TIFF: Tokenized Incentive for Federated Learning

Jingoo Han[†], Ahmad Faraz Khan[†], Syed Zawad[§],
Ali Anwar[||], Nathalie Baracaldo Angel[¶], Yi Zhou[¶], Feng Yan[§], Ali R. Butt[†]
[†]Virginia Tech, [§]University of Nevada, Reno, [||]University of Minnesota, [¶]IBM Research–Almaden
[†]{jingoo, ahmadfk, butta}@vt.edu, [§]{szawad, fyan}@nevada.unr.edu,
[||]aanwar@umn.edu, [¶]{baracald, yi.zhou}@ibm.com

***Abstract*—In federated learning (FL), clients collectively train a global machine learning model with their own local data. Without sharing sensitive raw data, each client in FL only sends updated weights to consider privacy and security concerns. Most of existing FL works focus mainly on improving model accuracy and training time, but only a few works focus on FL incentive mechanisms. To build a high performance model after FL training, clients need to provide high quality and large amounts of data. However, in real FL scenarios, high-quality clients are reluctant to participate in FL process without reasonable compensation, because clients are self-interested and other clients can be business competitors. Even participation incurs some cost for contributing to the FL model with their local dataset. To address this problem, we propose TIFF, a novel tokenized incentive mechanism, where tokens are used as a means of paying for the services of providing participants and the training infrastructure. Without payment delays, participation can be monetized as both providers and consumers, which promotes continued long-term participation of high-quality data parties. Additionally, paid tokens are reimbursed to each client as consumers according to our newly proposed metrics (such as token reduction ratio and utility improvement ratio), which keeps clients engaged in FL process as consumers. To measure data quality, accuracy is calculated in training without additional overheads. We leverage historical accuracy records and random exploration to select high-utility participants and to prevent overfitting. Results show that TIFF provides more tokens to normal providers by up to 6.9% and less tokens to malicious providers by up to 18.1%, achieving improvement of the final model accuracy by up to 7.4%, compared to the default approach.**

***Index Terms*—Federated learning, Privacy-aware machine learning, Incentive mechanism, Tokenized incentivization, Distributed deep learning**

I. Introduction

To build high-quality machine learning (ML) models, a massive amount of training data needs to be collected from various clients. With the growing usage of mobile and IoT devices, these large number of devices have become one of the main sources of user-generated data. This in turn enables ML models to become better over time. In addition, research institutes, government organizations, and industries can share their own data with others to build ML models collaboratively to get more complex yet accurate architectures. However, management of these locally-generated data also makes problems because it requires handling of private and secure data, which can leak private information [1]. Traditional distributed ML training methods [2]–[4] require large-scale training data to be moved to a central location such as the cloud-based servers [5]. However, deploying these traditional approaches into the cloud leads to cybersecurity risks and privacy concerns [6]. In order to meet privacy requirements, privacy laws and regulations [7], [8] have been enacted, which prevents data transfer to a centralized place. Recently, federated learning (FL) [9] has become important as a collaborative training approach to prevent the disclosure of private information because it does not require direct data transfer. Thus, ML models can be trained with a large number of clients without exposing raw data, which only sends local updates to a central server, referred to as an aggregator. In this way, data privacy can be protected because participating clients in the FL process do not need to send its own local data to other central locations.

Although FL has shown great potential in promising privacy-preserved ML, problems still exist in deploying FL. First, successful FL can come from continued long-term participation and availability of good-quality dataset [5]. One of the difficulties lies in data heterogeneity [9], where each client may have different data distribution. Unlike traditional distributed learning, in FL, it is highly possible that data distribution is not uniform [10], known as non-Identical Independent Distribution (non-IID data heterogeneity). As a result, contribution to model update by participating clients varies significantly according to their participation frequency and local data quality. Therefore, it is essential for clients to actively and reliably participate in FL process with high-quality data. Most existing works on FL [11]–[13] do not provide incentive mechanisms to promote participation of clients because they assume that all clients agreed to participate unconditionally and share their data voluntarily. However, without satisfactory rewards that compensate participating costs, each client is not willing to participate in FL process [14], [15]. Especially, clients with good quality data may be reluctant to share their local data with others. For achieving good model performance, it is necessary to incentivize local data owners to contribute large amounts of high-quality data. This leads us to the main question - *How can we incentivize data owners with good quality data to contribute consistently in the training process?* Our work focuses on providing the answer to this question.

While there have been recent works [5], [15], [16] to address incentive mechanisms for FL, these works give incentives to encourage high-quality clients to frequently and reliably participate in FL process. These works conduct clients selection

[||]Work performed while at IBM Research–Almaden.

based on quality estimation and prediction. However, the existing research does not address how to promote long-term participation of consumers, but only leverages historical quality records based on loss, instead of directly measuring accuracy. Besides, these works assume direct monetary transfer where participating clients exchange budget in a one-to-one manner, but there may be some delays before each client has enough budget to pay back [15].

In this paper, we propose **TIFF**, a **T**okenized **I**ncentive mechanism **F**or **F**ederated learning where tokens are a way of paying for the services of providing participants[1] and the training infrastructure. A third organization may provide secure and private aggregation and communication systems through tokenized FL incentive framework. Then, each client as a consumer should pay tokens to participate in training process because they can use the final trained model to generate revenue from commercialization of the model. Tokens can be monetized by the third organization without delays between training and commercialization of the trained model, where providers or consumers can exchange tokens using token-based pricing model, instead of direct monetary transfer between clients. Apart from the above-mentioned instant payback, the tokenized scheme can provide reasonable incentives to motivate both continued long-term participation and high-quality data contribution. Per training round, some of clients can be chosen as providers that train their own local data and send updates to an aggregator for updating a global model. To encourage participation of data providers with good quality data, tokens will be given to the selected providers proportionately according to their contribution to model update. To promote long-term active participation of providers, tokens will be given to providers who more frequently participated in training rounds. Moreover, as an incentive for continued participation of consumers, paid tokens can be reimbursed to each consumer when per-round accuracy of the model is not improved enough compared to expectation.

