# Degree of Staleness aware International Conference on Machine Learning (ICML 2025)

**Anonymous Authors**[1]

## Abstract

Handling data staleness still remains a significant challenge in federated learning with highly time-sensitive tasks, where data are generated continuously and data staleness largely depends on timeliness of data. Although recent works attempt to optimize data staleness by determining local data update frequency or client selection strategy with economic concern, none of them explore to take both data staleness and data volume into consideration. In this paper, we introduce a novel metric, *the Degree of Staleness (DoS)*, to quantify data staleness. Based on this metric, we propose *FedStream*, a DoS-aware incentive mechanism featuring an innovative local data update scheme manipulated by three knobs: the server's payment, outdated data conservation rate, and clients' fresh data collection volume, to coordinate staleness and volume of local data for best utilities. We model this mechanism as a two-stage Stackelberg game with dynamic constraint, afterward deriving the dynamic optimal strategies for clients in close form and the approximately optimal strategy for the server. Experimental results on real-world datasets demonstrates the significant performance of our approach.

## 1. Introduction

With the development of Internet of Things(IoT), many end devices are generating plenty of data each day. Due to the growing storage and computing power, it becomes more appealing to store data locally and push more compution function to them. It motivates the application of federated learning, which supports local storage and local training on each device without violating their privacy.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

A federated learning system comprises a parameter server and a large number of clients(i.e. end devices). The server maintains a global model while each client owns local statistic. During each iteration, clients download global model from the central server and makes local updates on their private data. Then the server will aggregate the updated models from the clients and generate a new global model. The training process will terminate when the accuracy of global model has reached the preset threshold. During the process, since clients share not their private data but the trained local model with the server, the federated learning system can protect clients' privacy efficiently.

Despite various advantages in enabling edge computing while protecting data privacy, it still faces two bottlenecks: 1) *Staleness of Data:* In most existing works, clients are assumed to hold a static dataset and use the same dataset to train local model over the time horizon. But in reality, there are many highly time-sensitive tasks with streaming data. That is, data are generated continuously and data valuation largely depends on timeliness of data (Xiao et al., 2023). Under this circumstance, outdated data used for training may deteriorate model parameters and reduce model service quality. 2) *Incentive Mechanism:* Incentive mechanism is attached great importance for stimulting clients to participate the training task. That's because clients consume various resources including computing capacity, communication bandwidth and battery charge while training. Besides, collecting fresh data frequently to meet the requirement of highly time-sensitive tasks also cause extra cost. Considering huge resource expenditure, clients may be not reluctant to provide fresh data or even reject to participate in federated learning tasks without economic compensation.

Efforts have been made in incentive mechanism design ranging from game theory (Lim et al., 2020) (Zhang et al., 2022) (Huang et al., 2024) to auction theory (Jiao et al., 2020) and contract theory (Kang et al., 2019) (Ding et al., 2020). However, most of these studies don't take into consideration the negative impact of outdated data on model performance. A few works have studied AoI-concerned incentive mechanisms against mobile crowdsensing (Xiao et al., 2023) (Wang et al., 2021). But they can't be applied in the context of federated learning directly. In the federated learning field,

some papers discuss the AoI-concerned incentive mechanism, focusing on controlling dynamic prices or update frequency (Wang & Duan, 2019) (Wu et al., 2023). But none of them explore to manipulate local data update by decaying stale data and collecting new data simultaneously considering value and volume of present data.

Motivated by the above considerations, we introduce a novel metric: degree of staleness (DoS) to measure data value, based on which we propose a DoS-aware incentive mechanism in federated learning, called FedStream. Different from previous works, this system allows the server to manipulate clients' local data update by launching monetary payment and enforced order at the same time according to DoS and data volume. Specifically, the server controls two variables: payment intented to encourage clients to collect new data and conservation rate intented to force clients to abandon outdated data. It targets to minimize model accuracy loss with plenty of fresh data and as low monetary payment as possible. For a particular client, it controls the volume of new data collected from local data stream dynamically according to the server's strategy. Before each communication round, it updates loal dataset by abandoning parts of outdated data as the server required and collecting some new data according to its own optimal strategy, then followed by local training. The objective of a client is to maximize the balance between payment related to its contribution share and various costs.

There are three key challenges in FedStream as follows. First, the server faces a two-variable optimization problem in an complicated form. Besides, the client faces a long-term optimization problem with dynamic constraint of update strategy. How to derive the optimal strategy for both server and clients to maximize their utility is of significance and challenging. Second, capturing the converged model performance before training is an important step to determine the optimal strategy of server. Intuitively, global model performance can be impacted by data volume and data staleness simultaneously. Specially, more fresh data will lead to better model performance, particularly in highly time-sensitive tasks. However, there is lack of quantitative relationship between global model performance and the freshness of data. Third, to derive clients' optimal strategies, they need to estimate their expected benefit before training, which depends on reward by the server and respective contributed share. However, in many real-world scenarios, the contributed share is unknown since clients cannot communicate with each other. Unknown information makes it challenging to make decision on client side.

To overcome the above challenges, we first derive the convergence upper bound of FedStream for the server, which reveals the relationship between converged model performance and data staleness. In addition, we introduce a mean-field term to estimate the unknown information for clients. Based on the above two approaches, the optimization problems on both server and client side can be constructed respectively. Then we use a Stackelberg game to model the interaction between both sides, where the server acts as a leader and clients are regarded as followers. Lastly, with the aid of backward reduction approach, we derive the optimal solution for this Stackelberg game.

The main contributions in this paper are summarized as follows:

- We propose a DoS-aware new federated learning system with the local data update scheme at the core. To fit the proposed update scheme, we define a novel concept of DoS to measure the degree of data staleness. Based on the definition, we conduct convergence analysis and secure the upper bound of converged upper bound for FedStream.
- On the basis of convergence upper bound, we construct the optimization problems for both the server and clients, where the server controls reward and conservation rate to minimize total cost, and clients control volume of new data to maximize their balance.
- We model the interaction between the server and clients as a Stackelberg game. Under the backward reduction, we derive clients' optimal strategy with Hamilton equation. Based on this ,we dervie the server's strategy by adopting a search algorithm.
- We carry out extensive experiments on two dataset: MNIST and FMNIST, to illustrate the extraordinary performance of FedStream.

The remainder of the paper is organized as follows: In Section 2, we introduce related work. We provide problem formulation and corresponding convergence analysis in Section 3. System model and methodology are demonstrated in Section 4 and 5 respectively. Then we conduct experiments in Section 6. The paper is concluded in Section 7. Proofs of theorems and remarks are moved to the Appendix.

## 2. Related Work

### 2.1. Incentive Mechanism

Incentive mechanism designed for federated learning has been widely investigated in previous works. They can be assorted into three catagories: (1) *game theory:* (Lim et al., 2020) proposes a hierarchical incentive mechanism based on coalitional game theory approach, where multiple workers can form various federations. (Zhang et al., 2022) builds a incentive mechanism utilizing repeated game theory to

enable long-term cooperation among participants in cross-silo federated learning. (Huang et al., 2024) designs a novel incentive framework based on Stackelberg game to model the collaboration behaviour among server and clients in federated learning with difference privacy. (2) *auction theory:* (Jiao et al., 2020) designs two auction mechanisms for the federated learning platform to maximize the social welfare of the federated learning services market. (3) *contract theory:* (Kang et al., 2019) proposes an effective incentive mechanism combining reputation with contract theory to motivate high-reputation mobile devices with high-quality data to participate in model learning. (Ding et al., 2020) presents an analytical study on the server's optimal incentive mechanism design by contract theory, in the presence of users' multi-dimensional private information. However, the above studies don't take into consideration the negative impact of outdated data on model performance and they can't be applied in highly time-sensitive tasks.

### 2.2. Data Staleness Optimization

Many efforts have been devoted to data staleness optimization. For example, (Tripathi & Modiano, 2021) consider the problem of minimizing age of information in general single-hop and multihop wireless networks. (Fang et al., 2021) devises a joint preprocessing and transmission policy to minimize the average AoI and the energy consumption at the IoT device. Only a few works among them study the AoI optimization with economic consideration. For example, (Xiao et al., 2023) investigates the incentive mechanism design in MCS systems that take the freshness of collected data and social benefits into concerns. (Wang et al., 2021) considers a general multi-period status acquisition system, aiming to maximize the aggregate social welfare and ensure the platform freshness. However, they can't be applied in the context of federated learning. In the federated learning field, (Wang & Duan, 2019) proposes dynamic pricing for the server to offer age-dependent monetary returns and encourages clients to sample information at different rates over time. (Wu et al., 2023) aims to minimize the loss of global model for FL with a limited budget by determining a client selection strategy under time-sensitive scenarios. But none of them explore to manipulate local data update by decaying stale data and collecting new data simultaneously considering value and volume of present data.

