Incentive Mechanism for Differentially Private Federated Learning in Industrial IoT

Yin Xu, Mingjun Xiao, *Member, IEEE*, Haisheng Tan, *Senior Member, IEEE*, An Liu, *Member, IEEE*, Guoju Gao, *Member, IEEE,* and Zhaoyang Yan

***Abstract*—Federated Learning (FL) is a newly-emerging distributed machine learning paradigm, whereby a server can coordinate multiple clients to jointly train a learning model by using their private datasets. Many researches focus on designing incentive mechanisms in FL, but most of them cannot allow that clients flexibly determine privacy budgets by themselves. In this paper, we propose a privacy-preserviNg InCentive mEchanism (NICE) based on Differential Privacy (DP) and Stackelberg game for FL systems in industrial IoT. First, we design a flexible privacy-preserving mechanism for NICE, in which clients can add a Laplace noise into the loss function according to a customized privacy budget. Under this mechanism, we design two incentive utility functions for the server and clients. Next, we model the utility optimization problems as a two-stage Stackelberg game by seeing the server as a leader and the clients as followers. Finally, we derive an optimal Stackelberg equilibrium solution for the both stages of the whole game. Based on this solution, NICE can make the server and all clients achieve their maximum utilities simultaneously. In addition, we conduct extensive simulations on real-world datasets to demonstrate the significant performance of the proposed mechanism.**

***Index Terms*—Federated learning, Industrial IoT, Privacy preservation, Stackelberg game.**

I. Introduction

The past few years have witnessed the proliferation of intelligent industry by integrating Machine Learning (ML) and industrial Internet of Things (IoT). The success of ML for IoT stems from the availability of big real-time data and enormous computation power [1], [2]. However, in many industrial IoT applications, data might be generated by distributed advanced sensors owned by different individuals or institutes, even involving sensitive information. Moreover, it becomes difficult to aggregate massive data for energy-intensive centralized training owing to the increasing data scale. To reconcile these concerns, Federated Learning (FL), as a compelling eye-catching distributed ML paradigm, is employed which can make multiple parties jointly train an ML model using their private datasets locally.

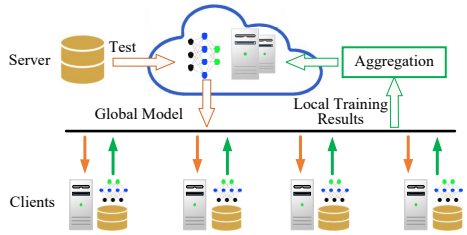


Fig. 1. The Federated Learning Framework

A typical FL system consists of a server and some clients, each of which has a private dataset generated by industrial IoT (called industrial dataset), and the server can coordinate clients to jointly train a global ML model, as shown in Fig. 1. We take driver drowsiness monitoring as an example to expound its advantages. A company intends to train a driver drowsiness detection model to increase road safety. Through shifting the ML training from the centralized cloud to drivers' end devices (e.g., smart in-car cameras), the company can fully leverage drivers' computation resources to unload the training burden. More importantly, each driver locally trains the local model and only transmits the model parameters instead of disclosing their private driving data (e.g., the expression feature and gesture), thereby protecting sensitive data from being eavesdropped upon by malicious attackers. Finally, the company can acquire a global model with the desired accuracy by repeatedly aggregating local models. Currently, many works have been dedicated to different FL issues [3]–[6].

One of the most important issues in industrial IoT applications is how to protect the industrial data privacy of clients from being revealed during the whole ML model training. Many techniques have been employed to solve the problem, including encryption, secure multi-party computation, and Differential Privacy (DP) [7]. Among them, DP attracts much attention since it is a lightweight tool. The most common way to achieve DP is to add noises into industrial datasets according to a certain privacy budget. The smaller the privacy budget, the more the noise needs to be injected, and the higher the privacy protection level. For instance, [8] proposed an anonymous and differentially private FL scheme, in which DP-based Gaussian mechanisms are leveraged to protect the privacy of industrial big data. [9] applied local DP by injecting random noises into the updated local model to protect UAVs' privacy. In most of these works, the privacy budget of each client is preassigned, which implies that the added noises are fixed during the whole process. Consequently, when the privacy budgets are small, the corresponding clients need to add more noises into industrial datasets, producing low-

This research was supported in part by the National Key R&D Program of China (Grant No. 2018AAA0101204), the National Natural Science Foundation of China (NSFC) (Grant No. 62172386, 61572457, 61872330, 61936015, 62102275, 61772489, 61572336), the Natural Science Foundation of Jiangsu Province in China (Grant No. BK20191194, BK20131174, BK2009150, BK20210704, BK20211307), Anhui Initiative in Quantum Information Technologies (Grant No. AHY150300), and the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Grant No. 21KJB520025).

Y. Xu and M. Xiao (*Corresponding author*) are with the School of Computer Science and Technology / Suzhou Institute for Advanced Research, University of Science and Technology of China (USTC), Hefei, P. R. China.

H. Tan is with the LINKE Lab and CAS Key Laboratory of Wireless-Optical Communications, University of Science and Technology of China (USTC), Hefei, P. R. China.

A. Liu and G. Gao are with the School of Computer Science and Technology, Soochow University, Suzhou, P. R. China.

Z. Yan is with P.C. Rossin College of Engineering and Applied Science, Lehigh University, U.S.

accurate local models and making the global model converge very slowly or even fail to converge. Therefore, it is necessary to design a flexible privacy-preserving mechanism to satisfy different privacy demands of clients.

In this paper, we allow clients to flexibly determine their privacy budgets by themselves. Moreover, we endeavor to design an incentive mechanism, which can incentivize clients to select a suitable privacy budget so as to achieve a balance between privacy protection and model accuracy. Many remarkable incentive mechanisms for FL have been proposed [10]–[13], among of which only a few studies have taken preserving privacy into consideration, e.g., [12] designed an incentive mechanism to award clients under DP. However, none of the previous works has addressed how to motivate clients to properly discard some privacy in exchange for higher model accuracies (i.e., more rewards) by customizing privacy budgets. Actually, there are two trade-offs in this problem. On one hand, each client wants to select a smaller privacy budget to improve the privacy protection level. However, the smaller privacy budget means more noise injected into the local model, which leads to a lower local model accuracy. Consequently, the server will pay a less reward to the client. Therefore, there is a trade-off between monetary rewards and privacy protection levels. On the other hand, there is also a trade-off for the server between the total payment to clients and gain from the model accuracies. When the server gives a low payment, clients are unwilling to sacrifice a large amount of privacy and choose small privacy budgets, which will result in low-accurate local models. Designing a privacy-preserving incentive mechanism, which can achieve privacy-flexibility and make a balance of two trade-offs, is much challenging.

To address the above challenges, we propose a privacy-preserviNg InCentive mEchanism (NICE) based on DP and Stackelberg game, which integrates a flexible privacy-preserving mechanism and an incentive mechanism. We first design the flexible privacy-preserving mechanism (Section III). Each client can choose a sequence of proper privacy budgets by itself, according to which the client adds Laplace noises into the loss function so as to achieve the DP property. To determine the privacy budget for each client, we next propose the incentive mechanism (Section IV). Specifically, we design two utility functions for the server and clients, respectively. The incentive problem is modeled as a two-stage Stackelberg game, in which the server is a leader and clients work as followers. We derive the optimal payment strategy for the server and the optimal privacy budget strategy for each client to maximize their utilities and balance their trade-offs, respectively. Additionally, we prove that these optimal strategies constitute a unique Stackelberg equilibrium of the game, based on which the incentive mechanism can incite clients to make an optimal balance between training accurate ML models and preserving the privacy of their local industrial datasets. Overall, our salient contribution is multifold as follows:

- We propose a novel privacy-preserving incentive mechanism, namely NICE, which integrates Stackelberg game and DP skillfully. Different from the existing studies, NICE facilitates the trade-off between monetary rewards and privacy protection levels so as to solve privacy-flexibility and utility maximization for FL systems in industrial IoT.
- We design a flexible privacy-preserving mechanism for sensitive data in industrial IoT, in which each client can decide how much privacy to disclose to the server (indicated by the privacy budget) according to the personalized privacy demands. What's more, we prove that it satisfies the DP property and demonstrate the convergence analysis.
- We model the incentive problem as a two-stage Stackelberg game, and derive the optimal strategies for the server and clients. Additionally, we prove that these optimal strategies form a unique Stackelberg equilibrium, which guarantees that no one will deviate from the optimal strategy and maximizes all parties' utilities simultaneously.
- We conduct extensive simulations on real datasets to corroborate the good performance of NICE.

TABLE I
DESCRIPTION OF MAJOR NOTATIONS

Variable	Description
$i, \mathcal{N}, N$	the index of the client i, the set of all clients, and the number of clients, respectively.
$(\boldsymbol{x_j}, y_j)$	the vector $\boldsymbol{x_j} \in \mathbb{R}^d$ is a training sample with d features, and $y_j \in \mathbb{R}$ is the corresponding label.
D_i, D_s	the dataset of client i and the server.
$\mathcal{M}$	the functionality of the ML model.
$\mathbf{w}_g^t, \mathbf{w}_i^t$	model parameters of the global and local model.
$L(\mathbf{w}_i^t; D_i)$	the loss function with model parameters and data.
θ^t	the accuracy of the t-th round of global model.
Ψ, Γ	the privacy-preserving and incentive mechanism.
Γ^t, Γ_i^t	the total payment and the reward of the i-th client paid by the server in the t-th round.
$\epsilon_i^t, \boldsymbol{\epsilon_{-i}^t}$	the privacy budget, vector $\langle \epsilon_1^t, ..., \epsilon_N^t \rangle$ expect ϵ_i^t.
$\mathcal{U}_i^t, \mathcal{U}^t$	the utility of client i and the server in round t.
c_i^{pv}, c_i^{cp}, c_i^{cm}	the privacy cost, the computation cost, and the communication cost of the i-th client.
ξ_i, ζ	the cost per unit ϵ_i^t and a system parameter.

