applied sciences

MDPI

Article

FedNow: An Efficiency-Aware Incentive Mechanism Enables Privacy Protection and Efficient Resource Utilization in Federated Edge Learning

Jianfeng Lu [1], Wenxuan Yuan [1], Shuqin Cao [1,*] and Pan Zhou [2]

[1] School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430065, China; lujianfeng@wust.edu.cn (J.L.); wenxuan@wust.edu.cn (W.Y.)
[2] School of Cyber Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China; panzhou@hust.edu.cn
* Correspondence: shuqincao@wust.edu.cn

Abstract: Federated edge learning (FEL) has recently attracted great interest due to its real-time response and energy-efficient characteristics. Most existing work focuses on designing algorithms to improve model performance, ignoring the malicious behavior and personal decision-making of self-interested edge servers. Although some efforts have been devoted to incentivizing honest edge server engagement by compensating training costs, this rarely considers resource efficiency and often assumes that edge servers provide complete information to the platform, which may lead to the risk of private attribute leakage. Hence, we aim to achieve an incentive mechanism that promotes secure and efficient model training between the platform and edge servers. However, edge servers' multi-dimensional private attributes and training strategies make the optimization problem nonconvex, and incomplete information further increases the complexity of the analysis. In order to address these challenges and by integrating contract theory and exponential mechanism, we propose an efficiency-aware incentive mechanism, FedNow, which enables edge servers to personally determine their local training rounds while motivating their participation without giving access to their true training strategies and private attributes. Specifically, we enabld edge servers to add noise to their submitted training strategy to hide their true training rounds; then, we carefully designed an efficiency score function to select honest and efficient edge servers without disclosing their private attributes. In order to demonstrate that FedNow strictly outperforms existing schemes in terms of total costs, we theoretically derived sufficient conditions for making the total costs of FedNow lower than existing schemes and designed a greedy algorithm that uses the Monte Carlo method to find feasible near-optimal solutions in polynomial time. Our extensive experimental assessment using synthetic and real datasets shows the superiority of FedNow.

Keywords: federated edge learning; exponential mechanism; resource efficiency; contract-based incentive mechanism

Citation: Lu, J.; Yuan, W.; Cao, S.; Zhou, P. FedNow: An Efficiency-Aware Incentive Mechanism Enables Privacy Protection and Efficient Resource Utilization in Federated Edge Learning. *Appl. Sci.* **2024**, *14*, 494. https://doi.org/10.3390/app14020494

Academic Editor: Christos Bouras

Received: 11 December 2023
Revised: 29 December 2023
Accepted: 3 January 2024
Published: 5 January 2024

1. Introduction

With the development of the internet of things, a huge number of smart devices, such as mobile phones and sensing devices, generate a massive amount of data every day [1–3]. Relying entirely on a central cloud for global model aggregation will consume considerable bandwidth resources and may result in significant response delays due to the long-haul transmission of massive data [4]. Besides, data privacy is prone to the invasion of unauthorized devices during transmission [5]. Federated edge learning (FEL) is a highly promising paradigm for implementing federated learning (FL)-based solutions in edge computing systems [6]. Compared to FL, FEL enables partial storage and computation resources to be offloaded from the data center (e.g., the cloud platform or one of the edge

servers) to the network edge servers (e.g., clients or communication base stations), which reduces the communication frequency and the risk of data leakage [7].

Although FEL has natural advantages in resisting a single point of failure and malicious attacks, the uploaded local models or gradients still pose a risk in terms of privacy leakage. For example, an untrusted cloud platform may infer the sensitive attributes of edge servers from the gradient updates [8], which hinders the willingness of edge servers to participate in FEL. Moreover, the inefficient resource utilization caused by device heterogeneity and the low-quality updates from inactive participants remain open issues in FEL. On the one hand, the performance of FEL systems can be affected by various factors related to both computation and communication [9], such as the data processing and transmission capability of edge servers and the network channel delay of data-collection devices. These factors may prevent each edge server's training progress in global training rounds from being synchronized, which introduces a *synchronization barrier* in FEL, particularly when requiring uniform update frequencies across all edge servers [4] (*The synchronization barrier*, i.e., the total training delay of an FEL system, is dominated by the slowest edge server, which leads to useless waiting times and the under-utilization of computation resources for efficient edge servers [10]). Some existing efforts have explored strategies, such as dynamic batch-size methods [11] and staleness-bounded updates [12,13], to alleviate the negative impact of the synchronization barrier. However, a dynamic batch-size method may not adapt well to the heterogeneous computing ability of edge servers, and the staleness-bounded update may have a slower convergence rate due to the potentially stale gradients of stragglers [14]. On the other hand, many existing works that focus on improving system efficiency usually assume unconditional learning resource sharing between the edge servers and the cloud platform [15,16]. This may not always be possible, as edge servers belong to different entities (e.g., organizations or enterprises) that are unwilling to contribute any computing resource without sufficient financial compensation. It is worth noting that some early and important works incentivize edge servers' effective participation by using auctions [17], contracts [18], and Stackelberg games [19], but they overlook the waiting time of heterogeneous edge servers [18] or focus on motivating low-cost learners [20], which may result in low resource utilization and inefficient model training.

In order to address the joint optimization problem of training efficiency and participation incentive in situations where the platform may be untrusted, we need to design a secure and efficient incentive mechanism that considers device heterogeneity and training costs (i.e., computation, communication, and data collection costs), as well as the personal decision-making of strategic edge servers. Nevertheless, edge servers' multi-dimensional private attributes and training strategies make the joint optimization problem nonconvex, and incomplete information further increases the complexity of the analysis. In order to tackle these challenges, we propose an efficiency-aware incentive mechanism, FedNow, which enables edge servers to personally determine their local training rounds while motivating their participation without access to their true training strategies and private attributes. Our main contributions can be condensed as follows:

- We propose an efficiency-aware incentive mechanism, FedNow, which allows strategic edge servers to personally decide their local training rounds to improve resource utilization while compensating their training costs. Compared to existing works that only focus on improving model performance or incentivizing client engagement, we jointly optimize the training efficiency and expenditure costs, forming a joint optimization objective. Moreover, we take the multi-dimensional heterogeneous attributes, decision-making, and incomplete information of edge servers into account, which makes our work more practical.
- We model the optimization problem of FedNow as a mixed-integer nonlinear programming problem, which is nonconvex due to the involvement of mixed-integer variables and multiple incentive constraints. In order to solve this complex problem, we use contracts to determine the optimal payments for each type of edge server and design a scoring function to prioritize the selection of more efficient edge servers

that are able to provide more data at lower costs. Then, to demonstrate that FedNow strictly outperforms existing schemes in terms of joint optimization objectives, we theoretically derive sufficient conditions for making the above situation true. Based on that, by integrating the Monte Carlo method, FedNow can obtain near-optimal feasible solutions and model parameters within a finite number of training rounds.
- We perform extensive experimental evaluations on one synthetic and three real datasets to demonstrate the advantages of FedNow in terms of resource utilization and training efficiency. The numerical results indicate that FedNow can converge at least 42.3% faster than our baselines when achieving the same model accuracy. Moreover, FedNow improves training efficiency and resource utilization by at least 15.53% and 22.78%, respectively.

The remaining part of this paper is organized as follows. Section 2 provides a review of the related work. The system model and design objectives of FedNow are described in Section 3. Section 4 elaborates on the design and optimization details of our efficiency-aware incentive mechanism, and Section 5 shows our experimental evaluations. Section 6 draws the conclusions.

2. Related Work

FEL combines the characteristics of privacy protection in FL and the high-energy efficiency of edge intelligence, which can reduce dependence on central aggregation and implement collaborative learning in large network systems [21]. Although promising, the leakage of clients' sensitive attributes due to an untrusted platform, the insufficient resource utilization caused by device heterogeneity, and the economic incentives required by practical deployment are important but still under-explored problems in FEL.

