Incentivized Federated Learning and Unlearning

Ningning Ding, Zhenyu Sun, Ermin Wei, and Randall Berry, *Fellow, IEEE*

Abstract—To protect users' *right to be forgotten* in federated learning, federated unlearning aims at eliminating the impact of leaving users' data on the global learned model. The current research in federated unlearning mainly concentrates on developing effective and efficient unlearning techniques. However, the issue of incentivizing valuable users to remain engaged and preventing their data from being unlearned is still under-explored, yet important to the unlearned model performance. This paper focuses on the incentive issue and develops an incentive mechanism for federated learning and unlearning. We first characterize the leaving users' impact on the global model accuracy and the required communication rounds for unlearning. Building on these results, we propose a four-stage game to capture the interaction and information updates during the learning and unlearning process. A key contribution is to summarize users' multi-dimensional private information into one-dimensional metrics to guide the incentive design. Interestingly, we prove that allowing federated unlearning can result in reduced payoffs for both the server and users, compared to a scenario without unlearning. Numerical results demonstrate the necessity of unlearning incentives for retaining valuable leaving users, and also show that our proposed mechanisms decrease the server's cost by up to 53.91% compared to state-of-the-art benchmarks.

Index Terms—incentive mechanism, federated learning, federated unlearning

1 Introduction

1.1 Background and Motivations

Federated learning is a promising distributed machine learning paradigm, in which multiple users collaborate to train a shared model under the coordination of a central server [1]. This approach allows users to keep their local data on their own devices and only share the intermediate model parameters, which helps protect their raw data. However, despite these measures, it may not provide sufficient privacy guarantees [2], [3].

A stronger privacy guarantee is to ensure a user's "right to be forgotten" (RTBF), which has been explicitly stated in the European Union General Data Protection Regulation (GDPR) [4] and the California Consumer Privacy Act (CCPA) [5]. That is, a user has the right to request deletion of his private data and its impact on the trained model, if he no longer desires to participate in the platform. Users may seek to leave a platform for a variety of reasons. For example, they may feel that the benefits from the platform are not sufficient to compensate for their potential privacy leakage from participation. Furthermore, until they participate in the platform, they may not have full knowledge of these benefits and costs due to incomplete information about other users' data. For instance, users' privacy costs in federated learning depend on how unique their data is [6], which they can infer from their training loss after training [7].

To remove data from a trained federated learning model, the concept of *federated unlearning* has recently been proposed [8]. In this concept, after some users request to revoke their data, the remaining users will perform additional training or calculations to eliminate the impact of leaving users' data and obtain an unlearned model. A simple yet costly approach is to retrain the model from scratch with the requested data being removed from the training dataset [9]. To be more efficient and effective, existing literature (e.g., [7], [10], [11]) has focused on alternative federated unlearning methods that obtain a model similar (in some distance metrics) to a retrained model with lower computational costs. However, these studies usually assumed that users are willing to participate in federated learning and unlearning. This assumption may not be realistic without proper incentives since users incur various costs during the training process (e.g., time, energy, and privacy costs). Our goal in this paper is to develop incentive mechanisms to help retain valuable leaving users and create a sustainable learning platform for both the users and the server.

To design an incentive mechanism for federated learning and unlearning, there are several challenges to tackle. First, different leaving users will lead to different unlearned model performances and unlearning costs, the relationship among which is still an open problem yet essential for designing incentives. Second, it is difficult for the server to design incentives for a large number of heterogeneous users, when users have multi-dimensional private information (e.g., training costs and privacy costs) and unknown information (e.g., users' training losses before federated learning). Third, unlearning incentives for retaining valuable leaving users require careful design. High incentives may encourage strategic users to intentionally request revocation to obtain retention rewards, while low incentives may fail to retain valuable users. It is also crucial for the server to distinguish between high-quality leaving users (e.g., with rare and valuable data) and low-quality ones (e.g., with erroneous data), both of which can lead to high training losses. Fourth, both learning and unlearning incentives affect the server's and users' payoffs but are determined in different

- *Ningning Ding is with the Data Science and Analytics Thrust and the Internet of Things Thrust, Information Hub, Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511453, China (email: ningningding@hkust-gz.edu.cn).*
- *Zhenyu Sun and Randall Berry are with the Department of Electrical and Computer Engineering, Northwestern University, Evanston, IL 60208, USA.*
- *Ermin Wei is with Electrical and Computer Engineering Department and Industrial Engineering and Management Sciences Department, Northwestern University, Evanston, IL 60208, USA.*

stages - before or after federated learning. Meanwhile, there are different information asymmetry levels in each stage, as the federated learning process can reveal some information such as users' training losses and contributions. Thus, the mutual influence of the incentives and dynamic information asymmetry further complicate the incentive mechanism design.

The above discussion motivates us to answer the following interesting question:

Question 1. *Considering leaving users' impact, what is the server's optimal incentive mechanism for federated learning and unlearning, when heterogeneous users have strategic data revocation decisions and multi-dimensional private and unknown information?*

Furthermore, although federated unlearning is important for protecting users' right to be forgotten and data privacy, existing work lacks the understanding of whether allowing federated unlearning is economically beneficial to the server or users by comparing the following two scenarios:

- *Unlearning-Allowed Scenario.* The federated learning server allows users to revoke data and will perform federated unlearning;
- *Unlearning-Forbidden Scenario.* The federated learning server does not allow users to revoke data after they decide to participate in the federated training.

Different unlearning scenarios will lead to different optimal incentive mechanisms, as well as the server's and users' payoffs. When unlearning is optional, studying the outcomes of each scenario will facilitate the server's and users' selection of which scenario to participate in. The performance comparison also provides insights into the policy design of a market regulator. This motivates the second key question of this paper:

Question 2. *Compared with the unlearning-forbidden scenario, is the federated unlearning-allowed scenario more beneficial to the server and users in terms of their payoffs?*

1.2 Contributions

We summarize our key contributions below.

- *Incentive mechanism design for federated learning and unlearning.* We propose a four-stage Stackelberg game to analyze the optimal incentives of the server and the optimal strategies of users within this game. To the best of our knowledge, we are the first to analytically study the incentive mechanism and economic benefit of federated unlearning.
- *Theoretical characterization of global model accuracy and unlearning communication rounds.* We theoretically derive bounds on the global model optimality gap given non-IID data for federated learning algorithms (Scaffold [12] and FedAvg [1]) and the number of global communication rounds required for a federated unlearning method.
- *Optimal incentives and revocation decisions under multi-dimensional incomplete information.* Due to the complex interaction, users' multi-dimensional private information, and dynamically updated knowledge, the server's optimization problem in Stage I of the four-stage game is highly complex. We summarize users' multi-dimensional heterogeneity into several one-dimensional metrics and develop an efficient algorithm with linear complexity, to handle the exponentially large number of possible cases involved in optimal mechanism design. We also identify and analyze a supermodular game among the users to obtain their optimal data revocation decisions.
- *Comparison of unlearning-allowed and unlearning-forbidden scenarios.* We show that (i) when users' unlearning costs in the unlearning-allowed scenario are large, the server needs to compensate them with large incentives and thus prefers the unlearning-forbidden scenario. Surprisingly, users prefer the unlearning-allowed scenario in which they have large costs, due to the excess rewards they obtain under information asymmetry. (ii) When users' perceived privacy costs in the unlearning-forbidden scenario are large, the server prefers the unlearning-allowed scenario while users prefer the unlearning-forbidden scenario for similar reasons as in (i).
- *Insights and Performance Evaluation.* We show that high costs and training losses motivate users to leave, while the server will retain the leaving users who make significant contributions to model accuracy but not necessarily low training losses, as small losses of retained users will reduce privacy costs yet increase unlearning costs. We numerically show that compared with state-of-the-art benchmarks, our proposed incentive mechanism decreases the server's cost by up to 53.91%. Moreover, the results demonstrate that it is beneficial for the server to retain valuable leaving users and jointly optimize the federated learning and unlearning incentive mechanisms.

1.3 Related Work

The concept of *machine unlearning*, which refers to the process of removing the impact of a data sample from a trained model, was first introduced by Cao et al. in 2015 [13]. Most related literature was about centralized machine unlearning (e.g., [9], [14], [15]), in which the unlearned model (not retrained from scratch) was trained on summarized (e.g., aggregates of summations) or partitioned subsets rather than individual training samples. As a result, the model only needed to be updated on the subset(s) of data that are associated with the requested samples.

Centralized unlearning methods are not suited to federated learning, due to (i) lack of direct data access, (ii) the fact that the global model is updated based on the aggregated rather than the raw gradients, and (iii) the possibility that different users may have similar training samples [7]. This motivated the emergence of *federated unlearning*, which focuses on deleting the impact of revoked data in federated learning.

Only a few studies proposed federated unlearning mechanisms using methods such as gradient subtraction (e.g., [8], [10]), gradient scaling (e.g., [7], [16]), isolation (e.g., [17]), null space calibration (e.g., [18]), or knowledge distillation (e.g., [11]). Albeit with good numerical performance, there is usually no theoretical guarantee of these proposed federated unlearning methods. Several papers (e.g., [19], [20]) have performed theoretical convergence analysis on federated unlearning. However, they didn't reveal the relationship

between the model accuracy loss and staying users' data or the relationship between unlearning time and leaving users' data. To fill this gap, we derive theoretical bounds on the model optimality gap and communication rounds for one approach to federated unlearning in this paper.

Additionally, there is a wide spectrum of literature on incentive mechanisms for various systems, including crowdsensing (e.g., [21]), wireless networks (e.g., [22]), data trading (e.g., [23]), and energy sharing (e.g., [24]). Some authors have studied incentive mechanism design for federated learning to discourage valuable clients from leaving (e.g., [25], [26], [27], [28], [29], [30]). However, very few of them considered users' multi-dimensional private information[1] (e.g., [26]), and none of them incorporated the unique aspects of federated unlearning (e.g., unlearning costs) or the dynamics of users' payoffs (e.g., pre-/post-training and before/after some users leave). This paper focuses on incentive mechanism design for both federated learning and unlearning.

Regarding the mechanism design for federated unlearning, Xia *et al.* presented four desirable properties for the data valuation with the sharded structure in machine unlearning and proposed S-Shapley value to measure the contribution of data effectively and efficiently [31]. Ding *et al.* studied users' strategic data revocation in federated unlearning [32]. However, these studies didn't consider the incentive issue in federated unlearning. Lin *et al.* in [33] and our prior work in [34] proposed incentive mechanisms for federated unlearning, without comparing with the unlearning forbidden scenario or investigating whether allowing unlearning is economically beneficial. This work is the first to study the economic benefit of incentivized federated unlearning.

The rest of the paper is organized as follows. In Section 2, we characterize some models of federated learning and unlearning, which form the basis for the system model described in Section 3. We give the optimal incentive mechanisms in the unlearning-allowed and unlearning-forbidden scenarios in Sections 4 and 5, respectively. We provide simulation results in Section 6 and conclude in Section 7.

2 Characterization of Federated Learning and Unlearning Models

Before modeling the game-theoretic interaction between the server and the users in the next section, we first discuss federated learning and unlearning models in this section as a preliminary. Specifically, we specify the learning and unlearning objectives in Sections 2.1 and 2.2, respectively. Then, we derive bounds on global model accuracy and federated unlearning time in Section 2.3.

2.1 Federated Learning Objective

Consider an example of data (x_a, y_a), where x_a is the input (e.g., an image) and y_a is the label (e.g., the object in the image). The objective of learning is to find the proper model parameter w that can predict the label y_a based on the input x_a. Let us denote the prediction value as $\tilde{y}(x_a; w)$. The gap between the prediction $\tilde{y}(x_a; w)$ and the ground truth label y_a is characterized by the prediction loss function $f_a(w)$. If user i selects a set of local data with data size d_i to train the model, the loss function of user $i \in \mathcal{I}$ is the average prediction loss on all his training data:

$$F_i(w) = \frac{1}{d_i} \sum_{a=1}^{d_i} f_a(w). \tag{1}$$

The purpose of federated learning is to compute the model parameter w by using all users' local data. The optimal model parameter w^* minimizes the global loss function, which is an average of all users' loss functions [12], [35]:[2]

$$w^* = \arg\min_w F(w) \triangleq \arg\min_w \frac{1}{I} \sum_{i \in \mathcal{I}} F_i(w), \tag{2}$$

where I denotes the number of users in $\mathcal{I}$.