II. MODEL AND DESIGN GOALS

A. System Model

We consider an FL system for industrial IoT applications with the flexible privacy-preserving mechanism and incentive mechanism, which consists of a server and N clients, denoted by $\mathcal{N} = \{1, 2, \cdots, N\}$. Each client $i \in \mathcal{N}$ has a local private dataset $D_i = \{(\boldsymbol{x_j}, y_j) | j = 1, 2, \cdots, |D_i|\}$ collected by industrial IoT, where the vector $\boldsymbol{x_j} \in \mathbb{R}^d$ is a training sample with d features, $y_j \in \mathbb{R}$ is the corresponding label, and $|D_i|$ is the number of samples. The server has an auxiliary dataset D_s for validation. We divide the whole FL process into many rounds, and let $t = \{1, 2, \cdots\}$ indicate the rounds. In each round, clients use their local data to train local ML models. The server can aggregate these local models to derive a global model and verify its accuracy by using the validation dataset. We denote the global model and the i-th local model in the t-th round as $\mathcal{M}(\mathbf{w}_g^t; \cdot)$ and $\mathcal{M}(\mathbf{w}_i^t; \cdot)$, respectively. Here, $\mathcal{M}$ represents the functionality of the ML model. Matrices $\mathbf{w}_g^t$ and $\mathbf{w}_i^t$ are model parameters, where subscript g means the global model and i indicates the client. The joint training process with NICE in FL systems can be roughly described as the following steps:

1) Initialization: At first, the server constructs an initial global model $\mathcal{M}(\mathbf{w}_g^0; \cdot)$ and broadcasts it to all clients together with the requirements and payment rules. Here,

the superscript $t=0$ means that this is an initial model. In the requirements, the server can specify the approach to be adopted, such as Stochastic Gradient Descent.

2) Privacy-preserving Local Training: Each client i receives the global model $\mathcal{M}(\mathbf{w}_g^{t-1};\cdot)$ of last round from the sever. Then, the client i lets $\mathbf{w}_i^{t-1}=\mathbf{w}_g^{t-1}$ and uses its private industrial dataset D_i to repeatedly train the model $\mathcal{M}(\mathbf{w}_i^{t-1};\cdot)$ according to the requirements specified by the server. As the output, the client will get a local model $\mathcal{M}(\mathbf{w}_i^t;\cdot)$, where t is the current round. Essentially, the local training is to determine a new model parameter matrix which can minimize a predefined loss function:
$$\mathbf{w}_i^t=\arg\min L(\mathbf{w}_i^{t-1};D_i). \quad (1)$$
Here, L is the loss function, including the total error of the training model over all data items in D_i. Note that, the model parameter matrix $\mathbf{w}_i^t$ is derived using the local private data. Thus, in order to prevent any sensitive information from being inferred through $\mathbf{w}_i^t$, the client i adopts a privacy-preserving mechanism to make the local training produce a disguised model parameter matrix $\tilde{\mathbf{w}}_i^t$. Denote the privacy-preserving mechanism as a function Ψ. Then, this process can be represented:
$$\tilde{\mathbf{w}}_i^t=\Psi(\mathbf{w}_i^{t-1};D_i). \quad (2)$$
Next, the client i will send the disguised model parameter matrix $\tilde{\mathbf{w}}_i^t$ back to the server.

3) Incentive Mechanism: The server collects the local training models of all clients, i.e., receiving the disguised model parameter matrix $\tilde{\mathbf{w}}_i^t$ from each client i. Then, the server pays each client i a reward Γ_i^t. The value of Γ_i^t is determined by an incentive mechanism. For simplicity, we denote the incentive mechanism as follows:
$$\Gamma=\{\Gamma_i^t|i\in\mathcal{N},t=1,2,\cdots\},\ \text{and}\ \Gamma^t=\textstyle\sum_{i\in\mathcal{N}}\Gamma_i^t. \quad (3)$$

4) Aggregation and Model Updating: The server aggregates all local model parameters to derive a model parameter matrix based on the following formula:
$$\mathbf{w}_g^t=\textstyle\sum_{i=1}^{N}\tilde{\mathbf{w}}_i^t r_i/\sum_{i=1}^{N} r_i, \quad (4)$$
where r_i is the reliability of the i-th client and can be derived from historical records. Here, as an example, we adopt the simple weighted averaging operation as the aggregation function. Actually, our work can also support other complex aggregation operations. After the aggregation, the server will update the global model as $\mathcal{M}(\mathbf{w}_g^t;\cdot)$. Next, the server evaluates the accuracy of the global model, i.e., $\theta^t=\sum_{(\boldsymbol{x_j},y_j)\in D_s}1_{\{\mathcal{M}(\mathbf{w}_g^t;\boldsymbol{x_j})=y_j\}}/|D_s|$. Here, θ^t is the accuracy of the t-th round of global model, and $1_{\{\cdot\}}$ is an indicator function. Note that we take a classification model as an example and NICE can also apply for other ML models. If the accuracy of the global model converges or is larger than a given threshold, the server will end the process. Otherwise, it will broadcast the updated $\mathbf{w}_g^t$ to all clients to start the next round.

B. Design Goals

In the above model, we aim to design a privacy-preserving incentive mechanism NICE=$\langle\Psi,\Gamma\rangle$, which integrates a flexible privacy-preserving mechanism Ψ and an incentive mechanism Γ under Ψ. Specifically, our objectives are as follows.

First, *we devote to designing a flexible privacy-preserving mechanism Ψ,* which satisfies the differential privacy property. The security model is essentially the privacy-preserving model based on differential privacy, defined as follows:

Definition 1 (Differential Privacy, DP [14])**.** *A randomized mechanism $\mathcal{A}$ has ϵ-differential privacy if for any two input sets D_1 and D_2 differing on at most one element, and for any set of outcomes $O\subseteq Range(\mathcal{A})$, we have $Pr[\mathcal{A}(D_1)\in O]\leq exp(\epsilon)\times Pr[\mathcal{A}(D_2)\in O]$. $\epsilon>0$ is the privacy budget: the smaller ϵ, the stricter protection and lower data availability.*

As long as a mechanism satisfies ϵ-DP, it can possess the DP-based security. Each client i has a privacy budget ϵ_i, which implies the privacy protection level for the sensitive data in industrial IoT (i.e., the quantity of noises to be added for its local model). Note that, the flexibility means that each client is allowed to determine a proper privacy budget by itself in each round of model training. That is, the heterogeneous privacy needs of different clients can be satisfied.

Second, *we aim to propose an incentive mechanism Γ to determine each client's optimal privacy budget (i.e., the parameter ϵ for Ψ) and the server's optimal payment, which can balance the following trade-offs:*

The first trade-off is on the clients' side: monetary rewards vs. privacy protection levels. On one hand, each client wishes to protect local sensitive data with a higher privacy-preserving level, thus it will add more noises (i.e., select a smaller privacy budget) to local ML models; on the other hand, it is unwilling to decrease the local model accuracy due to injecting more noises, resulting in less rewards from the server.

The second trade-off is on the server's side: the total payment to clients vs. gain from the model accuracies. If the server wants to save its cost by reducing Γ_i^t, the client i might be reluctant to bear more risk of privacy leakage and add more noises into its local model. Consequently, the server's gain might become smaller from these low-accurate local models.

In order to balance the two trade-offs, we need an incentive mechanism Γ under Ψ, in which each client holds its optimal privacy budget and the server meets its optimal payment.

III. Privacy-Preserving Mechanism

In this section, we design the flexible privacy-preserving mechanism Ψ, through which each client i can protect the privacy of local sensitive dataset D_i from being revealed. We adopt the DP approach in the design, because it has a relatively small computation and communication cost. The basic idea is to disguise the model parameters $\mathbf{w}_i^t$ by adding Laplace noises into the loss function of training model $L(\mathbf{w}_i^{t-1};D_i)$. Here, we suppose that the privacy budget ϵ_i for each client i is known in advance. For notational simplicity, we omit the superscript of ϵ_i^t. In the next section, we will introduce the detailed method to determine the privacy budget of each client.

A. Mechanism Design

Unlike traditional works, our flexible privacy-preserving mechanism Ψ directly adds Laplace noises into the loss function of training model, not into the model parameters. This is because there is a direct and apparent relationship between the loss function and the model accuracy. The whole DP mechanism mainly includes the following steps:

First, we need to determine the sensitivity of the loss function, which is defined as follows.

Definition 2 (Sensitivity)**.** *For any pair of input datasets D_i and D_i' which differ on at most one data item, the sensitivity δ_L of the loss function $L(\boldsymbol{w}_i^{t-1}; D_i)$ satisfies:*

$$\delta_L = \max_{\langle D_i, D_i' \rangle} |L(\boldsymbol{w}_i^{t-1}; D_i) - L(\boldsymbol{w}_i^{t-1}; D_i')|. \quad (5)$$

According to Definition 2, the sensitivity value is irrelative with the dataset and only relies on the loss function. Different training models and loss functions will have different sensitivity values. Fortunately, J. Zhang et al. and J. Ding et al. introduced a generalized method to derive the sensitivity values of different loss functions of ML models [15]. By using the Stone-Weierstrass theorem [16], any differentiable and continuous loss function can be rewritten as a polynomial about the ML model parameters. Thus, the authors turn the loss function into a polynomial and derive a closed formulation on the sensitivity. Once the loss function is given, the corresponding sensitivity can be determined. In this paper, we assume that the loss function is differentiable and continuous, but do not confine the concrete model for generality. So, we can directly use the above method to know the sensitivity δ_L.

Next, the client i produces the Laplace noise and adds the noise into the loss function. Denote the disguised loss function as $\tilde{L}(\mathbf{w}_i^{t-1}; D_i)$. Then, it can be formulated as follows:

$$\tilde{L}(\mathbf{w}_i^{t-1}; D_i) = L(\mathbf{w}_i^{t-1}; D_i) + Lap(\delta_L/\epsilon_i). \quad (6)$$

Here, $Lap(\delta_L/\epsilon_i)$ is the Laplace noise, which is actually drawn from a Laplace distribution with mean zero and scale δ_L/ϵ_i. The corresponding probability density function is denoted by $f(x) = (\epsilon_i/2\delta_L)\exp((-\epsilon_i/\delta_L)|x|)$.