Existing solutions have often focused on improving system efficiency by optimizing the allocation scheme of communication and computational resources [15]. For instance, the authors in [22] presented a compromise solution, named stale synchronous parallel (SSP), to create a trade-off between full synchronization and no synchronization, which sets a pre-defined threshold to enable some efficient devices to upload the model updates earlier than the stragglers. The study of [12] suggested that the proportion of time that each device spends on computation and training be increased by introducing the SSP strategy. However, the SSP method can increase the performance gap between efficient devices and straggler devices, leading to the global model being affected by outdated models and causing convergence issues [13]. Some other efforts attempted to leverage the dynamic batch-size method to alleviate the negative impact of a synchronization barrier [11,23,24]. As exemplified by [11], the authors proposed an adaptive algorithm to dynamically adjust the batch size based on the predicted loss and training progress during model training. Although these methods can achieve better system efficiency, they cannot adapt well to the heterogeneous computing capabilities of edge servers. By taking device heterogeneity into account, some methods based on device selection were proposed to mitigate the straggler effect [25–27]. For instance, Xia et al. [25] proposed a device selection scheme to choose more efficient devices without disclosing private device information, thus ensuring a satisfactory convergence rate. Unfortunately, reducing the platform's waiting time for all heterogeneous devices by excluding the engagement of some straggler devices may come at the cost of more epochs for model convergence and greater communication overhead. Moreover, there is still incomplete resource utilization among the selected heterogeneous edge devices. By drawing inspiration from this, our objective is to select edge servers with higher efficiency to improve training efficiency while making the selected edge servers maximize their resource utilization (e.g., training data size or local training rounds) to compensate for the accuracy loss and performance degradation caused by stragglers.

Another important practical challenge faced by FEL comes in the form of economic incentives. Many existing works derived their results based on the assumption that edge servers will participate in FL without seeking any rewards, which may not be practical enough since the communication and computation process often causes huge energy con-

sumption [11,22,25]. There are some important and early works that focus on compensating for clients' training costs to incentivize their participation, but they rarely pay extra attention to resource utilization and also have some limitations. For example, the authors in [28] modeled the client's utility based on one-dimensional attributes, i.e., training data size, to incentivize clients' data contribution. Sarikaya et al. [29] assumed a complete information scenario, where the server has full knowledge of clients' private information. The assumption of complete information may pose a risk of privacy leakage when the parameter server is untrusted, while the corresponding solutions are also not easily generalized when clients possess multi-dimensional attributes and keep them private. Kang et al. [18] considered incomplete information and proposed an incentive mechanism that integrates contract and reputation to select high-quality clients. However, they did not consider clients' personal decision-making and only incentivized clients with reputations above the threshold to maximize the economic utilities of the server, which may result in a significant loss of data and ultimately lead to poor model performance. Considering the joint optimization objectives of expenditure costs and model accuracy loss, the study of [30] saw the design of a multi-dimensional contract to achieve efficient FL. Although taking the personal decision-making of clients into account, they assumed that all heterogeneous clients have identical local update frequency, which leads to useless waiting times and leaves room for further optimization in resource efficiency.

In contrast to the existing works mentioned above, we aim to solve the joint optimization problem of efficiency and incentive from a game-theoretic perspective. Additionally, we consider the edge servers' personal decision-making, multi-dimensional private attributes, and incomplete information, which makes our incentive mechanism more general and practical.

3. System Model

In this section, we first present an overview of FedNow, which includes the framework and training process of FedNow. We then model the utility function of both edge servers and the cloud platform to formulate contracts, and finally, we give the design objectives of FedNow.

3.1. Overview

We plotted the framework diagram of FedNow, as shown in Figure 1, where we consider a typical FEL system with three major components: a data collecting network, a set of edge servers ($\mathcal{I} = \{1, \cdots, I\}$), and one cloud platform. The platform incentivizes edge servers to participate in FEL by compensating for their training and data collection costs. Considering the fact that the costs of edge servers represent private information that is unknown to the platform, the platform will send a set of contract items to edge servers. Each edge server will independently decide whether to participate in training and, if so, which training strategy to adopt, i.e., selecting a contract item and determining the number of times they complete contracts in each global round. Then, each edge server downloads the initial model from the cloud platform and conducts model training using its local data samples. At the beginning of each global round, each edge server first receives a batch of data samples from its associated data-collecting network and then updates its local model using this batch of data before the global model aggregation with other edge servers.

In the training process of FEL, we denote D_i as the data samples received by the edge server $i \in \mathcal{I}$. Generally, data samples received by different edge servers may not follow the same distribution, i.e., $D_i \neq D_x, \forall i \neq x \in \mathcal{I}$. If edge server i utilizes the data of size s_i in its dataset D_i to train its local model, the local loss function of edge server i is the average prediction loss for all data $d \in D_i$, i.e.,

$$F_j(\omega) = \frac{1}{s_i} \sum_{d \in D_i} f_d(\omega), \tag{1}$$

where $f_d(\omega)$ is the loss function for the data sample. FEL aims to find the optimal global model ω^* that minimizes the global loss function $F(\omega)$, which is a weighted average of all clients' loss functions [31]:

$$\omega^* = \underset{\omega}{\operatorname{argmin}}\, F(\omega) = \underset{\omega}{\operatorname{argmin}} \sum_{j=1}^{J} \frac{s_j}{s} F_j(\omega), \tag{2}$$

where s is the total data size of all edge servers.

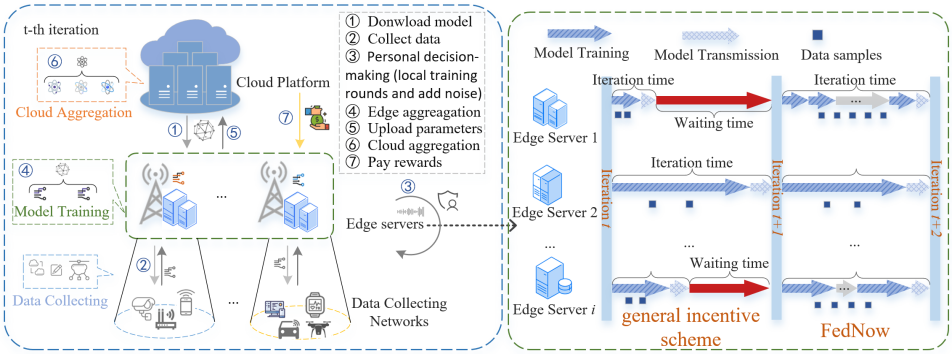


Figure 1. Framework of the efficiency-aware incentive mechanism for FEL.

In this paper, we propose FedNow, which considers personal decision-making in local training rounds and the self-interest of strategic edge servers, aiming to alleviate the performance bottlenecks associated with synchronous incentive schemes in FEL (Compared to asynchronous updates, a synchronous update scheme has provable convergence [14] and stronger anti-attack capability [32]). FedNow allows edge servers to determine their local training rounds in each global iteration independently, thereby reducing useless waiting times. However, conducting more local iterations will also lead to higher training costs. Moreover, heterogeneous edge servers usually have different capabilities in data collecting and processing. In the following, we model the cost of edge servers and the platform, and we formulate contracts to incentivize the truthful contributions of the edge servers. Table 1 summarizes the key notations used in this article.

Table 1. Key notations used in this paper.

Variable	Physical Meanings
$\mathcal{I}$, $\mathcal{J}$, $\mathcal{J}'$	The number set, type set, and incentivized type set of edge servers.
Φ, ϕ_j	The set of contract items; contract items for type-j edge servers.
c_j, r_j	The unit training costs and local training rounds of type-j edge servers.
s_j, p_j	The required data size and corresponding payments for type-j edge servers.
$\bar{\theta}$, t_a	The upper limit of local training rounds; the longest allowable training time.
e_j, $\tilde{e}_j$	The margin efficiency costs of type-j edge servers; the approx marginal efficiency costs.
$P(r'\|r)$	The conditional probability of edge servers with training strategy r choosing r' training strategy.
$\bar{e}_j$	The serial number corresponding to the type-j edge servers in score queue $\tilde{e}_j$.

3.2. Incentive Model

Cost model for FEL: The training costs of edge servers include computation, communication, and data collection. The computation and data-collection cost is proportional to the data-usage size s_i [30], while the communication cost is related to the size of the compressed uploaded model, which we assume to be a fixed model size, S_d [33]. For simplicity, we denote c_i^p, c_i^l. and c_i^m as the unit computation cost, unit data-collection cost, and unit communication cost of edge server i, respectively. The total training costs can be expressed as

$$c_i^t = (c_i^p + c_i^l)s_i + c_i^m S_d. \tag{3}$$

When considering that edge servers may perform several local training rounds but only one communication process in each global epoch, we assumed that training time is the predominant factor in each global epoch and set the edge server's unit training costs as c_i.