## 3. Problem Formulation

### 3.1. Federated Learning with Data Stream

We assume that there is a central server and $N$ clients in the federated learning system and they are arranged to conduct $T$ rounds in the training task. Different from the static local dataset in previous works, each client $k$ has a local data stream $\mathcal{D}_k$ which generates new data points continuously

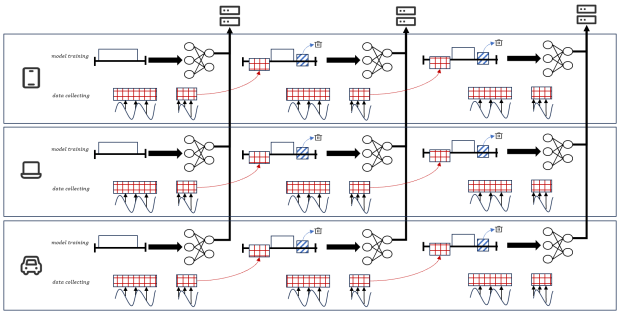


*Figure 1.* The framework of buffer update scheme. Tilde denotes local data stream. Blank blocks symbolize buffered data, while red blocks and blue blocks symbolize new data to be collected and stale data to be decayed respectively.

over the time horizon. In addition, due to the storage limit, each client maintains a buffer to store data points collected from its local data stream $\mathcal{D}_k$ and used for model training. We denote the stored data points in the buffer of client $k$ at round $t$ as $\mathcal{D}_k(t)$ and $|\mathcal{D}_k(t)| = D_k(t)$. During each round, the buffer of client $k$ can be updated by decaying part of stale data points and collecting new ones from its local data stream, and then used for local model training. Therefore, the updating scheme of client $k$ at round $t$ can be formulated as

$$D_k(t+1) = \theta D_k(t) + \Delta_k(t), \tag{1}$$

where $\theta \in [0, 1]$ is the conservation rate of stale data points and it decides how many buffered data can be conserved. Besides, $\Delta_k(t)$ is the volume of new data to be collected during round $t$, namely increment.

Note that considering the uncertainty of data stream, data collection cannot be completed immediately, which means the traditional "collecting-training" paradigm may lead to severe time latency for federated learning task. To improve the task efficiency, data collection is designed to proceed in parallel with model training in our framework, and the new data collected during present round will be used to update buffer and train model for next round, as depicted in Fig 1.

Then, the loss function of client $k$ based on global model $w(t)$ using stored data points $\mathcal{D}_k(t)$ at round $t$ can be represented as

$$F_k(w(t), \mathcal{D}_k(t)) = \frac{1}{D_k(t)} \sum_{j=1}^{D_k(t)} f(w(t), x_k^j), \tag{2}$$

where $f(w(t), x_k^j)$ is the loss function of each data point $\{x_k^j, y_k^j\} \in \mathcal{D}_k(t)$. Then client $k$ updates its local model by

$$w_k(t+1) = w(t) - \eta \nabla F_k(w(t), \mathcal{D}_k(t)), \tag{3}$$

where $\eta$ is the learning rate and $\nabla F_k(w(t), \mathcal{D}_k(t))$ is the loss gradient of client $k$ at round $t$. Until each client com-

pletes its local update and sends $w_k(t+1)$ to the central server, the central server will aggregate them by

$$w(t+1) = \sum_{k=1}^{N} \frac{D_k(t)}{D(t)} w_k(t+1), \tag{4}$$

where $\mathcal{D}(t) = \cup_{k=1}^{N} \mathcal{D}_k(t)$ and $D(t) = \sum_{k=1}^{N} D_k(t)$. Subsequently, the central server launches the new global model $w(t+1)$ to each client for the next round's training. The global loss function at round $t$ can be represented as

$$F(w(t), \mathcal{D}(t)) = \sum_{k=1}^{N} \frac{D_k(t)}{D(t)} F_k(w(t), \mathcal{D}_k(t)). \tag{5}$$

The ultimate goal is to find optimal parameters $w(t), t \in [0, \cdots, T-1]$ to minimize the global loss function in each round $t$, which can be expressed as

$$\arg\min_{w(t)} F(w(t), \mathcal{D}(t)) = \sum_{k=1}^{N} \frac{D_k(t)}{D(t)} F_k(w_k(t), \mathcal{D}_k(t)). \tag{6}$$

### 3.2. Convergence Analysis for Federated Learning with Data Stream

Convergence analysis for model performance is provided in this section. In practice, it's difficult to measure the model performance of FedStream directly due to data dynamity, data heterogeneity and so on. Therefore, we provide a convergence upper bound for FedStream, considering the impact of data volume and data staleness on model performance. Before that, the concept of degree of staleness (DoS) for buffered data is introduced in Definition 3.1.

**Definition 3.1.** We denote $S_k(t)$ as the degree of staleness(DoS) for client $k$'s buffered data at round $t$. The recursive definition is provided as

$$S_k(t) = \begin{cases} \frac{\theta D_k(t-1)}{D_k(t)}(S_k(t-1)+1) + \frac{\Delta_k(t-1)}{D_k(t)}, & t > 0; \\ 1, & t = 0, \end{cases} \tag{7}$$

where $S_k(t)$ is the weighted sum of conserved buffered data's DoS and new data's DoS. The conserved buffered data's DoS should be updated by adding to 1 once stepping into the next time slot, while new data's DoS is set to be 1.

*Remark* 3.2. $S_k(t)$ increases with conservation rate $\theta$ and decreases with increment $\Delta_k(t)$, which reflects the fact that less conserved buffered data and more fresh data contribute to DoS reduction.

*Remark* 3.3. The general formula of $S_k(t)$ can be further provided as

$$S_k(t) = \sum_{\tau=0}^{t} \frac{\theta^{t-\tau} D_k(\tau)}{D_k(t)}. \tag{8}$$

The detailed proof is provided in Appendix A.1.

Next, we introduce some assumptions on local loss function $F_k(w)$ which have been widely used in previous works before the presentation of convergence analysis (Wu et al., 2023) (Wei et al., 2020).

**Assumption 3.4.** Loss function of a particular client $k$ meets the following properties:

- $F_k(w)$ is $\rho$-Lipschitz, i.e., $F_k(w) - F_k(w') \leq \rho \|w - w'\|_2$.
- $F_k(w)$ is $\beta$-Lipschitz smooth, i.e., $\|\nabla F_k(w) - \nabla F_k(w')\| \leq \beta \|w - w'\|_2$.
- $F_k(w)$ is $\mu$-strong contex, i.e., $F_k(w)$ satisfies $F_k(w) - F_k(w^*) \leq \frac{1}{2\mu} \|\nabla F_k(w)\|_2^2$.
- The stochastic gradient is unbiased and variance-bounded, that is, $E[\nabla F_k(w(t)|D_k(t))] = \nabla F_k(w(t)|\mathcal{D}_k)$ and $E\|\nabla F_k(w(t)|D_k(t)) - \nabla F_k(w(t)|\mathcal{D}_k)\|^2 \leq \frac{\psi^2}{D_k(t)}$, where $\psi$ is a constant.
- The expected square norm of stochastic gradient is bounded, i.e., $E\|\nabla F_k(w(t)|D_k(t))\|^2 \leq G_k^2 + S_k(t)\sigma^2$, where $\sigma$ is a constant to measure the time sensitivity of the server's tasks.

**Theorem 3.5.** *Under Assumption 3.4, with $\eta \leq \frac{1}{2\beta}$, the convergence upper bound after $T$ rounds can be formulated as*

$$\begin{aligned} &E[F(w(T)|D(T)) - F(w^*)] \\ &\leq \underbrace{\kappa_1^T E[F(w(0)|D(0)) - F(w^*)]}_{(1)} \\ &+ \sum_{t=0}^{T-1} \kappa_1^{T-1-t} \Bigg[ \underbrace{\kappa_2 \frac{N\psi^2}{D(t)}}_{(2)} + \underbrace{\kappa_3 \sum_{k=1}^{N} \frac{D_k(t)}{D(t)} S_k(t)\sigma^2}_{(3)} + \underbrace{\Omega_t}_{(4)} \Bigg]. \end{aligned} \tag{9}$$

*where $\Omega_t = F(w(t+1)|D(t+1)) - F(w(t+1)|D(t))$. In addition, $\kappa_1 = 1 + 4\mu\beta\eta^2 - 2\mu\eta$, $\kappa_2 = 2\beta\eta^2$, $\kappa_3 = \beta\eta^2$.*

The detailed proof is provided in Appendix A.2.

*Remark* 3.6. In Theorem 3.5, the second term and the third term measures the influence of data volume and staleness on model performance respectively. Intuitively, the more and fresher buffered data are, the better global model performs. But in FedStream, staleness reduction partly depends on decaying stale data more intensively, which may lead to decrease of data volume, too. Thus, there exists a trade-off

between data volume and staleness. In addition, the greater $\sigma$ is, the more the third term weighs in the entire upper bound, which reflects the fact that highly time-sensitive tasks are affected more heavily by data staleness.