Contributions. This paper makes the following contributions:

- We present TIFF, a novel token-based incentive mechanism for FL systems, which uses tokens to monetize FL-as-a-Service for participants and the training infrastructure. Unlike direct monetary transfer, participating providers are paid back without delays through token-based pricing model.
- We design a method to incentivize high-quality data providers and encourage long-term participation, which gives free tokens using our newly created algorithm based on their per-round contribution and participation frequency.
- We introduce a reimbursement scheme to promote continued long-term participation of consumers where tokens are reimbursed to consumers according to utility[2] improvement. To systemize amount of reimbursement tokens, we propose token reduction ratio and utility improvement ratio.
- We suggest how to select good providers per round using both historical accuracy records and random exploration.
- We implement and evaluate the proposed FL incentive mechanism to demonstrate that the proposed incentive approach provides reasonable incentives for achieving better model performance and increasing participation of high-quality clients.

II. Background

A. Federated Learning

With the development of artificial intelligence technologies and growing size of data in modern applications, distributed learning methods [2]–[4] have been explored and developed to address newly created large-scale dataset in our daily life. However, due to privacy legislation [7], [8], private data should not be exposed or uploaded to a central server without any privacy consideration. In recent years, federated learning (FL) [9] has been proposed to collaboratively train a shared global model without explicitly sharing their local data with others. The basic concept of FL is to allow multiple clients to locally train the global model and to update the global model by iteratively aggregating model updates from these clients. FL system consists of two main components such as the clients and the aggregator. Before training starts, eligible clients need to be registered to a parameter server called the aggregator. The model training is processed synchronously in rounds by the aggregator. At the beginning of each training round, the aggregator randomly selects a subset of clients and distributes the global model to clients in the subset. Then, each client trains the model on its own local datasets and sends local model updates to the aggregator for model aggregation. Let us assume that N and D_k are the number of clients and local dataset on client k. In each round t, each client k independently trains its local model with its own local dataset D_k for local epochs and updates its model parameters w_t^k. Then, each client k sends its own model weight difference Δ_t^k, which is defined as:

$$\Delta_t^k = w_t^k - w_{t-1}^k \tag{1}$$

After the aggregator receives the weight difference from all clients in the subset, the aggregator updates the global model by the Federated Averaging (FedAVG) algorithm [9] with the learning rate η as follows:

$$w_{t+1} = w_t - \eta \sum_{k=1}^{N} \frac{1}{N} \Delta_t^k \tag{2}$$

The above steps will be repeated until the trained model reaches a target accuracy or the number of training rounds reaches the predetermined round.

In practice, FL faces the challenge of data heterogeneity [11]. In traditional distributed learning, data is collected at a central location and the classes of the training dataset are evenly distributed across clients, called Independent Identical Distribution (IID). However, in FL, training dataset is not

[1] We use terms ’clients’ and ’participants’ interchangeably in the paper.

[2] We use the term ‘utility’ to represent a function of the accuracy of the trained global model, as the previous works [16], [17] define.

uniformly distributed among clients [10] because training data on a given client depends on the client's experience and preference, known as non-Identical Independent Distribution (non-IID). Thus, contribution to model performance changes according to discrepancy of data quality between clients [18].

B. Incentive Mechanism

Incentive mechanisms have been studied in other areas such as crowdsensing [19], [20], but these works have not been directly applied to FL area [5]. Game theory and auction can be used as approaches to provide incentives for FL [14], [21]. Yu et al. [15] address FL incentives as auctions where the payment to data owners is determined by the auction mechanism that models cost, contributions, and regret of data owners. FAIR [5] proposes an auction-based incentive mechanism that models interaction between the clients and the aggregator as reverse-auction where each client submits bid information and the aggregator calculates optimal set of clients for maximizing model performance within limited budget. The Stackelberg game [22] is one of game theories, which formulates the hierarchical and competitive interactions between a leader and follower [23], where the leader decides the action predicting follower's response and the follower chooses actions based on the leader's action while maximizing their own profits. Khan et al. [21] addresses FL incentives as the Stackelberg game to model the competitive interactions between the clients and the aggregator, where clients focus on maximizing their resource utility while the aggregator focuses on maximizing the model performance.

III. Incentivized Federated Learning

In this section, we describe the design of our tokenized incentivization for federated learning (FL). The major idea here is that tokens are provided based on accuracy improvement and participation frequency as a means of encouraging reliable participation of high-quality data providers. This can be achieved by designing our system to fulfill the following objectives:

- *Compensating providers proportional to the quality of their data* - Data providers that can contribute more to the training process (i.e., their participation speeds up convergence or increases accuracy during training at a faster rate), must be compensated properly as a means of reward.
- *Incentivizing long-term participation* - Apart from good quality providers, we must also encourage long-term participation. We do so by rewarding tokens to clients that have been selected as providers more frequently. Eventually, additional tokens will be rewarded to clients which have participated for longer periods.
- *Reimbursing tokens for lower utility improvements* - If the improvement in model performance is not sufficient after each training round, we reimburse some of the tokens since the participants could not get the appropriate value out of participation.

For the next few subsections, we will further elaborate on the system design features that enable us to achieve these goals.

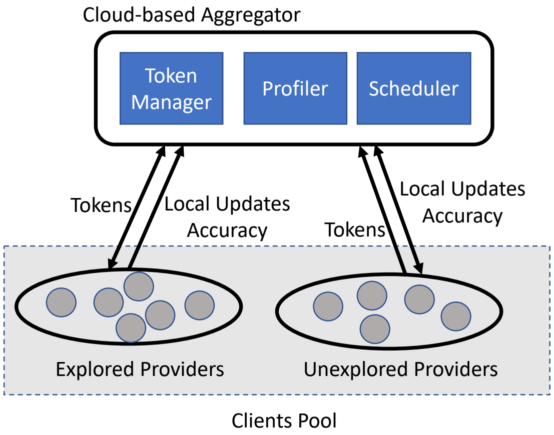


Fig. 1: Overview of incentivized FL with tokens.

We provide stronger definitions of utility, participation rates, token values, etc. and describe the algorithms that use them to accomplish the goal of incentivizing participation of clients such that the overall model performance is improved.

A. Overall Design

The overall architecture of the proposed tokenized FL incentivization is presented in Figure 1. All clients in the client pool can act as both consumer and provider. From a consumer's perspective, each client can benefit from a trained global model. From a provider's perspective, each client can contribute to training the global model. First, if clients want to participate in FL training process as consumers, they should buy tokens from a secure institution (e.g., IBM, Google, Apple) that orchestrates the FL process and provides a framework for FL. We use tokens as a form of credit for providing incentives to the providers. To the best of our knowledge, this is the first to use tokenization for offering FL as a service. Then, clients in the client pool can be chosen by an aggregator as providers. After completing a round of training, the selected clients receive tokens as incentives according to their contributions to the model performance. We classify all participating clients into two categories of providers: *unexplored* providers, and *explored* providers. At first, all participating clients do not have accuracy history because these clients are not trained, and these untrained clients are called unexplored providers. Then, after training starts, some of clients are selected as providers. These selected clients train on their local data, and then send back the local updates and test accuracies to the aggregator. Thus, these trained clients have their accuracy history and are then called explored providers.

