Data-Free Evaluation of User Contributions in Federated Learning

Hongtao Lv[*], Zhenzhe Zheng[*], Tie Luo[†], Fan Wu[*], Shaojie Tang[‡], Lifeng Hua[§], Rongfei Jia[§], Chengfei Lv[§]

[*]Shanghai Jiao Tong University [†]Missouri University of Science and Technology

[‡]University of Texas at Dallas [§] Alibaba Group

Email: {lvhongtao, zhengzhenzhe}@sjtu.edu.cn, tluo@mst.edu, wu-fan@sjtu.edu.cn,

shaojie.tang@utdallas.edu, {issac.hlf, rongfei.jrf, chengfei.lcf}@alibaba-inc.com

Abstract—Federated learning (FL) trains a machine learning model on mobile devices in a distributed manner using each device's private data and computing resources. A critical issues is to *evaluate individual users' contributions* so that (1) users' effort in model training can be compensated with proper incentives and (2) malicious and low-quality users can be detected and removed. The state-of-the-art solutions require a representative test dataset for the evaluation purpose, but such a dataset is often unavailable and hard to synthesize. In this paper, we propose a method called *Pairwise Correlated Agreement* (PCA) based on the idea of *peer prediction* to evaluate user contribution in FL without a test dataset. PCA achieves this using the statistical correlation of the model parameters uploaded by users. We then apply PCA to designing (1) a new federated learning algorithm called Fed-PCA, and (2) a new incentive mechanism that guarantees truthfulness. We evaluate the performance of PCA and Fed-PCA using the MNIST dataset and a large industrial product recommendation dataset. The results demonstrate that our Fed-PCA outperforms the canonical FedAvg algorithm and other baseline methods in accuracy, and at the same time, PCA effectively incentivizes users to behave truthfully.

Index Terms—Peer prediction, correlated agreement.

I. INTRODUCTION

Recent years have seen a large variety of mobile applications that have changed or improved our ways of communication, shopping, commuting, traveling, and lifestyle. To provide personalized services in these applications, machine learning techniques have been increasingly adopted. For that purpose, a common practice is to upload user data to a central server or cloud, which then trains a machine learning model to make predictions such as product recommendation. This centralized approach evokes many privacy concerns as users' sensitive data could be eavesdropped by malicious parties during data transmission, or be misused by an untrusted server. To address this privacy issue, a new learning paradigm called *Federated Learning* (FL) [1] was proposed to perform distributed machine learning over a large number of devices

without requiring data to leave the devices or data owners. In the training process, each user trains a local model using her own data on her own device, and uploads the local model parameters instead of the original data to the server. The server then aggregates the received models into a global model and distributes the global model back to the users. The above steps repeat until the global model converges. By doing so, federated learning preserves user privacy and reduces network traffic. As a result, it has attracted substantial attention from both academia and industry recently.

Unlike traditional distributed machine learning [2], in which the machines for model training are fully controlled by a central server, federated learning works with autonomous mobile users who decide by themselves whether and how to participate in model training. This makes FL vulnerable to selfish and malicious users who may manipulate the training process such as falsifying the model parameters or send random models without any training effort. Therefore, a critical issue in FL is to *evaluate individual users' contributions* in the model training process so that strategic users can be detected and truthful users can receive rewards proportional to their real contributions.

A popular approach, as recently proposed in [3]–[5], uses the *Shapley value* to measure user contribution in federated learning. This approach computes the marginal increase of average accuracy of the model due to the addition of data points contributed by a user in model training. However, this has two major drawbacks: 1) computing the Shapley value involves permutation operation which has an exponential computational complexity; even though some approximate methods have been proposed, the computation is still costly [4]; 2) more importantly, it relies on a representative test dataset to evaluate the model accuracy, but such a dataset is rarely available because no one knows which dataset perfectly mimics the distribution of future unseen data.

To overcome these issues, we propose a *data-free* approach to evaluate user contribution in FL. This approach only uses users' uploaded models and does not need any extra training or test data. However, there are several key challenges. First, the uploaded models (typically neural networks) have complex structures and the correlation among model parameters is too intricate to express. Second, there may exist dishonest

or malicious users who falsify their data or even directly manipulate model parameters [6], and detecting such behavior is hard because the server does not have access to user data. These challenges set the problem of evaluating user contribution in FL distinct from the data quality evaluation problem in mobile crowdsensing [7] and crowdsourcing [8].

To this end, we propose a *Pairwise Correlated Agreement* (PCA) method to evaluate user contribution in FL. The basic idea is that, although the uploaded models are different among mobile users due to their non-i.i.d. (non-independent and identically distributed) data [9], we can still extract certain internal correlation between the models *pairwise*, and exploit the correlation using an idea based on *peer prediction* [10]. More specifically, we characterize user contribution by how much a model uploaded by a user can predict the models uploaded by the others via the internal correlations.

With our PCA method, we apply it to two fundamental aspects of FL. First, we design a new *model aggregation* algorithm called Fed-PCA, which uses the user contribution evaluation results as the model weights for model aggregation performed by the server. This Fed-PCA algorithm accounts for *quality* of data, rather than just the user-reported data size (quantity); as a result, it offers significant potential to improve the accuracy of the aggregated model, and *counter the falsification of user-reported data size*. Second, we apply PCA to designing an incentive mechanism which is strategy-proof to the following undesirable user behaviors: (i) *free riding*, where a user randomly generates model parameters without performing the actual model training, and (ii) *overly privacy-preserving*: adding excessive noises to model parameters and thus substantially degrading model prediction performance.