Contract formulation: We distinguish edge servers based on their unit training costs, c_i. Specifically, we define an edge server with unit training costs c_i as a type-i edge server and categorize all edge servers in set $\mathcal{I} = \{1, \cdots, I\}$ into $\mathcal{J} = \{1, \cdots, J\}$ types, where each type j contains I_j edge servers, and $\sum_{j\in\mathcal{J}} I_j = |\mathcal{I}|$. We use a smaller index number to indicate smaller unit training costs, such that $c_1 < c_2 < \cdots < c_J$. It should be noted that the total number of participating edge servers, $|\mathcal{I}|$, is public information, and the platform can know the number, I_j, of edge servers for each type based on market research or past training records [34], but for privacy considerations, each edge server will not report its privacy attributes, such as unit training costs, c_j. In light of this, the platform requires the design of a set of contracts, $\Phi \triangleq \{\phi_j\}_{j\in\mathcal{J}}$, to motivate edge server participation. Each contract item, $\phi_j \triangleq (s_j, p_j)$, corresponds to edge server type, j, expressing the relationship between the data size (required for local training) of each type-j edge server and the corresponding rewards.

Edge server utility: We represent the CPU cycle frequency of the type-j edge server as f_j. We assume the number of CPU cycles for edge servers to compute one sample of data in local training is τ. Therefore, the computation time for a type-j edge server to perform one round of local training is $T_j^p = \frac{\tau s_j}{f_j}$. By representing he longest allowable training time of the platform in each global round as t_a, the local iteration rounds of the strategic edge servers are denoted as $r_i = \lfloor \frac{t_a}{T_i^p} \rfloor = \{1, 2, \cdots, \bar{\theta}\}$, where $\bar{\theta}$ is the maximum local training round that the edge servers can execute in each global epoch. Since edge server type is private information, they can choose a contract item designed for other types. Hence, by determining the contract item ϕ_j and the corresponding training strategy as r_j, an edge server's total utilities are the difference between the reward, p_j, and the costs, $c_j s_j$, multiplied by r_j:

$$U_j(c_j, \phi_j, r_j) = r_j(p_j - c_j s_j) \tag{4}$$

The platform's utility: We denote the platform's total costs as our joint optimization objectives, which is equivalent to the sum of the global model's accuracy loss and the total payments for edge servers. According to [35,36], the upper bound of model accuracy loss after T training rounds is $O(1/\sqrt{DT} + 1/T)$, where D is the total amount of data contributed by all edge servers in each global epoch ($D = \sum_{j\in\mathcal{J}} r_j I_j s_j$). In the case where FEL can converge in finite rounds, i.e., a fixed T, minimizing the upper bound of accuracy loss is mathematically equivalent to minimizing $1/\sqrt{\sum_{j\in\mathcal{J}} r_j I_j s_j}$ [30]. Then, given that all edge servers choose their corresponding contract items and determine their local training rounds, the total rewards are $\sum_{j\in\mathcal{J}} r_j I_j p_j$. It is clear that the model accuracy loss decreases with the increase in the edge servers' total training data size, while the total payments increase as the data size increases. Therefore, we expect to achieve the best trade-off between accuracy loss and expenditure cost to minimize the total cost of the platform:

$$U_S = \frac{1}{\sqrt{\sum_{j\in\mathcal{J}} r_j I_j s_j}} + \xi \sum_{j\in\mathcal{J}} r_j I_j p_j, \tag{5}$$

where ξ is a weight coefficient that balances the amount of accuracy loss and expenditure costs. A smaller ξ indicates that the platform places more emphasis on improving model performance and less on reducing expenditure costs.

3.3. Design Objectives

Based on the above, our objective was to design an incentive mechanism that improves resource utilization while minimizing the platform's costs, formulated as follows:

$$(\mathbb{P}_1) \quad \min_{\Phi, r_j} U_S = \frac{1}{\sqrt{\sum_{j\in\mathcal{J}} r_j I_j s_j}} + \xi \sum_{j\in\mathcal{J}} r_j I_j p_j, \quad \text{s.t.} \begin{cases} 1 \le r_j \le \bar{\theta}, \\ 0 < s_j \le s_{\max}, \forall j \in \mathcal{J}. \end{cases} \tag{6}$$

where these two constraints, respectively, limit the range of local training rounds and the required data size.

Moreover, an incentive mechanism that can motivate the truthful data contribution of edge servers should guarantee the following properties:

- Individual rationality (IR). A contract satisfies IR if each type of edge server can receive a non-negative payoff by choosing the contract item, ϕ_j, designed for that type, i.e., $U_j(c_j, \phi_j, r_j) \ge 0, \forall j \in \mathcal{J}$ [30,37].
- Incentive compatibility (IC). A contract satisfies IC if each type of edge server receives its maximum payoffs by choosing the contract item, ϕ_j, designed for that type, i.e., $U_j(c_j, \phi_j, r_j) \ge U_m(c_m, \phi_m, r_m), \forall j \ne m \in \mathcal{J}$ [30,37].
- Computational efficiency (CE). The contract mechanism can be terminated in polynomial time [33].

The IR constraints provide the necessary incentive for making edge servers sign the contract. The IC constraints dictate that edge servers can maximize their utilities when choosing the contract item designed for their true types. When IC conditions are satisfied, each edge server's type is revealed to the platform, which is called "self-reveal". The contract item is feasible if it satisfies both IC and IR constraints [20,37]. However, in this work, as strategic edge servers can adjust their local training rounds, simply imposing IC solely on each edge server's total utility may result in a decrease in the unit-round utility of edge servers that conduct more local training rounds, which is unfair [38–40]. Hence, we relax the incentive compatibility of the edge servers' total utility into incentive compatibility for their unit-round utility [33].

Definition 1 (μ-Incentive compatibility)**.** *A contract satisfies μ-IC if each type of edge server can receive its maximum unit payoff by accepting the contract item, ϕ_j, designed for that type, i.e.,*

$$u_j(c_j, \phi_j) = p_j - c_j s_j \ge u_m(c_m, \phi_m), \forall j \ne m \in \mathcal{J}. \tag{7}$$

If a contract satisfies IR, it also satisfies μ-IR since $r_j > 0$. Therefore, μ-IR can be regarded as an equivalent transformation constraint of IR, expressed as follows:

Definition 2 (μ-Individual rationality)**.** *A contract satisfies μ-IR if each type of edge server can receive a non-negative unit payoff by accepting the contract item, ϕ_j, designed for that type, i.e.,*

$$u_j(c_j, \phi_j) = p_j - c_j s_j \ge 0, \forall j \in \mathcal{J}. \tag{8}$$

4. Optimal Design of Efficiency-Aware Incentive Mechanism

In this section, we present the design details of FedNow and explore its optimal design.

4.1. Optimal Rewards

By integrating the constraints of IR and IC, the cost optimization problem, $(\mathbb{P}_1)$, of the incentive mechanism can be transformed as a non-collaborative cost optimization problem,

($\mathbb{P}_2$), which aims to motivate edge servers to make more data contributions (i.e., more training rounds and greater data size) at the lowest possible costs, i.e.,

$$(\mathbb{P}_2) \quad \min_{\Phi, r_j} U_S = \frac{1}{\sqrt{\sum_{j\in\mathcal{J}} r_j I_j s_j}} + \xi \sum_{j\in\mathcal{J}} r_j I_j p_j, \\ \text{s.t.} \begin{cases} 1 \le r_j \le \bar{\theta}, \\ 0 < s_j \le s_{\max}, \\ u_j(c_j, \phi_j) \ge 0, \\ u_j(c_j, \phi_j) \ge u_m(c_m, \phi_m,), \forall j \ne m \in \mathcal{J}. \end{cases} \tag{9}$$

Solving the above problem involves two challenges. First, due to the enormous number of μ-IR and μ-IC constraints (i.e., $\mathcal{J}^2$ [18]), directly solving the problem is significantly complex. Second, the multi-dimensional decisions of edge servers (i.e., ϕ_j and r_j) will affect their total payoffs and, consequently, influence the platform's optimal incentive strategy $(\phi_j^*, \bar{\theta}^*)$, leading to a nonconvex mixed-integer nonlinear programming problem. In order to overcome this complex problem, we first equivalently transform μ-IR&μ-IC constraints into a simplified set with fewer constraints (i.e, $\mathcal{J}$, Lemma 1) and then derive the optimal rewards for different types of edge servers, thereby simplifying the platform's strategies, $(\phi_j^*, \bar{\theta}^*)$, into two-dimensional variable decisions that only involve $(s_j^*, \bar{\theta}^*)$.