Unlike existing FL systems, our proposed system includes additional modules, residing at the aggregator side: *token manager*, *profiler*, and *scheduler*. The role of the *token manager* is to collect tokens from consumers and to pay back these collected free tokens to providers and consumers. The token manager keeps track of the tokens that have been distributed and collected between consumers and providers. The token manager issues tokens to providers according to contributions and reimburses tokens to consumers depending on utility improvement. The *profiler* monitors accuracy of all explored providers to measure quality of their local data. When a group

Algorithm 1 Accuracy-based provider selection with random exploration.

```
Input: $n$: Total number of providers, $L$: List of all providers,
       $E$: List of explored providers, $U$: List of unexplored
       providers, $Acc^k_{test}$: local test accuracy of client k, $N_R$:
       Number of clients to be selected randomly in each
       round, $N_A$: Number of clients to be selected based on
       accuracy rank in each round, $r$: Current round
Sort all providers in $E$ according to $Acc^k_{test}$
$S = SortDesc([E])$
$R$ = Ranked list of providers in the sorted list $S$ in
  descending order
$Q_a$ = Deterministically selected $N_A$ clients for current
  round $r$ from $E$ based on their ranks in $R$
$B$ = Number of clients in $U$
if $B > N_R$ then
   $Q_r$ = list of $N_R$ clients selected randomly from $U$ for
   training in current round $r$, $Q_r \subseteq U$
else
   $Q_r$ = list of $N_R$ clients selected randomly from $L$ for
   training in current round $r$, $Q_r \subseteq L$
end if
$Q_s = Q_a + Q_r$
for each data provider $i \in Q_s$ do
   if $i \notin E$ then
      Add $i$ to $E$
   end if
   if $i \in U$ then
      Remove $i$ from $U$
   end if
end for
return $Q_s$
```

Algorithm 2 Incentivized FL training with tokens.

```
Input: $L_P$: List of all providers, $L_C$: List of all consumers,
       $R$: Total training rounds
for each round $r$ = 0 to $R$ - 1 do
   Token manager collects tokens from consumers for
   training in round $r$
   Scheduler selects providers for training in round $r$
   according to both accuracy-based selection and random
   exploration (Algorithm 1)
   Each provider performs local training
   Each provider sends local update and local accuracy
   back to the aggregator
   Aggregator calculates utility improvement in round $r$
   using a global accuracy
   Token manager reimburses tokens to consumers accord-
   ing to utility improvement (Equation 7)
   Token manager pays tokens back to the selected
   providers according to local accuracy (Algorithm 3)
   Token manager distributes remaining tokens to all
   providers according to participation frequency (Algo-
   rithm 4)
end for
```

of selected providers finish a round of training, the profiler requests accuracy from the selected providers. Then, each selected provider sends its own local accuracy to the profiler, and the profiler stores the collected local accuracy into history records. Moreover, the profiler measures the accuracy of the trained global model per round, and then the measured global accuracy is used for reimbursing tokens to consumers. In each training round, the *scheduler* sorts providers in descending order based on the accuracy history records and deterministically selects providers. The accuracy history of each provider is used for selecting high-quality providers. On top of that, the scheduler also randomly selects unexplored providers for training. The detailed description of provider selection and reimbursement scheme will be given in the following subsections.

B. Provider Selection with Accuracy History and Random Exploration

Selecting providers out of all participating clients can be conducted based on both accuracy history and random exploration (Algorithm 1). At the beginning of each round, clients are sorted based on accuracy history records and half of the number of selected clients (Q_a) are chosen from the explored clients. The accuracy-based provider selection allows high-quality providers to contribute to training a global model. Since local data quality is tracked in real-time by profiling local accuracy of each provider, the selection scheme can dynamically adapt to changing data conditions in a timely manner. Remaining half of the selected clients (Q_r) are chosen from the unexplored clients by random exploration. The random exploration can fairly explore untrained clients and allow the scheduler to choose high-quality providers among them in the next selection round. List of selected clients (Q_s) is combined from both accuracy-based selection (Q_a) and random exploration (Q_r). However, when there is not enough number of explored clients at early rounds, client selection is done by random selection without accuracy-based selection. After training of current round r, clients chosen from the unexplored clients are moved from list of unexplored providers (E) to list of explored providers (U). When all providers are explored, half of the number of selected clients (Q_r) are still randomly chosen from list of explored providers (E). We adopt this approach to select from this diverse set of providers so that overfitting issues are prevented. This is because if global model is trained only with specific clients, it leads to training bias [11].

C. Incentivized Federated Training with Tokens

Algorithm 2 gives a detailed procedure of how our proposed incentive mechanism performs FL training with tokens. At the beginning of each round, the token manager collects tokens from consumers and the scheduler selects a set of providers from the pool of candidate providers that agree to participate in that training round. Then, the aggregator sends a global

Algorithm 3 Token distribution based on local accuracy.

```
Input: n: Total number of selected providers, L: List of selected providers, R: Rank of provider s, $Acc_s$: Local accuracy of provider s
$D = \frac{n(n+1)}{2}$
Sort the selected providers according to their local accuracy $Acc_s$
$S = SortDesc([L])$
$i =$ rank for current data provider
$F =$ Total number of free tokens available
$i = 0$
for each data provider $s$ = 0 to $S$ - 1 do
    Free tokens for $s = (n - i) \times \frac{F}{D}$
    $i++$
end for
```

model to those providers to perform local training. The tokens paid by consumers are called free tokens (T_{free}). As we mentioned above, these free tokens are collected and managed by the token manager. After the scheduler chooses providers by accuracy-based selection and random exploration, each selected provider conducts local training. With the training completed, the selected providers send back their own local updates and local accuracy to the aggregator. The profiler at the aggregator measures a global accuracy and calculates utility improvement. Utility improvement is presented as per-round improvement of a global accuracy. Then the token manger reimburses tokens to consumers according the utility improvement. The detailed explanation of the reimbursement algorithm is given in the next subsection. After reimbursement, according to their *local accuracy* (Algorithm 3), tokens are given to the *selected* providers that conduct local training for the current round. Then remaining tokens are distributed to *all* providers according to their *participation frequencies* (Algorithm 4). In this way, regardless of contribution for model training, providers are rewarded according to their continued long-term participation in federated training.