*Remark* 3.7. In Theorem 3.5, the fourth term of $\Omega_t$ captures the expected difference of global loss function based on the same global model $w(t)$ between buffered data at present round $\mathcal{D}(t)$ and at previous round $\mathcal{D}(t-1)$. Note that $\Omega_t$ is time-varying and it measures the influence of data dynamity has on model performance, and describes the generalization of global model towards new data points. The more fluctuating data dynamity is, the larger the convergence bound is and the worse the global model performs.

## 4. System Model

In this section, we design an incentive mechanism for FedStream. First, we construct optimization problems for both server and clients in 4.1 and 4.2. Then we model them as a two-stage Stackelberg game with dynamic constraint of local update.

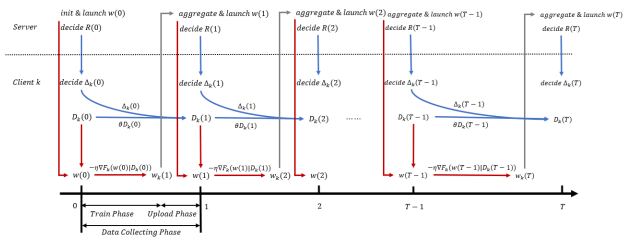



*Figure 2.* The overview of incentive mechanism. Red lines denote model parameter flow while blue lines symbolize decision and data flow.

### 4.1. Cost Function of Server

The cost of the server comprises two parts: accuracy loss of model and payment to clients. In reality, it's extremely hard to obtain accurate form of model accuracy loss directly. Yet we can approximate it with the convergence upper bound provided in Theorem 3.5. Note that (1) and (5) in Theorem 3.5 can't be controlled by the server, thus we extract the valid part of (2) and (3) to the cost function. In addition, total payment during the training period offered to clients can be represented as $TR$. Then, cost function of the server can be constructed as

$$U(\mathcal{D}, R, \theta) = \sum_{t=0}^{T-1} \gamma R + (1-\gamma)\kappa_1^{T-1-t} \times \left[ \left( \kappa_2 \frac{N\psi^2}{D(t)} + \kappa_3 \sum_{k=1}^{N} \frac{D_k(t)}{D(t)} S_k(t)\sigma^2 \right) \right], \tag{10}$$

where $\gamma \in [0,1]$ is a factor to balance model performance and cost payment. When $\gamma$ trends to 1, the server pays more attention to model performance, and vice versa.

(10) is dominated by payment $R$, conservation rate $\theta$ and total data volume $\boldsymbol{D}$. The server controls two variables: $R$ intented to encourage clients to collect new data, and $\theta$ intented to order clients to abandon parts of buffered data. It targets to minimize cost function in (10). Thus the optimality problem on the server side can be constructed as

$$\min_{R,\theta} U(\boldsymbol{D}, R, \theta), \tag{11}$$

In (11), greater reward incurs more fresh data for model training but increases monetary payment, while smaller conservation rate forces to abandon more stale data but decreases data volume at the same time. This is non-trivial and the server needs to minimize cost by elaborately determining the strategy of $(R, \theta)$.

### 4.2. Utility Function of Clients

For a client who participates into the federated learning task, it needs to spend various types of resource for data sampling and model training. Here, we use $\alpha_k \Delta_k(t)^2$ and $\beta_k D_k(t)^2$ to depict the expenditure for data collecting and training, respectively, where $\alpha_k$ is unit cost for data collecting and $\beta_k$ is the counterpart for model training. It should be noted that we choose the quadratic form of $\Delta_k(t)^2$ and $D_k(t)^2$ to reflect the fact that a client's sampling and training expenditure should convexly increase with its increment and scale, respectively (Wang & Duan, 2019). Therefore, the cost can be expressed as

$$C_k(t) = \alpha_k \Delta_k(t)^2 + \beta_k D_k(t)^2. \tag{12}$$

Meanwhile, a client can obtain some reward. In FedStream, the reward a particular client can secure depends on its buffered data compared with other participating clients, which can be formulated as

$$P_k(t) = \frac{D_k(t)}{\sum_{i=1}^{N} D_i(t)} R. \tag{13}$$

Then, utility function of a particular client $k$ is constructed as the sum of balance over the time horizon:

$$U_k(\boldsymbol{D_k}, \boldsymbol{D_{-k}}, R, \theta) = \sum_{t=0}^{T-1} \left( P_k(t) - C_k(t) \right), \tag{14}$$

(14) is dominated by its own buffered data volume vector $\boldsymbol{D_k}$, others buffered data volume vector $\boldsymbol{D_{-k}}$, the server's payment $R$ and conservation rate $\theta$. Client $k$ only controls its own increment $\Delta_k(t)$, a time-varying variable to influence its data volume indirectly. It aims to maximize utility

function in (14) under the constraint in (1). Hence, the optimization problem for a particular client $k$ can be constructed as

$$\max_{\substack{\boldsymbol{\Delta_k}=[\Delta_k(t)]\\ t\in[0,\cdots,T-1]}} U_k(\boldsymbol{D_k}, \boldsymbol{D_{-k}}, R, \theta), \\ \text{s.t.} D_k(t+1) = \theta D_k(t) + \Delta_k(t), \tag{15}$$

### 4.3. Stackelberg Game Formulation

Based on the discussions in 4.1 and 4.2, we use a two-stage Stackelberg Game to model the interaction between the server and clients as

$$\begin{aligned} &\text{Stage I}: \min_{R,\theta} U(\boldsymbol{D}, R, \theta), \\ &\text{Stage II}: \max_{\substack{\boldsymbol{\Delta_k}=[\Delta_k(t)]\\ t\in[0,\cdots,T-1]}} U_k(\boldsymbol{D_k}, \boldsymbol{D_{-k}}, R, \theta), \\ &\qquad \text{s.t.} D_k(t+1) = \theta D_k(t) + \Delta_k(t), \end{aligned} \tag{16}$$

where the server acts as a leader and clients are regarded as followers. The overview of incentive mechanism is depicted in Figure 2. Before each round, the server launches its optimal strategy of $(R, \theta)$ to clients in Stage I. Then in Stage II, according to $(R, \theta)$ announced by the server, each client $k \in \mathcal{N}$ decides its optimal strategy of $\Delta_k(t)$. As a result, the equilibrium of this game guarantees mutual optimality between the server and clients.

## 5. Methodology

In this section, we derive the equilibrium of the Stackelberg game in last section. By means of backward reduction, we provide the optimal strategy for clients and the server progressively.

### 5.1. Introduction of Mean-Field Term

For a particular client $k$, it needs to learn about the server's strategy $(R, \theta)$ and the total buffered data volume $\sum_{i=1}^{N} D_i(t)$ before deriving the optimal strategy at round $t$. However, in many real-world scenarios, the term of $\sum_{i=1}^{N}$ keeps unknown to client $k$ due to information isolation among clients. To tackle this problem, we introduce a mean-field term $\phi(t) = \sum_{i=1}^{N} D_i(t)$. From the perspective of mathematics, $\phi(t)$ is a given function and it can be considered as a known term. We replace $\sum_{i=1}^{N} D_i(t)$ in (14) with it, and the optimization problem of client $k$ can be reformulated as

$$\max_{\substack{\boldsymbol{\Delta_k}=[\Delta_k(t)]\\ t\in[0,\cdots,T-1]}} \sum_{t=0}^{T-1} \left( \frac{D_k(t)}{\phi(t)} R - \alpha_k \Delta_k(t)^2 - \beta_k D_k(t)^2 \right), \\ \text{s.t.} D_k(t+1) = \theta D_k(t) + \Delta_k(t). \tag{17}$$

Besides, the estimation of $\phi(t)$ is arranged in Section 5.4.

### 5.2. Optimal Strategy of Clients in Stage II

In stage II, provided the strategy $(R, \theta)$ launched by the server, we derive the optimal strategy of $\Delta_k(t)$ for client $k$ at round $t$. However, client $k$ faces a long-term optimization problem with dynamic constraint of update strategy. Thus, we adopt Hamilton equation to solve the problem in stage II. Then we have the following proposition.

**Proposition 5.1.** *For any client $k$ at arbitray round $t$, the optimal increment $\Delta_k(t)$ is*

$$\Delta_k(t) = \begin{cases} \max\left\{ \frac{1}{2\alpha_k} \sum_{\tau=t+1}^{T-1} \theta^{\tau-t-1} \left( \frac{R}{\phi(\tau)} - 2\beta_k D_k(\tau) \right), 0 \right\}, & t \in [0, T-2]; \\ 0, & t = T-1. \end{cases} \tag{18}$$

*Furthermore, the optimal buffer size $D_k(t)$ is*

$$D_k(t) = \begin{cases} D_0, & t = 0; \\ \theta^t D_k(0) + \sum_{\tau=0}^{t-1} \theta^{t-1-\tau} \Delta_k(\tau), & t \in [1, T-1]. \end{cases} \tag{19}$$

The detailed proof is provided in Appendix A.3.