In summary, our main contributions are as follows:

- We propose a Pairwise Correlated Agreement (PCA) method to evaluate user contributions in federated model training without using a test dataset. For motivation purposes, we also demonstrate the importance of contribution evaluation by showing the potential harm of strategic user behaviors on federated model training using our experiments conducted on an industrial dataset.
- Based on PCA, we design an algorithm called Fed-PCA which uses the user contributions computed by PCA as the model weights in model aggregation. We also apply PCA to designing a strategy-proof incentive mechanism which resists two types of common strategic behaviors of users: free riding and overly privacy-preserving.
- We conduct extensive experiments with the widely used MNIST dataset and an industrial dataset obtained from Taobao, one of the largest commercial mobile recommender system in China. The results clearly demonstrate the effectiveness of PCA in detecting users' strategic behaviors. In addition, counter-intuitively, our Fed-PCA achieves equal or even better prediction accuracy as compared to FedAvg on different datasets, without knowing the training data size of each user.

II. PRELIMINARIES

A. Federated Learning

In the canonical FL framework [1], there is a set of users (clients) U and a central server. Each user i has a private local dataset D^i, such as the historical dataset of click behaviors related to goods or videos in mobile recommender systems. In each round t, k users are randomly chosen to participate in the model training, and are denoted by the set $\{1, ..., k\}$. At the beginning of round t, the server sends the global model M_{t-1} to the selected k users, and each user trains a new local model M_t^i using her own data D^i, and uploads the model update $x_t^i = M_t^i - M_{t-1}$ to the server. Besides the model update, each user also reports n_t^i, the size of training data used at round t. The server then calculates the weight of each user i as $w_t^i = n_t^i / \sum_{j=1}^{k} n_t^j$, and aggregates the model updates by $\overline{x}_t = \sum_{i=1}^{k} w_t^i x_t^i$. Finally, the global model is updated as $M_t = M_{t-1} + \overline{x}_t$, which is sent back to users. The above process repeats until a stopping condition (e.g. a certain number of rounds) is met. As model compression with the quantization technique (e.g., [11]) is widely used in FL to reduce communication cost, we take each parameter of the model, indexed by p, to be discrete and finite without loss of generality, i.e., $x_{t,p}^i \in \{1, 2, ..., h\}$. Note, however, that our proposed approach can be used for continuous parameter values as well, in which case we can perform an additional quantization of the model update on the server only for the contribution evaluation phase, and it does not affect the model aggregation (i.e., the model update of each user i does not need to be quantized in the model aggregation).

B. Undesirable User Strategies

In an ideal FL framework, each user would participate in the model training, upload her model truthfully. However, there may exist strategic users who can manipulate this process to achieve their own interests, e.g., by uploading a fake model update $\hat{x}^i \neq x^i$ where x^i is the true model update (omitting subscript t for notation simplicity). We account for two main types of strategic user behaviors in FL: 1) free riding [6], which generates random model parameters without actually training to save training costs, such as computing power and storage; and 2) overly privacy-preserving, which adds excessive noises to the model parameters for privacy protection [11]–[13]).

To observe the effects of these strategic behaviors on the model training in FL, we have conducted two experiments on Deep Interest Network (DIN) [14], a deep learning-based click-through rate (CTR) prediction model, with an industrial dataset from Taobao. We adopt the classic performance metric in machine learning: area under the curve (AUC), and the baseline of AUC is 0.5. The detailed experimental setup can be found in Section V-B. In Fig. 1(a), we investigate the effect of free riders, where a free rider randomly generate her model parameter $\hat{x}_p^i \sim \mathcal{N}(0, \sigma_1^2)$, i.e., the Gaussian distribution with variance $\sigma_1 = 0.01$. As shown in Fig. 1(a), just 25% free riders are enough to degrade the performance substantially. In Fig.
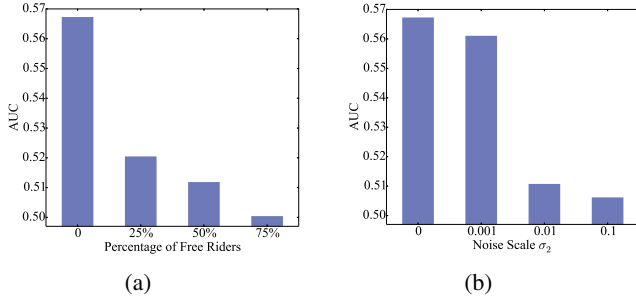
Fig. 1: Performance in terms of area under the curve (AUC) with two types of strategic users: (a) free riders in different percentages, (b) privacy-preserving users in different noise scales σ_2.

1(b), we investigate the performance with privacy-preserving behaviors. Each user further strengthen the privacy by adding noises into the model parameters, i.e., $\hat{x}_p^i = x_p^i + \mathcal{N}(0, \sigma_2^2)$, where $\mathcal{N}(0, \sigma_2^2)$ is Gaussian distribution with variance (noise scale) σ_2. The result in Fig. 1(b) shows that the AUC depends heavily on the noise scale σ_2: when $\sigma_2 = 0.001$, the performance has a slight degradation, but when $\sigma_2 = 0.01$ or larger, the AUC drops significantly, which demonstrates the harmfulness of overly privacy-preserving behaviors (due to excessive noise). With these observations, a contribution evaluation method is highly needed to detect and suppress the free riding behaviors and overly privacy-preserving behaviors.

To formulate the above strategic behaviors in a unified manner, we define the strategy of a user as $F_{r,a} = P(\hat{x}_p = r | x_p = a)$ for any $r, a \in \{1, 2, ..., h\}$ and parameter p, where r is the reported parameter value and a is the true value.[1] In other words, the strategy is the probability of reporting r when the value of parameter p is a. Next, we formulate the above behaviors as informed and uninformed strategies.