2.2 Federated Unlearning Objective

A federated learning process maps users' data into a model space, while a federated unlearning process maps a learned model, users' data set, and the data set that is required to be forgotten into an unlearned model space. The goal of federated unlearning is to make the unlearned model have the same distribution as the retrained model (i.e., retrained from scratch using the remaining data).[3]

A natural method for federated unlearning is to let the remaining users (excluding leaving users) continue training from the learned model w^*, until it converges to a new optimal model parameter $\tilde{w}^*$ that minimizes the global loss function of remaining users:

$$\tilde{w}^* = \arg\min_w \frac{1}{I - I_{leave}} \sum_{i \in \mathcal{I} \setminus \mathcal{I}_{leave}} F_i(w), \tag{3}$$

where $\mathcal{I}_{leave}$ is the set of I_{leave} users who leave the system through federated unlearning. This method is typically more efficient than training from scratch, as the minimum point may not change much after some users leave.

2.3 Model Accuracy and Unlearning Time

Given the objectives of federated learning and unlearning, we analyze the model accuracy gap and unlearning time in the following.

We use two widely adopted algorithms, Scaffold [12] and FedAvg [1], as the federated learning algorithms when deriving the optimality gap of the global model. In each local iteration of the algorithm, every user computes a minibatch gradient with batch size s_i. A batch or minibatch refers to equally sized subsets of the training dataset over which the gradient is calculated. In this paper, we consider the widely adopted setting that users' batch sizes $\{s_i\}_{i \in \mathcal{I}}$ are in the same proportion to their data sizes $\{d_i\}_{i \in \mathcal{I}}$ (i.e., $s_i = \iota d_i, \forall i \in \mathcal{I}, \iota \in (0, 1)$) [9], [26], [36].

1. The multi-dimensional private information means several types of information (e.g., training costs and privacy costs) that is only known by users themselves and unknown to others.

2. This model treats each user equally. Some papers (e.g., [1]) adopted another objective, a weighted sum of all users' losses, where the weights (i.e. $d_i / \sum_{i=1}^{I} d_i$) reflect the differences in data size. The two objectives are equivalent when users' data sizes are the same. Our results can be easily extended to the weighted case.

3. The distribution is due to the randomness in the training process (e.g. randomly sampled data and random ordering of batches).

The following proposition presents bounds on the optimality gap for the global models trained with Scaffold or FedAvg:

Proposition 1. *Suppose each user's loss function F_i is μ-strongly-convex and L-Lipschitz-smooth. Consider federated learning algorithms Scaffold and FedAvg, where K_i is the number of local iterations per global round for user i with local step size η_i. Set $\bar{\eta} = \eta_i K_i$. Then, we have for Scaffold with $\bar{\eta} \leq \frac{1}{12L}$,*

$$\mathbb{E}\|w_{t+1} - w^*\|^2 \leq (1 - \frac{\mu\bar{\eta}}{2})\mathbb{E}\|w_t - w^*\|^2 + \frac{22\bar{\eta}^2\sigma^2}{I}\sum_{i\in\mathcal{I}}\frac{1}{s_i}, \quad (4)$$

where w_{t+1} and w_t represent the model parameter after global round $t+1$ and t, respectively, s_i is user i's local batch size, and σ^2 is a variance bound of each data sample.[4] For FedAvg with $\bar{\eta} \leq \frac{1}{12LB}$,

$$\mathbb{E}\|w^{t+1} - w^*\|^2 \leq (1 - \frac{\mu\bar{\eta}}{2})\mathbb{E}\|w_t - w^*\|^2 + 6\bar{\eta}^2G^2 + \frac{19\bar{\eta}^2\sigma^2}{I}\sum_{i\in\mathcal{I}}\frac{1}{s_i}, \quad (5)$$

when the bounded dissimilarity assumption [12] is satisfied, i.e., there exist some constants $G \geq 0$ and $B \geq 1$ such that $(1/I)\sum_{i=1}^{I}\|\nabla F_i(w)\|^2 \leq G^2 + B^2\|\nabla F(w)\|^2, \forall w$.

Moreover, by selecting $\bar{\eta} = \frac{c}{t+1}$ for some $c > 0$, we have that the expected optimality gap of the global model satisfies: for Scaffold with $c \leq \frac{1}{12L}$,

$$\mathbb{E}\|w_t - w^*\|^2 \leq \frac{1}{t+1}\left(\frac{b_1(c)\sigma^2}{I}\sum_{i\in\mathcal{I}}\frac{1}{s_i} + \|w_0 - w^*\|^2\right), \quad (6)$$

and for FedAvg with $c \leq \frac{1}{12LB}$,

$$\mathbb{E}\|w_t - w^*\|^2 \leq \frac{1}{t+1}\left(\frac{b_2(c)\sigma^2}{I}\sum_{i\in\mathcal{I}}\frac{1}{s_i} + b_2(c)G^2 + \|w_0 - w^*\|^2\right), \quad (7)$$

where $b_m(c), m = 1, 2$ are some monotonically increasing functions of c.

The proof of Proposition 1 is given in Appendix 1. As a large optimality gap $\|w_t - w^*\|^2$ means a high accuracy loss of the global model, Proposition 1 presents a relationship between the expected global model accuracy loss and the users' data sizes. As shown in (6) and (7), the expected accuracy loss of the global model decreases in the users' training batch sizes $\{s_i\}_{i\in\mathcal{I}}$ (and thus data sizes $\{d_i\}_{i\in\mathcal{I}}$). Moreover, we explain two asymptotic cases of (6) and (7) for better understanding. When the initial point is optimal (i.e., $w_0 = w^*$), the bound does not go to zero due to sample randomness. When batch size s_i is large enough, the randomness is then highly reduced and the bound is controlled by the initialization of the algorithm, i.e., the farther the initial point w_0 is from the optimal solution w^*, the more iterations are needed.

Then, after applying the result in Proposition 1 to the natural unlearning model introduced in Section 2.2, we have the following proposition about federated unlearning rounds:

4. To estimate the true gradient $\nabla F_i(w)$, we uniformly sample one data point to generate a gradient estimate $g_i(w)$ and assume $\mathbb{E}\|g_i(w) - \nabla F_i(w)\|^2 \leq \sigma^2$ for any w.

Proposition 2. *Consider the same conditions of Proposition 1 with diminishing step size $\bar{\eta}$ and suppose*

$$b_m(c) \leq \frac{1}{(I - I_{leave})\mu^2}\frac{\left(\sum_{i\in\mathcal{I}_{leave}}\|\nabla F_i(w^*)\|\right)^2}{G^2\mathbb{1}_{m=2} + \sum_{i\in\mathcal{I}\setminus\mathcal{I}_{leave}}\frac{1}{s_i}\sigma^2},$$

where $m = 1$ when using Scaffold and $m = 2$ for FedAvg. It will require at most

$$T_{unlearn} := \left\lfloor \frac{2(I-1)}{\epsilon^2\mu^2}\sum_{i\in\mathcal{I}_{leave}}\|\nabla F_i(w^*)\|^2 \right\rfloor \quad (8)$$

rounds of communication to guarantee $\mathbb{E}\|w_{T_{unlearn}} - \tilde{w}^*\| \leq \epsilon$ when starting from the original learned model w^*, where the new model $\tilde{w}^*$ is defined in (3).

The proof of Proposition 2 is given in Appendix 2. Each user's gradient $\|\nabla F_i(w^*)\|$ can represent his training loss (denoted as ℓ_i) because the calculated gradient increases in the loss. Hence, Proposition 2 reveals the relationship between the number of communication rounds required for federated unlearning and the training losses of leaving users. As indicated in (8), a larger total training loss of the leaving users $\sum_{i\in\mathcal{I}_{leave}}\ell_i^2$ (i.e., a larger $\sum_{i\in\mathcal{I}_{leave}}\|\nabla F_i(w^*)\|^2$) requires more communication rounds $T_{unlearn}$ to achieve unlearning.

We will apply the derived results about model accuracy loss and unlearning rounds in building the system model in the next section.

3 System Model

We consider a federated learning and unlearning system consisting of a set of heterogeneous users with private data and a central server. As illustrated in Fig. 1, the server first incentivizes users to participate in a federated learning phase through a contract. However, some users may later choose to revoke their data and leave the system. In response, the server can provide further incentives to retain valuable users. Upon the final exit of some users from the system, the remaining users collectively execute an algorithm to unlearn the leaving users' data.

In the following, we first divide the heterogeneous users into different types for the convenience of incentive design, then formulate a multi-stage game between the strategic server and users, and finally specify the payoffs of the server and the users (i.e., their optimization objectives) in two unlearning scenarios, respectively.

3.1 User Type

We consider a set $\mathcal{I} \triangleq \{1, 2, ..., I\}$ of users in the system with two-dimensional private information: marginal cost for training effort θ and marginal perceived privacy cost ξ.[5] We refer to a user with (θ_j, ξ_j) as a type j user. We further assume that the I users belong to a set $\mathcal{J} \triangleq \{1, 2, ..., J\}$ of J types. Each type j has I_j users, with $\sum_{j\in\mathcal{J}} I_j = I$. The total number of users I and the number of each type I_j are

5. The ξ represents how much a user values the privacy of his data. If two users have the same data, the user with a higher ξ has a greater concern for privacy than the one with a lower ξ.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3557857

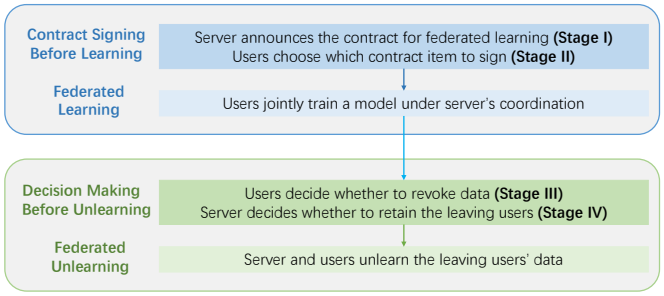


Fig. 1. Framework of federated learning and unlearning system with incentive mechanisms.

public information, but each user's specific type is private information.[6]

Under private information, it is difficult for the server to predict users' strategies. To this end, we propose to design a contract mechanism for the server to elicit information.

3.2 Games and Strategies

We use a multi-stage Stackelberg game to model the interaction between the server and users in each of the two scenarios.

3.2.1 *Unlearning-Allowed Scenario*

When unlearning is allowed, we consider the following four-stage game that captures the move sequence of the server and the users:

- Stage I: The server designs a federated learning incentive contract $\boldsymbol{\phi} \triangleq \{\phi_j\}_{j\in\mathcal{J}}$, which contains J contract items (one for each user type). Each contract item $\phi_j \triangleq \left(d_j, r_j^L\right)$ specifies the relationship between the required data size d_j of each type-j user (for local computation) and the corresponding learning reward r_j^L.[7]
- Stage II: Users decide which contract item to choose or to not participate in. Then, they jointly implement the federated learning algorithm (Scaffold or FedAvg).
- Stage III: Users decide whether to revoke data after federated learning. We denote a user i's revocation decision as

$$x_i = \begin{cases} 0, & \text{if user } i \text{ does not revoke data,} \\ 1, & \text{if user } i \text{ revokes his data,} \end{cases} \tag{9}$$

and denote the set of users who revoke their data as $\mathcal{I}_u$. If a type-j user revokes his data, then he needs to fully return the reward r_j^L to the server.[8] We consider that the server will announce users' training losses $\{\ell_i\}_{i\in\mathcal{I}}$ (without specifying users) after federated learning to help users decide whether to revoke data.[9]
- Stage IV: The server decides the set of leaving users to retain $\mathcal{I}_r$ and designs the corresponding retention incentives $\left\{r_i^U\right\}_{i\in\mathcal{I}_r}$, such that those receiving the retention incentives will choose to stay in the system and those without will leave.[10] The remaining users and server collectively implement federated unlearning.

In Stage III, we use $\ell_i = ||\nabla F_i(w_T)||$ to represent the training loss, where w_T is the solution obtained after T iterations of Scaffold or FedAvg. We assume T is large enough, such that w_T and w^* are close. A large ℓ_i implies the federated solution is far away from the minimizer of local loss function F_i and therefore a larger training loss.