Finally, since the client i needs to submit the model parameters instead of the loss function to the server after each round of local training, it needs to derive the disguised model parameters from the disguised loss function. This can be represented as $\tilde{\mathbf{w}}_i^t = \arg\min \tilde{L}(\mathbf{w}_i^{t-1}; D_i)$. Based on this, when the client i replaces the loss function L with $\tilde{L}$ in the local training, it can get the disguised model parameters $\tilde{\mathbf{w}}_i^t$.

The above flexible privacy-preserving mechanism Ψ can be formulated as follows.

$$\Psi(\mathbf{w}_i^{t-1}; D_i) = \arg\min \left[L(\mathbf{w}_i^{t-1}; D_i) + Lap(\tfrac{\delta_L}{\epsilon_i})\right]. \quad (7)$$

B. Theoretical Analysis

Theorem 1. *For each client $i \in \mathcal{N}$, the flexible privacy-preserving mechanism Ψ satisfies the ϵ_i-differential privacy, where ϵ_i is the privacy budget.*

Proof. Consider an arbitrary pair of local datasets D_i and D_i' which differ on at most one data item. Let $L(\mathbf{w}_i^{t-1}; \cdot)$ be the loss function with the sensitivity δ_L. Moreover, suppose that l is an arbitrary loss function value. Then, we have:

$$\begin{aligned}
&\frac{Pr[l = L(\mathbf{w}_i^{t-1}; D_i) + Lap(\delta_L/\epsilon_i)]}{Pr[l = L(\mathbf{w}_i^{t-1}; D_i') + Lap(\delta_L/\epsilon_i)]} \\
&= \frac{\frac{\epsilon_i}{2\delta_L}\exp(-\frac{\epsilon_i}{\delta_L}|l - L(\mathbf{w}_i^{t-1}; D_i)|)}{\frac{\epsilon_i}{2\delta_L}\exp(-\frac{\epsilon_i}{\delta_L}|l - L(\mathbf{w}_i^{t-1}; D_i')|)} \\
&= \exp((\epsilon_i/\delta_L)[|l - L(\mathbf{w}_i^{t-1}; D_i')| - |l - L(\mathbf{w}_i^{t-1}; D_i)|]) \\
&\le \exp((\epsilon_i/\delta_L)|L(\mathbf{w}_i^{t-1}; D_i) - L(\mathbf{w}_i^{t-1}; D_i')|) = \exp(\epsilon_i). \quad (8)
\end{aligned}$$

Each client conducts the mechanism Ψ by itself. On one hand, all DP mechanisms (i.e., Ψ) are independent of each other and there is no global DP mechanism. Therefore, the DP mechanisms of different clients do not need to be composed. On the other hand, it is straightforward that Ψ conducted in each round of each client satisfies the Parallel Composition theorem [17], since all batches are disjoint from each other in a training epoch. According to Definition 1, the privacy-preserving mechanism Ψ satisfies the ϵ_i-DP. □

Theorem 2. *When the number of samples $|D_i|$ is large adequately, the difference between the disguised model parameters $\tilde{\boldsymbol{w}}_i^t = \arg\min \tilde{L}(\boldsymbol{w}_i^{t-1}; D_i)$ and the real model parameters $\boldsymbol{w}_i^t = \arg\min L(\boldsymbol{w}_i^{t-1}; D_i)$ is arbitrarily close to 0.*

Proof. We use $\ell(\mathbf{w}_i^{t-1}; \boldsymbol{x_j}, y_j)$ to represent the loss function associated with the data sample $(\boldsymbol{x_j}, y_j)$ and the model parameters. Then the local loss function at the client i can be represented as $L(\mathbf{w}_i^{t-1}; D_i) = \sum_{j=1}^{|D_i|} \ell(\mathbf{w}_i^{t-1}; \boldsymbol{x_j}, y_j)$. Owing to $\tilde{L}(\mathbf{w}_i^{t-1}; D_i) = L(\mathbf{w}_i^{t-1}; D_i) + Lap(\delta_L/\epsilon_i)$, we can get the limit of $\tilde{L}(\mathbf{w}_i^{t-1}; D_i)$ as follows.

$$\begin{aligned}
&\lim_{|D_i|\to\infty} \tfrac{1}{|D_i|}\tilde{L}(\mathbf{w}_i^{t-1}; D_i) = \lim_{|D_i|\to\infty} \tfrac{1}{|D_i|}(L(\mathbf{w}_i^{t-1}; D_i) + Lap(\tfrac{\delta_L}{\epsilon_i})) \\
&= \lim_{|D_i|\to\infty} \tfrac{1}{|D_i|}(\textstyle\sum_{j=1}^{|D_i|} \ell(\mathbf{w}_i^{t-1}; \boldsymbol{x_j}, y_j) + Lap(\delta_L/\epsilon_i)) \\
&= \lim_{|D_i|\to\infty} \tfrac{1}{|D_i|} \textstyle\sum_{j=1}^{|D_i|} \ell(\mathbf{w}_i^{t-1}; \boldsymbol{x_j}, y_j) + \lim_{|D_i|\to\infty} \tfrac{Lap(\delta_L/\epsilon_i)}{|D_i|} \\
&= \lim_{|D_i|\to\infty} \tfrac{1}{|D_i|} \textstyle\sum_{j=1}^{|D_i|} \ell(\mathbf{w}_i^{t-1}; \boldsymbol{x_j}, y_j) + \lim_{|D_i|\to\infty} Lap(\tfrac{\delta_L}{|D_i|\epsilon_i}) \quad (9)
\end{aligned}$$

Since the sensitivity δ_L and the privacy budget ϵ_i are both finite real numbers, there is $\lim_{|D_i|\to\infty} Lap(\frac{\delta_L}{|D_i|\epsilon_i}) = 0$. What's more, the loss function can be rewritten as a polynomial and each $(\boldsymbol{x_j}, y_j)$ is an independent and identically distributed sample, so the limit of the coefficient of the polynomial will approach a constant when $|D_i| \to \infty$. Therefore, there exists $\lim_{|D_i|\to\infty} \frac{1}{|D_i|} L(\mathbf{w}_i^{t-1}; D_i) = \lim_{|D_i|\to\infty} \frac{1}{|D_i|} \tilde{L}(\mathbf{w}_i^{t-1}; D_i)$. According to the definition of $\tilde{\mathbf{w}}_i^t$, we further can get $\tilde{\mathbf{w}}_i^t = \arg\min_{|D_i|\to\infty} \tilde{L}(\mathbf{w}_i^{t-1}; D_i) = \mathbf{w}_i^t$ when $|D_i| \to \infty$. That is to say, $\tilde{\mathbf{w}}_i^t$ will be arbitrarily close to $\mathbf{w}_i^t$ when there are adequate training samples, so that the convergence performance after adopting the mechanism Ψ can be guaranteed. □

From the above security analysis, we have proved that the flexible privacy-preserving mechanism Ψ has a rigorous privacy guarantee, i.e., Ψ satisfies DP property. More importantly, although we inject the Laplace noise into the loss function of the local model, the gap between $\tilde{\mathbf{w}}_i^t$ and $\mathbf{w}_i^t$ is tolerable according to the convergence analysis on Ψ. Therefore, the degree of the global model accuracy degradation can be acceptable, which is consistent with the experimental results.

IV. Incentive Mechanism

In this section, we propose the incentive mechanism Γ to stimulate all parties to participate in the FL system. The server wishes to incentivize all clients to provide accurate local training models with low payments, while the clients wish to obtain high rewards by discarding a small amount of privacy. We model such an incentive problem as a two-stage Stackelberg game, in which the server is seen as a leader and the clients are followers. Moreover, we derive the optimal strategies for all parties based on a unique equilibrium.

A. Two-stage Stackelberg Game Modeling

To formulate the incentive problem, we design two utility functions for the server and clients, which indicate their net profits in each round of model training, respectively. The

server's utility depends on the model accuracy and the total payment. The utility of each client is influenced by the received reward and the costs. The server can adjust the payment while each client can flexibly control its privacy budget. All of them attempt to maximize their own utility values.

Utility Function of Client. The utility of each client i is its income minus the local training costs. Denote the client's utility in the t-th round as $\mathcal{U}_i^t$. The income only includes the reward paid by the server, i.e., Γ_i^t. For the fairness, the rewards of all clients are set to be proportional to their privacy budget values. Thus, we have $\Gamma_i^t = \Gamma^t \epsilon_i^t / \sum_{i\in\mathcal{N}} \epsilon_i^t$.

The local training costs consist of three parts: the privacy cost c_i^{pv}, the computation cost c_i^{cp}, and the communication cost c_i^{cm}. The privacy cost actually refers to the compensation that each client asks for bearing the privacy leakage risk. For the simplicity of presentation and the unification of cost computation, we treat such a compensation as a special kind of cost, which is related to the client's privacy protection level (i.e., the privacy budget ϵ_i), like in [18], [19]. The larger the privacy budget, the higher the privacy leakage risk, and consequently, the larger the privacy cost will be. More specifically, the privacy cost of each client i is proportional to its privacy budget, i.e., $c_i^{pv} = \xi_i \epsilon_i^t$. Here, ξ_i is a positive coefficient determined by each client itself, called unit privacy cost, which represents how much the client cares about its privacy loss. Inspired by [20], [21], the client's computation cost can be written as $c_i^{cp} = |D_i|\delta_l \delta_g M \alpha_i$, and communication cost is $c_i^{cm} = ((2^{R/QH_i} - 1)QH_i M \delta_g \beta_i)/h_i R$, where δ_l, δ_g, R are hyper-parameters, M is the model size, Q is the channel bandwidth, H_i is the number of channels requested by client i, h_i is the normalized channel power gain, α_i is the client's unit computation cost, and β_i is the client's unit communication cost. Hence, the utility of client i is:

$$\mathcal{U}_i^t(\Gamma^t, \epsilon_i^t, \boldsymbol{\epsilon}_{-i}^t) = \Gamma_i^t - c_i^{pv} - c_i^{cp} - c_i^{cm} \tag{10}$$
$$= \frac{\Gamma^t \epsilon_i^t}{\sum_{i\in\mathcal{N}} \epsilon_i^t} - \xi_i \epsilon_i^t - |D_i|\delta_l \delta_g M \alpha_i - \frac{(2^{\frac{R}{QH_i}} - 1)QH_i M \delta_g \beta_i}{h_i R},$$

where $\boldsymbol{\epsilon}_{-i}^t$ denotes the vector $\langle \epsilon_1^t, \epsilon_2^t, ..., \epsilon_N^t \rangle$ expect ϵ_i^t.