For the next round, clients with high local accuracy records are chosen by the scheduler for training a global model. The local accuracy records can be a direct indicator of data quality of clients. By selecting and rewarding reliable providers with good data quality, the accuracy of the trained global model can be improved. On the other hand, providers with bad data quality are not frequently chosen and therefore less rewards will be given to them. While *loss* is an indication of accuracy improvement used by previous works [5], [24], we use *accuracy* for reimbursement and scheduling. In previous works that used loss, it causes additional overhead to measure accuracy from selected clients. However, we found in our environment that each selected client already calculates accuracy whenever they conduct local training. Thus, we directly use the local accuracy as a metric for accuracy improvement *without* additional overhead. We also observed that high quality data providers show better local accuracy than low quality data providers.

Algorithm 4 Token distribution based on participation frequency.

```
Input: n: Total number of all providers, L: List of all providers, R: Rank of provider s, $r_s$: Number of participating rounds of provider s
$D = \frac{n(n+1)}{2}$
Sort all providers according to the number of their participating rounds $r_s$
$S = SortDesc([L])$
$i =$ rank for current data provider
$F =$ Total number of free tokens available
$i = 0$
for each data provider $s$ = 0 to $S$ - 1 do
    Free tokens for $s = (n - i) \times \frac{F}{D}$
    $i++$
end for
```

D. Reimbursement

To keep consumers continuously engaged in FL process, tokens are reimbursed to consumers, depending on utility improvement. The general idea here is that more tokens will be returned back to consumers when the utility is less improved. Utility improvement is calculated by the profiler module at the aggregator side. Let us assume that T_{ret} is the reimbursed tokens and I_{util} is the utility improvement. Thus, the number of reimbursed tokens T_{ret} is inversely proportional to the utility improvement I_{util} as:

$$T_{ret} \propto \frac{1}{I_{util}} \tag{3}$$

Let us assume that Acc_r is a global accuracy of current round r and Acc_{max} is the maximum global accuracy achieved until the current round r. The profiler keeps track of both Acc_r and Acc_{max} during training process. Then the utility improvement of round r is calculated as:

$$I_{util} = \max(0.0, \frac{(Acc_r - Acc_{max})}{Acc_{max}}) \tag{4}$$

To calculate reimbursement tokens, we propose the two following metrics: (1) *utility improvement ratio* I_r, and (2) *token reduction ratio* T_r. Utility improvement ratio I_r and token reduction ratio T_r are calculated as:

$$I_r = \frac{I_{max} - \min(I_{max}, I_{util})}{I_{max}} \tag{5}$$

$$T_r = T_{max} \times I_r \tag{6}$$

Therefore, the reimbursed tokens T_{ret} is calculated as:

$$T_{ret} = T_{free} \times T_r \tag{7}$$

In equations 5 and 6, I_{max} is the maximum amount of utility improvement, and T_{max} is the maximum portion of tokens returned. Thus, utility improvement ratio I_r is ranged between 0 and I_{max}, and token reduction ratio T_r is ranged between 0 and T_{max} as below:

$$I_r \in [0, I_{max}] \tag{8}$$

$$T_r \in [0, T_{max}] \tag{9}$$

I_{max} and T_{max} have values between 0 and 1 as below:

$$I_{max} \in [0, 1] \tag{10}$$

$$T_{max} \in [0, 1] \tag{11}$$

T_{max} and I_{max} work as knobs to control the amount of reimbursement tokens. The maximum tokens of reimbursement is limited by T_{max}. For example, when T_{max} is set to 1 and I_r is reached to 0, all pre-paid tokens are reimbursed to consumers. Since different learning applications have different learning progress, utility improvement ratio I_r can be curved by I_{max} to provide a portion of incentive compensation. When I_{max} is set to lower values, utility improvement ratio I_r is higher. As a result, less tokens are reimbursed to consumers, which guarantees reasonable incentives to providers in case that learning curve is not steep enough.

E. Individual Rationality

Given the incentive mechanism, we can consider incentive properties such as individual rationality. Individual rationality is one of the necessary conditions that make clients more willing to stay in federation process. Unlike the existing FL incentive studies [5], [16], we guarantee individual rationality separately for both consumer and provider perspectives. For the consumer perspective, if there is no improvement of utility, paid token will be reimbursed to clients. Thus, each consumer only pays tokens when non-negative utility is obtained (i.e., $I_{util} > 0$). For the provider perspective, clients selected for a single training round are rewarded based on accuracy contribution and participation frequency. In this way, incentive should be given to clients, as long as the clients participate in federated training. Therefore, each provider should be rewarded when non-negative utility (i.e., $I_{util} > 0$) is satisfied after gradient aggregation.

F. Incentive Compatibility

The incentive mechanism is incentive compatible if each data provider receives proper incentive according to their true contribution. The profiler module collects accuracy results from providers to declare their contribution differently. To maximize utility, high quality data providers can be chosen more frequently and be given more rewards as incentives than others.

G. Budget Feasibility

The total compensation of tokens to participants does not exceed the total sum of tokens paid from participants. With a total budget B, B_i is assigned to i-th round of whole training process with total rounds R. Their total sum should be equal to or less than the total budget, i.e., $B \leq \sum_{i=1}^{R} B_i$.

IV. Evaluation

A. Experimental Setup

1) Testbed: Our evaluation testbed is built by deploying 50 clients on a CPU cluster, where each client has its own exclusive CPU core for local training. We implement all codes in the IBM FL framework [25] version 1.0.6 in our evaluation because it is on the market and publicly available. Additionally, it is highly modularized as python-based modules and supports various ML frameworks such as TensorFlow [2], PyTorch [26], and Keras [27]. In this paper, the Keras is chosen as a ML library that provides FL model implementation (e.g., model definition and model update) to the IBM FL framework.

2) Datasets: We use two image classification datasets for evaluating TIFF: MNIST [28], and CIFAR-10 [29]. MNIST is composed of handwritten digit images of 10 classes, each class corresponding to a digit from 0 to 9. CIFAR-10 dataset contains 60,000 RGB images, which are 32x32 resolution images of 10 classes. Each class corresponds to different categories of animals and objects.