*Remark* 5.2. (18) reveals that provided $R, \theta$ and $\phi(t)$, $\Delta_k(t)$ is affected by the subsequent buffered data volume of $D_k(\tau), \tau \in [t+1, \cdots, T]$, which in turn affects $\Delta_k(t)$ according to (19). Therefore, a close loop exists between $\boldsymbol{\Delta}_k$ and $\boldsymbol{D}_k$, where greater $\boldsymbol{\Delta}_k$ leads to greater $\boldsymbol{D}_k$, which in turn inhibits $\boldsymbol{\Delta}_k$.

*Remark* 5.3. A stationary state exists against increment $\Delta_k(t)$ in an infinite time horizon. The detailed proof is provided in Appendix A.4.

### 5.3. Optimal Strategy of Server in Stage I

In stage II, provided all clients' optimal strategy of $\boldsymbol{\Delta}_k, k \in [N]$, we derive the optimal strategy of $(R^*, \theta^*)$ for the server. But the server faces a two-variable optimization problem in complicated form, which makes it hard to derive a close-form solution of $(R^*, \theta^*)$ for the server. Thus, we explore the approximately optimal strategy for the server by adopting searching algorithms.

### 5.4. Estimation of Mean-Field Term

Ultimately, we try to estimate $\phi(t), t \in [0, \cdots, T-1]$, the mean-field term introduced in section 5.1 by means of fixed point algorithm. According to (19), $D_k(t)$ is a function of $\Delta_k(\tau), \tau \in [0, \cdots, t-1]$. By inserting (18) into (19), $D_k(t)$ is actually a function of $\phi(t)$ and $D_k(t), t \in [0, \cdots, T-1]$. We define this function as

$$D_k(t) = \Psi_{k,t}(\phi(0), \phi(1), \cdots, \phi(T-1), \tag{20}$$
$$D_k(0), D_k(1), \cdots, D_k(T-1)). \tag{21}$$

Furthermore, $\phi(t)$ is a function of $D_k(t), k \in [1, \cdots, N]$ at round $t$, so it can be derived that $D_k(t)$ is a function of

**Algorithm 1** `Client-Strategy`

**Input:** $\phi, R, \theta$.
**Output:** $\Delta_k(t), D_k(t), t \in [0, \cdots, T-1], k \in [1, \cdots, N]$.
**Initialize:** $D_k^0(t), t \in [0, \cdots, T-1], k \in [1, \cdots, N], j = 0, \epsilon$.
**repeat**
  **for** $t = 0$ **to** $T-1$ **do**
    **for** $k = 1$ **to** $N$ **do**
      Calculate $\Delta_k^j(t)$ using $\phi, R, \theta$ and $D_k^j(t)$ according to (17).
      Calculate $D_k^j(t)$ using $\Delta_k^j(t)$ according to (18).
    **end for**
  **end for**
  $j \leftarrow j + 1$.
**until** $D_k^{j+1}(t) - D_k^j(t) \leq \epsilon, k \in [1, \cdots, K], t \in [0, \cdots, T-1]$.

$D_k(t), k \in [1, \cdots, N], t \in [0, \cdots, T-1]$. That is

$$\begin{aligned} D_k(t) = \Psi_{k,t}(&D_1(0), D_1(1), \cdots, D_1(T-1), \\ &D_2(0), D_2(1), \cdots, D_2(T-1), \\ &\cdots, \\ &D_N(0), D_N(1), \cdots, D_N(T-1)). \end{aligned} \tag{22}$$

For ease of reading, we denote the parameter matrix in (22) as $A$. Then we have $D_k(t) = \Psi_{k,t}(A)$. To summarize buffered data volume $D_k(t)$ over the time horizon $t \in [0, \cdots, T-1]$ and $k \in [1, \cdots, N]$, we have the following vector function as

$$\begin{aligned} A =&\Psi(A) \\ =&(\Psi_{1,0}(A), \Psi_{1,1}(A), \cdots, \Psi_{1,T-1}(A) \\ &\Psi_{2,0}(A), \Psi_{2,1}(A), \cdots, \Psi_{2,T-1}(A) \\ &\cdots, \\ &\Psi_{K,0}(A), \Psi_{K,1}(A), \cdots, \Psi_{K,T-1}(A)), \end{aligned} \tag{23}$$

which is a mapping from $A$ to $A$. Then we have the following proposition.

**Proposition 5.4.** *$\Psi$ has a fixed point.*

The detailed proof is provided in Appendix A.5 Based on Proposition 5.4, we can adopt fixed point algorithm to estimate $\phi(t)$ precisely.

### 5.5. Description of Algorithm

**Algorithm 2** `Server-Strategy`

**Input:** $\phi$.
**Output:** $R, \theta$.
**def** Func$(R, \theta)$:
  $D_k(t) \leftarrow$ `Client-Strategy`$(\phi, R, \theta)$.
  Calculate *res* using $D_k(t)$ according to (16).
  **return** *res*.
$R^*, \theta^* \leftarrow$ *`PSO(Func)`*.
***return*** $R^*, \theta^*$.

**Algorithm 3** `Estimate-MFT`

**Input:** None.
**Output:** $\phi_k(t), k \in [1, \cdots, N], t \in [0, \cdots, T-1]$.
**Initialize:** $\phi_k^0(t), k \in [1, \cdots, N], t \in [0, \cdots, T-1], j = 0, \epsilon$.
**repeat**
  $R, \theta \leftarrow$ `Server-Strategy`$(\phi)$.
  $D_k(t) \leftarrow$ `Client-Strategy`$(\phi, R, \theta)$.
  **for** $t = 0$ **to** $T-1$ **do**
    Calculate $\phi_k^j(t)$ using $D_k^j(t)$ according to (20).
  **end for**
  $j \leftarrow j + 1$.
**until** $D_k^{j+1}(t) - D_k^j(t) \leq \epsilon, k \in [1, \cdots, K], t \in [0, \cdots, T-1]$.

## 6. Experiment

In this section, we evaluate the performance of proposed FedStream by numerical experiments.

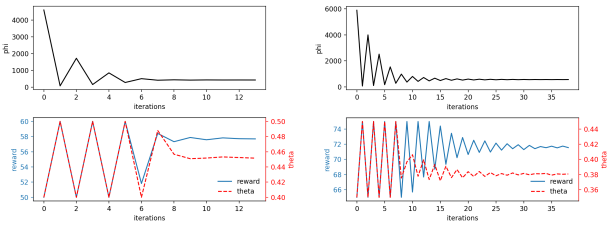
### 6.1. Experimental Setup

**Datasets and Models:** We conduct our experiments in two widely used real datasets: MNIST and FMNIST. The MNIST dataset contains 60,000 training samples and 10,000 test samples for handwrittern digit recognization. The FMNIST dataset comprises 50,000 training samples and 10,000 test samples for fashion item recognization. To simulate the impact of DoS in practical training, we mislabel parts of local buffer data of each client before each time slot, which is similar to the method adopted in (Xu et al., 2024). The proportion of mislabeling depends on time sensitivity coefficient $\sigma$. If we set a greater $\sigma$, which corresponds to a higher time-sensitive FL task, then the proportion of mislabeling will rise and vice versa. Besides, to simulate the setting of local data stream $\mathcal{D}$, we assign the dataset into all clients beforehand in a IID method. During the task, each client will sample new data from local dataset to its buffer according to the optimal strategy. Lastly, we use CNN as the training model in our experiments, which is made up of two sets of convolution layers and max pooling layers, and then two fully-connected layers and a RELU layer.

**Hyperparameters Settings:** We consider a federated learning setting with the number of rounds $T = 20$. In each round, there are 10 clients participating in the training. For a particular client $k$, it holds an initial buffer data with volume $D_k(0) = 700$ by default. During training, the unit price of

**Algorithm 4** `Main Procedure`

**Input:** $T, N$.
**Output:** $w(T)$.
**Initialize:** $w(0)$.
$\phi \leftarrow$ `Estimate-MFT()`.
$R, \theta \leftarrow$ `Server-Strategy`$(\phi)$.
$\Delta, D \leftarrow$ `Client-Strategy`$(\phi, R, \theta)$.
**for** $t = 0$ **to** $T - 1$ **do**
  Server distributes global model $w(t)$ to clients.
  **for** each client $k$ **in parallel do**
    Abandon $\theta D_k(t-1)$ data points randomly from buffer: $\widetilde{D}_k(t) = D_k(t-1) - \theta D_k(t-1)$.
    Collect $\Delta_k(t-1)$ data points from local data stream: $D_k(t) = \widetilde{D}_k(t) + \Delta_k(t-1)$.
    Execute local update with global model $w(t)$: $w_k(t+1) = w(t) - \eta \nabla F_k(w(t)|D_k(t))$.
    Upload local model $w_k(t+1)$ back to the server.
  **end for**
  Server aggregates local model $w_i(t+1), i \in [N]$: $w(t+1) = \sum_{i=1}^{N} \frac{D_i(t+1)}{\sum_{i=1}^{N} D_i(t+1)} w_i(t+1)$.
**end for**
**return** $w(T)$.