After federated learning, the server and users have more information in Stages III and IV compared with Stages I and II. For example, the users will know their training losses $\{\ell_i\}_{i\in\mathcal{I}}$. The server can evaluate the users' contribution to the global model (denoted by $\{v_i\}_{i\in\mathcal{I}}$), and it will know each user's type by observing users' contract item choices. We summarize their knowledge about some key information in the four stages in Table 1 and list the key notations

TABLE 1
The Server and Users' Knowledge in Different Stages

Stage	Known	Unknown
Server in Stage I	$\mathcal{J}, \{I_j\}_{j\in\mathcal{J}}$	$\{\theta_i, \xi_i, \ell_i, v_i\}_{i\in\mathcal{I}}$
User in Stage II	his own type (θ_i, ξ_i)	other users' types, $\{\ell_i, v_i\}_{i\in\mathcal{I}}$
User in Stage III	his own type (θ_i, ξ_i), $\{\ell_i\}_{i\in\mathcal{I}}$	other users' types, $\{v_i\}_{i\in\mathcal{I}}$
Server in Stage IV	$\mathcal{J}, \{I_j\}_{j\in\mathcal{J}}$, $\{\theta_i, \xi_i, \ell_i, v_i\}_{i\in\mathcal{I}}$	—

TABLE 2
Key Notations

θ_j	Marginal training cost of type-j users
ξ_j	Marginal perceived privacy cost of type-j users
I_j	Number of type-j users
$j/\mathcal{J}$	Index/Set of user types in the system
$i/\mathcal{I}$	Index/Set of users in the system
$\mathcal{I}_u$	Set of users who revoke their data in Stage III
$\mathcal{I}_r$	Set of users who are retained by the server in Stage IV
ϕ_j	Contract item designed for type-j users
d_j	Required data size for each type-j user in the contract
r_j^L	Learning reward for each type-j user in the contract
r_i^U	Unlearning reward (retention incentive) for user i
x_i	User i's data revocation decision
p_j	Historical revocation rate of type-j users
q_j	Historical retention rate of type-j users
T	Number of communication rounds of federated learning
λ	Coefficient related to unlearning communication rounds
ϱ	Coefficient related to expected accuracy loss
γ	Server's weight on incentive rewards
v_i	User i's contribution to global model accuracy
ℓ_i	User i's training loss (representing $||\nabla F_i(x^*)||$)

6. Note that users (even in the same user type) have different data (i.e., training losses and contributions to the model). In reality, the server can always divide users with different training and privacy costs into several groups based on market research, and each group with similar costs can be approximated as a super-type to simplify the server's contract design for the convenience of implementation. The server can have knowledge about statistics of type information through market research and past experiences, but it may be hard for it to know each user's private type (such as devices, computational capacities, and privacy preferences). Our framework can also be easily adapted to higher dimensional information scenarios.

7. We consider that users can generate the required amount of data if they participate. The server's primary goal is to improve model performance, so the contract in this paper ties training data contributions to rewards.

8. If there is no such return policy, every user can first participate to get rewards and then revoke data to reduce costs, resulting in a catastrophic failure of model training collaboration and a huge cost to the server.

9. It is not obvious that a strategic server would make such an announcement, but it can be stipulated by regulations for protecting users' right to be forgotten. If we do not make this assumption, the problem will be even simpler. As we shall see in the analysis in Section 4.2, we just need to replace other users' training losses $\{\ell_k\}_{k\in\mathcal{I}}$ in (23) with the same expected loss $\mathbb{E}[\ell]$ and solve the problem through a similar approach.

10. In this case, $\mathcal{I}_u \backslash \mathcal{I}_r$ is the set of users who finally leave the system, and $\mathcal{I} \backslash (\mathcal{I}_u \backslash \mathcal{I}_r)$ is the set of users who finally stay.

in this paper in Table 2.[11]

Moreover, in Stage IV, the server has enough information to know whether the users will accept the retention incentives. Therefore, we do not model a Stage V in which the users decide to accept or not accept the retention incentives. After that, as in Fig. 1, the staying users perform federated unlearning under the server's coordination, which makes staying users incur unlearning costs. We will specify the payoffs and costs of the server and users in each stage of the game in the next subsection.

3.2.2 Unlearning-Forbidden Scenario

Without the unlearning process, we consider a two-stage game that only includes Stages I and II from the unlearning-allowed case.

3.3 Payoffs in the Unlearning-Allowed Scenario

At each stage, every user or the server seeks to maximize his expected payoff (or minimize his expected cost) based on his current knowledge. As knowledge updates occur between stages, the payoffs of the users or the server (maximization or minimization objectives respectively) take different forms in each stage.

3.3.1 Server's Payoff in Stage I

The server's objective in Stage I is to minimize the sum of the expected accuracy loss of the global model and the expected total incentive rewards for users.

First, we specify the expected model accuracy loss, which depends on the data of users who finally stay in the system. Since the server cannot predict which users will leave and who to retain due to the lack of information in Stage I, it can only base its decision on knowledge of the typical user distribution. Specifically, we assume that according to the historical experience and market statistics, the server knows the probability of a type-j user revoking his data (i.e., his revocation rate) p_j and the probability that a type-j user who wants to revoke data is retained (i.e., his retention rate) q_j, where p_j and q_j are independent. Following Proposition 1, we model the server's expected accuracy loss after federated unlearning as:[12]

$$\frac{\varrho}{T}\sum_{j\in\mathcal{J}} I_j(1-p_j+p_jq_j)\frac{1}{d_j}, \tag{10}$$

where T is the number of communication rounds of federated learning, ϱ is a coefficient related to the sample variance, and $1-p_j+p_jq_j$ is the percentage of type j users remaining in the system in the end. This captures that the expected model accuracy loss decreases in the data sizes of all staying users.[13]

The server's payoff also includes the cost of all rewards it pays to users, which comprises the initial contract announced in Stage I and incentives offered to encourage leaving users to remain in Stage IV. If all users choose to participate in the contract and choose their corresponding contract items,[14] the expected total learning reward is $\sum_{j\in\mathcal{J}} I_j(1-p_j+p_jq_j)r_j^L$. Note that if a type-$j$ user successfully revokes his data, he needs to fully return the reward r_j^L to the server. The server's expected incentive for retaining leaving users is $\mathbb{E}[\sum_{i\in\mathcal{I}_r} r_i^U]$, which depends on p, q, and training losses and will be calculated through backward induction in Section 4.4.

Combining these terms, the server's expected cost in Stage I is

$$W^{s-1} = \frac{\varrho}{T}\sum_{j\in\mathcal{J}} I_j(1-p_j+p_jq_j)\frac{1}{d_j} + \gamma\left(\sum_{j\in\mathcal{J}} I_j(1-p_j+p_jq_j)r_j^L + \mathbb{E}\Big[\sum_{i\in\mathcal{I}_r} r_i^U\Big]\right), \tag{11}$$

where γ is how much weight the server puts on the incentive reward payments compared to the model accuracy loss. A smaller γ means that the server is less concerned about minimizing the incentive rewards and more concerned about reducing the accuracy loss.

3.3.2 Users' Payoffs in Stage II

In the overall game, there are three possible outcomes for a user (not revoke data, revoke and retained, revoke and not retained). However, in this stage, a user does not have enough information to know which outcome will realize, so he must calculate his expected payoff by considering three cases:

- *Case (a): not revoke.* With probability $1-p_j$, a type-j user will not revoke his data after federated learning. In this case, his expected payoff is the difference between the learning reward r_j^L and costs (including the learning cost, privacy cost, and unlearning cost):

$$U_{j,a}^{s-2} = r_j^L - \theta_j d_j T - \xi_j \mathbb{E}[\ell_j] d_j - \mathbb{E}\Big[\theta_j d_j \lambda \sum_{i\in\mathcal{I}_u\setminus\mathcal{I}_r} \ell_i^2\Big], \tag{12}$$

where $\theta_j d_j T$ is the total learning cost in T rounds. As we consider that each user's sampled data size in each local round is proportional to his total data size, the learning cost is linear in his data size d_j (e.g., [9], [26], [36]). Similarly, in the unlearning cost $\theta_j d_j \lambda \sum_{i\in\mathcal{I}_u\setminus\mathcal{I}_r} \ell_i^2$, $\lambda\sum_{i\in\mathcal{I}_u\setminus\mathcal{I}_r}\ell_i^2$ models the number of communication rounds for unlearning, which increases in the leaving users' training losses (according to Proposition 2).[15] A type j user's perceived privacy cost $\xi_j\mathbb{E}[\ell_j]d_j$ increases in his expected training loss $\mathbb{E}[\ell_j]$ and data size d_j. As

11. As analyzing the four-stage game is complicated, this paper does not model the information update in a fully Bayesian framework but specifies plausible beliefs that the players hold in each stage.

12. It is similar to the expected accuracy loss of the model retrained by the remaining users.

13. As the server aims to incentivize users to contribute data in federated learning, we only model the impact of data sizes and omit the independent term about the initial point w_0 in (6) and (7). Since we consider that users' batch sizes $\{s_i\}_{i\in\mathcal{I}}$ are in the same proportion to their data sizes $\{d_i\}_{i\in\mathcal{I}}$, it is equivalent to substitute s_i with d_i in (6) and (7).

14. As we shall see in Section 4.4, we will design the contract to ensure that each user will participate (i.e., individual rationality) and choose the contract item designed for his type (i.e., incentive compatibility).

15. We use the simplified model of (8) in Proposition 2 to capture the key relationship between the unlearning communication rounds $T_{unlearn}$ and leaving users' training losses (represented by $\|\nabla F_i(w^*)\|$).

a high training loss ℓ_j reflects a large distance of user j's data from the average of other users' distribution, we use it to measure the uniqueness of a user. Thus, the model captures that the privacy cost increases in the uniqueness and size of one's training data (e.g., [37], [38]). As each user cannot know his exact training loss ℓ_j before federated learning, we assume that he estimates the expected loss using the public distribution (with mean $\mathbb{E}[\ell_j]$ and variance $D(\ell_j)$).

- *Case (b): revoke but retained.* With probability $p_j q_j$, a type-j user will revoke his data after federated learning but will be retained by the server through more incentives r_j^U. In this case, his expected payoff is the difference between total rewards (including both learning and unlearning incentives) and costs:

$$U_{j,b}^{s-2} = r_j^L + \mathbb{E}\left[r_j^U\right] - \theta_j d_j T - \xi_j \mathbb{E}[\ell_j] d_j - \mathbb{E}\left[\theta_j d_j \lambda \sum_{i \in \mathcal{I}_u \setminus \mathcal{I}_r} \ell_i^2\right]. \quad (13)$$

  The unlearning incentive r_j^U will be determined by the server in Stage IV based on users' training losses, contributions, and data revocation, which are unknown in this stage. Thus, each user can only calculate the expectation of the unlearning incentive.
- *Case (c): revoke and not retained.* With probability $p_j(1-q_j)$, a type-j user will revoke his data and will not be retained by the server, i.e., the user's data will be unlearned. The user needs to return the reward r_j^L to the server but will not incur any privacy cost or unlearning cost. In this case, his expected payoff is

$$U_{j,c}^{s-2} = -\theta_j d_j T, \quad (14)$$

  which is the sunk training cost from federated learning.

In summary, a type-j user's expected payoff if he participates in Stage II is

$$U_j^{s-2} = (1-p_j)U_{j,a}^{s-2} + p_j q_j U_{j,b}^{s-2} + p_j(1-q_j)U_{j,c}^{s-2}. \quad (15)$$

If $U_j^{s-2} \geq 0$, the type-j user will choose to participate in the federated learning in Stage II.

3.3.3 Users' Payoffs in Stage III

After federated learning, each user i has knowledge about his training loss ℓ_i. If user i chooses not to revoke his data, his expected payoff in Stage III is (updating (12) in Case (a) with the realized training loss ℓ_i):

$$U_{i,a}^{s-3} = r_i^L - \theta_i d_i T - \xi_i \ell_i d_i - \mathbb{E}\left[\theta_i d_i \lambda \sum_{k \in \mathcal{I}_u \setminus \mathcal{I}_r} \ell_k^2\right]. \quad (16)$$

The reason for using expectation here is that users do not know the set of retained users $\mathcal{I}_r$ determined in Stage IV. Users' expected payoffs of Cases (b) and (c) in Stage III follow the same approach (i.e., updating (13) and (14) with the realized training loss ℓ_i).

Note that users of the same type may have different training losses and thus different payoffs, so the payoff in Stage III is user-specific instead of type-specific. Moreover, after some users leave, the remaining users' training losses may change as the global model will be updated. Since users cannot accurately predict their future expected loss even if they know all users' current losses, we assume that each user still approximates his future expected loss as equal to his current loss.

3.3.4 Server's Payoff in Stage IV

When some users want to leave the system, it is important for the server to know their contributions to the global model for retaining valuable users.