Utility Function of Server. The utility of the server is defined as the gain brought by the trained ML model, which is related to the model accuracy, minus the total rewards paid to clients. Let $\mathcal{U}^t$ denote the server's utility in the t-th round.

In general, the gain of the server can be seen as a function on the model accuracy. In this paper, we use the typical logarithmic function as the gain of the server, i.e., $\zeta \cdot \ln(1 + \sum_{i\in\mathcal{N}} \theta_i^t)$, which has been widely used in model designs [9], [22]. Furthermore, we construct the relationship between the model accuracy and privacy budget. We conduct the experiments to measure the model accuracy values under different privacy budgets based on the real-world MNIST dataset [23] and the CIFAR-10 dataset [24], respectively. In detail, the ML models used for training are introduced in Section V-A. As illustrated in Fig. 2, the fitted curves signify that the model accuracy θ_i^t can be regarded as a concave function with respect to the privacy budget ϵ_i^t. For generality, we directly adopt a continuous and reversible concave function $G : \epsilon_i^t \to \theta_i^t$ to indicate the relationship between the privacy budget and the model accuracy. Hence, we can define the utility of the server as follows:

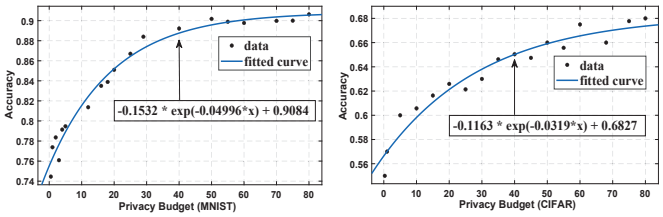


Fig. 2. Model Accuracy vs. Privacy Budget

$$\mathcal{U}^t(\Gamma^t, \epsilon_i^t, \boldsymbol{\epsilon}_{-i}^t) = \zeta(\ln(1 + \textstyle\sum_{i\in\mathcal{N}} G(\epsilon_i^t))) - \Gamma^t, \tag{11}$$

where $\zeta > 0$ is a system parameter.

Optimization Objectives. Both the server and the clients wish to maximize their own utilities in each round of model training. This can be seen as a two-stage Stackelberg game. The server is a leader. It will determine an optimal total payment Γ^t to maximize its utility in the first stage. In the second stage, each client will choose an optimal privacy budget ϵ_i^t for the local model training, so as to maximize its utility under the given payment Γ^t. These clients work as the followers. The whole process can be formulated as follows:

$$\text{Server's side:} \quad Maximize\, \mathcal{U}^t(\Gamma^t, \epsilon_i^t, \boldsymbol{\epsilon}_{-i}^t) \tag{12}$$
$$\text{Client's side:} \quad Maximize\, \mathcal{U}_i^t(\Gamma^t, \epsilon_i^t, \boldsymbol{\epsilon}_{-i}^t) \tag{13}$$

By means of the game, the server can incentivize each client to select its optimal privacy budget. In this way, each client will try to train the accurate local model under the premise that the risk of privacy leakage is acceptable. Consequently, the degree of the global model accuracy degradation will be acceptable.

B. The Optimal Equilibrium Solution

In this subsection, we derive the Stackelberg equilibrium whereby both the server and the clients can determine their optimal strategies to achieve the maximum utilities. The backward induction method is exploited to analyze the game. First, we analyze the second stage game to determine each client's optimal privacy budget under a given payment Γ^t. Next, we turn to the first stage to find the server's best payment Γ^t by maximizing its utility. Finally, we prove that there exists a Stackelberg equilibrium so that neither the leader (the server) nor the followers (clients) have incentives to unilaterally deviate its optimal decisions.

1) The Optimal Privacy Strategy: To derive the client i's optimal privacy budget in the second stage game, we compute the first-order and second-order derivatives of $\mathcal{U}_i^t(\Gamma^t, \epsilon_i^t, \boldsymbol{\epsilon}_{-i}^t)$ with respect to ϵ_i^t as follows:

$$\frac{\partial \mathcal{U}_i^t}{\partial \epsilon_i^t} = \Gamma^t \frac{\sum_{i\in\mathcal{N}} \epsilon_i^t - \epsilon_i^t}{(\sum_{i\in\mathcal{N}} \epsilon_i^t)^2} - \xi_i = \Gamma^t \frac{\sum_{k\in\mathcal{N}\setminus i} \epsilon_k^t}{(\sum_{i\in\mathcal{N}} \epsilon_i^t)^2} - \xi_i. \tag{14}$$

$$\frac{\partial^2 \mathcal{U}_i^t}{\partial (\epsilon_i^t)^2} = -2\Gamma^t \frac{\sum_{k\in\mathcal{N}\setminus i} \epsilon_k^t}{(\sum_{i\in\mathcal{N}} \epsilon_i^t)^3} < 0. \tag{15}$$

According to Eq. (15), the utility of each client is a strict concave function, where the maximum value can be obtained by solving $\partial \mathcal{U}_i^t / \partial \epsilon_i^t = 0$. Then, the optimal privacy strategy of the client i, denoted by $(\epsilon_i^t)^*$, satisfies

$$(\epsilon_i^t)^* = \sqrt{(\Gamma^t \textstyle\sum_{k\in\mathcal{N}\setminus i} \epsilon_k^t)/\xi_i} - \textstyle\sum_{k\in\mathcal{N}\setminus i} \epsilon_k^t. \tag{16}$$

From Eq. (16), we notice that $(\epsilon_i^t)^*$ depends on other clients' privacy budgets. So the following objective is to remove the dependence. According to Eq. (14), we let $\partial \mathcal{U}_i^t / \partial \epsilon_i^t = 0$ and then have

Algorithm 1 The whole joint training process with NICE

for each round t **do**
  //Initialization:
  The server broadcasts the global model $\mathcal{M}(\mathbf{w}_g^{t-1};\cdot)$ of the last round, the requirements, the global information, and the total payment $(\Gamma^t)^*$ to all clients.
  //Privacy-preserving Local Training:
  for each client i, $i \in \mathcal{N}$ **do**
    Update local model parameters $\mathbf{w}_i^{t-1} = \mathbf{w}_g^{t-1}$;
    Determine its optimal privacy budget $(\epsilon_i^t)^*$ according to Eq. (19) and Theorem 3;
    Determine the sensitivity δ_L of the loss function;
    Adopt the flexible privacy-preserving mechanism Ψ with its private industrial dataset to get the disguised model parameters $\tilde{\mathbf{w}}_i^t$:
    $\tilde{\mathbf{w}}_i^t = \arg\min\left[L(\mathbf{w}_i^{t-1}; D_i) + Lap(\frac{\delta_L}{(\epsilon_i^t)^*})\right]$.
    Upload $\tilde{\mathbf{w}}_i^t$ and $(\epsilon_i^t)^*$ to the server.
  end for
  //Incentive Mechanism Γ:
  The server computes each client i's reward Γ_i^t:
  $\Gamma = \{(\Gamma_i^t)^* = (\Gamma^t)^* \frac{(\epsilon_i^t)^*}{\sum_{i\in\mathcal{N}}(\epsilon_i^t)^*} | i \in \mathcal{N}, t = 1, 2, \cdots\}$.
  //Aggregation and Model Updating:
  The server aggregates all $\tilde{\mathbf{w}}_i^t$ to derive a new global model parameter matrix $\mathbf{w}_g^t$: $\mathbf{w}_g^t = \sum_{i=1}^N \tilde{\mathbf{w}}_i^t r_i / \sum_{i=1}^N r_i$.
  The server evaluates the accuracy of the global model θ^t: $\theta^t = \frac{\sum_{(\boldsymbol{x_j}, y_j)\in D_s} 1_{\{\mathcal{M}(\mathbf{w}_g^t; \boldsymbol{x_j}) = y_j\}}}{|D_s|}$.
  if $\theta^t >$ A given threshold **then**
    End the whole training process.
  end if
  if the utility of the server $\mathcal{U}^t \le 0$ **then**
    End the whole training process.
  end if
  if any client i's utility $\mathcal{U}_i^t \le 0$ **then**
    The client i refuses to participant in the training.
  end if
end for

$$\Gamma^t \textstyle\sum_{k\in\mathcal{N}\setminus i}(\epsilon_k^t)^* = \xi_i(\sum_{i\in\mathcal{N}}(\epsilon_i^t)^*)^2. \tag{17}$$

Summing up all Eq. (17) for each $i \in \mathcal{N}$, we can derive that

$$\Gamma^t(|\mathcal{N}|-1)((\epsilon_1^t)^* + ... + (\epsilon_{|\mathcal{N}|}^t)^*) = (\sum_{i\in\mathcal{N}} \xi_i)(\sum_{i\in\mathcal{N}}(\epsilon_i^t)^*)^2.$$

Therefore, we can get $\sum_{i\in\mathcal{N}}(\epsilon_i^t)^* = (\Gamma^t(|\mathcal{N}|-1))/\sum_{i\in\mathcal{N}}\xi_i$. By substituting this equation into Eq. (17), we have

$$\sum\nolimits_{k\in\mathcal{N}\setminus i}(\epsilon_k^t)^* = \frac{\xi_i(\sum_{i\in\mathcal{N}}(\epsilon_i^t)^*)^2}{\Gamma^t} = \frac{\Gamma^t\xi_i(|\mathcal{N}|-1)^2}{(\sum_{i\in\mathcal{N}}\xi_i)^2}. \tag{18}$$

Thus, we can derive the optimal privacy strategy of the client by substituting Eq. (18) into Eq. (16) as follows.

$$(\epsilon_i^t)^* = \frac{\Gamma^t(|\mathcal{N}|-1)(\sum_{i\in\mathcal{N}}\xi_i - |\mathcal{N}|\xi_i + \xi_i)}{(\sum_{i\in\mathcal{N}}\xi_i)^2}. \tag{19}$$

From Eq. (19), we can observe that the optimal privacy budget of each client $(\epsilon_i^t)^*$ is only associated with the total payment Γ^t of the server and some open information (e.g., $|\mathcal{N}|, \sum_{i\in\mathcal{N}}\xi_i$). Thus, $(\epsilon_i^t)^*$ can be determined as long as the total payment Γ^t is given. Next, we will present the theorem to elaborate on how the server determines the optimal payment.

