and the system highly depends on customers' training results to obtain the predictive model in one global epoch. Unlike other crowdsourcing jobs, manufacturers in our system prefers customers to follow their lifestyle instead of rushing to finish the job to obtain the real status. As a result, customers who seldom use devices may postpone the overall crowdsourcing progress. This problem can be mitigated by using the incentive mechanisms. Yu *et al.* [34] designed a queue to store customers who submitted their models in order. Thus, customers who submit their locally trained models early get more rewards, so that to encourage people to submit their updates earlier.

VI. Experiments

To validate the effectiveness of our designed FL with differential privacy approach, we conduct experiments on MNIST handwritten image dataset [35].

A. Experiment Setup

MNIST dataset includes 60,000 training image samples and 10,000 test image samples. Each sample is a 28×28 gray scale image showing a handwritten number within 0 to 9. Our designed CNN network includes hidden layers that are responsible for feature extraction and fully connected layers for classification. We have two convolutional layers with 30 and 80 channels respectively. After each convolutional layer, we deploy a max-pooling layer to reduce spatial dimensions of the convolutional layers' output. Therefore, max-pooling layers accelerate the learning speed of the neural network. Normalization is used after all non-linear layers, i.e., convolutional layers. The normalization layer enables the computation of sensitivity in differential privacy to determine the amount of noise to add, speeds up the learning rate, and regularizes gradients from distraction to outliers. Then, we apply ϵ-DP noise to perturb the output of normalization layers to preserve the privacy of the extracted features.

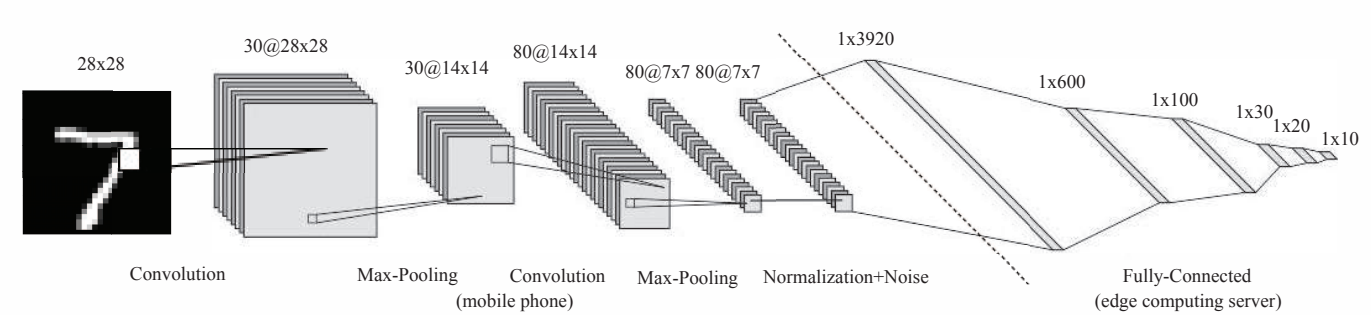

Figure 2: The CNN structure used in our experiments. The normalization layer and the noise followed are used to achieve differential privacy (DP) guarantee.

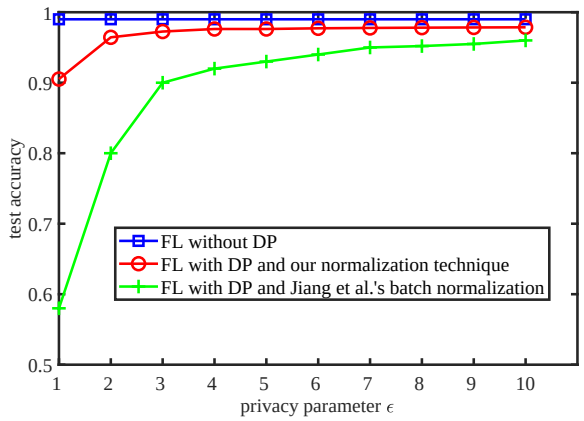

Figure 3: Comparing the test accuracies between federated learning (FL) without differential privacy (DP) and different DP-aware FL algorithms, including DP-aware FL using our normalization technique, and DP-aware FL using Jiang *et al.*'s [31] batch normalization.