Definition 1 (Informed and Uninformed Strategies). *A strategy is an uninformed strategy if it has $F_{r,a} = F_{r,b}$ for any $r, a, b \in \{1, 2, ..., h\}$, otherwise is an informed strategy if there exists $F_{r,a} \neq F_{r,b}$ for some r, a, b, it is an informed strategy.*

Intuitively, the uninformed strategy captures the free-rider behaviors, where the user does not spend resources or effort in training the model, but just report some random parameter values. Hence, the probability of the reported values does not depend on the true values. On the other hand, the informed strategy captures the privacy-preserving behaviors where the user has indeed trained the local model using her local data, but then obfuscated the true parameter value x_p^i into a reported value $\hat{x}_p^i$ with some probability. Note that truthfully reporting model updates is also an informed strategy, where $F_{a,a} = 1$ and $F_{r,a} = 0, \forall r \neq a$.

[1]Another possibility is to consider the probability conditional on parameter index p. However, it is usually not possible for a user to figure out the relation between a particular parameter in a neural network and her own interest, due to the complex structure of neural network models.

C. Peer Prediction and Correlated Agreement

First introduced in [15], the peer-prediction method is a classic mechanism for information elicitation problems without a ground truth, and has been employed in many scenarios, such as crowdsourcing [16] and peer grading [17]. The key idea of peer prediction is to compare the reported data of user i with that of other users. If their data satisfy some statistical correlation, i.e., the data of user i is predictive of the data of others, then her data is deemed to have a larger contribution, and vice versa. Peer prediction is naturally suitable for FL since there is no ground truth for the contribution evaluation.

For consistency with the terminology used in the peer-prediction literature, we also call each parameter in a FL model, indexed by p, a *task*. Each user fulfills a task by training the parameter with her private local data (using, e.g., the gradient descent algorithm). Note that in FL, the task set of each user is the same since all the users are training the same model. We call the value after performing the training task, x_p^i, a *signal*, whereby each signal belongs to the set $\{1, 2, ..., h\}$.

Next, we define a *delta matrix* Δ, which is an $h \times h$ matrix that captures the correlation between each pair of users on a certain task. We first consider the case of homogeneous user, i.e., the delta matrix is the same for each pair of the users. Thus, we omit the user index for the delta matrix. Let $P(a, b)$ denote the joint probability that one user gets signal a after training and the other user gets signal b on the same parameter, and $P(a)$ and $P(b)$ denote the corresponding marginal probabilities. An entry $\Delta(a, b)$ is then defined as

$$\Delta(a, b) \triangleq P(a, b) - P(a)P(b).$$

If $\Delta(a, b) > 0$, we have that the signals a and b are positively correlated; if $\Delta(a, b) = 0$, we have that a and b are independent; otherwise, they are negatively correlated. In addition, it can be easily verified that the sum of $\Delta(a, b)$ in each row or each column is always 0. We define $Sign(\Delta(a, b)) = 1$ if $\Delta(a, b) > 0$, and $Sign(\Delta(a, b)) = 0$ otherwise. Without loss of generality, we assume that there is at least one element in the matrix that is non-zero.

With the above definitions, we describe the *correlated agreement* (CA) method introduced by [10], [17] in the context of FL.

Definition 2 (CA Method for Homogeneous Users). *The CA method entails the following steps:*

1) *Randomly divide the parameters into a bonus parameter set M_1 and a penalty parameter set M_2.*
2) *For each user i and each bonus parameter $p \in M_1$, randomly pick a user $j \neq i$ as the peer of i, and randomly choose two different penalty parameters $q, q' \in M_2$ for users i and j, respectively.*
3) *The contribution or quality of parameter p of user i is evaluated by $Q_p^i = S(\hat{x}_p^i, \hat{x}_p^j) - S(\hat{x}_q^i, \hat{x}_{q'}^j)$, where the score matrix is $S = Sign(\Delta)$, and user i's contribution is $Q^i = \frac{1}{|M_1|} \sum_{p \in M_1} Q_p^i$.*

Algorithm 1: ComputeDelta(i, j, A, B)

Input: The focus user i, the peer user j, and the parameter sets A and B.

Output: The delta matrices $\Delta_A^{i,j}, \Delta_B^{i,j}$.

1 **for** *each pair of signals $a, b \in \{1, 2, ..., h\}$* **do**

2     $T_A^{i,j}(a, b) \leftarrow \frac{\sum_{p \in A} \mathbf{1}(\hat{x}_p^i = a, \hat{x}_p^j = b)}{|A|}$.

3     $T_A^i(a) \leftarrow \frac{\sum_{p \in A} \mathbf{1}(\hat{x}_p^i = a)}{|A|}$.

4     $T_A^j(b) \leftarrow \frac{\sum_{p \in A} \mathbf{1}(\hat{x}_p^j = b)}{|A|}$.

5 Repeat Lines 1 - 4 for parameter set B.

6 **for** *each pair of signals $a, b \in \{1, 2, ..., h\}$* **do**

7     $\Delta_A^{i,j}(a, b) \leftarrow T_A^{i,j}(a, b) - T_A^i(a) T_A^j(b)$.

8     $\Delta_B^{i,j}(a, b) \leftarrow T_B^{i,j}(a, b) - T_B^i(a) T_B^j(b)$.

The intuition in Step 3 is that if the reported signals by user i and her peer j on the same parameter p exhibit a positive statistical correlation (e.g., they both report a as in the above example), we give user i a reward of 1. On the other hand, if their signals on two different parameters q and q' are positively correlated, it suggests that the user i may have randomly uploaded an arbitrary signal for each parameter without actual training, so we give her a penalty of -1.