2) The Optimal Payment Strategy: In order to derive the optimal payment strategy, we prove that there exists a unique Stackelberg equilibrium, during which we will show that how the server determines the optimal payment. Before this, we define the equilibrium solution as follows.

Definition 3 (Nash Equilibrium, NE)**.** *The optimal local privacy budgets $\langle(\epsilon_1^t)^*, (\epsilon_2^t)^*, ..., (\epsilon_N^t)^*\rangle$ form a Nash Equilibrium if for any client i and any $\epsilon_i^t \ge 0$, we have*

$$\mathcal{U}_i^t(\Gamma^t, (\epsilon_i^t)^*, (\boldsymbol{\epsilon_{-i}^t})^*) \ge \mathcal{U}_i^t(\Gamma^t, \epsilon_i^t, (\boldsymbol{\epsilon_{-i}^t})^*). \tag{20}$$

Lemma 1. *[25] There exists a NE when the conditions are satisfied: 1) the player set is finite; 2) the strategy sets are closed, bounded, and convex; and 3) the utility functions are continuous and quasi-concave in the strategy space.*

Theorem 3. *There exists a unique Stackelberg equilibrium in the above two-stage Stackelberg game.*

Proof. The second stage can be considered as a non-cooperative game, since each client is rational and behaves in a selfish way to maximize its own utility. According to Eqs. (14), (15), and (19), it is straightforward to verify that Lemma 1 is satisfied. Thus, the optimal local privacy budgets $\langle(\epsilon_1^t)^*, (\epsilon_2^t)^*, ..., (\epsilon_N^t)^*\rangle$ form a Nash equilibrium (Definition 3) in the non-cooperative game among clients. Now, we only need to prove that all clients' optimal strategies generated by the second stage can also form a Nash equilibrium in the first stage. In this case, the whole two-stage Stackelberg game can form a Stackelberg equilibrium.

We first substitute all $(\epsilon_i^t)^*, i \in \mathcal{N}$ into Eq. (11) and get

$$\mathcal{U}^t(\Gamma^t, \epsilon_i^t, \boldsymbol{\epsilon_{-i}^t}) = \zeta(\ln(1 + \textstyle\sum_{i\in\mathcal{N}} G((\epsilon_i^t)^*))) - \Gamma^t. \tag{21}$$

The first-order derivative of $\mathcal{U}^t$ on the payment Γ^t is

$$\tfrac{\partial\mathcal{U}^t}{\partial\Gamma^t} = (\zeta\textstyle\sum_{i\in\mathcal{N}} \frac{\partial G}{\partial(\epsilon_i^t)^*}\frac{\partial(\epsilon_i^t)^*}{\partial\Gamma^t})/(1+\sum_{i\in\mathcal{N}} G((\epsilon_i^t)^*)) - 1. \tag{22}$$

The second-order derivative of $\mathcal{U}^t$ with respect to Γ^t is

$$\frac{\partial^2\mathcal{U}^t}{\partial(\Gamma^t)^2} = \zeta\frac{\sum_{i\in\mathcal{N}}(\frac{\partial^2 G}{\partial((\epsilon_i^t)^*)^2}(\frac{\partial(\epsilon_i^t)^*}{\partial\Gamma^t})^2 + \frac{\partial G}{\partial(\epsilon_i^t)^*}\frac{\partial^2(\epsilon_i^t)^*}{\partial(\Gamma^t)^2})}{(1+\sum_{i\in\mathcal{N}} G((\epsilon_i^t)^*))^2} \cdot (1+\sum\nolimits_{i\in\mathcal{N}} G((\epsilon_i^t)^*)) - \zeta\frac{(\sum_{i\in\mathcal{N}}\frac{\partial G}{\partial(\epsilon_i^t)^*}\frac{\partial(\epsilon_i^t)^*}{\partial\Gamma^t})^2}{(1+\sum_{i\in\mathcal{N}} G((\epsilon_i^t)^*))^2}$$
$$= \zeta\frac{\sum_{i\in\mathcal{N}}(\frac{\partial^2 G}{\partial((\epsilon_i^t)^*)^2}(\frac{\partial(\epsilon_i^t)^*}{\partial\Gamma^t})^2)}{1+\sum_{i\in\mathcal{N}} G((\epsilon_i^t)^*)} - \zeta\frac{(\sum_{i\in\mathcal{N}}\frac{\partial G}{\partial(\epsilon_i^t)^*}\frac{\partial(\epsilon_i^t)^*}{\partial\Gamma^t})^2}{(1+\sum_{i\in\mathcal{N}} G((\epsilon_i^t)^*))^2}.$$

Due to the concavity of the function G, we have $\partial^2 G/\partial((\epsilon_i^t)^*)^2 < 0$. Thus, we can derive that $\partial^2\mathcal{U}^t/\partial(\Gamma^t)^2 < 0$. That is to say, there is a unique optimal payment solution for $\partial\mathcal{U}^t/\partial\Gamma^t = 0$, denoted as $(\Gamma^t)^*$. By substituting Eqs. (19) and (22) into $\partial\mathcal{U}^t/\partial\Gamma^t = 0$, we can get an equation on the optimal payment $(\Gamma^t)^*$. Note that $G(\cdot)$ is a continuous and reversible concave function on $(\epsilon_i^t)^*$ as well as $(\epsilon_i^t)^*$ is a linear function of $(\Gamma^t)^*$. By using the two properties, we can derive the solution of $\partial\mathcal{U}^t/\partial\Gamma^t = 0$ to get $(\Gamma^t)^*$. If solving the equation $\partial\mathcal{U}^t/\partial\Gamma^t = 0$ cannot obtain a closed-form optimal payment due to the complexity of the function G, we can adopt some mathematical approximating methods (e.g., the bisection or Newton's method) to acquire an approximation of $(\Gamma^t)^*$.

Because both the server and the clients can find the optimal strategies to maximize their own utilities, the two-stage Stackelberg game possesses a Stackelberg equilibrium. □

According to Theorem 3, the server can derive the optimal payment $(\Gamma^t)^*$. Then, each client $i \in \mathcal{N}$ can determine its own

optimal privacy budget $(\epsilon_i^t)^*$ by substituting $(\Gamma^t)^*$ into Eq. (19). Now, the incentive mechanism can be instantiated as:

$$\Gamma = \{(\Gamma_i^t)^* = (\Gamma^t)^* \frac{(\epsilon_i^t)^*}{\sum_{i\in\mathcal{N}}(\epsilon_i^t)^*} | i \in \mathcal{N}, t = 1, 2, \cdots\}. \quad (23)$$

By means of this incentive mechanism, both the server and clients can determine their optimal strategies (i.e., the optimal payment $(\Gamma^t)^*$ and the optimal privacy budget $(\epsilon_i^t)^*$). $(\epsilon_i^t)^*$ ensures that the client i can obtain a maximum utility while compensating its privacy cost. Actually, this is the best privacy budget that the client i can accept, i.e., the maximum effort that the client is willing to make.

We illustrate the whole joint training process pseudocode in Algorithm 1, which mainly contains the designed mechanisms (i.e., Ψ and Γ). To be specific, the process of the flexible privacy-preserving local training with the sensitive industrial dataset is corresponding to the section III (Lines 4∼12), and the section IV has depicted the design of the incentive mechanism in great detail (Lines 13∼15). Moreover, the computation complexity of Algorithm 1 is $O(TNT'|D|d)$, where T is the number of global rounds, T' is the required number of local training epochs, $|D|$ is the required dataset size of each client, and d is the dimension of each training sample.

V. Performance Evaluation

In this section, we conduct extensive simulations to verify the performance of NICE based on real-world datasets. We first examine our theoretical results to validate the effectiveness of the proposed incentive mechanisms, and then provide numerical results for FL using TensorFlow [26].

A. Experiment Setup

Simulation Settings: In the simulations, we let the number of clients (i.e., N) be selected from [20, 500], and let $N = 20$ by default. We generate the system parameter (i.e., ζ) from [180, 220] and let $\zeta = 200$ by default. When generating the unit privacy cost of each client (i.e., ξ_i), we adopt the truncated Gaussian distribution with the mean value 2 and the standard deviation σ. The value of σ will change from [0.1, 1] to evaluate its influence. Also, we set $\sigma = 0.5$, $c_i^{cp} = c_i^{cm} = 0.1$ by default and use $\mathcal{U}^t(\Gamma^t, \epsilon_i^t, \boldsymbol{\epsilon_{-i}^t}) = \zeta(\ln(1+\sum_{i\in\mathcal{N}}(\epsilon_i^t)^*)) - \Gamma^t$ to denote the utility of the server for simplicity. In addition, to evaluate the training effect of FL, we use two representative datasets which are simulated as big data collected from industrial IoT: a handwritten digit dataset MNIST which contains 60,000 training samples of 10 digits and 10,000 testing samples [23], and a three-channel image dataset CIFAR-10 which contains 50,000 training samples in 10 classes and 10,000 test samples [24]. For the classification problem, we construct a common neural network model and train the model in a differentially-private manner. For MINST, we implement a neural network architecture containing two auto-encoder layers, a fully connected hidden layer, and an output layer. For the more complex dataset CIFAR-10, we leverage the network containing eight layers (i.e., five convolutional layers and three fully-connected layers) based on AlexNet [27]. The details of the parameter settings are summarized in Table II for clarity. Due to the limited space, it is unnecessary to observe the effect of NICE for each client so that we only choose four clients (i.e., client-2, 9, 14, 18) for illustration.