For different data quality distributions, we use the following two different kinds of providers with different data quality levels: a) *normal* provider: the provider trains a model on the training dataset with the original unchanged labels to normally train the model. b) *malicious* provider: the provider trains a model on the training dataset with the incorrect labels to maliciously train the model.

3) Models: We use a simple CNN [30] model for evaluating our proposed incentive algorithm, TIFF. The CNN model consists of 5 layers: a 3x3 convolution layers with 32 channels and ReLu activation, a 3x3 convolution layers with 64 channels and ReLu activation, a MaxPooling layer of size 2x2, a fully-connected layer with 128 units and ReLu activation, and a fully-connected layer with 10 units and softmax activation.

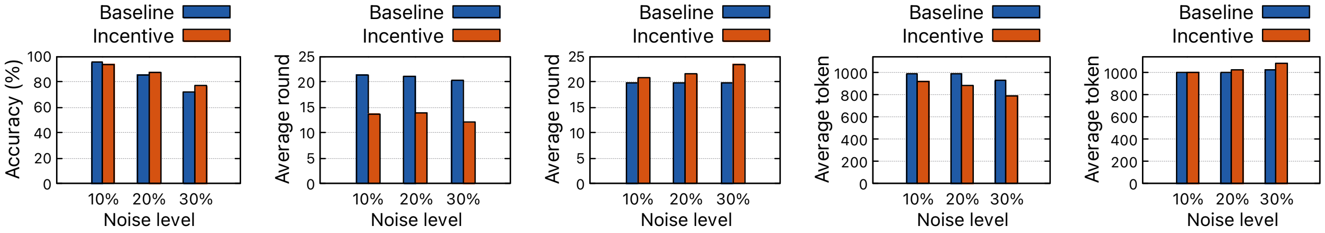
4) Methodology: We use SGD [31] as the optimizer for training local data in each local provider. We train a CNN model on either MNIST dataset or CIFAR-10 dataset for total 100 rounds. We set the number of local epochs to 1. We run experiments five times and report the averages.

5) Metrics: We measure final model accuracy, the number of participating rounds, and the total sum of tokens as our metrics.

B. Results for MNIST dataset

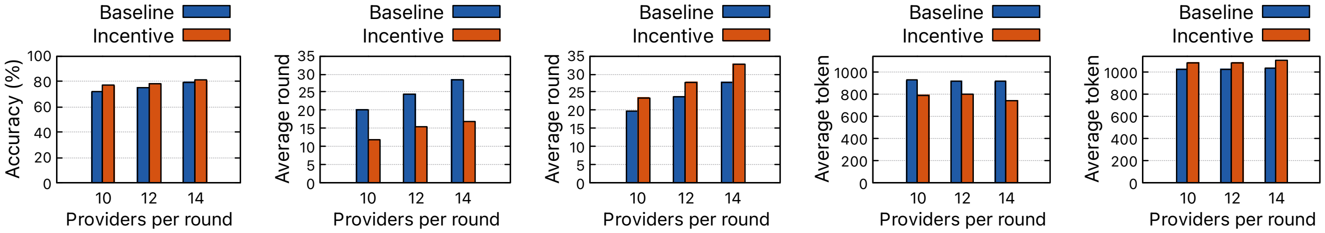
We evaluate and demonstrate the effectiveness and efficiency of TIFF incentive mechanism. We compare the performance of the proposed approach with the the Federated Averaging (FedAVG) algorithm [9] that is one of the most widely used FL algorithms [32]. In this baseline case, the FedAVG randomly selects clients without considering data quality of participating providers, and the free tokens are distributed *equally* to selected providers per round, regardless of their contribution to the learning progress.

Figure 2 shows the final model accuracy, the average round of malicious and normal providers, and the average token of malicious and normal providers with different noise levels. The model is trained with 50 providers under different noise level conditions: a) *normal* provider: each provider has its original unchanged training dataset. b) *malicious* provider: each provider has corrupted training dataset with random labels. The noise level refers to the percentage of malicious providers that have mislabeled data. We consider three noise levels such as 10%, 20%, and 30%. For very heterogeneous (non-IID) class distributions, each client of MNIST has a local dataset with



(a) Accuracy. (b) Round (Malicious). (c) Round (Normal). (d) Token (Malicious). (e) Token (Normal).

Fig. 2: Comparison results for different noise levels on MNIST.



(a) Accuracy. (b) Round (Malicious). (c) Round (Normal). (d) Token (Malicious). (e) Token (Normal).

Fig. 3: Comparison results for different provider numbers on MNIST.

2 classes. Each client has 1000 tokens and pays 10 tokens per round to participate in FL process as a consumer, thus providing 500 new free tokens in each training round. For all noise levels, 10 clients are selected as providers for training a single round. When a client is selected as a provider per round, the number of participating rounds of the client is increased by one. After running 100 rounds, we measure the accuracy, rounds numbers, and amount of tokens of each provider. Then, we average rounds numbers and amount of tokens in both groups of providers (i.e., malicious and normal providers).

The Figure 2a shows accuracy results with different noise levels. The baseline reaches final model accuracy of 95.0%, 85.5%, and 71.5%, whereas TIFF reaches final model accuracy of 93.5%, 87.4%, and 76.8%. In general, accuracy decreases as noise level increases as expected, because training with malicious parties has a negative effect on model training. TIFF improves the final accuracy by up to 7.4%, as compared with the the baseline. Specifically, the accuracy of 20% and 30% noise levels increases by 2.3% and 7.4%, respectively. This is because our incentive method selects high-quality providers more frequently based on accuracy contribution, while low-quality providers have less chances to be selected. However, it does not give performance improvement in the case of 10% noise level. The baseline already reaches very high accuracy (i.e., 95.0%), and thus there is no room to improve accuracy by accuracy-based selection scheme. Thus, our incentive approach, TIFF, is more effective in terms of accuracy in more noisy environment.

The Figures 2b and 2c show average participating rounds of malicious and normal providers. Overall, the numbers of rounds are similar regardless noise levels. In the case of malicious providers, the baseline shows average rounds between 20.2 and 21.3, whereas TIFF shows average rounds between 12.0 and 13.9. Thus, our incentive scheme reduces average rounds of malicious provider by up to 40.9%, which indicates that when the data quality of providers are not good, these providers have much lower chances to be chosen than high-quality providers by our accuracy-based selection. In the case of normal providers, the baseline shows average rounds between 19.7 and 19.9, while TIFF shows average rounds between 20.7 and 23.4, providing improvement of up to 17.4%.