III. Pairwise Correlated Agreement (PCA)

The above CA method assumes that all the users are homogeneous, and the delta matrix is the same for all the user pairs. However, one distinctive feature of FL is that users are heterogeneous with non-i.i.d. or imbalanced data [1]. For instance, in mobile recommender systems, the click behaviors of users can be substantially different. The work [18] extended the CA method to the heterogeneous users setting, which divides users into several groups and computes the delta matrix for each pair of groups. However, when there are a large number of heterogeneous users, which is likely to occur in FL, the number of groups would be quite huge, and then this method suffers from a prohibitive computing complexity for both the clustering procedure and the calculation of delta matrices. To address this issue, we propose a *pairwise correlated agreement* (PCA) method, and explores the underlying internal correlation of signal distributions to evaluate the contribution of heterogeneous users.

Before introducing PCA, we make a reasonable assumption that the signal of different parameters are independent and identically distributed (i.i.d.), which is based on our observation from practical data sets. To validate this assumption, we again conduct two experiments on DIN with an industrial dataset from taobao. We first test the *Spearman's Correlation* [19] between the signals of two randomly chosen parameters to validate the independence hypothesis. Second, we conduct the *Kolmogorov-Smirnov Test* [20] between them to validate the hypothesis of identical distribution. The resulting p-values are 0.185 and 0.343, respectively, and both of them are larger than 0.05, which adequately supports the i.i.d. assumption. We

---

Algorithm 2: Pairwise Correlated Agreement (PCA) Method for Heterogeneous Users

---

Input: The set of selected users K_t, the reported model updates $\hat{x}^i$ of each user $i \in K_t$, and the number of peers m.

Output: The contribution Q^i of each user $i \in K_t$.

1 Randomly divide model parameters into bonus parameter set M_1 and penalty parameter set M_2.

2 **for** *each user $i \in K_t$* **do**

3     $PR^i \leftarrow$ (a set of randomly selected m users in $K_t \setminus i$ as peers).

4   **for** *each peer $j \in PR^i$* **do**

5   Randomly divide parameters into two sets A, B of the same size.

6         $\Delta_A^{i,j}, \Delta_B^{i,j} \leftarrow$ ComputeDelta(i, j, A, B).

7   **for** *each parameter $p \in M_1 \cap A$* **do**

8   Randomly choose two different parameters $q, q' \in M_2 \cap A$.

9             $Q_p^{i,j} \leftarrow$ $Sign(\Delta_B^{i,j}(\hat{x}_p^i, \hat{x}_p^j)) - Sign(\Delta_B^{i,j}(\hat{x}_q^i, \hat{x}_{q'}^j))$.

10   Repeat Lines 7 - 9 for each parameter $p \in M_1 \cap B$ with $\Delta_A^{i,j}$.

11     $Q^i \leftarrow \frac{1}{m|M_1|} \sum_{p \in M_1} \sum_{j \in PR^i} Q_p^{i,j}$.

---

note that this assumption is also used in many related works on sensitivity analysis of neural networks [21]–[23].

Under this assumption, we conduct Algorithm 1 to estimate the delta matrix for each pair of heterogeneous users. As shown later in the proof of Theorem 1, the estimation error could be arbitrarily small with a large number of samples. In algorithm 1, parameters are split into two sets, A, B, of the equal size, and we compute a delta matrix for each parameter set and each pair of user i and her peer j. The function $\mathbf{1}(\cdot)$ in Lines 2 to 4 denotes the indicator function. We use $T_A^{i,j}(a, b)$ as the observed frequency of jointly reporting a, b from users i, j for the parameter set A, and $T_A^i(a)$ and $T_A^j(b)$ as the corresponding marginal probabilities (Lines 1 - 5). Lines 6 - 8 compute the delta matrix of each parameter set A and B.

Then, PCA is given in Algorithm 2. For each focus user i, we randomly choose m peers (Line 3). With a larger m, the accuracy of Q^i could be improved while the expectation remains the same since we would take the average among them. Then, for each parameter set A, B of user i and her peer j, we apply Algorithm 1 to obtain different estimated delta matrices $\Delta_B^{i,j}$ and $\Delta_A^{i,j}$, respectively (Lines 5 - 6). Note that we use swapped statistical correlations, i.e., the delta matrices, to calculate the score matrix Q (Line 9). As such, the score matrix is independent of the reported parameters themselves.[2]

---

[2]To understand this, suppose a signal r occurs only once on parameter p. Then since we compute the delta matrix statistically, the signal r will always receive a reward of 1 on this parameter because the correlation is calculated with itself.

Algorithm 3: Fed-PCA: An improved FL aggregation algorithm using weights calculated by PCA

Input: The set of users U, the initialized model $\boldsymbol{M}_0$, the number of learning rounds T.

Output: The global model $\boldsymbol{M}_T$.

1 **for** *each round* $t \in \{1, 2, ..., T\}$ **do**
2      $K_t \leftarrow$ (a set of k users randomly selected from N).
3   **for** *each user* $i \in K_t$ ***in parallel*** **do**
4   Obtain new local model $\boldsymbol{M}_t^i$ and model update $\boldsymbol{x}^i = \boldsymbol{M}_t^i - \boldsymbol{M}_{t-1}$.
5   Report $\hat{\boldsymbol{x}}^i$ to the server.
6   Get the contribution Q^i for each user $i \in K_t$ by Algorithm 2.
7   **for** *each user* $i \in K_t$ **do**
8          $w^i \leftarrow \exp(\alpha Q^i) / \sum_{j=1}^k \exp(\alpha Q^j)$.
9   **for** *each parameter p in the model* **do**
10          $\overline{x}_p \leftarrow w^i \hat{x}_p^i$.
11      $\boldsymbol{M}_t \leftarrow \boldsymbol{M}_{t-1} + \overline{\boldsymbol{x}}$.

Lastly, we compute the overall contribution of user i's model update in Line 11.

Intuitively, PCA estimates and takes advantage of the correlation between the uploaded models of heterogeneous users.

IV. APPLICATIONS OF PCA

In this section, we describe two potential applications of PCA: weight calculation in model aggregation and incentive mechanism design.