A fair and effective method to compute a user's contribution to a coalition is the Shapley value [39]. Wang et al. [40] introduced a related concept called federated Shapley value to evaluate each user's contribution in a federated learning setting.[16] The federated Shapley value for user i, denoted as v_i, is calculated by the server during the federated learning process and is unknown to the users.

After obtaining users' contributions (federated Shapley values), the server can calculate its realized cost in Stage IV. This cost is the sum of two factors: the realized accuracy loss, which is estimated by the sum of federated Shapley values of all users who remain in the system, and the realized incentives, i.e.,

$$W^{s-4} = \sum_{i \in \mathcal{I} \setminus (\mathcal{I}_u \setminus \mathcal{I}_r)} v_i + \gamma \left(\sum_{i \in \mathcal{I} \setminus (\mathcal{I}_u \setminus \mathcal{I}_r)} r_i^L + \sum_{i \in \mathcal{I}_r} r_i^U \right). \quad (17)$$

The first term in (17) represents the model accuracy loss, the second is the learning reward paid to all remaining users for participation in federated learning, and the last term is the total retention incentive. The additivity property of federated Shapley values allows the server to compare all the possible sets of users to retain and find the optimal one. Note that a smaller federated Shapley value is better, as it means a larger contribution to the accuracy of the global model, and the federated Shapley values can be negative.

3.4 Payoffs in the Unlearning-Forbidden Scenario

Similar to Stages I and II in the unlearning-allowed scenario, we now specify the users and the server's payoffs in the unlearning-forbidden scenario. The difference here is that there are no unlearning considerations (e.g., data revocation or retention incentives). We will use the superscript $'$ for the unlearning-forbidden scenario to differentiate the notations in the two scenarios.

3.4.1 Server's Payoff in Stage I

The server needs to minimize the sum of the expected accuracy loss and the incentive rewards paid for federated learning:

$$W = \frac{\varrho}{T} \sum_{j' \in \mathcal{J}'} \frac{I'_{j'}}{d'_{j'}} + \gamma \sum_{j' \in \mathcal{J}'} I'_{j'} r_{j'}^{L'}. \quad (18)$$

16. This method calculates the marginal contribution of each user by assessing how much value that user adds to the model accuracy across all possible subsets of users. Due to the space limit, we present the detailed method and properties in Appendix 12. Our mechanisms can also be applied to other contribution measurement methods.

3.4.2 Users' Payoffs in Stage II

A type-j' user's payoff is the difference between the learning reward and training costs (including the learning cost and perceived privacy cost)

$$U_{j'} = r_{j'}^{L\prime} - \theta_{j'} d'_{j'} T - \xi'_{j'} \mathbb{E}[\ell'_{j'}] d'_{j'}. \tag{19}$$

Note that the marginal perceived privacy cost $\xi'_{j'}$ here should be no smaller than that in the unlearning-allowed scenario ξ_j for the same user. This is because a user cannot revoke his data once he decides to participate in the unlearning-forbidden scenario and thus may incur larger privacy concerns.

Next, we will use the standard backward induction to analyze the server and users' optimal strategies in the two unlearning scenarios.

4 Optimal Incentive Mechanism in Unlearning-Allowed Scenario

In this section, we analyze an optimal incentive mechanism for the unlearning-allowed scenario. Based on backward induction, we will derive the optimal strategies from Stage IV to Stage I in Sections 4.1-4.4, respectively.

4.1 Server's Retention Strategies in Stage IV

Given the server's contract ϕ in Stage I, the users' contract item choices in Stage II, and the users' revocation decisions $\mathcal{I}_u$ in Stage III, the server needs to determine which users to retain $\mathcal{I}_r$ and the corresponding retention incentives $\{r_i^U\}_{i\in\mathcal{I}_r}$ in Stage IV.

As we discussed in Section 3.3.4, the server seeks to minimize the cost in (17) in Stage IV, which can be formulated as follows:

Problem 1 (Server's Optimization Problem in Stage IV)**.**

$$\min \sum_{i\in\mathcal{I}\setminus(\mathcal{I}_u\setminus\mathcal{I}_r)} v_i + \gamma \left(\sum_{i\in\mathcal{I}\setminus(\mathcal{I}_u\setminus\mathcal{I}_r)} r_i^L + \sum_{i\in\mathcal{I}_r} r_i^U \right) \tag{20a}$$

$$\text{s.t. } r_i^U + r_i^L - \theta_i d_i T - \xi_i \ell_i d_i - \theta_i d_i \lambda \sum_{k\in\mathcal{I}_u\setminus\mathcal{I}_r} \ell_k^2 \geq -\theta_i d_i T, \forall i \in \mathcal{I}_r \tag{20b}$$

$$\text{var. } \mathcal{I}_r \subseteq \mathcal{I}_u, \{r_i^U\}_{i\in\mathcal{I}_r}. \tag{20c}$$

The constraint (20b) is to ensure that the retention incentives are enough to make the target users stay in the system. The left-hand side of the constraint is a user i's payoff after accepting the retention incentive (including unlearning reward, learning reward, learning cost, privacy cost, and unlearning cost), and the right-hand side is his payoff of not accepting (i.e., he has to return the learning reward to the server and only has sunk learning cost).

The following proposition presents the solution to Problem 1.

Proposition 3. *The server's optimal set of users to retain is*

$$\mathcal{I}_r^* = \arg\min_{\mathcal{I}_r\subseteq\mathcal{I}_u} \sum_{i\in\mathcal{I}_r} \left(v_i + \gamma\theta_i d_i \lambda \sum_{k\in\mathcal{I}_u\setminus\mathcal{I}_r} \ell_k^2 + \gamma\xi_i \ell_i d_i \right), \tag{21}$$

and the optimal retention incentives are

$$r_i^{U*} = \theta_i d_i \lambda \sum_{k\in\mathcal{I}_u\setminus\mathcal{I}_r^*} \ell_k^2 + \xi_i \ell_i d_i - r_i^L, \forall i \in \mathcal{I}_r^*. \tag{22}$$

The proof of Proposition 3 is given in Appendix 3. Proposition 3 highlights a trade-off regarding the retention of users and their training losses. Users who have larger training losses incur higher privacy costs and thus require higher incentives to retain (indicated by $\gamma\xi_i\ell_i d_i$ in (21)). However, retaining such users also helps reduce the unlearning costs since the objective in (21) increases with the aggregated loss of the leaving users. Furthermore, the server has the incentive to retain users who contribute more to the model accuracy, which corresponds to smaller values of v_i. Additionally, users with smaller marginal costs θ_i and ξ_i are also desirable to reduce unlearning incentives.[17]

4.2 Users' Revocation Decisions in Stage III

Considering the server's optimal retention strategies in Stage IV, each user i decides whether to revoke his data in Stage III given the information announced in Stages I and II.

Based on the server's optimal retention incentives (22) and the user's payoffs in Stage III (i.e., the updated (13) and (14) with realized losses), a user i's payoff after revoking data is $-\theta_i d_i T$, regardless of whether the user is retained by the server or not. Thus, user i's expected payoff in Stage III can be rewritten as

$$U_i^{s-3}(x_i; x_{-i}) = x_i \left(-\theta_i d_i T\right) + (1 - x_i) \left[r_i^L - \theta_i d_i T - \xi_i \ell_i d_i - \theta_i d_i \lambda \sum_{k\in\mathcal{I}} x_k (1-q) \ell_k^2 \right], \tag{23}$$

where $x_{-i} = \{x_k\}_{k\in\mathcal{I}\setminus\{i\}}$ is the revocation decisions of all users except user i and $q = \mathbb{E}[q_j]$ is the expected retention rate of all users, as users do not know each other's type.[18] As shown in (23), each user's payoff depends on the other users' revocation decisions, so users engage in a non-cooperative game in Stage III.

We formally define users' non-cooperative sub-game as follows.

Sub-Game 1 (Users' Revocation Sub-Game in Stage III)**.**

17. Note that in (21), the server may not only include users with a negative value in the brackets, as retaining some users with positive values may reduce the server's objective through the aggregated losses. This is an integer programming problem. When the number of leaving users I_u is large, the server can reduce the complexity by classifying the leaving users into several categories to retain, each category with similar contributions and costs.

18. Here we use the historical retention rate q to calculate the expected payoffs instead of the retention rate obtained in Stage IV (i.e., $|\mathcal{I}_r^*|/|\mathcal{I}_u|$). This is because users do not know their federated Shapley values and cannot calculate $\mathcal{I}_r^*$. If they calculate the expectation $\mathbb{E}[\mathcal{I}_r^*]$ based on type statistics, according to (21), the result will be user type retention instead of user retention (e.g., retain all type-i users and not retain all type-j users regardless of different data distributions and losses of the same type of users), which is not true. Conversely, historical rates ranging between $[0, 1]$ allow for more realistic partial retention of same-type users. Therefore, we assume that the users have a belief at this stage in the retention rate which is the same as the historical rate. In the following analysis in Stages I and II, we will also use the historical rates for calculating the expected cost/payoffs for similar reasons.

- *Players: all users in set $\mathcal{I}$.*
- *Strategy space: each user $i \in \mathcal{I}$ decides whether to revoke his data, i.e., $x_i \in \{0, 1\}$ (0: not revoke, 1: revoke).*
- *Payoff function: each user $i \in \mathcal{I}$ maximizes his payoff in* (23).

The following proposition characterizes the Nash equilibrium (NE) of Sub-Game 1:

Proposition 4. *Sub-Game 1 is a supermodular game, where pure NE exists but may not be unique. Algorithm 1 converges to one NE.*

Algorithm 1: Users' optimal revocation decisions

Input : $\{r_i^L, \xi_i, \ell_i, d_i, \theta_i\}_{i\in\mathcal{I}}, \lambda, q$
Output: Optimal revocation decisions $\{x_i^*\}_{i\in\mathcal{I}}$
1 Initialize $x_i^* \leftarrow 0, i \in \mathcal{I}$;
2 **while** $\exists x_i^* =$ 0 & $r_i^L - \xi_i \ell_i d_i - \theta_i d_i \lambda (1-q) \sum_{k\in\mathcal{I}\setminus\{i\}} x_k \ell_k^2 < 0$ **do**
3 $\quad x_i^* \leftarrow 1, \forall i$ satisfying conditions in line 2;
4 **end**

The proof of Proposition 4 is given in Appendix 4. Based on Algorithm 1, we can find the set of users who revoke data in one NE, i.e., $\mathcal{I}_u^* = \{i : x_i^* = 1, i \in \mathcal{I}\}$. Basically, Algorithm 1 corresponds to doing the best response updates of the users starting from all users choosing not to revoke (i.e., 0). It is well known that for supermodular games, these updates will converge monotonically to an NE. Algorithm 1 will terminate within I iterations.[19] The resulting equilibrium strategies and insights will be illustrated through simulation in Section 6.1.2.

4.3 Users' Contract Item Choices in Stage II

Based on the analysis in Stages III and IV, a type-j user's expected payoff in Stage II (15) can be rewritten as:

$$U_j^{s-2} = (1-p_j) r_j^L - \kappa_j d_j, \tag{24}$$

where

$$\kappa_j \triangleq (1-p_j)\xi_j \mathbb{E}[\ell_j] + \theta_j T + \theta_j (1-p_j)\lambda \sum_{m\in\mathcal{J}} I_m p_m (1-q_m) \left(\mathbb{E}[\ell_m]^2 + D(\ell_m)\right), \tag{25}$$

and $D(\ell_m)$ is the variance of type-m users' training losses.

Each type-j user in Stage II will choose a contract item that gives him a maximum non-negative expected payoff, leading to the constraints that the server needs to consider in Stage I.

4.4 Server's Contract in Stage I

In Stage I, the server designs a contract to minimize its expected cost, considering the results in Stages II-IV.