Lemma 1. *In a scenario of information asymmetry between the cloud server and edge servers, a contract is feasible if and only if the following conditions hold:*

(i) $p_J - c_J s_J \ge 0$;
(ii) $p_1 \ge p_2 \ge \cdots \ge p_J \ge 0$ *and* $s_1 \ge \cdots \ge s_J \ge 0$;
(iii) $p_{j+1} + c_j(s_j - s_{j+1}) \le p_j \le p_{j+1} + c_{j+1}(s_j - s_{j+1})$.

Proof. See Appendix A. □

Constraint *(i)* is the equivalent conversion of the μ-IR constraints. Constraints *(ii)* and *(iii)* jointly correspond to the μ-IC constraints. Constraint *(i)* ensures that if the edge server with the largest training cost can obtain non-negative utility by choosing the contract item intended for its type, then each type-j edge server can also receive a non-negative payoff by choosing the contract item of type-J. This is because $p_J - c_j s_J \ge p_J - c_J s_J \ge 0, \forall j \in \{1, \ldots, J\}$. Constraint *(ii)* indicates that the platform should provide more rewards for the edge servers with a lower training cost and require more data in return. Constraint *(iii)* specifies the relationship between any two adjacent contract items. Based on the equivalent relaxation of μ-IR&μ-IC, we can derive the optimal rewards for each type of edge server, shown as follows:

Lemma 2. *For any data size,* $s = \{s_j\}_{j\in\mathcal{J}}$*, and any edge server type,* $j \in \mathcal{J}$*, the optimal choice for the platform is to select the rewards that satisfy the following criteria:*

$$(p_j)^* = \begin{cases} c_j s_j, & \text{if } j = J, \\ c_j s_j + \sum_{i=j+1}^{J} (c_i - c_{i-1}) s_i, \\ \qquad \text{if } j \ne J \text{ and } j = \{1, \cdots, (J-1)\}. \end{cases}$$

Proof. See Appendix B. □

When deriving the optimal rewards for each type of edge server, most of the existing works tend to prioritize incentivizing low-cost edge servers [20,30,41]. However, they often overlook the importance of resource efficiency, i.e., when the low-cost edge servers are providing more data for local model training, some expensive but computationally powerful edge servers may have already completed the local training, resulting in significant waiting times and resource waste for the entire participating group. In contrast, if the

platform tends to choose the edge servers with the strongest computing ability, it may lead to excessive payment costs and, thus, low profits. In order to solve this dilemma, we expect to achieve a better trade-off between economic incentives and training efficiency.

4.2. Efficiency Score Function Design

Before presenting our efficiency score function, we first calculate the total costs of the general incentive mechanism that does not consider multiple rounds of local training, and we take this as the upper-bound cost of our optimization scheme. Hence, we obtain

Theorem 1. *For any solution, from* $(s_j', \mathcal{J}')$ *to* $(\mathbb{P}_2)$*, it is a feasible solution if and only if the platform's costs under such a solution are lower than the upper-bound cost,* $\overline{U_S}$*, i.e.,*

$$\begin{aligned} U_S'(s_j', \mathcal{J}') < \overline{U_S} = & \frac{1}{\sqrt{\sum_{j\in\mathcal{J}} \frac{1}{(2\xi c_j)^{\frac{2}{3}}}}} + \xi|\mathcal{J}| \sum_{j\in\mathcal{J}} \frac{c_j}{(2\xi c_j)^{\frac{2}{3}}} \\ & + \xi \sum_{j=1}^{\mathcal{J}-1} I_j \cdot \sum_{i=j+1}^{\mathcal{J}} (c_i - c_{i-1}) \frac{1}{I_i (2\xi c_i)^{\frac{2}{3}}}. \end{aligned} \tag{10}$$

Proof. See Appendix C. □

After deriving the upper-bound cost of problem $(\mathbb{P}_2)$ (Theorem 1), a general method that can be used is to relax integer constraints and then obtain a feasible solution that satisfies the constraints by rounding, but it cannot guarantee both the existence and the quality of the feasible solutions. Inspired by [25,42], we consider $(\mathbb{P}_2)$ as a high-efficiency participant selection problem, which aims to select an appropriate number of edge servers with the best score to form the incentivized set. We consider the edge servers that can generate higher benefits (i.e., lower model accuracy loss) at lower costs as high-efficiency participants and define marginal efficiency costs (MECs) as our efficiency score function, expressed as follows:

$$\begin{aligned} e_j &= U_S(\mathcal{J}) - U_S(\mathcal{J}/\{j\}) \\ &= \frac{1}{\sqrt{\sum_{j\in\mathcal{J}} r_j I_j s_j}} + \xi \sum_{j\in\mathcal{J}} r_j I_j p_j - \frac{1}{\sqrt{\sum_{j\in\mathcal{J}/\{j\}} r_j I_j s_j}} - \xi \sum_{j\in\mathcal{J}/\{j\}} r_j I_j p_j', \end{aligned} \tag{11}$$

where p_j' denotes the optimal rewards for type set $\mathcal{J}/\{j\}$ (since the number of types changes, the optimal rewards that decrease (by type) will also change accordingly). Given s_j, it is evident that the MECs are influenced by the local training rounds r_j. However, the edge servers' CPU-cycle frequency f is private information and unobservable by the platform. In order to calculate the efficiency score of the edge servers without access to their private attributes, we introduce an exponential mechanism [33], defined as follows:

Definition 3 (Exponential mechanism)**.** *There exists a random mechanism* $\mathcal{M}_e$ *and a score function* $q(r, v)$*, where* $r \to \mathcal{R}$ *is the input and* $v \to \mathcal{V}$ *is the output of the mechanism;* $\mathcal{M}_e$ *satisfies* ϵ*-differential privacy if*

$$\Pr[\mathcal{M}_e(r) = v] \propto \exp\left(\frac{\epsilon q(r, v)}{2\Delta q}\right), \tag{12}$$

where Δq *is the global sensitivity of the score function, which is defined as the maximum distance between any two input scores, i.e.,* $\Delta q = \max|q(r, v) - q(r^*, v)|$.

The definition of the exponential mechanism indicates that each edge server has a probability, $P(r'_j|r_j)$, to determine its local training round as r'_j. Moreover, the closer r'_j and r_j are, the greater the probability $P(r'_j|r_j)$ is. As a result, $q(r_j, r'_j)$ can be defined as

$$q(r_j, r'_j) = -|r_j - r'_j|^{\frac{1}{2}}. \tag{13}$$

Thus, the global sensitivity $\Delta q = \max|q - q'| = (r_{max} - r_{min})^{\frac{1}{2}} = (\overline{\theta} - 1)^{\frac{1}{2}}$. For any local training round $r_j \in \mathcal{R} = \{1, 2, \cdots, \overline{\theta}\}$, the platform can calculate the mapping probability $P(r'_j|r_j)$ as

$$P(r'_j|r_j) = \frac{exp(\epsilon \frac{-|r_j - r'_j|^{\frac{1}{2}}}{2\Delta q^{\frac{1}{2}}})}{\sum_{r^* \in \mathcal{R}} exp(\epsilon \frac{-|r_j - r^*|^{\frac{1}{2}}}{2\Delta q^{\frac{1}{2}}})}. \tag{14}$$

For the mapping probability $P(r'_j|r_j)$, the platform can use the expected rounds, $\mathbb{E}(r_j)$, to approximate the edge servers' true local training rounds r_j, i.e., given $P(r'_j|r_j)$, $r' \in \mathcal{R}$, the expected local training rounds can be calculated by

$$\mathbb{E}(r_j) = \sum_{r' \in \mathcal{R}} r'_j P(r'_j|r_j) = \sum_{r' \in \mathcal{R}} r'_j \frac{exp(\epsilon \frac{-|r_j - r'_j|^{\frac{1}{2}}}{2\Delta q^{\frac{1}{2}}})}{\sum_{r^*_j \in \mathcal{R}} exp(\epsilon \frac{-|r_j - r^*_j|^{\frac{1}{2}}}{2\Delta q^{\frac{1}{2}}})}. \tag{15}$$

Therefore, the approximate marginal efficiency cost (AMEC) can be expressed as

$$\begin{aligned} \tilde{e}_j = & \frac{1}{\sqrt{\sum_{j \in \mathcal{J}} \mathbb{E}(r_j) I_j s_j}} + \xi \sum_{j \in \mathcal{J}} \mathbb{E}(r_j) I_j p_j - \\ & \frac{1}{\sqrt{\sum_{j \in \mathcal{J}/\{j\}} \mathbb{E}(r_j) I_j s_j}} - \xi \sum_{j \in \mathcal{J}/\{j\}} \mathbb{E}(r_j) I_j p'_j. \end{aligned} \tag{16}$$