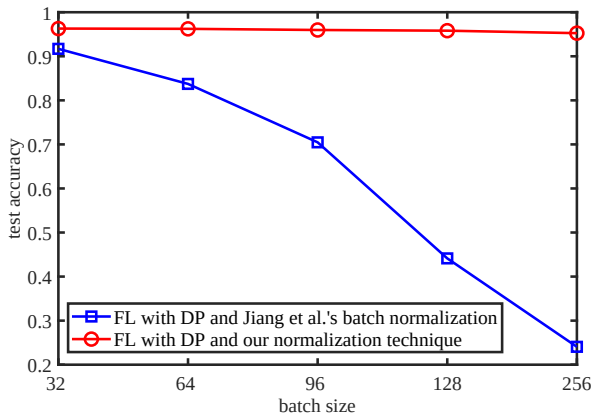

Figure 4: Impact of the batch size on the test accuracy of the FL model with ϵ-differential privacy for ϵ and local epochs being 2 and 40 and the global epoch $= 1$.

Perturbed features serve as inputs of fully connected layers for classification in the MEC server. In our designed model, fully connected layers include four hidden layers. The dimension decreases from 3920 to the dimension of the label which is 10. Finally, there is a softmax layer to predict label and compute loss. The architecture of CNN is shown in Figure 2. We simulate the FL by constructing the model using the averaged parameters of multiple locally trained model parameters.

In our experiment, we set the hyperparameters of CNN as follows. The learning rate is 0.01, and the batch size N is 64. Then, we set the range of privacy parameter ϵ to be $[1, 10]$. Before training, we separate the training image dataset into equally ten parts, meaning that each participant gets 6000 training images randomly. We normalize each dimension of the feature to the interval $[-\sqrt{N-1}, \sqrt{N-1}]$ for N denoting the batch size, so that the sensitivity of the normalized feature vector when one dimension of the feature changes is $2\sqrt{N-1}$. Then, according to Laplace mechanism [7], the independent zero-mean Laplace noise with scale $2\sqrt{N-1}/\epsilon$ is added to each dimension of the normalized features to protect features under ϵ-differential privacy.

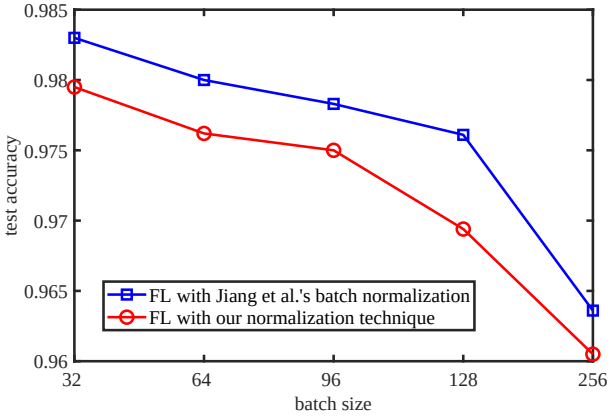

Figure 5: Comparing test accuracies between FL using our normalization technique and FL using Jiang *et al.*'s [31] batch normalization when the global epoch is 1, ϵ-differential privacy parameter $\epsilon = 2$ and the number of local epochs is 40.

B. Experimental Results

Figure 3 compares the test accuracies between federated learning (FL) without differential privacy (DP) and different DP-aware FL algorithms, including DP-aware FL using our normalization technique, and DP-aware FL using Jiang *et al.*'s [31] batch normalization. Figure 3 shows the

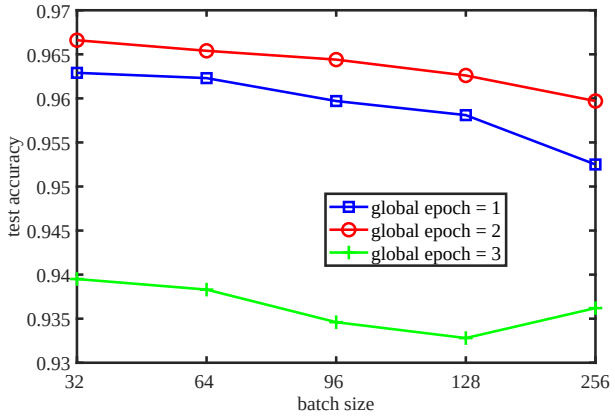

Figure 6: Impact of the global epoch and batch size on the test accuracy of the federated learning model under ϵ-differential privacy for ϵ and local epochs being 2 and 40 respectively.