When designing the contract, the server needs to ensure that each user achieves a non-negative payoff, so that the user will accept the corresponding contract item. Moreover, since the server does not know each user's type in Stage I, the server also needs to make a user choose the contract item intended for him (i.e., the user does not misreport his type).[20] In other words, a contract is feasible if and only if it satisfies Individual Rationality (IR) and Incentive Compatibility (IC) constraints:

Definition 1 (Individual Rationality)**.** *A contract is individually rational if each type-j user receives a non-negative payoff by accepting the contract item $\phi_j = \left(d_j, r_j^L\right)$ intended for his type, i.e.,*

$$(1-p_j) r_j^L - \kappa_j d_j \geq 0, \forall j \in \mathcal{J}. \tag{26}$$

Definition 2 (Incentive Compatibility)**.** *A contract is incentive compatible if each type-j user maximizes his own payoff by choosing the contract item $\phi_j = \left(d_j, r_j^L\right)$ intended for his type, i.e.,*

$$(1-p_j) r_j^L - \kappa_j d_j \geq (1-p_j) r_m^L - \kappa_j d_m, \forall j, m \in \mathcal{J}. \tag{27}$$

Considering the constraints in Definitions 1 and 2, the server in Stage I seeks to design the contract $\boldsymbol{\phi} = \{(d_j, r_j^L)\}_{j\in\mathcal{J}}$ to minimize its expected cost in (11), which is rewritten as follows after combining the results in Stages II-IV:

Problem 2.

$$\begin{aligned} \min \sum_{j\in\mathcal{J}} \Bigg(& \frac{\varrho I_j (1-p_j+p_j q_j)}{T d_j} + \gamma I_j (1-p_j) r_j^L \\ & + \gamma I_j p_j q_j (\alpha\theta_j + \xi_j \mathbb{E}[\ell_j]) d_j \Bigg), \\ \text{s.t. } & (1-p_j) r_j^L - \kappa_j d_j \geq 0, \forall j \in \mathcal{J}, \\ & (1-p_j) r_j^L - \kappa_j d_j \geq (1-p_j) r_m^L - \kappa_j d_m, \forall j, m \in \mathcal{J}, \\ \text{var. } & \left\{\left(d_j, r_j^L\right)\right\}_{j\in\mathcal{J}}, \end{aligned} \tag{28}$$

where

$$\alpha \triangleq \lambda \sum_{j\in\mathcal{J}} I_j p_j (1-q_j) \left(\mathbb{E}[\ell_j]^2 + D(\ell_j)\right). \tag{29}$$

Solving Problem 2 involves two challenges. First, users' multi-dimensional heterogeneity leads to a challenging multi-dimensional contract design for the server. We will simplify the analysis by summarizing users' multi-dimensional heterogeneity into several one-dimensional metrics, to guide the server's design of the optimal rewards and data sizes in the contract. Second, as the total number of IR and IC constraints is large (i.e., J^2), it is challenging to obtain the optimal contract directly. To overcome such a complexity issue, we will first transform the constraints into a smaller number of equivalent ones (Lemma 1). Then, for any given data size $\boldsymbol{d} = \{d_j\}_{j\in\mathcal{J}}$, we derive the server's optimal reward $\{r_j^{L*}(\boldsymbol{d})\}_{j\in\mathcal{J}}$ (Lemma 2) in Section 4.4.1. Finally, we derive the optimal data size $\boldsymbol{d}^*$ (Proposition 5 and Theorem 1) in Section 4.4.2.

19. We can also initialize all the users' decisions as 1 and check whether there exists a user who wants to change his action from 1 to 0 for payoff improvement. If the equilibrium is the same as that found by Algorithm 1, it is the unique NE, as Game 1 is a supermodular game.

20. The revelation principle demonstrates that if a social choice function can be implemented by an arbitrary mechanism, then the same function can be implemented by an incentive-compatible-direct-mechanism (i.e. in which users truthfully report types) with the same equilibrium outcome. Thus, requiring IC will simplify the mechanism design without affecting optimality.

4.4.1 Optimal Rewards in Contract

Without loss of generality, we assume that users are indexed in ascending order of

$$\pi_j \triangleq \frac{\kappa_j}{1-p_j} = \xi_j \mathbb{E}[\ell_j] + \frac{\theta_j T}{1-p_j} + \theta_j \lambda \sum_{m\in\mathcal{J}} I_m p_m (1-q_m)\left(\mathbb{E}[\ell_m]^2 + D(\ell_m)\right),$$

which can be regarded as a type-j user's aggregated marginal cost. That is,

$$\pi_1 \le \pi_2 \le ... \le \pi_J. \tag{30}$$

In the following Lemma 1, we present an equivalent version of the IR and IC constraints to simplify Problem 2.

Lemma 1. *A contract $\boldsymbol{\phi} = \{(d_j, r_j^L)\}_{j\in\mathcal{J}}$ is feasible (i.e., satisfies IR and IC constraints) if and only if the contract items satisfy the following three constraints:*

a) $r_J^L - \pi_J d_J \ge 0$;
b) $r_1^L \ge ... \ge r_J^L \ge 0$ *and* $d_1 \ge ... \ge d_J \ge 0$;
c) $r_{j+1}^L + \pi_j(d_j - d_{j+1}) \le r_j^L \le r_{j+1}^L + \pi_{j+1}(d_j - d_{j+1})$, $j \in \mathcal{J}$.

The proof of Lemma 1 is given in Appendix 5. Constraint (a) ensures that each user can get a non-negative payoff by accepting the contract item of type-J users, corresponding to the IR constraints. Both constraints (b) and (c) are related to IC constraints. Constraint (b) shows that the server should request more data from a user type with a lower marginal cost π and provide a larger reward in return. Constraint (c) characterizes the relationship between any two neighboring contract items.

Based on Lemma 1, the following Lemma 2 characterizes the server's optimal learning rewards for any feasible data size:

Lemma 2. *For any given data size $\boldsymbol{d} = \{d_j\}_{j\in\mathcal{J}}$ (even if it is not optimal), the unique optimal reward for a type j user is:*

$$r_j^{L*}(\boldsymbol{d}) = \begin{cases} \pi_j d_j, & \text{if} j = J; \\ \pi_j d_j + \sum_{m=j+1}^{J} (\pi_m - \pi_{m-1}) d_m, & \text{if} j = 1, ..., J-1. \end{cases} \tag{31}$$

The proof of Lemma 2 is given in Appendix 6. Lemma 2 indicates that all user types except the boundary type J will obtain positive expected payoffs (type-J users receive zero expected payoff), which can be interpreted as the *information rent* in economics due to information asymmetry.

4.4.2 Optimal Data Sizes in Contract

Based on Lemma 2, we can significantly simplify Problem 2 but still need to derive the optimal values of J variables $\{d_j\}_{j\in\mathcal{J}}$ under J constraints $d_1 \ge ... \ge d_J \ge 0$.

For the convenience of presentation, we define

$$A_j \triangleq \frac{\varrho I_j (1 - p_j + p_j q_j)}{T}, \tag{32}$$

$$B_j \triangleq \gamma I_j \left(p_j q_j \left(\alpha\theta_j + \xi_j \mathbb{E}[\ell_j]\right) + (1-p_j)\pi_j\right) + \sum_{m=1}^{j-1} \gamma I_m (1-p_m)(\pi_j - \pi_{j-1}). \tag{33}$$

Based on these two metrics, we first present two special cases of the optimal data sizes, which we call all-independent and all-dependent.

Proposition 5. *Two special cases of the optimal data sizes follow:*

- All-independent. *If*

$$\frac{A_1}{B_1} \ge \frac{A_2}{B_2} \ge ... \ge \frac{A_J}{B_J}, \tag{34}$$

then the optimal data sizes in the contract are

$$d_j^* = \sqrt{\frac{A_j}{B_j}}, j \in \mathcal{J}. \tag{35}$$

- All-dependent. *If*

$$\frac{\sum_{m\in\mathcal{J}} A_m}{\sum_{m\in\mathcal{J}} B_m} > \frac{\sum_{m=1}^{j} A_m}{\sum_{m=1}^{j} B_m}, \forall j = 1, 2, ..., J-1, \tag{36}$$

then the optimal data sizes in the contract are

$$d_j^* = \sqrt{\frac{\sum_{m\in\mathcal{J}} A_m}{\sum_{m\in\mathcal{J}} B_m}}, j \in \mathcal{J}. \tag{37}$$

The proof of Proposition 5 is given in Appendix 7. The all-independent case means that if $\{A_j/B_j\}_{j\in\mathcal{J}}$ follow a descending order, then the optimal data size for each type-j user only depends on his own parameters (A_j, B_j). The condition for the all-dependent case means that for any type j, there always exists at least one type $m > j$ with A_m/B_m larger than A_j/B_j (i.e., not in descending order). In this case, each type's optimal data size depends on all types' parameters $\{(A_j, B_j)\}_{j\in\mathcal{J}}$.

Next, we give an efficient algorithm to compute the optimal data sizes in any possible case based on the insights in Proposition 5.

Theorem 1. *For a fixed J, there are 2^{J-1} possible cases of the optimal data sizes depending on the values of $\{(A_j, B_j)\}_{j\in\mathcal{J}}$. For any given $\{(A_j, B_j)\}_{j\in\mathcal{J}}$, the unique optimal data sizes can be calculated by Algorithm 2.*

The proof of Theorem 1 is given in Appendix 8. The computation complexity of Algorithm 2 is $\mathcal{O}(\sum_{x=1}^{X} J_x)$, which is no larger than $\mathcal{O}(J)$. We can interpret Algorithm 2 as greedily merging non-descending types based on A_j/B_j, so that all merged types have $\sum_j A_j/\sum_j B_j$ in a descending order. The optimal data sizes of the merged types are the same and follow the dependent form (37) in Proposition 5, while the optimal data sizes of the not-merged types follow the independent form (35).

For example, if $\frac{A_1}{B_1} \ge \frac{A_4}{B_4} \ge \frac{A_3}{B_3} \ge \frac{A_2}{B_2} \ge \frac{A_5}{B_5} \ge \frac{A_8}{B_8} \ge \frac{A_6}{B_6} \ge \frac{A_7}{B_7}$, then $X = 2$, $\mathcal{J}_1 = \{2, 3, 4\}$, and $\mathcal{J}_2 = \{6, 7, 8\}$, which corresponds to Lines 2-4 in Algorithm 2. Further, according to Lines 5-7 in Algorithm 2, if $\frac{A_6}{B_6} \ge \frac{A_7+A_8}{B_7+B_8}$, then $\mathcal{J}_2$ can be divided into two subsets $\{6\}$ and $\{7, 8\}$. The optimal data sizes in this example are $d_j^* = \sqrt{\frac{A_j}{B_j}}, j = 1, 5, 6$, $d_j^* = \sqrt{\frac{A_2+A_3+A_4}{B_2+B_3+B_4}}, j = 2, 3, 4$, and $d_j^* = \sqrt{\frac{A_7+A_8}{B_7+B_8}}, j = 7, 8$. Eventually, we have $\frac{A_1}{B_1} \ge \frac{A_2+A_3+A_4}{B_2+B_3+B_4} \ge \frac{A_5}{B_5} \ge \frac{A_6}{B_6} \ge \frac{A_7+A_8}{B_7+B_8}$.

Algorithm 2: Optimal data sizes in contract

Input : Parameters $\{(A_j, B_j)\}_{j\in\mathcal{J}}$ indexed based on (30)

Output: Optimal data sizes $\{(d_j^*)\}_{j\in\mathcal{J}}$

Initialize $d_j^* \leftarrow \sqrt{\frac{A_j}{B_j}}, j \in \mathcal{J}$;

Find all non-descending types $\{j : \exists m > j, \frac{A_m}{B_m} > \frac{A_j}{B_j}$ or $\exists m < j, \frac{A_m}{B_m} < \frac{A_j}{B_j}\}$;

Put each group of non-descending types that have adjacent indexes into one auxiliary set $\mathcal{J}_x$;

$X \leftarrow$ the number of these auxiliary sets;// i.e., $x \in \{1, 2, ..., X\}$

for $x = 1; x \le X; x++$ **do**

  check($\mathcal{J}_x$);// divide each auxiliary set $\mathcal{J}_x$ into subsets $\{\mathcal{J}_x^y\}$ that satisfy (36)

end

Function check($\mathcal{J}$)**:**

  Reindex the types in $\mathcal{J}$ with $1_\mathcal{J}, 2_\mathcal{J}, ..., J_\mathcal{J}$.

  if $|\mathcal{J}| \neq 1$ **then**

    flag←1;

    for $m = 1_\mathcal{J}$ *to* $(J-1)_\mathcal{J}$ **do**

      if $\frac{\sum_{j\in\mathcal{J}} A_j}{\sum_{j\in\mathcal{J}} B_j} \le \frac{\sum_{j=1_\mathcal{J}}^{m} A_j}{\sum_{j=1_\mathcal{J}}^{m} B_j}$// $\mathcal{J}$ does not satisfy (36) **then**

        flag←0;

        $d_j^* = \sqrt{\frac{\sum_{n=1_\mathcal{J}}^{m} A_n}{\sum_{n=1_\mathcal{J}}^{m} B_n}}, j \in \{1_\mathcal{J}, ..., m\}$;

        $check(\{m+1, ..., J_\mathcal{J}\})$;

        break;

      end

    end

    if *flag=1* **then**

      $d_j^* = \sqrt{\frac{\sum_{m\in\mathcal{J}} A_m}{\sum_{m\in\mathcal{J}} B_m}}, j \in \mathcal{J}$;// $\mathcal{J}$ satisfies (36)

    end

  end

5 Optimal Incentive Mechanism in Unlearning-Forbidden Scenario

In this section, we first derive the server's optimal incentive mechanism in the unlearning-forbidden scenario in Section 5.1, then we compare the unlearning-allowed and unlearning-forbidden scenarios based on the server's expected cost and users' expected payoffs in Section 5.2.