$N = 3, T = 5$ $\quad$ $N = 5, T = 10$

*Figure 3.* The iteration of $\phi$, $R, \theta$ and $\Delta$

data collection and training is set to be $\alpha_k \sim U(10^{-4}, 10^{-3})$ and $\beta_k \sim U(5^{-6}, 5^{-5})$ respectively. Each client conducts one local update at a round with learning rate $\eta = 10^{-2}$. In addition, the server's cost balance factor is $\gamma = 10^{-4}$, with time sensitivity coefficient of $\sigma = 1$ and strategy space of $\theta \in [0, 1]$, $R \in [0, 100]$.

As for the hyperparameters of $\kappa_2, \kappa_3$ and $\kappa_4$, we take the similar method adopted in (Wang et al., 2019) and (Huang et al., 2024), both of which estimate these values with a simple FL training procedure. By conducting the simple procedure, we obtain $\kappa_2 = 1$ and $\kappa_3 = \kappa_4 = 10^{-2}$.

### 6.2. Experimental Results

**Iteration of Mean-Field Term:**

**Effect of Client's Strategy on its Utility:** In this paragraph, we analyse the effect of increment strategy $\Delta_k(t)$ on

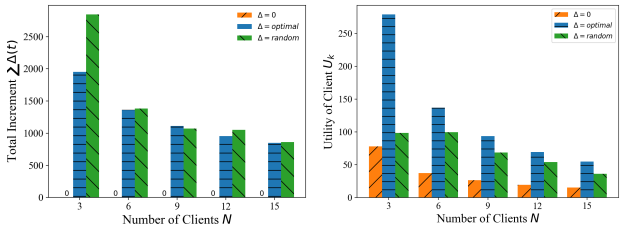



*Figure 4.* Comparative analysis of total increment $\sum_{t=0}^{T-1} \Delta_k(t)$ (**left**) and utility function $U_k$ (**right**) against a particular client $k$ versus number of clients $N$ across three increment strategies, including $\Delta_k(t) = 0$, optimal and random.

clients' utility and the results are plotted in Figure 4. For comparison, we implement two auxiliary strategies: zero strategy and random strategy. For a particular client $k$, zero strategy means it doesn't collect any new data through the training procedure, while random strategy refers that it collects new data randomly in each time slot. Note that in order to keep comparison meaningful, the total amount of new data collected over the time horizon under random strategy is set to be roughly consistent with that under optimal strategy. We can find optimal strategy helps client $k$ secure the highest utility compared with other two strategies. Provided clients are selfish, this indicates that each of them will follow the optimal strategy, thereby the mutually best response strategies are reached simultaneously, which meets the requirement the Nash equilibrium solution of Stage II. In addition, we can find under the same increment strategy, the client's utility $U_k$ decreases with number of clients $N$. The underlying reason is that clients expansion may intensify the competition for a fixed reward among them, thereby leads to reward reduction and utility reduction.

**Effect of Server's Strategy on Average Buffer Data:** In this paragraph, we study the effect of server's strategy of both reward $R$ and conservation rate $\theta$ on average buffer data from three terms: average increment $\Delta(t)$, average data size $D(t)$ and average data staleness $S(t)$. The results are depicted in Figure 5, 6 and 7 respectively. Specially, we fix the reward $R_0 = 61.62$ and consider four types of strategies: $\pi_1 = (R_0, 0.1)$, $\pi_2 = (R_0, 0.5)$, $\pi_3 = (R_0, 0.7)$, $\pi_4 = (R_0, 0.9)$. Besides, we also fix $\theta_0 = 0.3$ and consider another four strategies: $\pi_5 = (30, \theta_0)$, $\pi_6 = (60, \theta_0)$, $\pi_7 = (100, \theta_0)$, $\pi_8 = (140, \theta_0)$.

According to Figure 5, provided a fixed $R$, $\Delta(t)$ decreases with $\theta$. The underlying realistic meaning is that the more valuable previous data are, the greater $\theta$ is set by central server to conserve previous data, thereby the less necessarily central server recruits new data. In contrast, given a fixed $\theta$, $\Delta(t)$ increases with $R$. That's because a greater reward can effectively stimulate clients to collect more new data for training. In addition, we notice that $\Delta(t)$ steps into a stationary state after several communication rounds no mat-

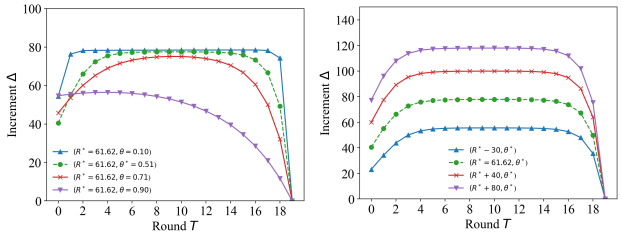



*Figure 5.* Comparative analysis of increment $\Delta_k(t)$ against a particular client $k$ versus communication rounds $t$ across various value of $\theta$ ranging from 0 to 1 (**left**) and $R$ ranging from 0 to 150 (**right**).

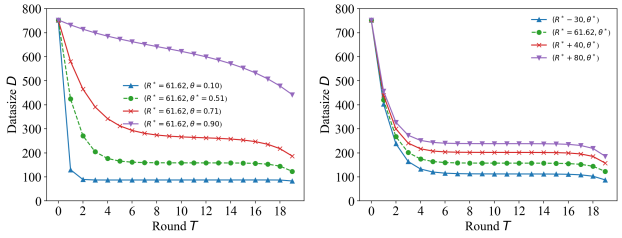



*Figure 6.* Comparative analysis of data size $D_k(t)$ against a particular client $k$ versus communication rounds $t$ across various value of $\theta$ ranging from 0 to 1 (**left**) and $R$ ranging from 0 to 150 (**right**).

ter the strategy $\pi$, and then converges to 0 before the end of training. The observation of stationary state is substantiated by remark 5.3, wherein it is illustrated that a stationary state exists in an dynamic circumstance where clients adopts updated method based on (1). And the convergence trend to 0 is also in line with the boundary condition proposed in (18).

For data volume $D(t)$, as shown in Figure 6, the volume of buffer data $D(t)$ increases with both $\theta$ and $R$ since greater $\theta$ means to conserve more previous data and greater $R$ means to collect more new data from local data stream. Besides, the stationary state of $D(t)$ keeps pace with that of $\Delta(t)$ according to (19).

The observation against staleness $S(t)$ is not the same as that against volume. In Figure 7, we can find as $\theta$ grows, $S(t)$ soars to an unacceptable level because of rapid accumulation of deteriorated previous data. Yet it seldom effected by the change of $R$. Thus, the observation demonstrates that $S(t)$ is dominated by $\theta$ rather than $R$. It's not trivial to strike the balance between data volume and staleness by adjusting $\theta$ for a better model performance.

**Effect of Server's Strategy on its Cost:** In this paragraph, we analyse the effect of server's strategy $(R, \theta)$ on its cost. Under the hyperparameters settings in experimental setup, the optimal strategy is $\pi^* = (61.62, 0.51)$. For comparison, we fix the optimal reward $R^*$ and consider three types of strategies: $\pi_1 = (R^*, \theta^* - 0.4), \pi_2 = (R^*, \theta^* + 0.2), \pi_3 = (R^*, \theta^* + 0.4)$. Besides, we also fix the optimal theta $\theta^*$ and consider another three types of strategies: $\pi_4 = (R^* -$

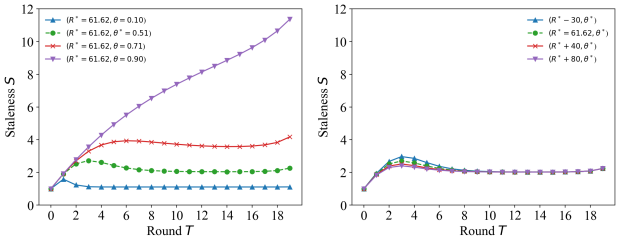



*Figure 7.* Comparative analysis of data staleness $S_k(t)$ against a particular client $k$ versus communication rounds $t$ across various value of $\theta$ ranging from 0 to 1 (**left**) and $R$ ranging from 0 to 150 (**right**).

$30), \pi_5 = (R^*, \theta^* + 40), \pi_6 = (R^*, \theta + 80)$.

The effect of $\theta$ and $R$ on test accuracy and cost is shown in Figure 8 and 9. We can find that $\pi^*$(solid line) outperforms $\pi_1, \pi_2$ and $\pi_3$(dotted line) in terms of test accuracy on the MNIST and CIFAR-10 datasets. The underlying reason is that strategy $\pi_2, \pi_3$ with improperly great $\theta$ contributes to huge but stale buffer data, while $\pi_1$ with improperly small $\theta$ contributes to fresh but tiny ones. Both of them leads to worse model performance. Under the same payment to clients, $\pi^*$ reach the lowest cost compared with other strategies.