A. Weight Calculation for Model Aggregation

In order to reduce the negative effects of low-quality uploaded models, we leverage the contribution value provided by PCA in model aggregation and propose the improved FL algorithm called Fed-PCA. In particular, Fed-PCA aims to achieve two goals: 1) to prevent falsification of the data size n^i, which is a serious issue in FL since it directly affects the weights in model aggregation; 2) to improve the performance of FL as the traditional model aggregation does not take into account the model/data contribution of users.

As shown in Algorithm 3, we map the value of user i's contribution from the interval $[-1, 1]$ to a non-negative weight $w^i = \exp(\alpha Q^i) / \sum_{j=1}^k \exp(\alpha Q^j)$ in Line 8. Here, we use the exponential function with a controlled parameter α, which determines the variance of user weights in model aggregation. The local models are then aggregated into the global model using the new normalized user weights (Lines 9 - 11). Note that our contribution values could be used in not only the weighted averaging aggregation process as shown here, but also the recent median-based mechanisms proposed to defend Byzantine attacks [24], [25]. We evaluate the performance for both the averaging and median-based aggregation methods in Section V.

B. Incentive Mechanism Design

We next discuss how to prevent the strategic behaviors through incentives, where an essential property for this issue is *incentive compatibility*. We say that incentive compatibility is satisfied when it is in all users' best interests (obtaining the highest rewards) to report their true models, i.e.,

$$U^i(\boldsymbol{x}^i, \{\hat{\boldsymbol{x}}^j\}^{j \neq i}) \geq U^i(\hat{\boldsymbol{x}}^i, \{\hat{\boldsymbol{x}}^j\}^{j \neq i})$$

for any reported $\hat{\boldsymbol{x}}^i$, where U^i denotes the utility obtained by user i, which will be defined later, and $\{\hat{\boldsymbol{x}}^j\}^{j \neq i}$ is the set of reported model updates of all the other users. We emphasize that incentive compatibility could be quite important for FL, because otherwise, users may upload falsified models and the learning problem would be ill-defined.

Next, based on the definition of informed and uninformed strategies in Section II-B, we introduce the concept of informed incentive compatibility and ϵ-informed incentive compatibility for the above defined strategic behaviors. We denote by $\mathbb{I}^i$ and $\{\mathbb{I}\}^{j \neq i}$ the truthful strategy of user i and that of other users, respectively.

Definition 3 (**Informed Incentive Compatibility and ϵ-Informed Incentive Compatibility**). *If for every user i and any informed strategies F, G, and some $\epsilon \geq 0$, we have*

$$U^i(\mathbb{I}^i, \{\mathbb{I}\}^{j \neq i}) \geq U^i(F^i, \{G\}^{j \neq i}) - \epsilon,$$

and for any uninformed strategy F', we have

$$U^i(\mathbb{I}^i, \{\mathbb{I}\}^{j \neq i}) > U^i(F'^i, \{G\}^{j \neq i}),$$

then the mechanism is ϵ-informed incentive compatible. If $\epsilon = 0$, the mechanism is informed incentive compatible.

The definition of ϵ-informed incentive compatibility reflects that (i) if a user adopts an uninformed strategy, her contribution is strictly lower than that of truthful reporting; (ii) if a user adopts an informed (yet still untruthful) strategy, her contribution is at best ϵ more than that of truthful behavior, where ϵ is a small constant number. Therefore, an incentive mechanism with this property can prevent strategic users from uninformed or informed strategies.

Aiming to illustrate the application of PCA in a clearest manner, we assume that the utility of each user is simply her received reward, regardless of other factors, that is, $U^i(\hat{\boldsymbol{x}}^i, \{\hat{\boldsymbol{x}}^j\}^{j \neq i}) = R^i(\hat{\boldsymbol{x}}^i, \{\hat{\boldsymbol{x}}^j\}^{j \neq i})$ where $R^i(\hat{\boldsymbol{x}}^i, \{\hat{\boldsymbol{x}}^j\}^{j \neq i})$ is the reward received by user i. It should be noted that PCA could be easily integrated into most of other existing incentive mechanisms, such as that in [26], [27], and more factors could be taken into account, e.g., the training cost, the privacy cost and the budget balance property.

On top of this simplified utility model of users, we propose that the following simple mechanism could guarantee the ϵ-informed incentive compatibility: allocating a reward proportional to her contribution, i.e., $R^i(\hat{\boldsymbol{x}}^i, \{\hat{\boldsymbol{x}}^j\}^{j \neq i}) = f(Q_i(\hat{\boldsymbol{x}}^i, \{\hat{\boldsymbol{x}}^j\}^{j \neq i}))$ where f could be any positive monotonic increasing function. We posit the following main theorem,
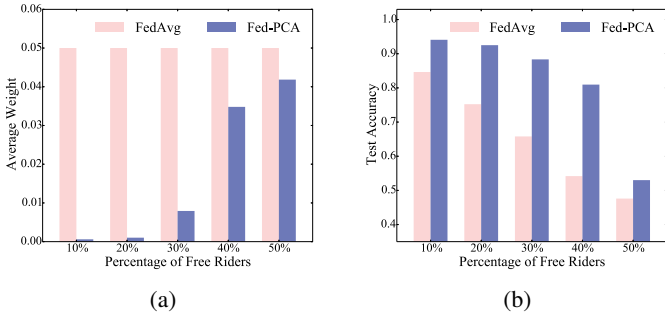
Fig. 2: Results on MNIST with free-riding users: (a) Average weight assigned to free-riding users; (b) Test accuracy.
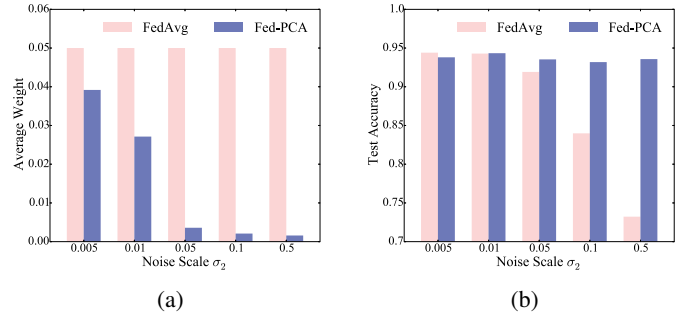
Fig. 3: Results on MNIST with (overly) privacy-preserving users: (a) Average weight assigned to privacy-preserving users. (b) Test accuracy with 25% privacy-preserving users.

which shows that this mechanism is close to informed incentive compatibility.