5.1 Server's Optimal Contact Design

Similar to the server's contract design in the unlearning-allowed scenario (i.e., Problem 2), the server designs the contract to minimize its expected cost in (18) under the IR and IC constraints in the unlearning-forbidden scenario:

Problem 3.

$$\begin{aligned}
&\min \sum_{j'\in\mathcal{J}'} \left(\frac{\varrho I'_{j'}}{T d'_{j'}} + \gamma I'_{j'} r_{j'}^{L'} \right), \\
&\text{s.t. } r_{j'}^{L'} - \Pi_{j'} d'_{j'} \ge 0, \forall j' \in \mathcal{J}', \text{ (IR)} \\
&\quad r_{j'}^{L'} - \Pi_{j'} d'_{j'} \ge r_m^{L'} - \Pi_{j'} d'_m, \forall j', m \in \mathcal{J}', \text{ (IC)} \\
&\text{var. } \left\{ \left(d'_{j'}, r_{j'}^{L'} \right) \right\}_{j'\in\mathcal{J}'},
\end{aligned} \tag{38}$$

where

$$\Pi_{j'} \triangleq \theta_{j'} T + \xi'_{j'} \mathbb{E}[\ell'_{j'}]. \tag{39}$$

For the convenience of presentation, we re-indexed users with j' in an ascending order of Π, i.e.,

$$\Pi_{1'} \le \Pi_{2'} \le ... \le \Pi_{J'}, \tag{40}$$

and define

$$A'_{j'} \triangleq \frac{\varrho I'_{j'}}{T}, \tag{41}$$

$$B'_{j'} \triangleq \gamma \left(\Pi_{j'} \sum_{m=1'}^{j'} I'_m - \Pi_{j-1'} \sum_{m=1'}^{j-1'} I'_m \right). \tag{42}$$

After a similar analysis to Section 4.4, we obtain the following theorem about the server's optimal contract in the unlearning-forbidden scenario.

Theorem 2. *The optimal data sizes $\boldsymbol{d}'^*$ can be obtained from Theorem 1 by substituting $\{(A_j, B_j)\}_{j\in\mathcal{J}}$ with $\{(A'_{j'}, B'_{j'})\}_{j'\in\mathcal{J}'}$. The optimal rewards are*

$$r_{j'}^{L'*}(\boldsymbol{d}'^*) = \begin{cases} \Pi_{j'} d'^*_{j'}, & \text{if } j' = J'; \\ \Pi_{j'} d'^*_{j'} + \sum_{m=j+1'}^{J'} (\Pi_m - \Pi_{m-1}) d'^*_m, & \text{if } j' = 1', ..., J-1'. \end{cases} \tag{43}$$

The proof of Theorem 2 is given in Appendix 9.

5.2 Comparison

In this subsection, we compare the unlearning-allowed and unlearning-forbidden scenarios, to reveal the economic impact of federated unlearning.

Suppose that a type-j user in the unlearning-allowed scenario corresponds to type j' in the unlearning-forbidden scenario. For the convenience of presentation, we first introduce the following definitions:

$$\begin{aligned}
\Delta U_j = &\sum_{m=j+1}^{J} (1-p_j)(\pi_m - \pi_{m-1}) \sqrt{\frac{\sum_{m\in\mathcal{J}_m} A_m}{\sum_{m\in\mathcal{J}_m} B_m}} \\
&- \sum_{m=j+1'}^{J'} (\Pi_m - \Pi_{m-1}) \sqrt{\frac{\sum_{m\in\mathcal{J}'_m} A'_m}{\sum_{m\in\mathcal{J}'_m} B'_m}},
\end{aligned} \tag{44}$$

$$\begin{aligned}
\Delta W = &\sum_{j=1}^{J} \frac{2}{|\mathcal{J}_j|} \sqrt{\left(\sum_{m\in\mathcal{J}_j} A_m \right) \left(\sum_{m\in\mathcal{J}_j} B_m \right)} \\
&- \sum_{j'=1'}^{J'} \frac{2}{|\mathcal{J}'_{j'}|} \sqrt{\left(\sum_{m\in\mathcal{J}'_{j'}} A'_m \right) \left(\sum_{m\in\mathcal{J}'_{j'}} B'_m \right)}.
\end{aligned} \tag{45}$$

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3557857

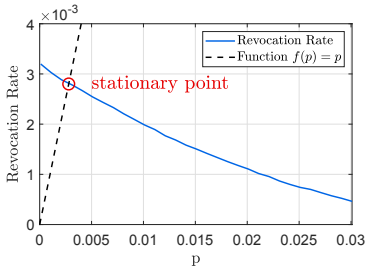


Fig. 2. Relationship between the revocation rate $|\mathcal{I}_u^*|/I$ and historical revocation rate p.

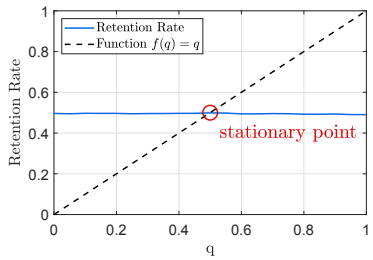


Fig. 3. Relationship between the retention rate $|\mathcal{I}_r^*|/|\mathcal{I}_u^*|$ and historical retention rate q.

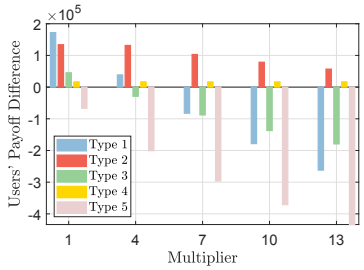


Fig. 4. Different types of users' expected payoff difference (unlearning-allowed minus unlearning-forbidden) versus the multiplier of $\boldsymbol{\xi}'$.

Given the definitions, we present the comparison results in Proposition 6.

Proposition 6. *If $\Delta U_j > 0$, then at equilibrium a type-j user has a larger payoff (i.e., more beneficial) in the unlearning-allowed scenario than the unlearning-forbidden scenario. If $\Delta W < 0$, then the server has a smaller cost (i.e., more beneficial) in the unlearning-allowed scenario than the unlearning-forbidden scenario.*

The proof of Proposition 6 is given in Appendix 10. We can obtain some insights from qualitative analysis:

- If users' perceived privacy costs (e.g., $\{\xi'_{j'}\}_{j'\in\mathcal{J}'}$) in the unlearning-forbidden scenario are very large but unlearning costs (e.g., λ and $D(\ell)$) are relatively small, then we will have $\Delta U_j < 0$ and $\Delta W < 0$, i.e., the unlearning-allowed scenario is worse for users but more beneficial to the server.
- If the unlearning costs are very large but users' perceived privacy costs in the unlearning-forbidden scenario are relatively small, then $\Delta U_j > 0$ and $\Delta W > 0$, i.e., the unlearning-allowed scenario is better for users but worse for the server.

It is counter-intuitive that users prefer the scenario where they have large costs. One explanation is that due to information asymmetry, the server would pay an exceedingly large reward to incentivize such users to participate. We will present detailed illustrations of the preferences of users and the server through simulations in Section 6.1.1.

6 Simulations

In this section, we use simulations to evaluate the performance of our proposed mechanism. Specifically, in Section 6.1, we validate the optimal strategies of the users and the server in unlearning-allowed and unlearning-forbidden scenarios, and we also compare our mechanism with state-of-the-art benchmarks. In Section 6.1.3, we conduct experiments based on public datasets to show the model performance under our mechanism. We discuss the mechanism application and implementation in Section 6.3.

6.1 Strategies and Payoffs

We consider $J = 5$ types of users with marginal training costs $\boldsymbol{\theta} = [1, 4, 6, 9, 10]$, marginal perceived privacy costs in the unlearning-allowed scenario $\boldsymbol{\xi} = [0.8, 1.7, 1.4, 2.2, 1.2] \times 10^3$,[21] and marginal perceived privacy costs in the unlearning-forbidden scenario $\boldsymbol{\xi}' = M \cdot \boldsymbol{\xi}$, where nominally the *multiplier* $M = 8$. Each type has $I_j = I/J = 1000$ users. Heterogeneous users' training losses follow a truncated normal distribution $N(0.5, 0.2)$ over the support $[0, 1]$, and users' federated Shapley values follow a normal distribution $N(5 \times 10^{-5}, 0.04)$.[22] Users perform $T = 100$ rounds of federated learning, and the unlearning rounds coefficient $\lambda = 4$. The server's accuracy loss coefficient $\varrho = 1$ and its weight on the incentives $\gamma = 10^{-10}$ (to balance different units of incentives and model accuracy loss).

We perform experiments to find the appropriate values of historical revocation rate p and retention rate q. As shown in Fig. 2 and Fig. 3, when we set different values of p and q, both the realized revocation rate $|\mathcal{I}_u^*|/I$ and retention rate $|\mathcal{I}_r^*|/|\mathcal{I}_u^*|$ at the equilibrium have a stationary point, i.e., $(2.8 \times 10^{-3}, 2.8 \times 10^{-3})$ in Fig. 2 and $(0.5, 0.5)$ in Fig. 3, respectively. Therefore, we take the historical revocation rate $p = 0.28\%$ and the historical retention rate $q = 50\%$ in the following simulations.

In Section 6.1.1, we compare the server's expected costs and users' expected payoffs under the optimal contracts in the unlearning-allowed and unlearning-forbidden scenarios. Then, we show users' optimal equilibrium revocation decisions and the server's optimal retention decision in Section 6.1.2. Finally, in Section 6.1.3, we present comparison results between our mechanism and two benchmarks.

6.1.1 Users' Expected Payoffs and Server's Expected Cost Comparison

In the following, we show the expected payoff/cost comparison considering three dimensions: privacy cost in the unlearning-forbidden scenario, unlearning cost, and training cost.

(i) *Impact of marginal perceived privacy cost in the unlearning-forbidden scenario $\boldsymbol{\xi}'$:*

Fig. 4 shows various types of users' payoff differences in unlearning-allowed and unlearning-forbidden scenarios, which indicate their preferences between the two scenarios. Different types of users may have different preferences, which are closely related to their type ranking (ranked by π in (30) and Π in (40), respectively) in the two scenarios.

Specifically, as shown in Fig. 5, a negative type ranking difference is more likely to lead to a positive payoff differ-

21. Different orders of magnitude are to balance different units of users' training costs and privacy costs.

22. In future work, we may use real-world datasets to calculate users' true training losses and federated Shapley values. The simulation data here can also demonstrate our results. As in Appendix 11, we further validate that if we change the simulation setting, we will obtain similar experiment results and insights.

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3557857

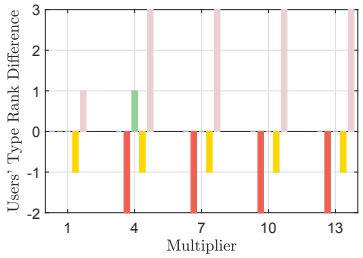


Fig. 5. Users' type ranking difference (unlearning-allowed minus unlearning-forbidden) versus the multiplier of $\boldsymbol{\xi'}$.

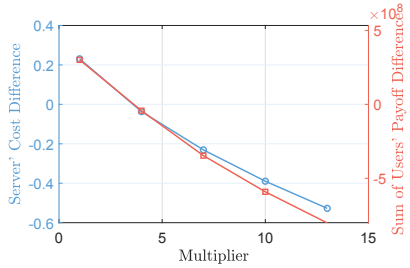


Fig. 6. Server's expected cost difference and users' expected total payoff difference (unlearning-allowed minus unlearning-forbidden) versus the multiplier of $\boldsymbol{\xi'}$.

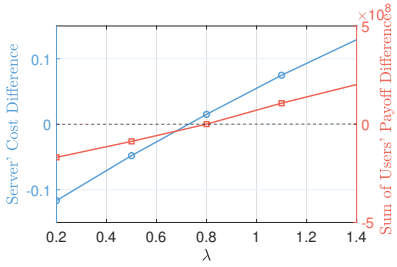


Fig. 7. Server's expected cost difference and users' expected total payoff difference (unlearning-allowed minus unlearning-forbidden) versus λ.