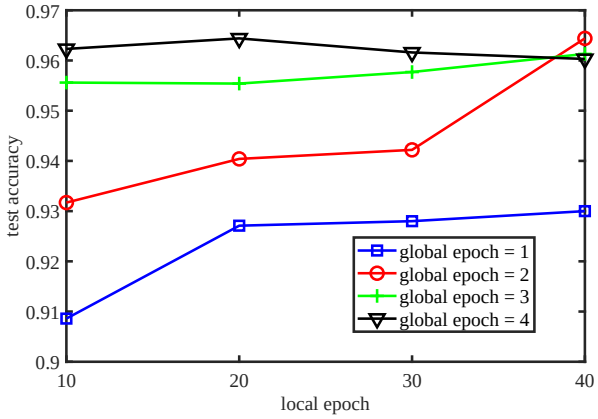

Figure 8: Impact of the local epoch on the test accuracy of the federated learning model with ϵ-differential privacy for ϵ and batch size being 2 and 64 respectively.

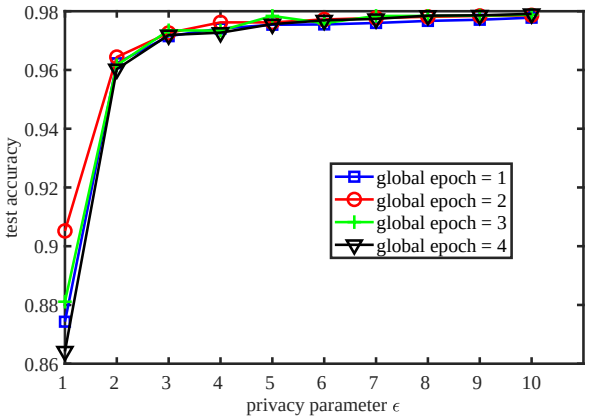

Figure 7: Impact of differential privacy (DP) parameter ϵ on the test accuracy when the number of global epochs $= 1, 2, 3$ and 4.

superiority of DP-aware FL using our normalization technique over DP-aware FL using Jiang *et al.*'s [31] batch normalization. Thus, we confirm that our normalization technique is useful when we add Laplace noise to features, because we relax constraints of normalization compared with batch normalization as stated in Section IV-B. A feature goes through batch normalization results in a smaller size than that goes through our normalization technique, so the value of feature is easily overwhelmed by the noise when using batch normalization. For each DP-aware FL, we also observe that the test accuracy gets closer to the test accuracy of FL without DP as the privacy parameter ϵ increases, because a larger privacy parameter ϵ means less privacy protection which equals that little noise is used. Thus, we conclude that our normalization technique outperforms the batch normalization under ϵ-differential privacy when training the FL model.

Figure 4 shows that the test accuracy of FL model decreases as the batch size increases. This is because we add Laplace noise to features, thus the added noise will increase as the batch size N increases, which results in a worse test accuracy. Moreover, due to the three-sigma rule in Gaussian distribution, most feature values normalized with batch normalization lie in $[-3\sigma, 3\sigma]$. But feature values normalized using our normalization technique lie in $[-\sqrt{N-1}, \sqrt{N-1}]$. However, Figure 5 shows that if no differential privacy noise is added, the test accuracy with the batch normalization outperforms that using our normalization technique. Moreover, as the batch size increases, the test accuracy will decrease. Therefore, we conclude that our normalization technique is more suitable for FL with DP protection. Moreover, Figure 6 illustrates that the test accuracy is better when the global epoch $= 2$ than the global epoch $= 1$ or 3 when $\epsilon = 2$ and the number of local epochs is 40. As the number of global epochs increases, the test accuracy increases. However, Laplace noise increases as the number of global epochs increases. In our case, when the privacy budget $\epsilon = 2$ and the number of local epochs is 40, the optimal number of global epochs is 2.

Figure 7 shows how the privacy parameter ϵ affects the test accuracy of the FL model. In our experiment, we train the FL with 4 global epochs to validate the practicality of our designed approach. The test accuracy increases as the privacy parameter ϵ increases. A larger ϵ means that less noise is added to features, so that the privacy protection is weaker. In the literature, typical ϵ values chosen for experimental studies are between 0.1 and 10 [36]. Our experiment shows that we can achieve at least 90% accuracy when the global epoch $= 2$ and the privacy parameter $\epsilon > 1$. Before training, we initialize the model with random parameters, and the model with initial parameters will be used by all parties for their local training. After the first global epoch, we obtain a new model by averaging all parties' model parameters. Then, in the second global epoch, parties start training using the model from the first global epoch. Through our experiment, we can verify that our designed FL method is effective. However, when the number of global epochs increases to 3 or 4, the test accuracy may decrease. The test accuracy decreases because the noise increases as the number of global epochs increases.