In addition, $\pi_5$ and $\pi_6$(dotted line) surpass the optimal strategy of $\pi^*$(solid line) with respect to test accuracy across all datasets because richer payment encourages clients to collect more fresh data, thereby enhance the model performance. However, $\pi^*$ still reach the lowest cost, which indicates that our proposed strategy has the ability to strike the trade-off between payment and accuracy loss simultaneously. The above two experiments verify the optimality of our proposed strategy.

**Effect of Hyperparameters:** In this paragraph, we explore the result of effect of hyperparameters such as time-sensitive coefficient $\sigma$ and initial volume of data $D(0)$ on exprements.

For generalization, we set three task modes: low time sensitivity, normal sensitivity and high sensitivity. The low time sensitivity mode usually corresponds to weak-dynamic scenarios where the data distribution is stable, such as long-term agriculture analysis or chronic disease analysis, while the high time sensitivity mode corresponds to strong-dynamic scenarios where the data distribution is unstable, such as real-time personalized recommendation and smart transporation. To simulate the above three modes in the experiments, we set the time sensitivity coefficient of $\sigma$ as $0.1, 0.3$ and $0.7$ respectively. The optimal strategies under various $\sigma$ are shown in Figure 10. As $\sigma$ increases, the optimal reward $R$ rises with optimal conservation rate $\theta$ drops. It matches the intution that when facing tasks with higher time sensitivity, the value of previous data decrease. Server

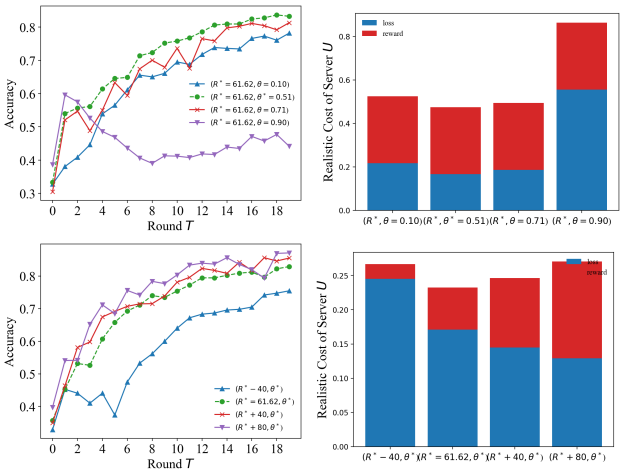



*Figure 8.* Comparative analysis of test accuracy (**left**) and cost (**right**) versus communication rounds $t$ across various value of $\theta$ ranging from 0 to 1(**top**) and $R$ ranging from 0 to 140(**bottom**) with CNN over MNIST. Left-Top: Test accuracy vs. Rounds, Right-Top: Cost vs. Rounds

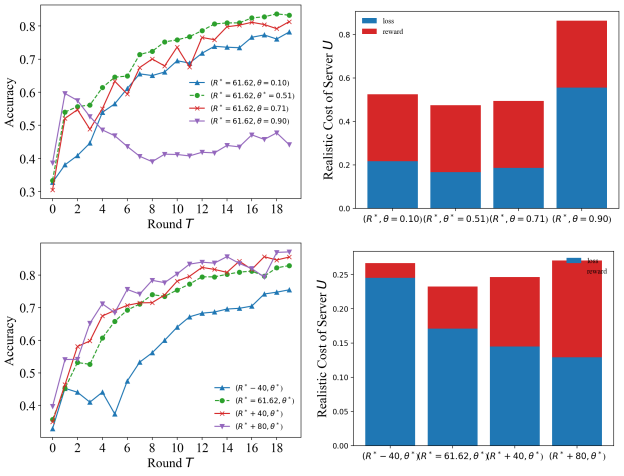



*Figure 9.* Comparative analysis of test accuracy (**left**) and cost (**right**) versus communication rounds $t$ across various value of $\theta$ ranging from 0 to 1(**top**) and $R$ ranging from 0 to 140(**bottom**) with CNN over FMNIST. Left-Top: Test accuracy vs. Rounds, Right-Top: Cost vs. Rounds

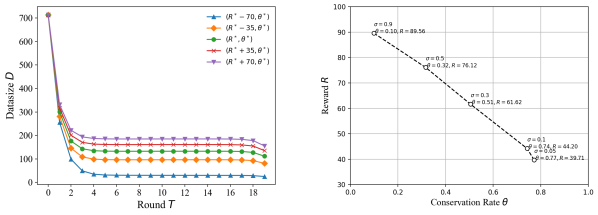



*Figure 10.* Comparative analysis of increment $\Delta_k(t)$ against a particular client $k$ versus communication rounds $t$ across various value of $\theta$ ranging from 0 to 1 (**left**) and $R$ ranging from 0 to 100 (**right**).

need to guide clients to abandon previous data as well as to collect new data in time. This illustrates that our proposed strategy can be applied in various different scenarios.

Next, we exhibite the effect of initial data size $D_k(0)$ on data volume. There is a key question about *whether the volume of data keeps dropping along with communication rounds*. It seems elusive from the perspective of intution. For comparison, We sampe three initial data volume: $D^1(0) = 50$, $D^2(0) = 100$ and $D^3(0) = 700$. The results is shown in Figure 10. We can observe that $D(0)$ only makes differneces on a few rounds at the begining of training, then they converge to the same stationary state and drop synchronously. From the aspect of mathematics, stationary state doesn't depend on $D(0)$ as demonstrated in remark 5.3. Therefore, the answer to the above question is that the trend of curve of data volumn banks on $D(0)$ and stationary state at the same time. If $D(0)$ is greater than stationary state, then $D(t)$ keeps rising before reaching the stationary state, and vice versa.

## References

Ding, N., Fang, Z., and Huang, J. Optimal contract design for efficient federated learning with multi-dimensional private information. *IEEE Journal on Selected Areas in Communications*, 39(1):186–200, 2020.

Fang, M., Wang, X., Xu, C., Yang, H. H., and Quek, T. Q. Computing-aided update for information freshness in the internet of things. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 1–7. IEEE, 2021.

Huang, G., Wu, Q., Sun, P., Ma, Q., and Chen, X. Collaboration in federated learning with differential privacy: A stackelberg game analysis. *IEEE Transactions on Parallel and Distributed Systems*, 2024.

Jiao, Y., Wang, P., Niyato, D., Lin, B., and Kim, D. I. Toward an automated auction framework for wireless federated learning services market. *IEEE Transactions on Mobile Computing*, 20(10):3034–3048, 2020.

Kang, J., Xiong, Z., Niyato, D., Xie, S., and Zhang, J. Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory. *IEEE Internet of Things Journal*, 6(6): 10700–10714, 2019.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lim, W. Y. B., Xiong, Z., Miao, C., Niyato, D., Yang, Q., Leung, C., and Poor, H. V. Hierarchical incentive mechanism design for federated machine learning in mobile networks. *IEEE Internet of Things Journal*, 7(10):9575–9588, 2020.

Tripathi, V. and Modiano, E. Age debt: A general framework for minimizing age of information. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 1–6. IEEE, 2021.

Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., and Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE journal on selected areas in communications*, 37(6):1205–1221, 2019.

Wang, X. and Duan, L. Dynamic pricing for controlling age of information. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pp. 962–966. IEEE, 2019.

Wang, Z., Gao, L., and Huang, J. Taming time-varying information asymmetry in fresh status acquisition. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pp. 1–10. IEEE, 2021.

Wei, K., Li, J., Ding, M., Ma, C., Yang, H. H., Farokhi, F., Jin, S., Quek, T. Q., and Poor, H. V. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE transactions on information forensics and security*, 15:3454–3469, 2020.

Wu, C., Xiao, M., Wu, J., Xu, Y., Zhou, J., and Sun, H. Towards federated learning on fresh datasets. In *2023 IEEE 20th International Conference on Mobile Ad Hoc and Smart Systems (MASS)*, pp. 320–328. IEEE, 2023.

Xiao, M., Xu, Y., Zhou, J., Wu, J., Zhang, S., and Zheng, J. Aoi-aware incentive mechanism for mobile crowdsensing using stackelberg game. In *IEEE INFOCOM 2023-IEEE Conference on Computer Communications*, pp. 1–10. IEEE, 2023.

Xu, Y., Xiao, M.-J., Wu, C., Wu, J., Zhou, J.-R., and Sun, H. Age-of-information-aware federated learning. *Journal of Computer Science and Technology*, 39(3):637–653, 2024.

Zhang, N., Ma, Q., and Chen, X. Enabling long-term cooperation in cross-silo federated learning: A repeated game perspective. *IEEE Transactions on Mobile Computing*, 22(7):3910–3924, 2022.

In the appendix, the complete proofs of theoretic results provided in the main text and more contents about experiment will be exhibited in detail.