The platform sorts each type of edge server according to their AMEC in ascending order. A smaller AMEC indicates a higher margin utility and higher quality. The platform will select edge servers according to their AMEC from low to high until finding a threshold type, x', such that $U_S(\mathcal{J}') \leq \overline{U_S}$ and $U_S(\mathcal{J}' \cup \{\bar{e}_{x'+1}\}) > \overline{U_S}$, where $\mathcal{J}'$ represents the set of selected edge servers (Assuming there are only five types of edge servers: {1, 2, 3, 4, and 5} and $x' = 4$. Since the order sorted by the edge servers' AMEC may not follow the same order as their contract types, e.g., AMEC: {2, 1, 4, 3, and 5}, then $\bar{e}_{x'} \triangleq 3$, $\bar{e}_{x'+1} \triangleq 5$, and $(\bar{e}_{x'} + 1) \triangleq 4$). For the selected set $\mathcal{J}'$, we denote $J' = |\mathcal{J}'|$ and re-index the edge servers' type in $\mathcal{J}'$ by $\{j\}_{j \in \{1, \cdots, J'\}}$ in ascending order of training cost, c_j. However, since the platform's cost, U_S, is affected by both r_j and s_j, even if the incentivized set $\mathcal{J}'$ is determined, the platform may be able to obtain a better feasible solution by adjusting its data requirements $s_j, \forall j \in \mathcal{J}'$. Based on the above analysis, we obtain

Proposition 1. *Given a set of contract items $\phi_j = (p_j, s_j)$ and $\overline{\theta}$, where $\forall j \in \mathcal{J}$,*

(i) if $\sum_{j \in \mathcal{J}'} r_j I_j s_j > \sum_{j \in \mathcal{J}} I_j s_j$ and $\sum_{j \in \mathcal{J}'} r_j I_j p'_j < \sum_{j \in \mathcal{J}} I_j p_j$, there exists a threshold type x', such that $U'_S(s_j, \mathcal{J}') < \overline{U_S}$, where

$$p'_j = \begin{cases} c_j s_j, & \text{if } j = J', \\ c_j s_j + \sum_{i=j+1}^{J'} (c_i - c_{i-1}) s_i, & \\ \qquad \text{if } j \neq J' \text{ and } j = \{1, \cdots, (J'-1)\}. \end{cases}$$

(ii) if the threshold type x' exists, there exists an optimal type x^* and required data size s_j^* such that $U_S^*(s_j^*, \mathcal{J}^*) \leq U_S'(s_j, \mathcal{J}')$, where $\mathcal{J}^*$ is the platform's optimal incentivized type set.

Proof. See Appendix D. □

Proposition 1*(i)* shows that, given s_j, the optimal choice for the platform is to incentivize the edge servers that can provide more total data at lower payment costs. In addition, the platform only provides positive contract items for the incentivized type of edge servers, whereas the non-incentivized edge servers will receive zero contract items. Proposition 1*(ii)* suggests that if a feasible solution exists, the platform can adjust its incentive strategy (i.e., required data size s_j), which may lead to a better solution that minimizes the platform's costs. According to the above analysis, we first designed an iterative algorithm to determine all incentivized edge servers and then characterized the training process of FedNow, which enables the edge servers to personally determine their local training rounds; we used the Monte Carlo method to obtain a better (near-optimal) response for the platform, including the incentivized set $\mathcal{J}^*$ and corresponding $\{s_j^*, p_j^*\}_{j\in\mathcal{J}^*}$. With $\mathcal{J}^*$ and $\{s_j^*, p_j^*\}_{j\in\mathcal{J}^*}$, the incentivized edge servers will honestly complete model training to compute the optimal global model parameters.

The details of searching for feasible solutions are shown in Algorithm 1. The platform first calculates all edge servers' AMEC (Line 2–4), then sorts the edge servers based on their AMEC in ascending order and searches for the threshold type x' (Line 5–18). The time complexity of calculating $\tilde{e}_j$ (Line 4) is $O(\mathcal{J})$, while the time complexity for checking whether the feasibility condition holds (Line 11) is also $O(\mathcal{J})$. Therefore, the time complexity of Algorithm 1 is $O(\mathcal{J}^2)$.

Algorithm 1 Calculation of the feasible solution $(x', \mathcal{J}')$.

Input: $\overline{\theta}$, c_j, I_j, $P(r_j'|r_j), \forall j \in \mathcal{J}; r_j, r_j' \in \mathcal{R}$.
Output: x', $\mathcal{J}'$.
1 Compute contract items $\Phi = (s_j, p_j), \forall j \in \mathcal{J}$;
2 **for** *j in range* $[1, J]$ **do**
3 Calculate $\mathbb{E}(r_j)$ of type-j edge servers based on Formula (15);
4 Calculate $\tilde{e}_j$ of type-j edge servers based on Formula (16);
5 Sort edge servers according to $\tilde{e}_j$ in ascending order;
6 **for** *j in range* $[1_{\tilde{e}}, J_{\tilde{e}}]$ **do**
7 Initialize $x' = j$;
8 Insert x' into $\mathcal{J}'$;
9 Check if Formula (10) holds;
10 **if** *yes* **then**
11 Check if $U_S(\mathcal{J}' \cup \{j+1\}_{\tilde{e}}) > \overline{U_S}$;
12 **if** *yes* **then**
13 Record x';
14 break;
15 **else**
16 continue;
17 **else**
18 $j++$;

Algorithm 2 describes the training process of FedNow when computing the optimal model parameters. The time complexity of the Monte Carlo method is $O(N)$, where N means the number of sampling times. For a strongly convex objective $F(\omega)$, the general upper bound of global iterations is expressed as $T_{\text{ite}}(\kappa, \zeta) = \frac{O(\log(1/\sigma))}{1-\zeta}$ [43], which is affected by both global accuracy κ and local accuracy ζ. Given a fixed global accuracy σ, the upper bound of the local iterations, i.e., $O(\log(1/\sigma))$, can be normalized to 1 so that

$T_{\text{ite}}(\zeta) = \frac{1}{1-\zeta}$ [16]. We represent the time used for one global iteration of FEL as $T_{\text{glob}}(\xi)$; the upper bound of the time complexity for Algorithm 2 is $O\left(max\{N, \frac{1}{1-\xi} \cdot T_{glob}(\zeta)\}\right)$.

Algorithm 2 The training process of FedNow

Input: t_w, $\mathcal{I}$, $\mathcal{J}$, T, $\boldsymbol{\eta}$, x', $\bar{\theta}$ and N.
Output: ω^T.

Use the Monte Carlo Method to determine the optimal incentivized set $\mathcal{J}^*$ and contract items $(p_j^*, s_j^*), \forall j \in \mathcal{J}^*$ based on (N, x');
Send contracts and initialize ω^0;
for *epoch* $t = 0; t < T; t++$ **do**
  Each edge server in $\mathcal{J}^*$ **executes:**
    Download global model ω^t from the platform;
    Collect data D_i from its associated edge devices;
    Train the model with data size $s_j^*(\leq D_i)$ and local training rounds $r_j^*(\leq \bar{\theta})$:

$$\omega_j^{t+1} \leftarrow \omega_j^t - \eta \cdot \nabla F_j\left(\omega_j^t\right),$$

    Upload model parameters ω_j^{t+1} within t_a;
  Cloud server executes:
    aggregate model parameters:

$$\omega^{t+1} \leftarrow \sum_{j \in \mathcal{J}^*} \frac{s_j}{s} \omega_j^{t+1}$$

    Pay rewards for the incentivized edge servers.

5. Experiments

In this section, we first exhibit the experimental setup and then show the experiments conducted on both the synthetic and real datasets to evaluate the performance of our proposed FedNow.