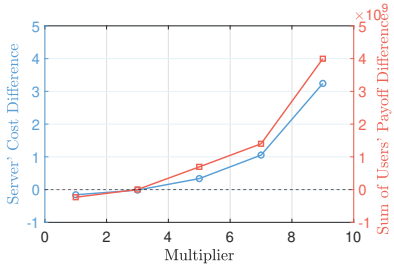


Fig. 8. Server's expected cost difference and users' expected total payoff difference (unlearning-allowed minus unlearning-forbidden) versus the multiplier of $\boldsymbol{\theta}$.

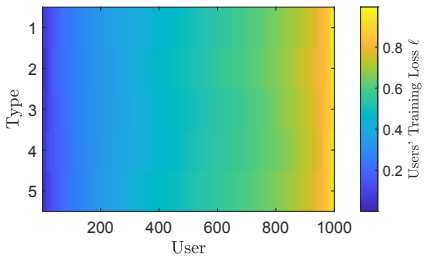


Fig. 9. Users' training losses $\{\ell_i\}_{i\in\mathcal{I}}$.

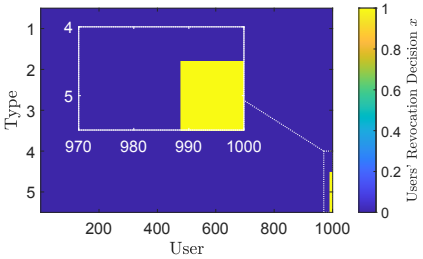


Fig. 10. Users' optimal revocation decisions $\{x_i^*\}_{i\in\mathcal{I}}$.

ence, i.e., a high ranking corresponds to a high payoff. This is consistent with the server's optimal rewards in Lemma 2 and Theorem 2.

Fig. 6 shows users' total expected payoff difference and the server's expected cost difference. When the perceived privacy cost in the unlearning-forbidden scenario $\boldsymbol{\xi'}$ increases, it is more likely that the unlearning-forbidden scenario is more beneficial to users (i.e., negative payoff difference) but worse for the server (i.e., negative cost difference). The server's preference is straightforward, as a larger $\boldsymbol{\xi'}$ means larger incentive costs for the server. However, it is counter-intuitive that users prefer the scenario where they have larger costs, as we may naturally presume that large costs will discourage users' participation. This is because the server will set the rewards larger than users' costs due to information asymmetry, and the gap (i.e., users' total payoff) increases in users' costs (as indicated in Lemma 2 and Theorem 2).

(ii) *Impact of unlearning cost:*

Increasing unlearning rounds coefficient λ or users' training loss variance $D(\ell)$ will both increase the unlearning cost, so we only simulate the impact of λ here. As shown in Fig. 7, when we increase the unlearning cost, it is more likely that the unlearning-allowed scenario is worse for the server but better for users. This is because larger unlearning costs mean larger incentive costs for the server but more rewards for users in the unlearning-allowed scenario.

Moreover, as shown in Fig. 7, the server and users' preferences are not always the same (i.e., one positive and the other negative) or different (i.e., the same sign). However, in most cases, they have different preferences.

(iii) *Impact of marginal training cost $\boldsymbol{\theta}$:*

We increase the value of the marginal training cost $\boldsymbol{\theta}$ by multiplying by a multiplier. As shown in Fig. 8, the insights are similar to that of unlearning cost. The training cost $\boldsymbol{\theta}$ affects both learning cost and unlearning cost. As both scenarios have learning costs, increasing $\boldsymbol{\theta}$ is similar to the effect of increasing the unlearning cost.

6.1.2 *Users' Revocation Decisions and Server's Retention Decision*

In Fig. 9, we visualize all users' training losses in federated learning by ranking each type of users in ascending order of their training losses. This is for the convenience of presenting insights in Fig. 10.

To obtain Fig. 10, we calculate each user's decision on whether to revoke data based on Algorithm 1. Output decision 1 means the user revoking data while 0 means not revoking data, so the yellow region is the users who revoke data and the blue region is users who do not revoke data. By referring to Fig. 9, Fig. 10 shows that at the equilibrium, users with larger aggregated marginal costs π (i.e., type 5) and training losses ℓ (i.e., users 986-1000) are more likely to revoke their data. This is because (i) users with larger costs receive smaller learning incentives from the server in the contract (Lemma 2); (ii) they do not know their high training losses before federated learning and their realized privacy costs (training losses) significantly exceed their expectations.

Fig. 11 illustrates the server's optimal retention decision. We rank the users who want to revoke their data in ascending order of their federated Shapley values $\{v_i\}_{i\in\mathcal{I}_u}$. Users with smaller federated Shapley values are more likely to be retained by the server, as smaller Shapley values represent larger contributions to the global model accuracy. Users with smaller training losses have lower privacy costs and may require fewer incentives from the server, compared to users with larger losses. However, Fig. 11 shows that the server does not necessarily retain users with smaller training

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3557857

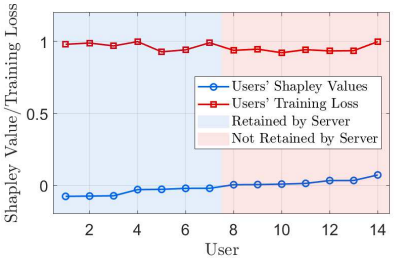


Fig. 11. Server's optimal retention decisions $\mathcal{I}_r^*$.

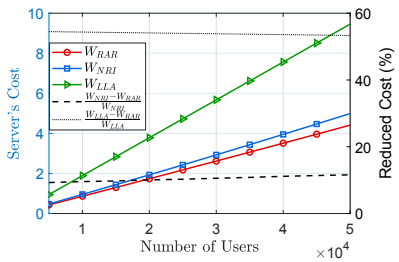


Fig. 12. Server's cost comparison of NRI, LLA, and RAR.

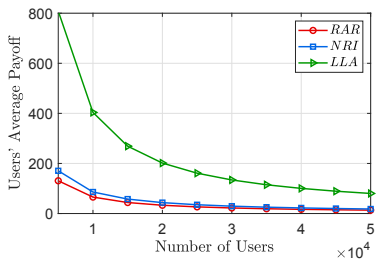


Fig. 13. Users' average payoff comparison of NRI. LLA. and RAR.

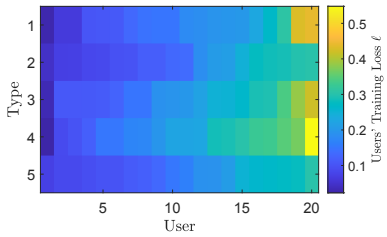


Fig. 14. Users' training losses $\{\ell_i\}_{i\in\mathcal{I}}$.

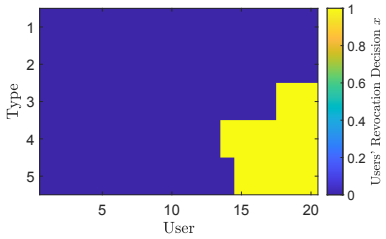


Fig. 15. Users' optimal revocation decisions $\{x_i^*\}_{i\in\mathcal{I}}$.

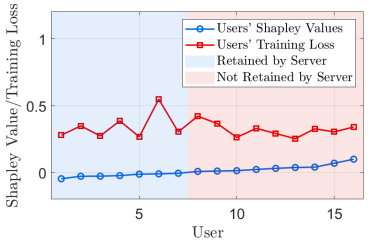


Fig. 16. Server's optimal retention decisions $\mathcal{I}_r^*$.

losses (i.e., "average" users). This is because given a fixed set of users, reducing the total training losses of retained users means increasing the total losses of leaving users, resulting in higher unlearning costs (Proposition 3).

6.1.3 Comparison with Benchmarks

We compare our incentive mechanism with two benchmarks to evaluate the performance.

- No Retention Incentive (NRI): the server does not retain users who want to revoke their data.
- Limited Look Ahead (LLA) (adapted from [26]): the server first optimizes the incentive mechanism for federated learning without considering the unlearning part, and then designs the retention incentive in unlearning (i.e., separate optimization).
- Our proposed incentive mechanism (RAR): the server is Rational in jointly optimizing both federated learning and unlearning And designs Retention incentive to retain valuable leaving users.

Fig. 12 shows the server's costs in the three mechanisms under different numbers of users. Our proposed RAR reduces the server's cost by around 53.91% (black dotted line) compared with LLA. The reduced cost of RAR compared with NRI can reach 11.59% (black dashed line) and will increase in the number of users, as the server retains more valuable users when the number of users increases. Therefore, it is beneficial for the server to retain valuable leaving users and make joint optimization of federated learning and unlearning incentive mechanisms. As the objective of our incentive mechanism design is to minimize the server's cost, the server's cost reduction is at the expense of users' payoffs (as shown in Fig. 13).

6.2 Model Performance

We perform experiments based on CIFAR-10 dataset with users possessing non-IID data. Specifically, we consider that there are $J = 5$ types of users, with marginal training costs $\boldsymbol{\theta} = [0.4, 1.2, 2.4, 2.8, 4]$ and marginal perceived privacy costs $\boldsymbol{\xi} = [2.8, 1.7, 3.4, 4.2, 1.2] \times 10^4$. Each type has $I_j = I/J = 20$ users, totaling $I = 100$ users. Each user is randomly assigned 2 labels, each containing 250 data points. Users perform $T = 450$ rounds of federated learning training and 150 rounds of unlearning. Users conduct local training using stochastic gradient descent (SGD) with an initial point = 0, epoch = 3, batch size = 10, and learning rate = 0.01. The server's accuracy loss coefficient $\varrho = 1$, and the server's weight on the incentives $\gamma = 10^{-10}$. Our convolutional neural network (CNN) model consists of two convolutional layers with 5×5 kernels, each followed by a ReLU activation and a 2×2 max pooling layer. After the convolutional layers, the network includes three fully connected layers: the first with 1024 neurons, the second with 512 neurons, both followed by ReLU activations, and a final output layer. We take the historical revocation rate $p = 17\%$ and the historical retention rate $q = 40\%$.

Fig. 14, Fig. 15, and Fig. 16 show the users' training losses, the users' revocation decisions, and the server's retention decisions under the new experiment setting, respectively. The key insights are consistent with our theoretical analysis (Lemma 2 and Proposition 3) and the experiment results under another setting in Section 6.1.2.

Fig. 17 shows the convergence performance about the training loss as the number of communication rounds increases. In Fig 18, we compare the test accuracy of our proposed mechanism and two benchmarks in the paper. The results show that, when users request to revoke their data in round 450, our mechanism RAR achieves the highest accuracy (i.e., lowest accuracy drop) in general. The LLA benchmark has a slightly worse performance, as it also retains leaving users but makes limited look ahead optimization. The NRI benchmark does not retain users, leading to the largest accuracy drop.

6.3 Discussion on Application and Implementation

Consider an application scenario of a mobile phone keyboard such as Gboard (Google Keyboard), where a large

This article has been accepted for publication in IEEE Transactions on Mobile Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TMC.2025.3557857

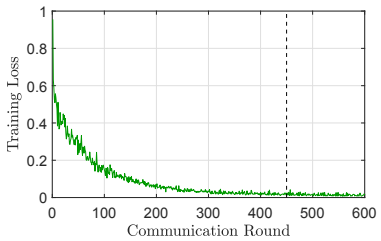


Fig. 17. Model training loss v.s. communication round in federated learning and unlearning.

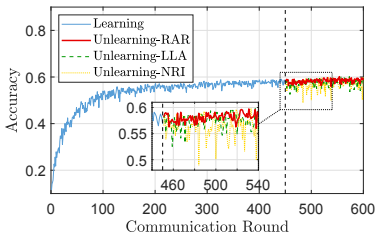


Fig. 18. Model test accuracy of NRI, LLA, and RAR in federated learning and unlearning.

amount of local data will be generated when users use the keyboard app on their mobile devices. Suppose that the Google server wants to train a next-word prediction model based on users' data through federated learning.