Theorem 1. *Let $\epsilon > 0$, and $\delta > 0$. If the number of samples is $g = O(9h^2 \log(1/\delta)/\epsilon^2)$, then with a probability of at least $1 - \delta$, the above incentive mechanism with heterogeneous users is ϵ-informed incentive compatible.*

We omit the proof due to space constraint.

V. PERFORMANCE EVALUATION

In this section, we report the evaluation results of our proposed approach and methods (PCA, Fed-PCA, and incentive mechanism) using the MNIST dataset and an industrial product recommendation dataset.

A. Experiments on the MNIST Dataset

Experimental Setting. We first study the performance of Fed-PCA for the MNIST digit-recognition dataset using a multi-layer perceptron with a hidden layer of 100 units. ReLU activations and dropout technique are adopted in the experiments. The network contains a total of 159,010 parameters. Similar to [1], we first sort the MNIST data by digit label, and divide them into 200 shards, each of which includes 300 data items. We assume there are 100 users in total in the FL system, each of which is randomly assigned two shards of data. This way, the data distribution among users is highly non-i.i.d.[3]

For the model updates, we choose a gradient quantization technique called cpSGD [11] to reduce the communication cost. In the FL training, $k = 20$ users are randomly chosen in one round. We adopt mini-batch stochastic gradient descent (SGD) with momentum as the optimization algorithm. The value of momentum is set to 0.5, the batch size is 10, the local epoch number in each round is 5, and the learning rate is 0.01. In the quantization process, we set $h = 8$, and $X^{max} = 0.1$. In PCA, the peer number m is 5, and the number of bonus parameters is $|M_1| = 1,000$. The parameter α, which converts the user contribution into her weight, is set to 10. Regarding the free riders in the system, we assume the parameters they

---

[3]Following [1], we adopt this balanced partition, but in our experiments on the industrial dataset, users are divided naturally in an unbalanced manner.

generate in their model updates to be $\hat{x}_p^i = \mathcal{N}(0, \sigma_1^2)$, where σ_1 is set to 0.01. In the experiments on privacy-preserving users, we assume that 25% of users add noises into their model parameters, each of which reports $\hat{x}_p^i = x_p^i + \mathcal{N}(0, \sigma_2^2)$. We will test the impact of different noise scales σ_2.

Experimental Results. We first compare the performance of our Fed-PCA with the original FedAvg algorithm with different percentages of free riders. For clarity, we illustrate the user contributions using their corresponding weights, as calculated in Algorithm 3. Fig. 2(a) shows the average weight of free riders. When the percentage of free riders is no more than 20%, PCA evaluates their contributions (and hence weights) as nearly 0, while FedAvg constantly assigns them a weight of 0.05 (recall there are 20 users). This result demonstrates that free riders are detected accurately by PCA, while further implying the effectiveness of our incentive mechanism. When the percentage of free riders increases to 50%, their average weight increases accordingly, but is always lower than that of FedAvg. This is because that the model quality of peers decreases with the percentage of free riders. Note that we could assume most users in FL are likely to be truthful, which provides a reference to detect free riders. This is a mild assumption in real life and has been widely adopted in previous literature [6], [28]. In Fig. 2(b), we study the test accuracy with different percentages of free riders. It is depicted in the figure that Fed-PCA decisively outperforms the FedAvg algorithm. The reason is that PCA detects free riders successfully and, thus, assigns them low weights in the model aggregation process. We also note that in the normal case without any free riders or privacy-preserving users, the accuracy of Fed-PCA is 0.9421 after 100 rounds, which is quite close to the baseline of FedAvg: 0.9458.

The impact of the noise scale σ_2 of privacy-preserving users is shown in Fig. 3. Fig. 3(a) explores the average weight of privacy-preserving users. When the noise scale σ_2 is no less than 0.05, the average weight of privacy-preserving users is nearly 0, while a lower noise scale leads to a higher weight. This result demonstrates that PCA is capable of detecting users who add large noises, but ignores the slight noises that do not affect the performance of FL, which is a good characteristic
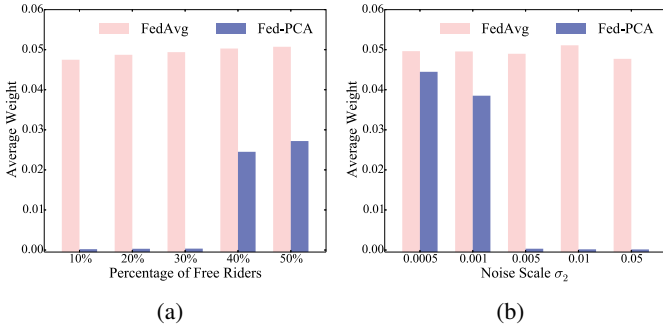
Fig. 4: Results on industrial dataset: (a) Average weight assigned to free-riding users. (b) Average weight assigned to privacy-preserving users.
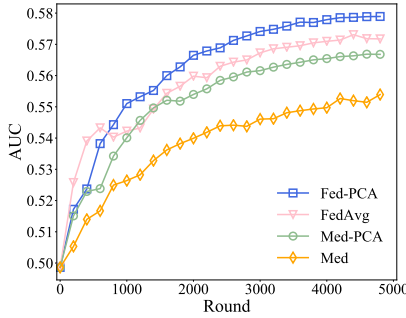

Fig. 5: Comparison of different model aggregation methods.

of the corresponding incentive mechanism. In Fig. 3(b), we show the test accuracy of Fed-PCA and FedAvg. The accuracy of FedAvg degrades seriously with the increase of σ_2, but our Fed-PCA remains nearly unaffected by users with large noises because they are assigned very low weights.