Figure 8 shows that the test accuracy of the FL model is affected by both the local epoch and the global epoch. The number of local epochs reflects the cost of devices' computing

resources locally. Normally, if a device trains the model with more epochs, and then the locally trained model will obtain a higher accuracy. Thus, the number of global epochs will reduce. However, we add ϵ-differential privacy noise during training, the test accuracy may drop if there is too much noise added each epoch. From Figure 8, when the number of local epochs equals 20 or 30, it takes 4 global epochs to achieve a similar accuracy. When the number of local epochs is 40, it takes 2 global epochs. But the test accuracy will start to drop if the number of local epochs is 40 and the number of global epochs is more than 2. Hence, to obtain a high test accuracy, it necessites optimal values to strike a good balance between the number of local epochs and global epochs for averaging locally uploaded models, which we leave as the future work.

VII. Discussion

To attract more customers to contribute their data to train the global model, our designed system should guarantee that customers' confidential information will not leak. There are some studies discussing potential risks of information leakage in FL [37,38]. Attackers are possible to obtain customers' private data from gradients. To prevent this scenario, we add differential privacy noise to features after the feature extraction and before classification in the fully connected layers. Thus, gradients are protected by the differential privacy. Hitaj *et al.* [37] demonstrated that a curious server could learn the private data using GAN (Generative Adversarial Network) if gradients were protected by a large privacy budget in the collaborate learning. But their experiments confirmed that GAN was invalid when the selected privacy parameter was smaller than 10, which is the upper bound of privacy parameter in our experiment. Hence, the privacy parameter assigned to the each gradient is relatively small, but we can achieve the accuracy of 97%. Therefore, our designed approach guarantees the accuracy and protects the privacy of local models. Yin *et al.* [38] introduced a DeepInversion method which could invert a trained neural network to a synthesized class-conditional input images starting from the random noise. However, they leveraged the information stored in the batch normalization layer. In our trained model, we add differential privacy noise to the batch normalization layer, so that attackers cannot obtain the true information stored in the normalization layer. Thus, their approach is ineffective against our trained model.

VIII. Conclusion and Future Work

In this paper, we present a reputation-based crowdsourcing FL system for IoT manufacturers to provide better services in the future. We use multiple state-of-the-art technologies to construct the system, including the mobile edge computing server, blockchain, distributed storage, and federated learning. Besides, our system enforces differential privacy to protect the privacy of customers' data. To improve the accuracy of the FL model, we design a new normalization technique which is proved to outperform the batch normalization if features' privacy is protected by the differential privacy. By designing a proper incentive mechanism for the crowdsourcing task, customers are more likely to participate in the crowdsourcing tasks. The blockchain will audit all customers' updates during the federated training, so that the system can hold malicious updates accountable.

In the future, we aim to conduct more experiments and test our system with real-world home appliance manufacturers. Moreover, we will strive to find the deterministically optimal balance between local epochs and global epochs to obtain the best test accuracy.

References

[1] S. R. Department, "Smart home - Statistics & Facts," 2020. [Online]. Available: https://www.statista.com/topics/2430/smart-homes/

[2] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "When edge meets learning: Adaptive control for resource-constrained distributed machine learning," in *IEEE Conference on Computer Communications (INFOCOM)*, 2018, pp. 63–71.

[3] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *IEEE Symposium on Security and Privacy (S&P)*, 2019.

[4] C. Fung, C. J. Yoon, and I. Beschastnikh, "Mitigating sybils in federated learning poisoning," *arXiv preprint arXiv:1808.04866*, 2018.

[5] Y. Zhang, T. Gu, and X. Zhang, "Mdldroid: a chainsgd-reduce approach to mobile deep learning for personal mobile sensing," *arXiv preprint arXiv:2002.02897*, 2020.

[6] J. Benet, "IPFS-content addressed, versioned, P2P file system," *arXiv preprint arXiv:1407.3561*, 2014.

[7] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography Conference (TCC)*, 2006, pp. 265–284.

[8] X. Qu, S. Wang, Q. Hu, and X. Cheng, "Proof of federated learning: A novel energy-recycling consensus algorithm," *arXiv preprint arXiv:1912.11745*, 2019.