The Figures 2d and 2e show average tokens of malicious and normal providers. In the baseline case, there is not a large difference between malicious and normal providers: malicious providers have average tokens between 929 and 995, and normal providers have average tokens between 1001 and 1030. On the other hand, TIFF shows up to 37% of difference between malicious and normal providers: malicious providers have average tokens between 793 and 926, and normal providers have average tokens between 1008 and 1088. Our incentive scheme reduces average tokens of malicious providers by up to 14.7% and increases average tokens of normal providers by up to 5.7%. Thus, it indicates that low-quality providers have much lower incentives than high-quality providers.

Figure 3 shows the final model accuracy, the average round of malicious and normal providers, and the average token of malicious and normal providers with different provider numbers. At the beginning of each round, the scheduler module at the aggregator chooses providers from the client pool to perform local training on each provider. In the Figure 3, we select different numbers of providers (i.e., 10, 12, and 14 providers) among 50 providers per each round and train the model with the chosen providers, until the predefined number of rounds (i.e., 100 rounds) is reached. The model is trained under the above-mentioned different noise conditions such as normal and malicious providers, but noise level is fixed as 30%. Each consumer pays 10 tokens per round and total 500 free tokens is reimbursed and distributed to consumers and providers as earlier experiments.

The Figure 3a shows accuracy results with different numbers of providers. The baseline reaches final model accuracy of 71.5%, 75.3%, and 79.3%, whereas our proposed incentive work reaches final model accuracy of 76.8%, 78.1%, and 81.0%. TIFF improves the final accuracy by up to 7.4%, as compared to the the baseline. The accuracy of 10, 12, and 14 providers increases by 7.4%, 3.7%, and 2.1%, respectively. Regardless of numbers of providers, our incentive method provides better accuracy than the baseline by selecting high-quality providers more than low-quality providers.

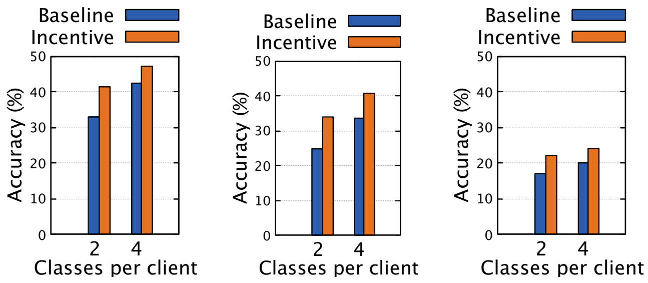
The Figures 3b and 3c show average participating rounds of malicious and normal providers. Generally, rounds numbers increases as number of providers increases, because more numbers of clients are chosen for training a single round. In the case of malicious providers, the baseline shows average rounds between 20.2 and 28.4, whereas TIFF shows average rounds between 12.0 and 16.9, reducing average rounds of malicious provider by up to 40.9%. In the case of normal providers, the baseline shows average rounds between 19.9 and 27.8, while TIFF shows average rounds between 23.4 and 32.8, providing improvement of up to 17.8%, as compared with the baseline approach.

The Figures 3d and 3e show average tokens of malicious and normal providers. In the baseline case, there is not a large difference between malicious and normal providers: malicious providers have average tokens between 916 and 929, and normal providers have average tokens between 1030 and 1035. On the other hand, our incentive work shows up to 48% of difference between malicious and normal providers: malicious providers have average tokens between 750 and 799, and normal providers have average tokens between 1085 and 1106. Average tokens of malicious providers is reduced by up to 18.1%, while average tokens of normal providers is increased by up to 6.9%. This result means that our incentive approach provides more incentives to high-quality providers than low-quality providers, regardless of the number of per-round providers.

We evaluate the scheduling and calculation overhead by comparing total completion time of the baseline and the incentive cases. The average runtime of the baseline approach is 3495 seconds, while the average runtime of the incentive approach is 3596 seconds. It means that our incentive approach causes little computation overhead such as 2.9%.

Therefore, TIFF selects high-quality providers frequently giving more tokens as incentives than low-quality providers through accuracy-based selection and random exploration, which allows reasonable rewards and improves model performance of the trained global model.

We evaluate the impact of varying class distribution per client on final model accuracy, and show the result in Figure 4. We distribute the dataset to clients so that each client has equal number of images from 2 classes and 4 classes, respectively. We train the CNN model on MNIST dataset and 14 providers are chosen per round. As we expected, 4-class per client shows higher accuracy than 2-class per client. This is because different non-IID distribution affects final model accuracy [11]. Higher data heterogeneity (i.e., 2-class per client) causes more accuracy drop. The Figure 4a shows accuracy results with 10% noise level. In the case of 2-class per client, our token-based incentive approach improves final accuracy by 25.8%, while in the case of 4-class per client, the incentive approach improves final accuracy by 11.5%, as compared with baseline. Similarly, as shown in Figures 4b and 4c, 2-class per client with 20% and 30% noise levels shows better improvement compared to 4-class per client. Overall, our incentive approach is effective regardless of the degree of non-IID-ness. While we do not present results in this paper, we observe similar patterns in terms of participating rounds and tokens of both malicious and normal provider groups.



(a) 10% noise. (b) 20% noise. (c) 30% noise.

Fig. 4: Accuracy with varying class distribution per client on MNIST.

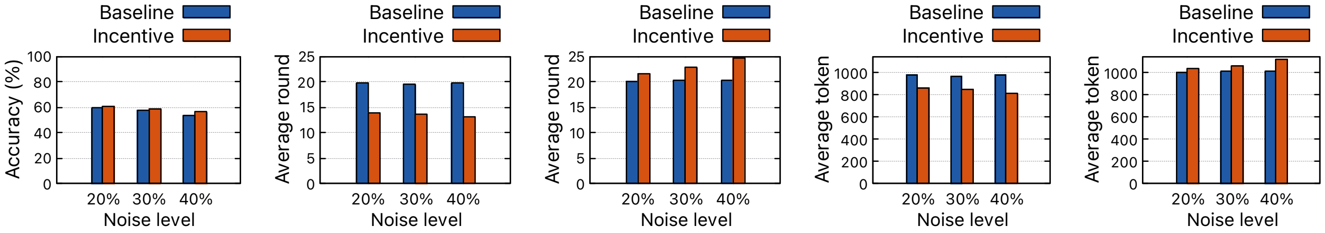
C. *Results for CIFAR-10 dataset*

We evaluate our proposed incentive mechanism to check if the mechanism can be applied to other dataset. Figure 5 shows the final model accuracy, the average round of malicious and normal providers, and the average token of malicious and normal providers with different noise levels. The model is trained with different noise levels of data providers: 20%, 30%, and 40% of malicious providers. As we did on MNIST dataset, we run 100 rounds of training on CIFAR-10 and measure the model accuracy, rounds and tokens of each provider, and then calculate the average values of rounds and tokens in both provider groups (i.e., malicious and normal providers).