TABLE II
Evaluation Settings

Parameter name	Values
number of clients, N	[20, 500] (20 in default)
system parameter, ζ	[180, 220] (200 in default)
the standard deviation, σ	[0.1, 1] (0.5 in default)
the strategy ϵ_9^t, Γ^t	[0, 25], [160, 230]
unit privacy cost of worker-9, ξ_9	[1.5, 3] (1.95 in default)

Algorithms in Comparison: Since NICE combines Stackelberg game and DP to solve privacy-flexibility and the incentive problem for FL systems in industrial IoT, we compare NICE with the existing state-of-the-art studies with incentive mechanism designs [19], [28]. However, the models and problems in these works are different from ours so that we cannot compare them directly. Therefore, we tailor the basic idea in these algorithms for our model and carefully design three incentive mechanisms for comparison: Contract-based algorithm [19], Auction-based algorithm [28], and Uniform cost. Here, the contract-based algorithm is based on contract theory, the auction-based scheme takes the advantage of the technique of game theory, and the uniform cost mechanism means that the unit privacy cost of each client is the same.

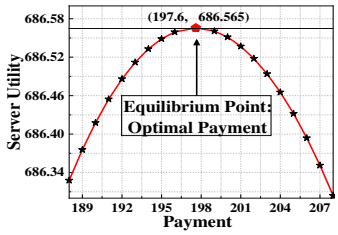
B. Experiment Results

We first evaluate the effect of various game strategies (i.e. Γ^t and ϵ_i^t) on the parties' utilities to verify the existence of the unique Stackelberg equilibrium. Next, we carefully measure the influence of system parameter ζ, the client-9's unit privacy cost ξ_9, the client-9's privacy budget ϵ_9^t, the payment of the server Γ^t, and the standard deviation σ. Then, we consider the effect of the number of clients N and compare NICE with other incentive mechanisms. Finally, we evaluate the training convergence based on the real-world datasets MINST and CIFAR-10 in FL. It is noteworthy that all of the points in the figures are in the equilibrium except for Fig. 3 and Fig. 4.

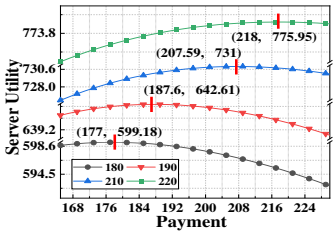
The Optimal Γ^t. We evaluate the impact of Γ^t on Server Utility (SU) in Fig. 3. As the utility of the server $\mathcal{U}^t$ is a concave function about Γ^t, there is a unique value $(\Gamma^1)^* = 197.6$ that can achieve the maximum utility $\mathcal{U}_{max}^1 = 686.565$. When we set different system parameters, the server can also maximize its own utility by adopting the optimal payment under the unique Stackelberg equilibrium in Fig. 3(b).

The Optimal ϵ_i^t. Fig. 4 shows the impact of privacy budget on Client Utility (CU) under different system parameters. We can notice that CU first climbs up and then declines as the privacy budget grows. It demonstrates that all clients can meet a Stackelberg equilibrium, i.e., there is a unique optimal privacy budget for each client to earn the maximum reward. The reason is that the utility function of each client $\mathcal{U}_i^t$ is strictly concave with regard to ϵ_i^t. Besides, the client-18's utility is higher than others because his privacy cost is lower. We can attribute this to the fact that a client with a lower cost would not care about privacy leakage or have a low sensitive dataset, and thus can attain a better income.

Effect of ζ. To illustrate the impact of the system parameter, we observe the variation of all parties' utilities and strategies along with ζ in Fig. 5. SU rises with the increase of ζ owing to getting a higher gain according to Eq. (11). Hence, the server is pleased to pay more to incentivize the clients to train the

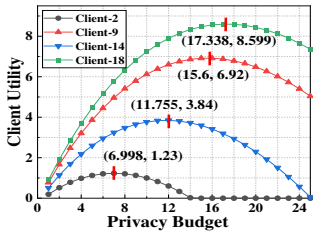


(a) SU vs. Payment Γ^t

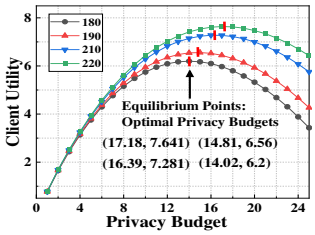


(b) SU vs. Γ^t under different ζ

Fig. 3. SU vs. Payment (The existence of unique Stackelberg equilibrium)

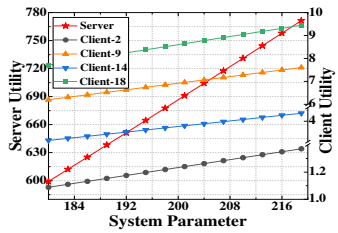


(a) CU vs. ϵ

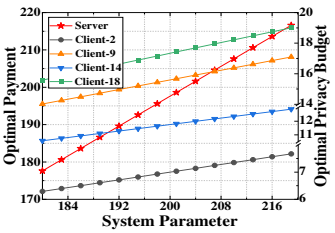


(b) CU vs. ϵ under different ζ

Fig. 4. CU vs. Privacy budget (The existence of unique Stackelberg equilibrium)

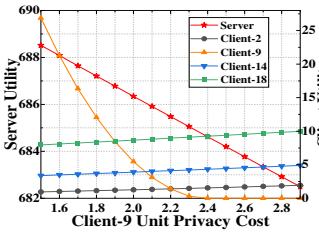


(a) SU & CU vs. ζ

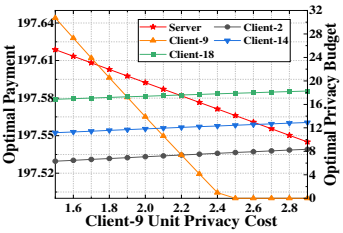


(b) Γ^t & ϵ_i^t vs. ζ

Fig. 5. Effect of the system parameter ζ

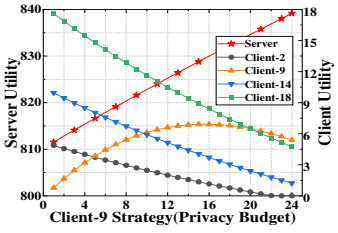


(a) SU & CU vs. ξ_9

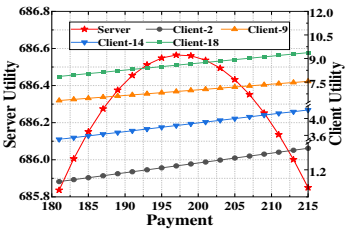


(b) Γ^t & ϵ_i^t vs. ξ_9

Fig. 6. Effect of the client-9's unit privacy cost ξ_9

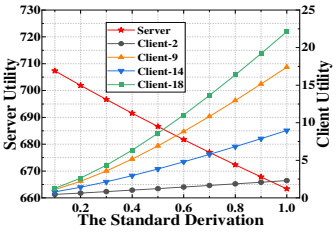


(a) SU & CU vs. ϵ_9^t

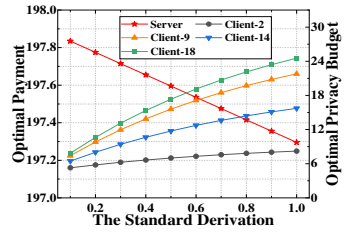


(b) SU & CU vs. Γ^t

Fig. 7. Effect of the non-optimal strategy



(a) SU & CU vs. σ



(b) Γ^t & ϵ_i^t vs. σ

Fig. 8. Effect of the standard deviation σ

more accurate models. CU has the same changing tendency as SU. This is because the payment Γ^t will increase along with the growth of ζ. Thus, each client tends to pick a higher privacy budget and its utility will grow up. Meanwhile, we can also find that a client who has a lower privacy cost ξ_i achieves a higher utility. The reason is similar to the results in Fig. 4.

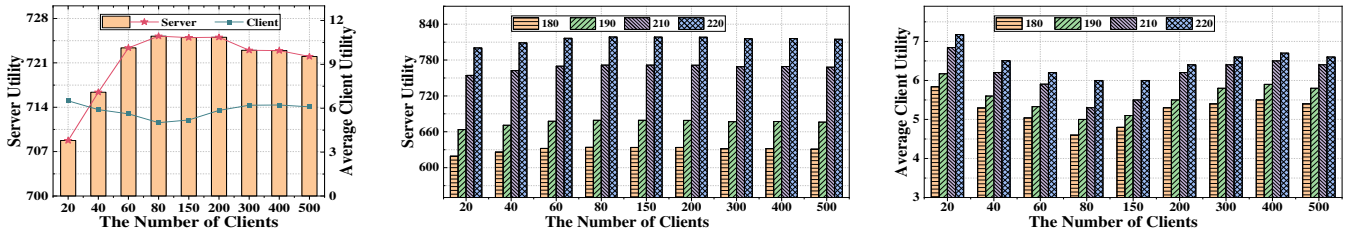
Effect of ξ_9. Fig. 6(a) reports the utility obtained by each party when we change the unit privacy cost ξ_9 of client-9 from 1.5 to 3. Correspondingly, Fig. 6(b) shows the changes of all parties' optimal strategies. We first note that SU and Γ^t decrease rapidly with the increase of ξ_9. Obviously, a client is not willing to participate in the FL training process when his local data has more sensitive information. Therefore, along with the improvement of the privacy cost, the gain of the server from client-9 drops off, and the server could cut down on the payment cost appropriately. When the privacy cost reaches a certain value, client-9's loss due to privacy leakage has far exceeded its reward. In such a case, client-9 might leave to avoid private information from being divulged, and the rewards of other clients will increase correspondingly.

Effect of ϵ_9^t and Γ^t. In Fig. 7, we plot the changing trend of SU and CU when client-9 or the server does not select the optimal strategy. Firstly, SU increases because the larger privacy budget means that the client will have a higher contribution to the global model. So, the server can receive a better local model from client-9. Secondly, the utilities of other clients go down as ϵ_9^t grows. The reason is that more rewards will be paid to client-9 according to Eq. (10). The utilities of all clients increase when the server pays more. Lastly, the client-9's utility and the server's utility first rise and then fall, which is consistent with Fig. 3 and Fig. 4.