B. Experiments on the Industrial Dataset

Experimental Setting. We next evaluate the performance of our Fed-PCA on an industrial dataset from Taobao, one of the largest a mobile recommendation system of products in China. In the dataset, there are 30-day impressions and click logs of users (dated from June 15 to July 15, 2019). We use the Deep Interest Network (DIN) [14] as the machine learning models. We refer the readers to [14] for more details on DIN and the dataset.

In the training of FL, the batch size is set to 2, the epoch number is set to 1, and the learning rate is initialized as 1.0 with an exponential decay rate of 0.999. In the quantization process, we set $h = 256$, and set X^{max} as 0.1 multiplied by the learning rate in the round. In PCA, the parameter α is set to 100 if not otherwise stated. The other parameters are the same as those of the experiments on the MNIST dataset.

Experimental Results. We first explore the average weight of free riders and privacy-preserving users in Fig. 4. In Fig. 4(a), we can see that when the percentage of free riders is no more than 30%, all of them are detected in PCA and, hence, are allocated merely 0 weight in the aggregation. When the percentage increases to 50%, Fed-PCA still has the ability

to detect them to some extent; thus, their average weight is about half that of truthful users. We investigate the impact of privacy-preserving users in Fig. 4(b). A clear noise scale boundary of 0.001 is found, above which the user would be detected and punished with a low weight. These results clearly suggest the effectiveness of PCA in the application of incentive mechanism design.

Then, we test the AUC with different aggregation methods in the system consisting of exclusively truthful users in Fig. 5. As median-based aggregation methods has attracted much attention in recent years as resistant to the attacks of malicious users [24], [28], we conduct experiments on both averaging and median aggregation. In the figure, the black line (Med) is the performance of the unweighted median-based aggregation method, and the red line (Med-PCA) represents a mix of PCA and the weighted median methods. It is depicted that the unweighted median-based aggregation method underperforms all other approaches since more information of the model updates from the users are lost due to the median operation, compared with the averaging operation. PCA helps alleviate this problem by allocating larger weights to the users with better performance in the median operation. Surprisingly, we can observe that, under the premise of preventing free riders and overly privacy-preserving users, the performance of Fed-PCA clearly surpasses all other aggregation methods, including FedAvg. This means that the contributions and, hence, the weights calculated by PCA are even more reasonable and effective than the traditional weights directly calculated by data sizes, in absence of the information about local datasets. We conjecture that this is because of the difference between the synthetic MNIST dataset and the industrial dataset. In MNIST, each data item has approximately the same value for the learning problem, and hence the data sizes could be used as a good proxy of weights in model aggregation. But in the industrial dataset, real users may provide many useless data items for the learning problem, such as unintended activations. Therefore, a well-designed weight assignment approach has the potential to evaluate the true contribution of each user, and hence to outperform the traditional FL algorithm.

VI. Related Work

The FL framework was first introduced by Google [1], [29] as a new paradigm of distributed machine learning, and it has been applied in a virtual keyboard named Gboard [30], [31]. Some recent works have taken incentive mechanisms of FL into account [3], [4], [26], [27], [32], [33]. For example, [34] proposes an incentive mechanism for FL using the contract theory, by considering the computation and communication costs of training the model. These studies focused on the costs of users, while the contribution of each user to the FL platform is simplified as a sandbox or a linear function of her cost, which is not so practical in real life. Some other work [3]–[5] account for the contribution using the Shapley value-based techniques. However, as stated above, the computation of Shapley values suffers from high time complexity and the requirement of a representative test dataset in FL, which

poses obstacles for fair contribution evaluation. Closely related works are [35], [36], which adopt peer prediction to compare the output of the updated models of users on the test data, and thus a representative test dataset is still required in their proposed approaches. Our work tackles this problem by leveraging the technique of peer prediction on the model parameters, whereby the contribution of each user is evaluated without either training data or test data.

VII. Conclusion

In this paper, we have proposed a pairwise correlated agreement (PCA) method to evaluate the contributions of users in federated learning, without requiring a test dataset. We have then applied PCA in (1) weight calculation for more robust model aggregation, where we have designed Fed-PCA as a better alternative to FedAvg, and (2) incentive mechanism design for FL. Extensive experiments are conducted using the MNIST dataset and a large industrial product recommendation dataset. The evaluation results demonstrate the effectiveness of our proposed approach in terms of both detecting strategic user behaviors and improving prediction accuracy.

References

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson *et al.*, "Communication-efficient learning of deep networks from decentralized data," in *AISTAT*, 2017.

[2] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "A survey of machine learning for big data processing," *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no. 1, p. 67, 2016.

[3] Y. Liu, S. Sun, Z. Ai, S. Zhang, Z. Liu, and H. Yu, "Fedcoin: A peer-to-peer payment system for federated learning," *arXiv preprint arXiv:2002.11711*, 2020.

[4] R. Jia, D. Dao, B. Wang, F. A. Hubis, N. Hynes, N. M. Gurel, B. Li, C. Zhang, D. Song, and C. Spanos, "Towards efficient data valuation based on the Shapley value," *arXiv preprint arXiv:1902.10275*, 2019.