As shown in Figure 5a, the baseline shows final model accuracy of 59.5%, 57.5%, and 53.4%, whereas TIFF mechanism shows final model accuracy of 61.0%, 59.0%, and 56.3%, which improves the final accuracy by 2.5%, 2.6%, and 5.4%, respectively, as compared to the the baseline. As we expected, noise level has negative effects on the final accuracy, because it is possible that poor quality of dataset can be chosen more frequently when noise level increases. However, since TIFF selects clients based on local accuracy of each client, it shows more improvement when noise level increases.

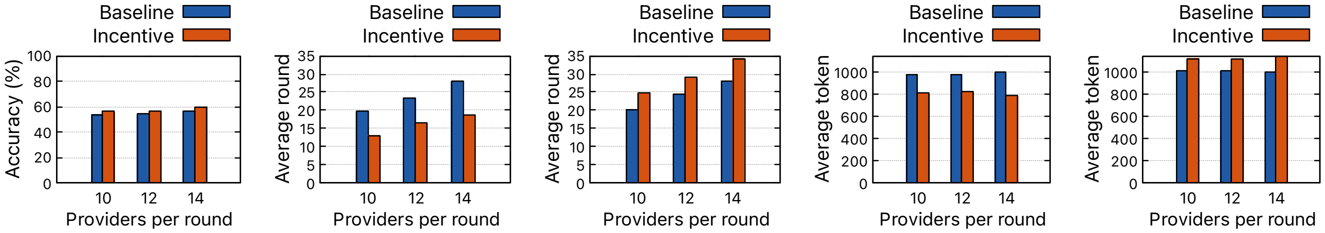
We measure average participating rounds of malicious and normal providers as Figures 5b and 5c. In the case of malicious providers, the baseline shows average rounds between 19.5 and 19.7, whereas TIFF shows average rounds between 13.0 and 13.9, reducing average rounds number of malicious providers by up to 34.0%. In the case of normal providers, the baseline shows average rounds between 20.0 and 20.2, while our incentive scheme shows average rounds between 21.5 and 24.7, providing improvement of up to 22.1%.

The Figures 5d and 5e show average tokens of malicious and normal providers. In the baseline case, there is not a large difference between malicious and normal providers:



(a) Accuracy. (b) Round (Malicious). (c) Round (Normal). (d) Token (Malicious). (e) Token (Normal).

Fig. 5: Comparison results for different noise levels on CIFAR-10.



(a) Accuracy. (b) Round (Malicious). (c) Round (Normal). (d) Token (Malicious). (e) Token (Normal).

Fig. 6: Comparison results for different provider numbers on CIFAR-10.

malicious providers have average tokens between 973 and 985, and normal providers have average tokens between 1004 and 1012. On the other hand, our incentive work shows difference between malicious and normal providers: malicious providers have average tokens between 813 and 863, and normal providers have average tokens between 1034 and 1124. Thus, TIFF scheme provides more tokens to normal providers than malicious providers, by choosing normal providers more frequently according to data quality.

The Figure 6a shows accuracy results with different numbers of providers: 10, 12, and 14. The baseline reaches final model accuracy of 53.4%, 54.2%, and 57.0%, while TIFF reaches final model accuracy of 56.3%, 56.4%, and 60.8%, achieving improvement of up to 5.4%.

The Figures 6b and 6c show average participating rounds of malicious and normal providers. As we expected, the number of participating rounds increases as the number of providers selected per round increases. In the case of malicious providers, the baseline shows average rounds between 19.7 and 28.0, whereas our incentive scheme shows average rounds between 13.0 and 18.6, reducing average rounds of malicious provider by up to 34.0% (i.e., 10 providers). In the case of normal providers, the baseline shows average rounds between 20.2 and 28.0, while our incentive scheme shows average rounds between 24.7 and 34.3, providing improvement of up to 22.4% (i.e., 14 providers).

The Figures 6d and 6e show average tokens of malicious and normal providers. In the baseline case, there is not a large difference between malicious and normal providers: malicious providers have average tokens between 985 and 999, and normal providers have average tokens between 1000 and 1012. On the other hand, our incentive work shows meaningful differences between malicious and normal providers: malicious providers have average tokens between 789 and 813, and normal providers have average tokens between 1119 and 1140. This result suggests that our incentive approach guarantees incentives to high-quality providers than low-quality providers, regardless of the number of per-round providers.

V. Related Work

Some works have proposed to optimize performance in terms of model accuracy and training time for federated learning (FL), which mainly focuses on heterogeneity in data distribution and system resources. McMahan et al., who first termed FL, suggest FederatedAveraging (FedAvg) using iterative model averaging [9]. Google focuses on reducing client-to-server communication overheads leveraging lossy compression [33]. There have been works for FL scheduling to address heterogeneity. To address resource heterogeneity of FL systems, FedCS [34] drops slow devices to minimize straggler problems. Bonawitz et al. propose large-scaled FL systems, which selects more clients (e.g., 130%) than the required number of clients and throws out slow clients [12]. Some researchers start to study on both resource and data heterogeneity for improving performance of FL. FedProx [13] addresses resource heterogeneity by updating partial results and data heterogeneity by adding a proximal term to provide stable convergence. TiFL [11] suggests tier-based parties according training latency to reduce straggler effect and introduces credits to avoid overfitting. Oort [24] suggests guided participants selection based on training latency and importance sampling to provide trade-off between resource and data heterogeneity. However, the above-mentioned works assume that participating clients agreed to share their local data and local resources voluntarily without reasonable incentives.

Recently, a few works on FL incentive have been proposed. FAIR [5] suggests a quality-aware incentive mechanism to motivate participation of high-quality clients. An incentive mechanism for public goods feature is introduced to maximize the social welfare and achieve budget balance [16]. Another work [15] proposes a context-aware incentive mechanism to promote participation of high-quality data owners by reducing temporal mismatch between contributions and rewards. Above-mentioned works usually focus on how to give incentives to *only providers* based on quality estimation, but do not consider how to encourage sustained long-term partitions as perspectives of *both consumers and providers*. Furthermore, these works require direct monetary transfer between parties, however, there may be some delays before the federation has enough budget for payment back to each client [15]. Unlike the above-mentioned studies, our work introduces a token-based incentive mechanism to provide instant monetization through our own token-based pricing model. Our work additionally suggests reimbursement to promote participation of parties as a consumer perspective, as well as a provider perspective.