5.1. Experimental Setup

Edge server parameters. By referring to [7,37], we considered a cloud platform, a set of $|\mathcal{I}| = 100$ edge servers for FEL. We leveraged two GPU devices (i.e., Lenovo Notebooks with NVIDIA RTX 3060 and NVIDIA GTX 1050Ti, Beijing, China) with multi-thread programming to simulate the training process of heterogeneous edge servers. For simplicity, the platform divides edge servers into several groups based on their training costs, c, and CPU frequency, f, and we approximate each group as a supertype [30]. By limiting computing resources to a range from 0.8 GHz to 2.7 GHz, we divided the edge servers into 10 groups, where the edge servers with (c_j, f_j) uniformly distributed over $([1.5, 2.4] \times [0.8, 0.9]) \cup ([2.5, 3.4] \times [1.0, 1.1]) \cup ([3.5, 4.4] \times [1.2, 1.3]) \cup ([4.5, 5.4] \times [1.4, 1.5]) \cup ([5.5, 6.4] \times [1.6, 1.7]) \cup ([6.5, 7.4] \times [1.8, 1.9]) \cup ([7.5, 8.4] \times [2.0, 2.1]) \cup ([8.5, 9.4] \times [2.2, 2.3]) \cup ([9.5, 10.4] \times [2.4, 2.5]) \cup ([10.5, 11.4] \times [2.6, 2.7])$. We took the midpoint of each group to represent the cost and CPU frequency of their supertype, e.g., approximately $(c_j, f_j) = (2.3, 0.8) \approx (2.0, 0.85)$ as the type-1 edge server (According to [30], approximating edge server type with reagrd to several supertypes will not significantly affect the platform's payment and utility.).

Training parameters. We constructed an FL model using MLP, CNN, and ResNet-18, respectively, and trained them by exploiting benchmark datasets: MNIST, Fashion MNIST, and CIFAR-10. The MLP network is formed by two convolutional layers with a max-pooling of 2, fully connected layers, a Relu activation function, and a softmax output layer, whereas the CNN model has two convolutional layers with a kernel size of 5, two dense layers, and a softmax output layer. The ResNet-18 model is made up of a feedfoward layer, a fully connected layer (size = 1000), and a softmax output layer, where the feedforward

layer includes four convolutional layers and eight residual blocks. We set the economic conversion parameter to $\xi = 9 \times 10^{-8}$ and the learning rate to 0.01.

Baselines. In order to demonstrate the performance of FedNow, we adopted two state-of-the-art solutions as the baselines, described as follows:

- *UB*: A vanilla contract incentive mechanism that assumes each edge server only conducts one round of local training (Theorem 1). We consider that it represents the **u**pper-**b**ound cost of any feasible strategy.
- *CD*: A **c**ost-**d**riven incentive mechanism that considers the multi-local training rounds of edge servers but prefers to motivate low-cost edge servers [20,30]. We compared it with FedNow to demonstrate the effectiveness of the efficiency score function.

Metrics. For the synthetic dataset, we used model accuracy loss: $acc.l = \frac{1}{\sqrt{\sum_{j\in\mathcal{J}} r_j I_j s_j}}$, payment costs: $P = \sum_{j\in\mathcal{J}} r_j I_j p_j$, and the platform's total costs: $U_S = acc.l + \xi P$ to assess the performance of FedNow in terms of economic incentives. In the real datasets, we used (i) model accuracy and accuracy loss, (ii) average waiting time, (iii) training efficiency, and (iv) completion time, respectively, to evaluate the performance of FedNow. Concretely, we recorded the changes in model accuracy over time and accuracy loss with epoch under different strategies. Moreover, for the different strategies, we recorded the average waiting time of all edge servers in each epoch. Based on this, we could calculate the overall average waiting time $\overline{\mathcal{W}}$ and the overall average epoch time $\overline{\mathcal{V}}$ for different strategies, thus deriving their training efficiency (defined as $1 - \overline{\mathcal{W}}/\overline{\mathcal{V}}$ [4]). $\overline{\mathcal{W}}$ and $\overline{\mathcal{V}}$ are represented as follows:

$$\begin{cases} \overline{\mathcal{W}} = \frac{1}{T}\sum_{t=1}^{T} \mathcal{W}^t, \\ \overline{\mathcal{V}} = \frac{1}{T}\sum_{t=1}^{T} \rho^t. \end{cases}$$

where $\mathcal{W}^t$ means the overall average waiting time of the edge servers in epoch t, and ρ^t denotes the time spent on completing the t-th epoch, which can be measured by recording the training time of the slowest edge server at the t-th epoch ($\rho^t \leq t_a, \forall t \in \{1, \cdots, T\}$). Moreover, we also recorded the completion time of Fednow and the two baselines when achieving the same training accuracy.

5.2. Evaluation of the Synthetic Dataset

For the first set of experiments, we used to verify the superiority and rationality of our mechanism in terms of economic utility. Concretely, we first demonstrated how FedNow selects more efficient edge servers, and we verified the rationality of the incentive design. Then, we performed experiments on the synthetic dataset to evaluate the performance of the different strategies.

Figure 2 shows the MECs and AMECs of different types of edge servers. Apparently, the MECs and AMECs of the type-1, -3, and -10 edge servers were both greater than 0, which means that the cost contribution of the type-1, -3, and -10 edge servers is positive, considering that the smaller the MEC/AMEC (i.e., smaller cost increment), the higher the value/efficiency of the corresponding edge servers. In contrast, the MEC and AMEC of the type-2, -4–9 edge servers are both less than 0, which means that they have negative effects on the platform's costs. In other words, when MEC/AMEC is less than zero, the smaller the MEC/AMEC, the lower the priority of selecting the corresponding type of edge server. In addition, since the training cost of type-10 edge servers is the largest, then, when calculating the efficiency score of the other types of edge servers, the type-10 edge servers are always in the incentivized set $\mathcal{J}'$, and the platform's optimal payment for them will just cover their training costs, making the type-10 edge servers always a critical payment type. When calculating the efficiency score of the type-10 edge servers, the critical payment type changes from 10 to 9, which may lead to a significant reduction in server payment costs. The above two factors explain the intense changes in efficiency scores between the type-9 and type-10 edge servers. Therefore, when the platform has complete information about the local training rounds of the edge servers, the preference order is $\{1, 3, 10, 5, 2, 4, 6, 7, 8, 9\}$, whereas the preference order under incomplete information is $\{3, 1, 10, 2, 5, 4, 6, 7, 8, 9\}$.

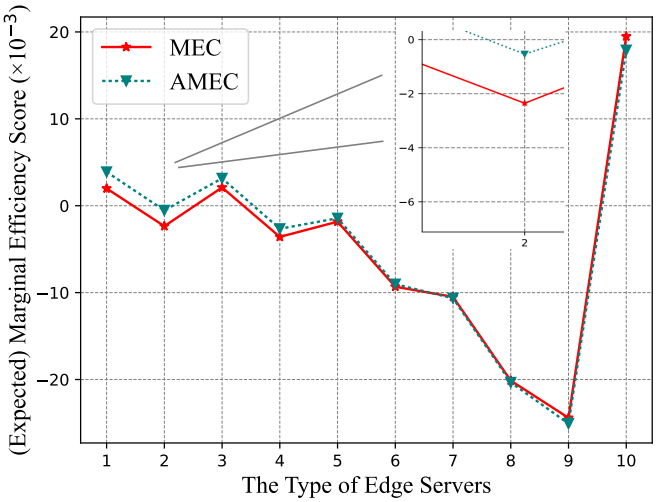


Figure 2. Marginal efficiency score.

Table 2 characterizes the performance differences of different strategies under their respective optimal incentive solutions. According to the control group, it is clear that the lack of data size leads to a significant accuracy loss, thus making the total costs higher than UB, even if the payment costs are the lowest. The cost-driven incentive strategy (CD) could incentivize more types of edge servers to participate, but this also leads to higher payment costs, whereas when reducing the incentivized types, the decrease in data size causes an increase in accuracy loss and may make the total cost U_S higher than UB. Moreover, in Table 2, we denote MEC as the efficiency-aware incentive mechanism with complete information about edge servers' local training rounds. Obviously, with the complete information about r_j, MEC achieves the best performance on both accuracy loss and payment costs, thereby minimizing the total costs of the platform. Due to the influence of incomplete information, although the preference order and the optimal incentivized set of FedNow are different from MEC, FedNow can achieve similar performance to MEC and outperform the cost-driven strategy.

Figure 3 shows that the platform provides positive contract items for the incentivized type of edge servers, and the critical type of edge servers (i.e., type-10 edge servers) only obtain 0 utility since the optimal rewards provided to type-10 edge servers by the platform just cover their training costs.