[9] 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*, 2019.

[10] P. Ramanan, K. Nakayama, and R. Sharma, "Baffle: Blockchain based aggregator free federated learning," *arXiv preprint arXiv:1909.07452*, 2019.

[11] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Communications Letters*, 2019.

[12] B. Yin, H. Yin, Y. Wu, and Z. Jiang, "FDC: A secure federated deep learning mechanism for data collaborations in the internet of things," *IEEE Internet of Things Journal*, 2020.

[13] 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.

[14] J. Weng, J. Weng, J. Zhang, M. Li, Y. Zhang, and W. Luo, "DeepChain: Auditable and privacy-preserving deep learning with blockchain-based incentive," *Cryptology ePrint Archive, Report 2018/679*, 2018.

[15] L. Lyu, J. Yu, K. Nandakumar, Y. Li, X. Ma, and J. Jin, "Towards fair and decentralized privacy-preserving deep learning with blockchain," *arXiv preprint arXiv:1906.01167*, 2019.

[16] M. Li, J. Weng, A. Yang, W. Lu, Y. Zhang, L. Hou, J.-N. Liu, Y. Xiang, and R. Deng, "CrowdBC: A blockchain-based decentralized framework for crowdsourcing," *IEEE Transactions on Parallel and Distributed Systems*, 2018.

[17] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[18] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *arXiv preprint arXiv:1909.11875*, 2019.

[19] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *arXiv preprint arXiv:1908.07873*, 2019.

[20] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.

[21] L. Lyu, H. Yu, and Q. Yang, "Threats to federated learning: A survey," *arXiv preprint arXiv:2003.02133*, 2020.

[22] Z. Liu, T. Li, V. Smith, and V. Sekar, "Enhancing the privacy of federated learning with sketching," *arXiv preprint arXiv:1911.01812*, 2019.

[23] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Transactions on Industrial Informatics*, 2019.

[24] K. Dolui, I. C. Gyllensten, D. Lowet, S. Michiels, H. Hallez, and D. Hughes, "Poster: Towards privacy-preserving mobile applications with federated learning–the case of matrix factorization," in *The 17th Annual International Conference on Mobile Systems, Applications, and Services, Date: 2019/06/17-2019/06/21, Location: Seoul, Korea*, 2019.

[25] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 739–753.

[26] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 2512–2520.

[27] J. Xu, S. Wang, B. K. Bhargava, and F. Yang, "A blockchain-enabled trustless crowd-intelligence ecosystem on mobile edge computing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3538–3547, 2019.

[28] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[29] W. Shi and S. Dustdar, "The promise of edge computing," *Computer*, vol. 49, no. 5, pp. 78–81, 2016.

[30] L. Lyu, J. C. Bezdek, X. He, and J. Jin, "Fog-embedded deep learning for the internet of things," *IEEE Transactions on Industrial Informatics*, 2019.

[31] L. Jiang, X. Lou, R. Tan, and J. Zhao, "Differentially private collaborative learning for the IoT edge," in *International Workshop on Crowd Intelligence for Smart Cities: Technology and Applications (CISC)*, 2018.

[32] F. Pukelsheim, "The three sigma rule," *The American Statistician*, vol. 48, no. 2, pp. 88–91, 1994.

[33] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling byzantine agreements for cryptocurrencies," in *ACM Symposium on Operating Systems Principles (SOSP)*, 2017, pp. 51–68.

[34] H. Yu, Z. Liu, Y. Liu, T. Chen, M. Cong, X. Weng, D. Niyato, and Q. Yang, "A fairness-aware incentive scheme for federated learning," in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 2020, pp. 393–399.

[35] Y. LeCun, C. Cortes, and C. Burges, "MNIST handwritten digit database," 2010, accessed on March 1, 2019. [Online]. Available: http://yann.lecun.com/exdb/mnist

[36] D. Sánchez, J. Domingo-Ferrer, and S. Martínez, "Improving the utility of differential privacy via univariate microaggregation," in *International Conference on Privacy in Statistical Databases*, 2014, pp. 130–142.

[37] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: information leakage from collaborative deep learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 603–618.

[38] H. Yin, P. Molchanov, Z. Li, J. M. Alvarez, A. Mallya, D. Hoiem, N. K. Jha, and J. Kautz, "Dreaming to distill: Data-free knowledge transfer via DeepInversion," *arXiv preprint arXiv:1912.08795*, 2019.