VI. Conclusion

In this study, we have proposed TIFF, a novel tokenized incentive mechanism for federated learning (FL) that uses tokens to monetize the contribution of participating clients and the training infrastructure, which effectively motivates the long-term participation of high-quality data providers. Unlike existing studies, we introduce incentivization to both providers and consumers and profiles data quality using accuracy measurement without additional overheads, instead of using loss measurement. Through our newly proposed metrics (i.e., token reduction ratio and utility improvement ratio) based on utility measurement, clients are reimbursed as consumers. Through historical accuracy records and random exploration, high-quality clients are frequently selected as providers with reasonable rewards, while preventing overfitting. The result shows that TIFF approach reduces the number of rounds and tokens of malicious providers by up to 40.9% and 18.1%, while increasing the number of rounds and tokens of normal providers by up to 17.8% and 6.9%, as compared with the baseline. Thus, TIFF shows up to 48% difference of tokens between normal and malicious providers, resulting in improvement of the final accuracy by up to 7.4%, as compared with the baseline.

Acknowledgment

This work is sponsored in part by the NSF under the grants: CSR-2106634, CCF-1919113, OAC-2004751, and CSR-1838271.

References

[1] R. Xu, N. Baracaldo, Y. Zhou, A. Anwar, and H. Ludwig, "Hybridalpha: An efficient approach for privacy-preserving federated learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 2019, pp. 13–23.

[2] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning." in *Proc. USENIX OSDI*, 2016.

[3] T. Chilimbi, Y. Suzue, J. Apacible, and K. Kalyanaraman, "Project adam: Building an efficient and scalable deep learning training system," in *Proc. USENIX OSDI*, 2014, pp. 571–582.

[4] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang *et al.*, "Large scale distributed deep networks," *NIPS*, vol. 25, pp. 1223–1231, 2012.

[5] Y. Deng, F. Lyu, J. Ren, Y.-C. Chen, P. Yang, Y. Zhou, and Y. Zhang, "Fair: Quality-aware federated learning with precise user incentive and model aggregation," in *IEEE INFOCOM*, 2021.

[6] S. Sharma, A. M. Alam, and K. Chen, "Image disguising for protecting data and model confidentiality in outsourced deep learning," in *Proc. IEEE CLOUD*, 2021.

[7] G. D. P. Regulation, "General data protection regulation (gdpr)," *Intersoft Consulting, Accessed in October*, vol. 24, no. 1, 2018.

[8] A. Act, "Health insurance portability and accountability act of 1996," *Public law*, vol. 104, p. 191, 1996.

[9] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, 2017, pp. 1273–1282.

[10] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *Proc. ICLR*, 2020.

[11] Z. Chai, A. Ali, S. Zawad, S. Truex, A. Anwar, N. Baracaldo, Y. Zhou, H. Ludwig, F. Yan, and Y. Cheng, "Tifl: A tier-based federated learning system," in *Proc. ACM HPDC*, 2020, pp. 125–136.

[12] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečný, S. Mazzocchi, B. McMahan *et al.*, "Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, 2019.

[13] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.

[14] Y. Zhan, J. Zhang, Z. Hong, L. Wu, P. Li, and S. Guo, "A survey of incentive mechanism design for federated learning," *IEEE Transactions on Emerging Topics in Computing*, 2021.

[15] H. Yu, Z. Liu, Y. Liu, T. Chen, M. Cong, X. Weng, D. Niyato, and Q. Yang, "A sustainable incentive scheme for federated learning," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 58–69, 2020.

[16] M. Tang and V. W. Wong, "An incentive mechanism for cross-silo federated learning: A public goods perspective," in *IEEE INFOCOM*, 2021.

[17] S. R. Pandey, N. H. Tran, M. Bennis, Y. K. Tun, A. Manzoor, and C. S. Hong, "A crowdsourcing framework for on-device federated learning," *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 3241–3256, 2020.

[18] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-iid data," *arXiv preprint arXiv:1806.00582*, 2018.

[19] X. Gong and N. Shroff, "Incentivizing truthful data quality for quality-aware mobile data crowdsourcing," in *Proc. ACM MobiHoc*, 2018.

[20] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing," in *Proc. ACM MobiCom*, 2012, pp. 173–184.

[21] L. U. Khan, S. R. Pandey, N. H. Tran, W. Saad, Z. Han, M. N. Nguyen, and C. S. Hong, "Federated learning for edge networks: Resource optimization and incentive mechanism," *IEEE Communications Magazine*, vol. 58, no. 10, pp. 88–93, 2020.

[22] R. B. Myerson, *Game theory*. Harvard university press, 2013.

[23] L. Jia, F. Yao, Y. Sun, Y. Xu, S. Feng, and A. Anpalagan, "A hierarchical learning solution for anti-jamming stackelberg game with discrete power strategies," *IEEE Wireless Communications Letters*, vol. 6, no. 6, pp. 818–821, 2017.

[24] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient federated learning via guided participant selection," in *Proc. USENIX OSDI*, 2021.

[25] H. Ludwig, N. Baracaldo, G. Thomas, Y. Zhou, A. Anwar, S. Rajamoni, Y. Ong, J. Radhakrishnan, A. Verma, M. Sinn *et al.*, "Ibm federated learning: an enterprise framework white paper v0. 1," *arXiv preprint arXiv:2007.10987*, 2020.

[26] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *NIPS*, vol. 32, pp. 8026–8037, 2019.

[27] N. Ketkar, "Introduction to keras," in *Deep learning with Python*. Springer, 2017, pp. 97–111.

[28] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[29] G. H. Alex Krizhevsky, Vinod Nair, "CIFAR-10 dataset," *https://www.cs.toronto.edu/ kriz/cifar.html*, 2014.

[30] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Handwritten digit recognition with a back-propagation network," *NIPS*, vol. 2, 1989.

[31] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[32] J. Yao, Z. Dou, and J.-R. Wen, "Fedps: A privacy protection enhanced personalized search framework," in *Proc. WWW*, 2021, pp. 3757–3766.

[33] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[34] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE ICC*, 2019.