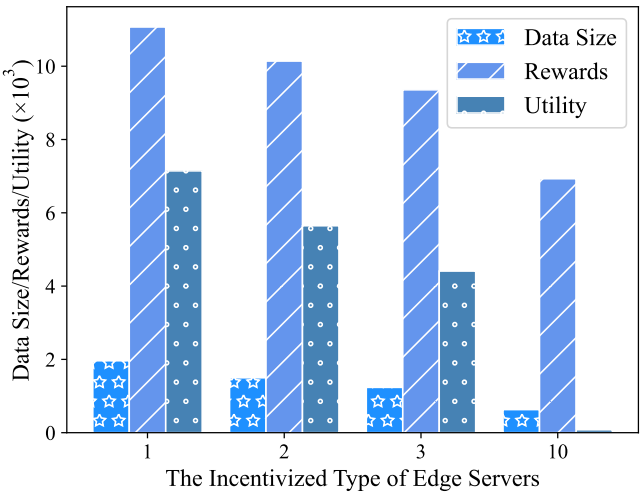


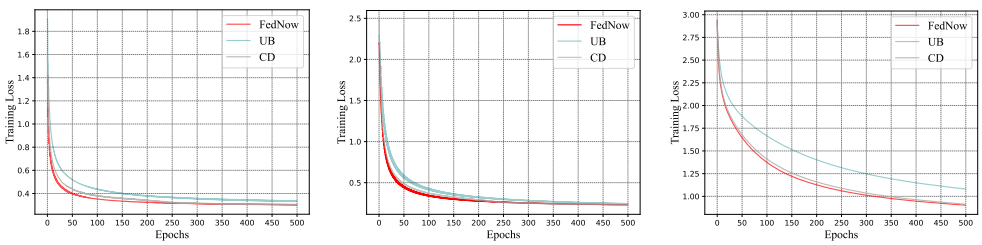
Figure 3. The platform's optimal contract for different types of edge servers.

Table 2. Differences in performance for different metric; *LR* means the local training rounds, and *D* denotes the overall data size contributed by the incentivized set $\mathcal{J}'$.

Ref.	$\mathcal{J}'$	*LR*	*D* ($\times 10^3$)	*acc.l* ($\times 10^{-2}$)	*P* ($\times 10^4$)	U_S ($\times 10^{-1}$)
UB	$\{1, \cdots, 10\}$	1	10.3583	0.3107	9.2625	0.1152
Control Group	{1, 2, 3}	$r_j, j \in \mathcal{J}'$	2.4602	0.6375	6.5252	0.1231
CD	{1, 2, 3, 4, 5}	$r_j, j \in \mathcal{J}'$	7.5857	0.363	7.5447	0.1048
MEC	{1, 3, 5, 10}	$r_j, j \in \mathcal{J}'$	10.4166	0.3098	7.0024	0.0946
FedNow	{1, 2, 3, 10}	$r_j, j \in \mathcal{J}'$	9.0836	0.3317	7.4569	0.1009

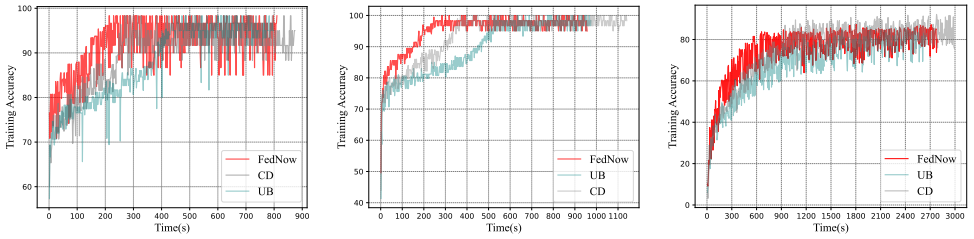
5.3. Evaluation Using the Real Datasets

The second set of experiments evaluated the training performance of our mechanism according to accuracy and loss when using the real datasets. We first used training loss to evaluate the performance of the three strategies. Figure 4 shows the training loss of three strategies trained with different datasets over the epochs. As training progresses, the training loss of FedNow and the two baselines both decrease with epoch, but the loss curve of FedNow clearly has the fastest decline and a faster convergence rate than the baselines. For example, in Figure 4a, when the epoch approaches 50, FedNow begins to converge, and its loss is below 0.4, while the training loss for the baselines is obviously higher than 0.4; it takes CD and UB an additional 50 and 100 epochs, respectively, to achieve a similar loss. Second, considering the fact that different strategies may have different training times in each epoch, we recorded the variation in training accuracy over time, as shown in Figure 5. Due to the ease of training with MNIST and FMNIST, the three strategies ultimately achieve similar training accuracies, as seen in Figure 5a,b; in contrast, CD achieves the highest training accuracy in Figure 5c, which may due to the fact that tending toward low-cost edge servers can motivate relatively more training samples with the same budgets. However, it is worth noting that since FedNow prioritizes selecting more efficient edge servers and allows them to personally decide their local training rounds (i.e., multiple local training rounds), the model trained with FedNow can converge over a time that is obviously shorter. For example, in FMNIST and CIFAR-10, the model trained with FedNow converges nearly 100 and 300 s earlier than CD, respectively. Moreover, CD allows multiple local training rounds but tends to choose low-cost edge servers, which leads to more training samples but also longer training times. UB requires each edge server to perform only one round of local training, which makes the overall training time under this strategy relatively short. However, this also leads to insufficient training samples/local training rounds for UB, resulting in the slowest convergence speed and the worst model accuracy. In Figure 6, in order to demonstrate the effectiveness of the efficiency-aware mechanism, we further intuitively represent the completion time when the model trained using the three strategies reaches the same accuracy. Excitingly, the completion time of FedNow is the lowest among the three training strategies. For instance, in Figure 6c, it takes 393 s, 681 s, and 801 s for a model to obtain the training accuracy of 0.75 when trained with FedNow, CD, and UB, respectively. In other words, for CIFAR10-ResNet18, FedNow saves more training time than CD and UB by as much as 42.3% and 51% when reaching an accuracy of 0.75.



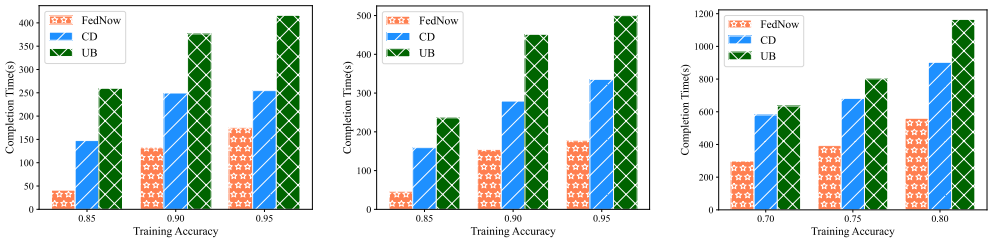
(**a**) (**b**) (**c**)

Figure 4. Training loss for the three strategies with epoch. (**a**) MNIST-MLP; (**b**) FMNIST-CNN; (**c**) CIFAR10-Resnet18.



(**a**) (**b**) (**c**)

Figure 5. Training accuracy of the three strategies with time. (**a**) MNIST-MLP; (**b**) FMNIST-CNN; (**c**) CIFAR10-Resnet18.



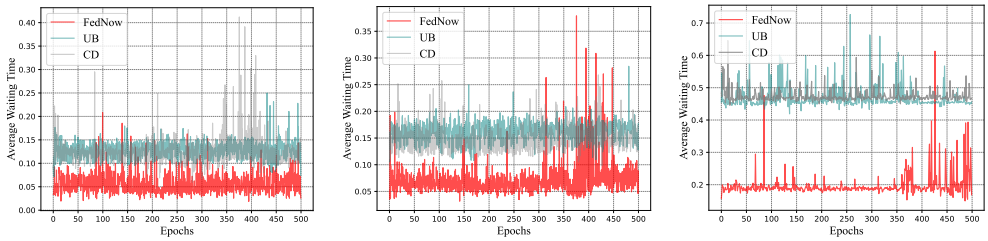
(**a**) (**b**) (**c**)

Figure 6. Completion time when reaching the same accuracy. (**a**) MNIST-MLP; (**b**) FMNIST-CNN; (**c**) CIFAR10-Resnet.

Based on the above results, we found that enabling personal decision-making during local training rounds could ensure satisfactory convergence speed and shorten completion time when reaching the same training accuracy. Therefore, even in a situation where resources are limited (e.g., the limited payment budgets), FedNow can improve model convergence speed and achieve a better final model accuracy than those solutions that only consider payment costs or ignore edge server personal decision-making.