Effect of σ. We change the standard deviation σ to form different distributions of the unit privacy cost and Fig. 8 presents the effect of σ on utilities and strategies. When N is large enough, ϵ_i^t will grow up according to Eq. (19) and the server does not have enough money to compensate these high-cost clients. Thus, the optimal payment and SU will decrease.

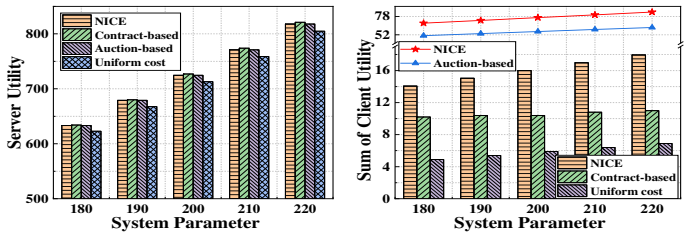
Effect of N. Fig. 9 demonstrates the influence of the number of clients under different system parameters. Because the gain of the server is modeled as a logarithmic function, the server's gain cannot keep growing indefinitely even if there are enough clients. Therefore, SU (resp. CU) shows an upward (resp. downward) trend and then keeps stable. Fig. 9(b) and Fig. 9(c) signify that a higher ζ will lead to a higher utility.

Incentive Mechanism Comparison. As illustrated in Fig. 10 and Fig. 11, we investigate the impact of ζ and N under different incentive mechanisms based on contract theory and game theory, respectively. We see that the value of SU among various mechanisms differs little and NICE performs much better than the compared algorithms, as shown in Fig. 10(a), Fig. 10(b), and Fig. 11(b). This is because that the contract-based algorithm and the auction-based algorithm only guarantee the non-negativity of clients' utilities. In Fig. 11(a), SU of the contract-based mechanism decreases but SU of other mechanisms slowly grow when N increases. The reason is that the number of contracts is limited and it cannot satisfy all clients' needs compared with NICE. Besides, when the unit privacy cost of each client is the same (i.e., uniform cost),



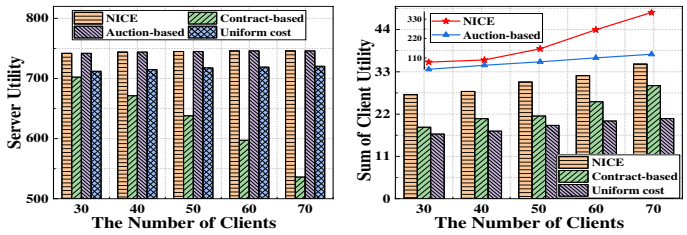
(a) SU & CU vs. N (b) SU vs. N under different ζ (c) CU vs. N under different ζ

Fig. 9. Effect of the number of clients N on SU & CU under different system parameters



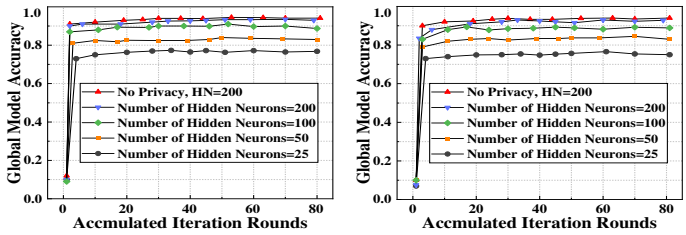
(a) SU vs. ζ (b) Sum of CU vs. ζ

Fig. 10. SU & Sum of CU vs. ζ under different incentive mechanisms



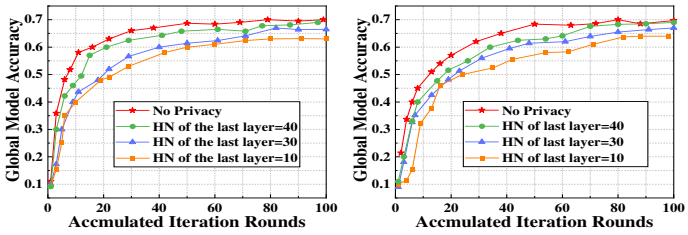
(a) SU vs. N (b) Sum of CU vs. N

Fig. 11. SU & Sum of CU vs. N under different incentive mechanisms



(a) The number of clients: N=30 (b) The number of clients: N=80

Fig. 12. Global Model Accuracy vs. Rounds (MNIST)



(a) The number of clients: N=30 (b) The number of clients: N=80

Fig. 13. Global Model Accuracy vs. Rounds (CIFAR-10)

both the utility of the server and clients are lowest since no client is willing to lower its privacy protection level.

Training Convergence and Comparison. Finally, we conduct a series of comparative experiments based on the real-world MNIST dataset and CIFAR-10 dataset. All clients adopt the optimal strategies determined by the incentive mechanism Γ. We assume that these strategies remain unchanged in each round and randomly select three or five clients in each round. Next, we alter the number of Hidden Neurons (HN) as $\{200, 100, 50, 25\}$ and then borrow the non-private approach as a baseline for comparison in Fig. 12. It is shown that there is a fast convergence rate at the early stage of training followed by a slight increase due to the simplicity of MNIST. The accuracy of global model raises gradually based on the CIFAR-10 dataset in Fig. 13. Although we employ the DP mechanism Ψ to upload disguised model parameters, the degree of the global model accuracy degradation can be acceptable. Besides, the convergence rate will be slower if we increase N, because the partition of clients' local datasets is related to N (i.e., the larger the N, the less the training data of each client).

More importantly, our proposed flexible privacy-preserving mechanism can achieve DP without consuming too much privacy budget, compared with adding noises into enormous model parameters. For example, [29] utilized the LDP in federated learning. Owing to the high variance of their mechanism, they need more than 200 communication rounds and spend much more privacy budgets (MINST with ϵ=500 and CIFAR-10 with ϵ=5000). [30] added artificial noise into parameters at the clients' side before aggregation, which would require protection levels ϵ=50/60/100. Truex et al. [31] proposed α-CLDP to achieve a higher accuracy by requiring a relatively privacy budget $\epsilon = \alpha * 2c * 10^{\rho}$ (e.g., $\alpha = 1, c = 1, \rho = 10$), which results in a weak privacy guarantee. Overall, the results prove that the NICE can work efficiently and suitably in real applications. Moreover, we can enhance the global accuracy through raising the number of hidden neurons to weaken the influence of noise. Given a fixed training set, we expect that increasing the number of hidden neurons leads to better learning performance, but has the greater computational complexity.

VI. Related Work

A. Privacy-Preserving Mechanisms in FL

Even though FL leaves clients' data locally and eases concerns about industrial privacy leaks to some extent, recent researches [32] have shown that some malicious attackers can still recover the sensitive information of industrial data (e.g., image tabs, memberships, industry location, etc) by exploiting the shared gradient and global parameters.

Existing privacy-preserving approaches are mostly evolved from three underlying techniques: encryption [7], Secure Multi-party Computation (SMC) [33], and Differential Privacy (DP). Despite encryption or SMC can ensure the confidentiality of individual information completely, they would lead to huge computation and communication overheads. On the contrary, DP has received growing attention since it is a lightweight tool. However, the great majority of DP-based privacy-preserving mechanisms [9], [30] do not consider the clients' different privacy preferences, namely, the privacy

budgets are preset and all clients are treated equally. Only a few adaptive DP schemes are studied recently [19], [34]–[36]. For example, [19], [34] designed a set of optimal contracts by considering the information asymmetry and the heterogeneous types of costs. [35], [36] can satisfy different privacy requirements for users by reusing the wasted budget or injecting adaptive noise. However, the contract designs in [19], [34] are manipulated by the server and the privacy budgets for clients to choose from are limited. [35], [36] attempt to harness adaptive DP-based algorithms for improving the privacy protection levels of all clients, but we aim to allow clients to determine their privacy protection levels by themselves according to their personalized privacy demands. Moreover, we take the game among all clients and the server into account.

B. Incentive Mechanisms in FL

Recently, a wide spectrum of remarkable incentive mechanisms have been designed for various FL systems. By means of different tools, these incentive mechanisms can motivate enough clients to participate in FL systems so as to achieve a satisfactory global model.

Game theory has been utilized as a powerful tool to study the incentive mechanisms for FL. For example, the study in [37] proposed a hierarchical incentive mechanism for FL using the coalitional game theory approach, in which multiple model owners can form various federations. The study of [38] incentivized unmanned aerial vehicles to participate in the FL training by designing a joint auction-coalition formation framework. Jiao et al. [28] introduced an auction framework for the wireless federated learning services market and a reverse auction mechanism to maximize the social welfare. The authors in [39] formulated a market model based on Stackelberg game to obtain the maximum utility of operators and solved a tradeoff between the revenue and energy consumption.

Some other tools (e.g., contract theory and deep reinforcement learning) have also been applied for the design of incentive mechanisms. For instance, Lim et al. [40] presented a contract-theoretic incentive mechanism to stimulate workers to update the data, which can balance the tradeoff between age of information (AoI) and service latency in FL systems. [10] designed a contract theory-based incentive mechanism to motivate high-reputation workers with high-quality data to participate in model training of FL. Zhan et al. [41] developed a deep reinforcement learning-based incentive mechanism, which can learn the best pricing strategy of the aggregator in a dynamic environment. However, none of them incorporates clients' privacy concerns into incentive mechanism design. Different from them, we propose an incentive mechanism taking clients' heterogeneous industrial privacy demands into account, which can maximize the utilities of all clients and the server simultaneously.

VII. Conclusion

In this paper, we propose a privacy-preserving incentive mechanism, named NICE. It adopts a flexible Laplace DP approach, in which each client can decide the privacy budget by itself according to its own personalized privacy demand, i.e., each client can customize how much noise to be added for protecting its data privacy. In NICE, we model the problem of determining the optimal privacy budget for each client as an incentive mechanism design problem, which is further formulated as a two-stage Stackelberg game. Moreover, we derive the optimal strategies for all parties in this game. Through rigorous theoretical analysis, we prove that the optimal strategies constitute a unique equilibrium, by which NICE can make the server and all clients achieve their maximum utilities simultaneously. Extensive simulations confirm the efficacy of NICE and indicate significant performance in accordance with the design expectations.