[5] G. Wang, C. X. Dang, and Z. Zhou, "Measure contribution of participants in federated learning," in *Proceedings of 2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2019, pp. 2597–2604.

[6] J. Lin, M. Du, and J. Liu, "Free-riders in federated learning: Attacks and defenses," *arXiv preprint arXiv:1911.12560*, 2019.

[7] T. Luo, J. Huang, S. S. Kanhere, J. Zhang, and S. K. Das, "Improving IoT data quality in mobile crowd sensing: A cross validation approach," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5651–5664, Jun. 2019.

[8] X. Gong and N. Shroff, "Incentivizing truthful data quality for quality-aware mobile data crowdsourcing," in *Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2018, pp. 161–170.

[9] 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.

[10] V. Shnayder, A. Agarwal, R. Frongillo, and D. C. Parkes, "Informed truthfulness in multi-task peer prediction," in *Proceedings of the 2016 ACM Conference on Economics and Computation (EC)*. ACM, 2016, pp. 179–196.

[11] N. Agarwal, A. T. Suresh, F. X. X. Yu, S. Kumar, and B. McMahan, "cpSGD: Communication-efficient and differentially-private distributed SGD," in *Proceedings of the 2018 Advances in Neural Information Processing Systems (NIPS)*, 2018, pp. 7564–7575.

[12] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2016, pp. 308–318.

[13] Z. Bu, J. Dong, Q. Long, and W. J. Su, "Deep learning with Gaussian differential privacy," *arXiv preprint arXiv:1911.11607*, 2019.

[14] G. Zhou, X. Zhu, C. Song, Y. Fan, H. Zhu, X. Ma, Y. Yan, J. Jin, H. Li, and K. Gai, "Deep interest network for click-through rate prediction," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*. ACM, 2018, pp. 1059–1068.

[15] N. Miller, P. Resnick, and R. Zeckhauser, "Eliciting informative feedback: The peer-prediction method," *Management Science*, vol. 51, no. 9, pp. 1359–1373, 2005.

[16] E. Kamar and E. Horvitz, "Incentives for truthful reporting in crowdsourcing," in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*. IFAAMAS, 2012, pp. 1329–1330.

[17] A. Dasgupta and A. Ghosh, "Crowdsourced judgement elicitation with endogenous proficiency," in *Proceedings of the 22nd international conference on World Wide Web*. ACM, 2013, pp. 319–330.

[18] A. Agarwal, D. Mandal, D. C. Parkes, and N. Shah, "Peer prediction with heterogeneous users," in *Proceedings of the 2017 ACM Conference on Economics and Computation (EC)*. ACM, 2017, pp. 81–98.

[19] C. Spearman, "The proof and measurement of association between two things." 1961.

[20] F. J. Massey Jr, "The kolmogorov-smirnov test for goodness of fit," *Journal of the American statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.

[21] S. W. Piche, "The selection of weight accuracies for Madalines," *IEEE Transactions on Neural Networks*, vol. 6, no. 2, pp. 432–445, 1995.

[22] H. Soula, G. Beslon, and O. Mazet, "Spontaneous dynamics of asymmetric random recurrent spiking neural networks," *Neural Computation*, vol. 18, no. 1, pp. 60–79, 2006.

[23] S.-S. Yang, C.-L. Ho, and S. Siu, "Computing and analyzing the sensitivity of MLP due to the errors of the iid inputs and weights based on CLT," *IEEE Transactions on Neural Networks*, vol. 21, no. 12, pp. 1882–1891, 2010.

[24] D. Yin, Y. Chen, K. Ramchandran, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," *arXiv preprint arXiv:1803.01498*, 2018.

[25] Y. Chen, L. Su, and J. Xu, "Distributed statistical machine learning in adversarial settings: Byzantine gradient descent," *ACM SIGMETRICS Performance Evaluation Review*, vol. 46, no. 1, pp. 96–96, 2019.

[26] 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 2020 AAAI/ACM Conference on AI, Ethics, and Society (AIES)*, 2020, pp. 393–399.

[27] R. Zeng, S. Zhang, J. Wang, and X. Chu, "Fmore: An incentive scheme of multi-dimensional auction for federated learning in MEC," *arXiv preprint arXiv:2002.09699*, 2020.

[28] L. Muñoz-González, K. T. Co, and E. C. Lupu, "Byzantine-robust federated machine learning through adaptive model averaging," *arXiv preprint arXiv:1909.05125*, 2019.

[29] J. Konečnỳ, 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.

[30] T. Yang, G. Andrew, H. Eichner, H. Sun, W. Li, N. Kong, D. Ramage, and F. Beaufays, "Applied federated learning: Improving Google keyboard query suggestions," *arXiv preprint arXiv:1812.02903*, 2018.

[31] A. Hard, K. Rao, R. Mathews, F. Beaufays, S. Augenstein, H. Eichner, C. Kiddon, and D. Ramage, "Federated learning for mobile keyboard prediction," *arXiv preprint arXiv:1811.03604*, 2018.

[32] N. Ding, Z. Fang, and J. Huang, "Incentive mechanism design for federated learning with multi-dimensional private information," in *2020 18th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT)*. IEEE, 2020, pp. 1–8.

[33] J. Huang, R. Talbi, Z. Zhao, S. Boucchenak, L. Y. Chen, and S. Roos, "An exploratory analysis on users' contributions in federated learning," *arXiv preprint arXiv:2011.06830*, 2020.

[34] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 700–10 714, 2019.

[35] Y. Liu and J. Wei, "Incentives for federated learning: a hypothesis elicitation approach," *arXiv preprint arXiv:2007.10596*, 2020.

[36] J. Weng, J. Weng, H. Huang, C. Cai, and C. Wang, "Fedserving: A federated prediction serving framework based on incentive mechanism," *arXiv preprint arXiv:2012.10566*, 2020.