The incentivized federated learning and unlearning process works as follows. The server first announces the learning project to users through the app and encourages their participation through incentives (e.g., money, credit, or advanced service). If the user thinks one incentive contract item is optimal and beneficial to him, he will choose the contract item and sign the contract with the server. Once enough users decide to participate, the server will start the training process by broadcasting an initial global model to all participating users. On behalf of the user, the app can download this global model and upload the model updates generated by the training on the user's local data. After finishing the model training project, some users may want to leave the system due to reasons like privacy concerns or training costs. Upon receiving users' unlearning requests, the server can retain certain valuable users by giving them some further incentives to keep them in the system. Users who are not retained by the server will leave the system, and their data will be unlearned by the server and remaining users. During this process, the server aims to achieve a small model accuracy loss while keeping the total incentives paid to users low. By using our mechanism, the server's cost (including both the model accuracy loss and incentives) can be reduced up to 53.91% in our experiment, which can greatly benefit the server and promote the sustainable development of the application.

Note that our mechanism can effectively avoid false data issues. The server evaluates users' contributions to the model accuracy by calculating their federated Shapley values, ensuring that rewards are assigned based on actual contributions. If a user with false or poor-quality data attempts to gain excessive rewards by threatening to leave after model convergence, even if he has a larger training loss, the server won't provide him incentives and will let him leave the system. This user will obtain nothing except for a sunk learning cost.

The implementation of our proposed method may be constrained by the limited budget of the server. Although we aim to minimize the server's total incentives paid to users, the server may still have some financial budget constraints on the incentives, which may be insufficient especially when the number of participating users is very large. Nevertheless, we can adapt our modeling and analysis to this case by adding the budget constraints to the server's optimization problems.[23] Moreover, the current mechanism compensates for privacy costs but cannot prevent data privacy leakage. In future work, we can extend our framework to incorporate more factors related to privacy protection, such as differential privacy techniques.

7 Conclusion

To the best of our knowledge, we are the first to analytically study the incentive mechanism and economic benefit of federated unlearning We derive theoretical bounds on the global model optimality gap and the number of communication rounds of natural federated unlearning, based on Scaffold and FedAvg algorithms, and use these to motivate a multi-stage game model. Our approach tackles a challenging problem in incentive design, by summarizing users' multi-dimensional heterogeneity into one-dimensional metrics and developing an efficient algorithm for an exponentially large number of possible cases. We compare the unlearning-forbidden and unlearning-allowed scenarios in terms of users' payoffs and the server's cost. Counterintuitively, users usually prefer the scenario where they have larger costs. This is because the server will give them even higher incentives than their costs due to information asymmetry. We also identify what types of users will leave the system or be retained by the server. The experiments demonstrate the superior performance of our proposed incentive mechanism and the benefits of unlearning incentives for retaining leaving users.

References

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

[2] S. Truex, N. Baracaldo, A. Anwar, T. Steinke, H. Ludwig, R. Zhang, and Y. Zhou, "A hybrid approach to privacy-preserving federated learning," in *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 2019.

[3] V. Mothukuri, R. M. Parizi, S. Pouriyeh, Y. Huang, A. Dehghantanha, and G. Srivastava, "A survey on security and privacy of federated learning," *Future Generation Computer Systems*, vol. 115, pp. 619–640, 2021.

[4] P. Voigt and A. Von dem Bussche, "The eu general data protection regulation (gdpr)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, vol. 10, no. 3152676, pp. 10–5555, 2017.

[5] E. L. Harding, J. J. Vanto, R. Clark, L. Hannah Ji, and S. C. Ainsworth, "Understanding the scope and impact of the california consumer privacy act of 2018," *Journal of Data Protection & Privacy*, vol. 2, no. 3, pp. 234–253, 2019.

23. Detailed discussions are in Appendix 13.

[6] Y. Jiang, J. Konečnỳ, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," *arXiv preprint arXiv:1909.12488*, 2019.

[7] X. Gao, X. Ma, J. Wang, Y. Sun, B. Li, S. Ji, P. Cheng, and J. Chen, "Verifi: Towards verifiable federated unlearning," *arXiv preprint arXiv:2205.12709*, 2022.

[8] Y. Liu, Z. Ma, X. Liu, and J. Ma, "Learn to forget: User-level memorization elimination in federated learning," *arXiv preprint arXiv:2003.10933*, 2020.

[9] L. Bourtoule, V. Chandrasekaran, C. A. Choquette-Choo, H. Jia, A. Travers, B. Zhang, D. Lie, and N. Papernot, "Machine unlearning," in *IEEE Symposium on Security and Privacy (SP)*, 2021.

[10] G. Liu, X. Ma, Y. Yang, C. Wang, and J. Liu, "Federaser: Enabling efficient client-level data removal from federated learning models," in *2021 IEEE/ACM 29th International Symposium on Quality of Service*, 2021.

[11] C. Wu, S. Zhu, and P. Mitra, "Federated unlearning with knowledge distillation," *arXiv preprint arXiv:2201.09441*, 2022.

[12] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International Conference on Machine Learning*, 2020.

[13] Y. Cao and J. Yang, "Towards making systems forget with machine unlearning," in *IEEE Symposium on Security and Privacy*, 2015.

[14] A. Ginart, M. Guan, G. Valiant, and J. Y. Zou, "Making ai forget you: Data deletion in machine learning," *Advances in neural information processing systems*, vol. 32, 2019.

[15] Z. Liu, H. Ye, C. Chen, and K.-Y. Lam, "Threats, attacks, and defenses in machine unlearning: A survey," *arXiv preprint arXiv:2403.13682*, 2024.

[16] Y. Liu, L. Xu, X. Yuan, C. Wang, and B. Li, "The right to be forgotten in federated learning: An efficient realization with rapid retraining," in *IEEE Conference on Computer Communications (INFOCOM)*, 2022, pp. 1749–1758.

[17] Y. Lin, Z. Gao, H. Du, D. Niyato, G. Gui, S. Cui, and J. Ren, "Scalable federated unlearning via isolated and coded sharding," *arXiv preprint arXiv:2401.15957*, 2024.

[18] H. Chen, T. Zhu, X. Yu, and W. Zhou, "Machine unlearning via null space calibration," *arXiv preprint arXiv:2404.13588*, 2024.

[19] Y. Tao, C.-L. Wang, M. Pan, D. Yu, X. Cheng, and D. Wang, "Communication efficient and provable federated unlearning," *arXiv preprint arXiv:2401.11018*, 2024.

[20] J. Shao, T. Lin, X. Cao, and B. Luo, "Federated unlearning: a perspective of stability and fairness," *arXiv preprint arXiv:2402.01276*, 2024.

[21] H. Xie and J. C. Lui, "Modeling crowdsourcing systems: Design and analysis of incentive mechanism and rating system," *ACM SIGMETRICS Performance Evaluation Review*, vol. 42, no. 2, pp. 52–54, 2014.

[22] N. Zhao, Y.-C. Liang, and Y. Pei, "Dynamic contract incentive mechanism for cooperative wireless networks," *IEEE transactions on vehicular technology*, vol. 67, no. 11, pp. 10 970–10 982, 2018.

[23] W. Wang, L. Ying, and J. Zhang, "The value of privacy: Strategic data subjects, incentive mechanisms and fundamental limits," in *ACM International Conference on Measurement and Modeling of Computer Science*, 2016.

[24] J. Wang, H. Zhong, J. Qin, W. Tang, R. Rajagopal, Q. Xia, and C. Kang, "Incentive mechanism for sharing distributed energy resources," *Journal of Modern Power Systems and Clean Energy*, vol. 7, no. 4, pp. 837–850, 2019.

[25] Y. Zhan, P. Li, Z. Qu, D. Zeng, and S. Guo, "A learning-based incentive mechanism for federated learning," *IEEE Internet of Things Journal*, 2020.

[26] N. Ding, Z. Fang, and J. Huang, "Optimal contract design for efficient federated learning with multi-dimensional private information," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 186–200, 2020.

[27] M. Zhang, E. Wei, and R. Berry, "Faithful edge federated learning: Scalability and privacy," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3790–3804, 2021.

[28] Z. Wang, L. Gao, and J. Huang, "Socially-optimal mechanism design for incentivized online learning," in *IEEE Conference on Computer Communications (INFOCOM)*, 2022.

[29] N. Zhang, Q. Ma, and X. Chen, "Enabling long-term cooperation in cross-silo federated learning: A repeated game perspective," *IEEE Transactions on Mobile Computing*, vol. 22, no. 7, pp. 3910–3924, 2022.

[30] S. Wang, M. Chen, C. G. Brinton, C. Yin, W. Saad, and S. Cui, "Performance optimization for variable bitwidth federated learning in wireless networks," *IEEE Transactions on Wireless Communications*, vol. 23, no. 3, pp. 2340–2356, 2023.

[31] H. Xia, J. Liu, J. Lou, Z. Qin, K. Ren, Y. Cao, and L. Xiong, "Equitable data valuation meets the right to be forgotten in model markets," *Proceedings of the VLDB Endowment*, vol. 16, no. 11, pp. 3349–3362, 2023.

[32] N. Ding, E. Wei, and R. Berry, "Strategic data revocation in federated unlearning," in *IEEE Conference on Computer Communications (INFOCOM)*, 2024, pp. 1151–1160.

[33] Y. Lin, Z. Gao, H. Du, D. Niyato, J. Kang, and X. Liu, "Incentive and dynamic client selection for federated unlearning," in *Proceedings of the ACM on Web Conference 2024*, 2024, pp. 2936–2944.

[34] N. Ding, Z. Sun, E. Wei, and R. Berry, "Incentive mechanism design for federated learning and unlearning," in *Proceedings of the Twenty-fourth International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2023.

[35] R. Pathak and M. J. Wainwright, "Fedsplit: An algorithmic framework for fast federated optimization," *Advances in Neural Information Processing Systems*, vol. 33, pp. 7057–7066, 2020.

[36] N. Tran, W. Bao, A. Zomaya, and C. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *IEEE Conference on Computer Communications (INFOCOM)*, 2019.

[37] Y.-A. De Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel, "Unique in the crowd: The privacy bounds of human mobility," *Scientific reports*, vol. 3, no. 1, pp. 1–5, 2013.

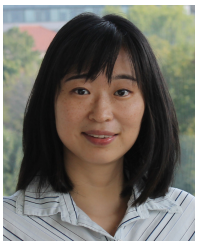
[38] D. Romanini, S. Lehmann, and M. Kivelä, "Privacy and uniqueness of neighborhoods in social networks," *Scientific reports*, vol. 11, no. 1, p. 20104, 2021.

[39] E. Winter, "The shapley value," *Handbook of game theory with economic applications*, vol. 3, pp. 2025–2054, 2002.

[40] T. Wang, J. Rausch, C. Zhang, R. Jia, and D. Song, "A principled approach to data valuation for federated learning," in *Federated Learning*. Springer, 2020, pp. 153–167.

Ningning Ding is a Tenure-Track Assistant Professor in the Data Science and Analytics Thrust and the Internet of Things Thrust within the Information Hub at the Hong Kong University of Science and Technology (Guangzhou). Before that, she was a Postdoctoral Scholar in the Department of Electrical and Computer Engineering at Northwestern University, USA. She received her Ph.D. in Information Engineering from The Chinese University of Hong Kong in 2022 and her B.S. degree in Information Science and Engineering from Southeast University in 2018. Her research focuses on interdisciplinary areas of artificial intelligence, network systems, and network economics, with a current emphasis on federated learning, machine unlearning, and data trading.

Zhenyu Sun received his B.Eng. in Electrical Engineering from Southwest Jiaotong University, Chengdu, China in 2021. He is currently a fourth-year Ph.D. student from the Department of Electrical and Computer Engineering at Northwestern University, Evanston, USA. His research interest generally lies in machine learning and optimization theory. He is focusing on distributed learning under networked systems, stochastic non-convex optimization, and reinforcement learning from both algorithmic and generalization perspectives.



Ermin Wei is an Associate Professor of Electrical and Computer Engineering, of Industrial Engineering and Management Sciences and by courtesy of Computer Science at Northwestern University. She completed her PhD studies in Electrical Engineering and Computer Science at MIT in 2014, advised by Professor Asu Ozdaglar, where she also obtained her M.S. Her team won the 2nd place in the GO-competition Challenge 1, an electricity grid optimization competition organized by Department of Energy. Wei's research interests include distributed optimization methods, convex optimization and analysis, smart grid, communication systems and energy networks and market economic analysis.

Randall Berry (Fellow, IEEE) is the Chair and John A. Dever Professor of Electrical and Computer Engineering at Northwestern University. He received his Ph.D. from MIT in 2000. He has served on the editorial boards of the IEEE Transactions on Wireless Communications and the IEEE Transactions on Information Theory and is currently an area editor for the IEEE Open Journal of the Communications Society. His research interests include network economics, wireless networking, and information theory.