References

[1] S. Bahrami, Y. C. Chen, and V. W. S. Wong, "Deep reinforcement learning for demand response in distribution networks," *IEEE Transactions on Smart Grid*, vol. 12, no. 2, pp. 1496–1506, 2021.

[2] X. Zhang, F. Fang, and J. Wang, "Probabilistic solar irradiation forecasting based on variational bayesian inference with secure federated learning," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7849–7859, 2021.

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

[4] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1205–1221, 2019.

[5] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning for digital twin edge networks in industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5709–5718, 2021.

[6] C. T. Dinh, N. H. Tran, T. D. Nguyen, W. Bao, A. Y. Zomaya, and B. B. Zhou, "Federated learning with proximal stochastic variance reduced gradient algorithms," in *ICPP*, 2020, pp. 1–11.

[7] L. Lyu, J. Yu, K. Nandakumar, Y. Li, X. Ma, J. Jin, H. Yu, and K. S. Ng, "Towards fair and privacy-preserving federated deep models," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 11, pp. 2524–2541, 2020.

[8] B. Zhao, K. Fan, K. Yang, Z. Wang, H. Li, and Y. Yang, "Anonymous and privacy-preserving federated learning with industrial big data," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 9, pp. 6314–6323, 2021.

[9] Y. Wang, Z. Su, N. Zhang, and A. Benslimane, "Learning in the air: Secure federated learning for uav-assisted crowdsensing," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1055–1069, 2021.

[10] 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.

[11] Y. Zhan, P. Li, Z. Qu, D. Zeng, and S. Guo, "A learning-based incentive mechanism for federated learning," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6360–6368, 2020.

[12] Y. Zhao, J. Zhao, L. Jiang, R. Tan, D. Niyato, Z. Li, L. Lyu, and Y. Liu, "Privacy-preserving blockchain-based federated learning for iot devices," *IEEE Internet of Things Journal*, vol. 8, no. 3, pp. 1817–1829, 2021.

[13] Y. Zhan, P. Li, S. Guo, and Z. Qu, "Incentive mechanism design for federated learning: Challenges and opportunities," *IEEE Network*, vol. 35, no. 4, pp. 310–317, 2021.

[14] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of cryptography conference*, vol. 3876, 2006, pp. 265–284.

[15] J. Zhang, Z. Zhang, X. Xiao, Y. Yang, and M. Winslett, "Functional mechanism: Regression analysis under differential privacy," *Proceedings of the VLDB Endowment*, vol. 5, no. 11, pp. 1364–1375, 2012.

[16] W. Rudin *et al.*, *Principles of mathematical analysis*. McGraw-hill New York, 1964, vol. 3.

[17] F. McSherry, "Privacy integrated queries: an extensible platform for privacy-preserving data analysis," in *ACM SIGMOD*, 2009, pp. 19–30.

[18] A. Ghosh and A. Roth, "Selling privacy at auction," *Games and Economic Behavior*, vol. 91, pp. 334–346, 2015.

[19] P. Sun, Z. Wang, Y. Feng, L. Wu, Y. Li, H. Qi, and Z. Wang, "Towards personalized privacy-preserving incentive for truth discovery in crowdsourced binary-choice question answering," in *IEEE INFOCOM*, 2020, pp. 1133–1142.

[20] N. H. Tran, W. Bao, A. Y. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *IEEE INFOCOM*, 2019, pp. 1387–1395.

[21] Z. Qu, S. Guo, H. Wang, B. Ye, Y. Wang, A. Zomaya, and B. Tang, "Partial synchronization to accelerate federated learning over relay-assisted edge networks," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2021.

[22] Q. Xu, Z. Su, and R. Lu, "Game theory and reinforcement learning based secure edge caching in mobile social networks," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3415–3429, 2020.

[23] 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.

[24] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," in *CIFAR-10 Dataset*, 2009.

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

[26] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, and S. G. et al., "Tensorflow: A system for large-scale machine learning," in *OSDI*, 2016, pp. 265–283.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[28] Y. Jiao, P. Wang, D. Niyato, B. Lin, and D. I. Kim, "Toward an automated auction framework for wireless federated learning services market," *IEEE Transactions on Mobile Computing*, vol. 20, no. 10, pp. 3034–3048, 2021.

[29] A. Bhowmick, J. C. Duchi, J. Freudiger, G. Kapoor, and R. Rogers, "Protection against reconstruction and its applications in private federated learning," *arXiv preprint arXiv:1812.00984*, 2018.

[30] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[31] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "Ldp-fed: Federated learning with local differential privacy," in *ACM International Workshop on Edge Systems, Analytics and Networking*, 2020, pp. 61–66.

[32] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "Verifynet: Secure and verifiable federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 911–926, 2020.

[33] B. Jayaraman, L. Wang, D. Evans, and Q. Gu, "Distributed learning without distress: Privacy-preserving empirical risk minimization," in *NeurIPS*, 2018, pp. 6346–6357.

[34] M. Wu, D. Ye, J. Ding, Y. Guo, R. Yu, and M. Pan, "Incentivizing differentially private federated learning: A multidimensional contract approach," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10 639–10 651, 2021.

[35] B. Niu, Y. Chen, B. Wang, Z. Wang, F. Li, and J. Cao, "Adapdp: Adaptive personalized differential privacy," in *IEEE INFOCOM*, 2021, pp. 1–10.

[36] X. Liu, H. Li, G. Xu, R. Lu, and M. He, "Adaptive privacy-preserving federated learning," *Peer-to-Peer Networking and Applications*, vol. 13, no. 6, pp. 2356–2366, 2020.

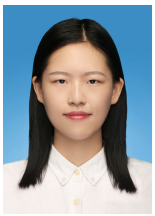
[37] W. Y. B. Lim, Z. Xiong, C. Miao, D. Niyato, Q. Yang, C. Leung, and H. V. Poor, "Hierarchical incentive mechanism design for federated machine learning in mobile networks," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9575–9588, 2020.

[38] J. S. Ng, W. Y. B. Lim, H. Dai, Z. Xiong, J. Huang, D. Niyato, X. Hua, C. Leung, and C. Miao, "Joint auction-coalition formation framework for communication-efficient federated learning in uav-enabled internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 4, pp. 2326–2344, 2021.

[39] J. Lee, D. Kim, and D. Niyato, "Market analysis of distributed learning resource management for internet of things: A game-theoretic approach," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 8430–8439, 2020.

[40] W. Y. B. Lim, Z. Xiong, J. Kang, D. Niyato, C. Leung, C. Miao, and X. Shen, "When information freshness meets service latency in federated learning: A task-aware incentive scheme for smart industries," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 1, pp. 457–466, 2022.

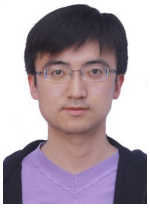
[41] Y. Zhan and J. Zhang, "An incentive mechanism design for efficient edge learning by deep reinforcement learning approach," in *IEEE INFOCOM*, 2020, pp. 2489–2498.



Yin Xu received her B.S. degree from the School of Computer Science and Technology at the Anhui University (AHU), Hefei, China, in 2019. She is currently a PhD student in the School of Computer Science and Technology at the University of Science and Technology of China (USTC), Hefei, China. Her research interests include federated learning, game theory, spatial crowdsourcing, reinforcement learning, mobile computing, privacy preservation, and incentive mechanism.

Mingjun Xiao is a professor in the School of Computer Science and Technology at the University of Science and Technology of China (USTC). He received his Ph.D. from USTC in 2004. His research interests include crowdsourcing, mobile social networks, vehicular ad hoc networks, mobile cloud computing, auction theory, data security and privacy. He has published more over 100 papers in referred journals and conferences, including TMC, TC, TPDS, TON, TKDE, TSC, INFOCOM, ICDE, ICNP, etc. He served as the TPC member of INFOCOM'21, IJCAI'21, INFOCOM'20, INFOCOM'19, ICDCS'19, DASFAA'19, INFOCOM'18, etc. He is on the reviewer board of several top journals such as TMC, TON, TPDS, TSC, TVT, TCC, etc.

Haisheng Tan received his B.E. degree in Software Engineering and B.S. degree in Management both from University of Science and Technology of China (USTC) with the highest honor. Then, he got his Ph.D. degree in computer science at the University of Hong Kong (HKU). He is currently an associate professor at USTC. His research interests lie primarily in Networking Algorithm Design and System Implementation, where he has published over 70 papers in prestigious journals and conferences. He recently received the awards of ACM China Rising Star (Hefei Chapter), the Distinguished TPC Member of INFOCOM 2019, the Best Paper Award in WASA'19, CWSN'20 and PDCAT'20.



An Liu is a professor in the Department of Computer Science and Technology at Soochow University. He received his Ph.D. degree in computer science from both City University of Hong Kong (CityU) and University of Science and Technology of China (USTC) in 2009. His research interests include spatial databases, crowdsourcing, data security and privacy, and cloud/service computing. He has published more than 80 papers in referred journals and conferences, including IEEE TKDE, IEEE TSC, GeoInformatica, KAIS, ICDE, WWW etc. He served as the Workshop Co-Chairs of WISE 2017 and DASFAA 2015. He is on the reviewer board of several top journals such as IEEE TKDE, IEEE TSC, IEEE TII, IEEE TCC, ACM TOIT, JSS, DKE, FGCS, WWWJ, and JCST.

Guoju Gao is an assistant professor in the School of Computer Science and Technology at the Soochow University. He received his Ph.D. degree in computer science and technology with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China. He visited the Temple University, USA, from Jan. 2019 to Jan. 2020. His research interests include mobile computing, network traffic measurement, and reinforcement learning. He has published 30 articles in referred journals and conferences, including IEEE TMC, ToN, TPDS, TSC, INFOCOM, ICPP, etc.

Zhaoyang Yan received her B.S degree from Shandong University (SDU), Weihai, China in 2020. He is currently a graduate student in P.C. Rossin College of Engineering and Applied Science at Lehigh University (LU), Bethlehem, U.S. His research interests include machine learning, smart city, mobile computing, reinforcement learning, IoT, parallel computing, incentive mechanism, computer vision, and big data analysis.