# A. Proof of Theoretic Results

## A.1. Proof of Remark 3.3

*Proof.* According to the definition of DoS, we have

$$\begin{aligned}
S_k(t+1) &= S_k(t)\frac{\theta D_k(t)}{D_k(t+1)} + 1 = S_k(t)\left(1 - \frac{\Delta_k(t)}{D_k(t+1)}\right) + 1 \\
&= \left(S_k(t-1)\frac{\theta D_k(t-1)}{D_k(t)} + 1\right)\frac{\theta D_k(t)}{D_k(t+1)} + 1 \\
&= S_k(t-1)\frac{\theta^2 D_k(t-1)}{D_k(t+1)} + \frac{\theta D_k(t)}{D_k(t+1)} + 1 \\
&= S_k(t-2)\frac{\theta^3 D_k(t-2)}{D_k(t+1)} + \frac{\theta^2 D_k(t-1)}{D_k(t+1)} + \frac{\theta D_k(t)}{D_k(t+1)} + 1 \\
&= \cdots \\
&= S_k(0)\frac{\theta^{t+1} D_k(0)}{D_k(t+1)} + \frac{\theta^t D_k(1)}{D_k(t+1)} + \cdots + \frac{\theta^2 D_k(t-1)}{D_k(t+1)} + \frac{\theta D_k(t)}{D_k(t+1)} + 1 \\
&= \sum_{\tau=0}^{t+1} \frac{\theta^{t+1-\tau} D_k(\tau)}{D_k(t+1)}.
\end{aligned} \tag{24}$$

□

## A.2. Proof of Theorem 3.5

*Proof.* Let's pay attention to the round $t+1$.

$$\begin{aligned}
&E[F(w(t+1)|D(t+1)) - F(w(t)|D(t))] \\
=&E[F(w(t+1)|D(t)) - F(w(t)|D(t))] + E[F(w(t+1)|D(t+1)) - F(w(t+1)|D(t))] \\
=&\underbrace{E[F(w(t+1)|D(t)) - F(w(t)|D(t))]}_{A} + \Omega_t,
\end{aligned} \tag{25}$$

where $\Omega_t = E[F(w(t+1)|D(t+1)) - F(w(t+1)|D(t))]$. It captures the expected difference of global loss function based on the same global model $w(t)$ between buffered data at present round $D(t)$ and at previous round $D(t-1)$. Then we focus on $A$. Since $F_k(w)$ is $\beta$-Lipschitz smooth, we have

$$\begin{aligned}
&E[F(w(t+1)|D(t)) - F(w(t)|D(t))] \\
\leq&E\left\langle \nabla F(w(t)|D(t)), w(t+1) - w(t)|D(t)\right\rangle + \frac{\beta}{2}E\|w(t+1) - w(t)|D(t)\|^2 \\
=&E\left\langle \nabla F(w(t)|D(t)), (-\eta)\nabla F(w(t)|D(t))\right\rangle + \frac{\beta}{2}E\|w(t+1) - w(t)|D(t)\|^2 \\
=&(-\eta)\|\nabla F(w(t)|D(t))\|^2 + \underbrace{\frac{\beta}{2}E\|w(t+1) - w(t)|D(t)\|^2}_{B}.
\end{aligned} \tag{26}$$

Next, we focus on bounding $B$. Because the stochastic gradient of $\nabla F_k(w)$ is unbiased and variance-bounded, we have

$$\begin{aligned}
&\frac{\beta}{2}E\|w(t+1)-w(t)|D(t)\|^2\\
=&\frac{\beta}{2}E\|(-\eta)\nabla F(w(t)|D(t))\|^2\\
\leq&\beta\eta^2 E\|\nabla F(w(t)|D(t))\|^2\\
=&\beta\eta^2 E\left\|\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}\nabla F_k(w(t)|D_k(t))\right\|^2\\
\leq&\beta\eta^2\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}E\|\nabla F_k(w(t)|D_k(t))\|^2\\
\leq&\beta\eta^2\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}\left(2\|\nabla F(w(t)|D(t))\|^2+\frac{\psi^2}{D_k(t)}+S_k(t)\sigma^2\right)\\
=&2\beta\eta^2\|\nabla F(w(t)|D(t))\|^2+\beta\eta^2\frac{N\psi^2}{D(t)}+\beta\eta^2\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}S_k(t)\sigma^2. \qquad (27)
\end{aligned}$$

Substituting (27) into (26), we have

$$\begin{aligned}
&E[F(w(t+1)|D(t))-F(w(t)|D(t))]\\
\leq&\underbrace{(2\beta\eta^2-\eta)E\|\nabla F(w(t)|D(t))\|^2}_{C}+2\beta\eta^2\frac{N\psi^2}{D(t)}+\beta\eta^2\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}S_k(t)\sigma^2. \qquad (28)
\end{aligned}$$

Now we bound $C$. We set $\eta<\frac{1}{2\beta}$, then $2\beta\eta^2-\eta<0$. Polyak Lojasiewicz condition holds for $F_k(w)$ due to strong convexity in assumptions:

$$E[F(w(t)|D(t))-F(w^*)]\leq\frac{1}{2\mu}E\|\nabla F(w(t)|D(t))\|_2^2. \qquad (29)$$

Thus the follwing inequality holds:

$$2\mu(2\beta\eta^2-\eta)E[(F(w(t)|D(t))-F(w^*))]\geq(2\beta\eta^2-\eta)E\|\nabla F(w(t)|D(t))\|_2^2. \qquad (30)$$

Substituting (30) into (28), we have

$$\begin{aligned}
&E[F(w(t+1)|D(t))-F(w(t)|D(t))]\\
\leq&2\mu(2\beta\eta^2-\eta)E[F(w(t)|D(t))-F(w^*)]+2\beta\eta^2\frac{N\psi^2}{D(t)}+\beta\eta^2\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}S_k(t)\sigma^2. \qquad (31)
\end{aligned}$$

Consequently, substituting (31) into (25), we have

$$\begin{aligned}
&E[F(w(t+1)|D(t+1))-F(w(t)|D(t))]\\
=&E[F(w(t+1)|D(t))-F(w(t)|D(t))]+\Omega_t\\
\leq&2\mu(2\beta\eta^2-\eta)E[F(w(t-1)|D(t))-F(w^*|D(t))]+2\beta\eta^2\frac{N\psi^2}{D(t)}+\beta\eta^2\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}S_k(t)\sigma^2+\Omega_t. \qquad (32)
\end{aligned}$$

Adding $E[F(w(t)|D(t))-F(w^*)]$ on both sides, we get

$$\begin{aligned}
&E[F(w(t+1)|D(t+1))-F(w^*)]\\
\leq&(1+4\mu\beta\eta^2-2\mu\eta)E[F(w(t)|D(t))-F(w^*)]+2\beta\eta^2\frac{N\psi^2}{D(t)}+\beta\eta^2\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}S_k(t)\sigma^2+\Omega_t. \qquad (33)
\end{aligned}$$

Then call (33) recursively, we have

$$
\begin{aligned}
&E[F(w(t+1)|D(t+1)) - F(w^*)] \\
\leq&(1+4\mu\beta\eta^2-2\mu\eta)E[F(w(t)|D(t)) - F(w^*)] + 2\beta\eta^2\frac{N\psi^2}{D(t)} + \beta\eta^2\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}S_k(t)\sigma^2 + \Omega_t \\
\leq&(1+4\mu\beta\eta^2-2\mu\eta)^2E[F(w(t-1)|D(t-1)) - F(w^*)] \\
&+ (1+4\mu\beta\eta^2-2\mu\eta)\left[2\beta\eta^2\frac{N\psi^2}{D(t-1)} + \beta\eta^2\sum_{k=1}^{N}\frac{D_k(t-1)}{D(t-1)}S_k(t-1)\sigma^2 + \Omega_{t-1}\right] \\
&+ \left[2\beta\eta^2\frac{N\psi^2}{D(t)} + \beta\eta^2\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}S_k(t)\sigma^2 + \Omega_t\right] \\
\leq&\cdots \\
\leq&(1+4\mu\beta\eta^2-2\mu\eta)^{t+1}E[F(w(0)|D(0)) - F(w^*)] \\
&+ \sum_{r=0}^{t}(1+4\mu\beta\eta^2-2\mu\eta)^r\left[2\beta\eta^2\frac{N\psi^2}{D(t-r)} + \beta\eta^2\sum_{k=1}^{N}\frac{D_k(t-r)}{D(t-r)}S_k(t-r)\sigma^2 + \Omega_{t-r}\right].
\end{aligned} \tag{34}
$$

For ease of representation, let $\kappa_1 = 1+4\mu\beta\eta^2-2\mu\eta$, $\kappa_2 = 2\beta\eta^2$ and $\kappa_3 = \beta\eta^2$. Thus, the convergence upper bound after $T+1$ rounds can be reformulated as

$$
\begin{aligned}
&E[F(w(T+1)|D(T+1)) - F(w^*)] \\
\leq&\kappa_1^{T+1}E[F(w(0)|D(0)) - F(w^*)] + \sum_{t=0}^{T}\kappa_1^t\left[\kappa_2\frac{N\psi^2}{D(T-t)} + \kappa_3\sum_{k=1}^{N}\frac{D_k(T-t)}{D(T-t)}S_k(T-t)\sigma^2 + \Omega_{T-t}\right] \\
=&\kappa_1^{T+1}E[F(w(0)|D(0)) - F(w^*)] + \sum_{t=0}^{T}\kappa_1^{T-t}\left[\kappa_2\frac{N\psi^2}{D(t)} + \kappa_3\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}S_k(t)\sigma^2 + \Omega_t\right].
\end{aligned} \tag{35}
$$

Further, the convergence upper bound after $T$ rounds is

$$
\begin{aligned}
&E[F(w(T)|D(T)) - F(w^*)] \\
=&\kappa_1^{T}E[F(w(0)|D(0)) - F(w^*)] + \sum_{t=0}^{T-1}\kappa_1^{T-1-t}\left[\kappa_2\frac{N\psi^2}{D(t)} + \kappa_3\sum_{k=1}^{N}\frac{D_k(t)}{D(t)}S_k(t)\sigma^2 + \Omega_t\right].
\end{aligned} \tag{36}
$$

□

### A.3. Proof of Proposition 5.1

*Proof.* For the long term optimization problem with dynamic constraint in (17), we construct a Hamilton equation $H_k(t)$ as follows:

$$
H_k(t) = \frac{\overline{\delta_k}^{-1}D_k(t)}{\phi(t)}R - \alpha_k\Delta_k(t)^2 - \beta_k D_k(t)^2 + \lambda_k(t+1)\left((\theta-1)D_k(t) + \Delta_k(t)\right). \tag{37}
$$

In addtion, $\frac{\partial^2 H_k(t)}{\partial\Delta_k(t)^2} = -2\alpha_k < 0$. That is, $H_k(t)$ is a concave function in $\Delta_k(t)$. To derive the optimal increment $\Delta_k(t)$ that minimize (17), it satisfies

$$
\frac{\partial H_k(t)}{\partial\Delta_k(t)} = 0, \tag{38}
$$

$$
\frac{\partial H_k(t)}{\partial D_k(t)} = \lambda_k(t) - \lambda_k(t+1). \tag{39}
$$

By solving the above two formulas, we have

$$\Delta_k(t) = \frac{1}{2\alpha_k}\lambda_k(t+1), \tag{40}$$

$$\lambda_k(t) = \theta\lambda_k(t+1) + \frac{\overline{\delta_k}^{-1}R}{\phi(t)} - 2\beta_k D_k(t). \tag{41}$$

In addition, it can be derived that $\Delta_k(T-1) = 0$ because $\Delta_k(T-1)$ decides round $T$'s increment for client $k$. The buffered data volume $D_k(T)$ will not be involved in the training rounds which ranges from 0 to $T-1$. Therefore $D_k(T)$ cannot bring any benefit for client $k$ under high collection expenditure. It's feasible for client $k$ to set $\Delta_k(T-1) = 0$ and stop data collection. Based on this, the boundary condition can be expressed as

$$\begin{aligned}\lambda_k(T-1) &= \frac{\partial\left(\frac{\overline{\delta_k}^{-1}D(T-1)}{\phi(T-1)}R - \beta_k D_k(T-1)^2\right)}{\partial D_k(T-1)} \\ &= \frac{\overline{\delta_k}^{-1}R}{\phi(T-1)} - 2\beta_k D_k(T-1).\end{aligned} \tag{42}$$

Combing (41) with (42), we have

$$\begin{aligned}\lambda_k(t) &= \theta\lambda_k(t+1) + \frac{\overline{\delta_k}^{-1}R}{\phi(t)} - 2\beta_k D_k(t) \\ &= \theta^2\lambda_k(t+2) + \theta\left(\frac{\overline{\delta_k}^{-1}R}{\phi(t+1)} - 2\beta_k D_k(t+1)\right) + \frac{\overline{\delta_k}^{-1}R}{\phi(t)} - 2\beta_k D_k(t) \\ &= \cdots \\ &= \theta^{T-1-t}\lambda_k(T-1) + \sum_{\tau=0}^{T-2-t}\theta^\tau\left(\frac{\overline{\delta_k}^{-1}R}{\phi(t+\tau)} - 2\beta_k D_k(t+\tau)\right) \\ &= \theta^{T-1-t}\left(\frac{\overline{\delta_k}^{-1}R}{\phi(T-1)} - 2\beta_k D_k(T-1)\right) + \sum_{\tau=0}^{T-2-t}\theta^\tau\left(\frac{\overline{\delta_k}^{-1}R}{\phi(t+\tau)} - 2\beta_k D_k(t+\tau)\right) \\ &= \sum_{\tau=0}^{T-1-t}\theta^\tau\left(\frac{\overline{\delta_k}^{-1}R}{\phi(t+\tau)} - 2\beta_k D(t+\tau)\right) \\ &= \sum_{\tau=t}^{T-1}\theta^{\tau-t}\left(\frac{\overline{\delta_k}^{-1}R}{\phi(\tau)} - 2\beta_k D_k(\tau)\right).\end{aligned} \tag{43}$$

and

$$\lambda_k(t+1) = \sum_{\tau=t+1}^{T-1}\theta^{\tau-t-1}\left(\frac{\overline{\delta_k}^{-1}R}{\phi(\tau)} - 2\beta_k D_k(\tau)\right). \tag{44}$$

Substituting (44) into (40), we obtain

$$\Delta_k(t) = \frac{1}{2\alpha_k}\sum_{\tau=t+1}^{T-1}\theta^{\tau-t-1}\left(\frac{\overline{\delta_k}^{-1}R}{\phi(\tau)} - 2\beta_k D_k(\tau)\right). \tag{45}$$

Substituting (45) into (1), we obtain

$$\begin{aligned}D_k(t+1) &= \theta D_k(t) + \Delta_k(t) \\ &= \theta^2 D_k(t-1) + \theta\Delta_k(t-1) + \Delta_k(t) \\ &= \cdots \\ &= \theta^{t+1}D_k(0) + \sum_{\tau=0}^{t}\theta^{t-\tau}\Delta_k(\tau).\end{aligned} \tag{46}$$

with $t \in [0, T-2]$ and $\Delta_k(T-1) = 0$. □

### A.4. Proof of Remark 5.3

*Proof.* Under the infinite time horizon $T \to \infty$, the optimal strategy of a particular client $k$ in (18) is

$$\Delta_k(t) = \frac{1}{2\alpha_k} \sum_{\tau=t+1}^{\infty} \theta^{\tau-t-1} \left( \frac{R}{\phi(\tau)} - 2\beta_k D_k(\tau) \right) \tag{47}$$

$$= \frac{1}{2\alpha_k} \left[ \left( \frac{R}{\phi(t+1)} - 2\beta_k D_k(t+1) \right) + \theta \left( \frac{R}{\phi(t+2)} - 2\beta_k D_k(t+2) \right) + \cdots + \theta^{\infty} \left( \frac{R}{\phi(\infty)} - 2\beta_k D_k(\infty) \right) \right], \tag{48}$$

Thereby, when $t$ trends to $\infty$, the optimal strategy is as follows:

$$\lim_{t\to\infty} \Delta_k(t) = \frac{1}{2\alpha_k} \left[ \left( \frac{R}{\phi(\infty))} - 2\beta_k D_k(\infty) \right) + \theta \left( \frac{R}{\phi(\infty)} - 2\beta_k D_k(\infty) \right) + \cdots + \theta^{\infty} \left( \frac{R}{\phi(\infty)} - 2\beta_k D_k(\infty) \right) \right] \tag{49}$$

$$= \frac{1}{2\alpha_k(1-\theta)} \left( \frac{R}{\phi(\infty)} - 2\beta_k D_k(\infty) \right). \tag{50}$$

□

### A.5. Proof of Proposition 5.4

*Proof.* According to Brouwer's fixed point theorem, we need to prove that $\Psi$ is a continuous mapping from a closed set to itself. First, we prove that $\Psi$ is a mapping from a close set to itself. We bound $D_k(t)$ as [0, U], where $D_k(t) \geq 0$ means buffered data volume must be non-negative, and $D_k(t) \leq U$ means the storage capacity of buffer can't exceed $U$. Then, the domain of $\Psi$ can be bounded as

$$\Pi = [0, U] \times [0, U] \times \cdots \times [0, U]. \tag{51}$$

Therefore, $\Psi$ is a mapping from a close set $\Pi$ to itself.

Then, we prove that $\Psi$ is a continuous mapping in $\Pi$. It's obvious that $\Psi_k(t)$ is continuous because $D_k(t)$ is continuous. $\Psi$ is a linear combination of $\Psi_k(t)$, which refers that it's continuous. □