The third set of experiments was performed to evaluate the training efficiency of the different strategies. Figure 7 characterizes the average waiting time for the different strategies with epochs. Figure 7 indicates that the average waiting time for the three strategies remains relatively stable, and Fednow's performance in resource utilization is still the best. For example, for MNIST-MLP in Figure 7a and FMNIST in Figure 7b, the average waiting time for the two baselines is around 0.125 s and 0.15 s, respectively, whereas that of FedNow is 0.05 s and 0.75 s, which indicates that FedNow reduces the average waiting time by at least 50% when compared to the baseline method. Moreover, the average waiting time of CD is only slightly lower than that of UB, which indicates that a preference for low-cost edge servers cannot effectively solve the problem of low training efficiency. Last but not least, we present the training efficiency and testing accuracy of the three strategies in Table 3.

Based on the above results, we found that FedNow could improve resource utilization and reduce useless waiting times. This is because the performance gaps between heterogeneous devices are narrowed by allowing for different local training rounds and data requests. Hence, comprehensively considering an edge server's personal decision-making and individual efficiency can alleviate the negative effect of a synchronization barrier for those edge devices with heterogeneous computation capacities.



(**a**) (**b**) (**c**)

Figure 7. The average waiting time of three strategies over epochs. (**a**) MNIST-MLP; (**b**) FMNIST-CNN; (**c**) CIFAR10-Resnet.

Table 3. Training efficiency and test accuracy of three strategies

	MNIST-MLP		FMNIST-CNN		CIFAR10-RESNET	
	Efficiency	**Test_acc**	**Efficiency**	**Test_acc**	**Efficiency**	**Test_acc**
UB	0.4109	0.9546	0.4132	0.9800	0.4787	0.7874
CD	0.6672	0.9667	0.6780	0.9822	0.7010	0.8435
FedNow	0.8247	0.9833	0.8027	0.9838	0.8357	0.8332

6. Conclusions

In this work, we propose a novel, efficiency-aware incentive mechanism, FedNow, that can motivate data contribution and efficient resource utilization without accessing edge servers' private attributes. On the basis of contract incentive boundaries, we have carefully designed an efficiency score function to prioritize more efficient edge servers. Subsequently, we have derived sufficient conditions for making FedNow outperform existing schemes in terms of joint optimization objectives. Extensive simulations on both synthetic and real datasets show that our efficiency-aware incentive mechanism can improve training efficiency and resource utilization by at least 15.53% and 22.78%, respectively.

In future research, we consider that this work can be extended/deepened in multiple directions and mention two ideas as examples. First, when compared to the solution that requires edge servers to maintain the same local update frequency, allowing multiple rounds of local training can narrow the performance gap between heterogeneous edge servers, which, to some extent, enhances the fairness of the FEL system. As far as we know, although there have been some efforts made to focus on controlling client participation frequency to achieve better fairness [44], few of these introduce the axiomatized fairness concept or rigorously derive optimal solutions based on maximizing social fairness. Therefore, improving training efficiency while strictly ensuring social fairness remains an open topic that, although challenging, is of great significance for promoting the sustainable co-operation of clients in FEL. Second, in this work, to resist privacy breaches that may be caused by an untrusted platform, we introduced an exponential mechanism and designed FedNow to motivate clients' contributions without accessing their private attributes and true training strategies. In future work, we expect to generalize our work to situations where both the platform and clients are untrusted while still incentivizing efficient co-operation between honest clients and the platform. The explanations of mathematical terms used in all formulas are shown in Table 4.

Table 4. Expand Table 1: An explanation of the mathematical terms used in the formulas.

Mathematical Terms	Physical Meanings
$f_d(\omega)$, $F_j(\omega)$, ω, ω^*	The prediction loss on a pair of data samples d, the loss function of edge server j, the model parameter, and the optimal model parameters.
c_i^t, c_i^p, c_i^l, c_i^m	The total costs of edge server i in t-th global epoch; the unit computation cost, unit data-collection cost, and unit communication cost of edge server i.
S_d, I_j	The global model size (fixed), and the number of edge servers contained in the type set j.

Table 4. *Cont.*

Mathematical Terms	Physical Meanings
T_j^p, τ, f_j	The computation time for a type-j edge server to perform one local training round, the number of CPU cycles required for performing one sample of data, and the CPU-cycle frequency of a type-j edge server.
$U_j(c_j, \phi_j, r_j)$, $u_j(c_j, \phi_j)$	The total utilities of type-j edge servers; the unit-round utility of type-j edge servers, equivalent to the utility of completing contract per time.
ξ, s_j^*, p_j^*	The platform's weight regarding the rewards and the optimal data size/rewards for type-j edge servers.
U_S, $\overline{U_S}$	The utility function of the platform, equivalent to the platform's total costs; the cost upper-bound of the platform.
Δq	The global sensitivity of the score function.
r_j, r_j', $\mathcal{R}$	The real training rounds of type-j edge servers, the expected training rounds of type-j edge servers, the set of possible training rounds.
$U_S(\mathcal{J})$, $U_S(\mathcal{J}/\{j\})$	The total costs of the platform when the incentivizing set is $\mathcal{J}$ and $\mathcal{J}/\{j\}$, respectively.
$T_{\text{ite}}(\kappa, \zeta)$	The general upper bound of global iterations; κ: global accuracy, ζ: local accuracy.
$\mathcal{W}^t$, $\overline{\mathcal{W}}$	Waiting time in the t-th epoch and the overall average waiting time.
$\mathcal{V}^t$, $\overline{\mathcal{V}}$, ρ^t	Epoch time in the t-th epoch, the overall average epoch time, and the training time of the slowest edge server in the t-th epoch.

Author Contributions: Conceptualization, J.L. and S.C.; methodology, J.L. and W.Y.; software, J.L.; validation, W.Y.; formal analysis, J.L. and W.Y.; investigation, J.L.; resources, J.L.; data curation, S.C.; writing—original draft preparation, W.Y.; writing—review and editing, J.L., W.Y., S.C. and P.Z.; visualization, J.L. and W.Y.; supervision, J.L., S.C. and P.Z.; project administration, J.L.; funding acquisition, J.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (No. 62372343, 62072411), the Zhejiang Provincial Natural Science Foundation of China (No. LR21F020001), the Key Research and Development Program of Hubei Province (No. 2023BEB024), and in part by the fund from Hubei Province Key Laboratory of Intelligent Information Processing and Real-time Industrial System (Wuhan University of Science and Technology).

Data Availability Statement: Publicly available datasets were used in this study. These datasets can be found here: 1. http://yann.lecun.com/exdb/mnist/ (accessed on 10 December 2023); 2. http://www.cs.toronto.edu/~kriz/cifar.html (accessed on 10 December 2023).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

In Lemma 1, we propose that contracts are feasible if and only if the following three conditions hold:

(i) $p_J - c_J s_J \geq 0$,
(ii) $p_1 \geq p_2 \geq \cdots \geq p_J \geq 0$ *and* $s_1 \geq s_2 \geq \cdots \geq s_J \geq 0$,
(iii) $p_{j+1} + c_j(s_j - s_{j+1}) \leq p_j \leq p_{j+1} + c_{j+1}(s_j - s_{j+1})$,

where $j \in \{1, 2, \cdots, J\}$.

The contract items are feasible only when both IR and IC constraints are satisfied. Therefore, we prove the equivalence between the above three conditions and the IR&IC constraints from necessity and sufficiency perspectives.

(i) Necessity:

(a) Condition (i) $p_J - c_J s_J \geq 0$: according to the IR constraint, i.e., $p_j - c_j s_j \geq 0, \forall j \in \mathcal{J}$, we have

$$p_J - c_J s_J \geq 0, \text{ where } J = |\mathcal{J}|, \tag{A1}$$

So, Condition (i) holds.

(b) Condition (ii) $p_1 \geq p_2 \geq \cdots \geq p_J \geq 0$ and $s_1 \geq s_2 \geq \cdots \geq s_J \geq 0$: according to IC constraints, the following condition for the type-j edge servers must hold:

$$p_j - c_j s_j \geq p_m - c_j s_m \Leftrightarrow c_j(s_m - s_j) \geq p_m - p_j, \tag{A2}$$

where $j, m \in \{1, \cdots, J\}$ and $j \neq m$. Formula (A2) shows that if $s_m \leq s_j$, then $0 \geq c_j(s_m - s_j) \geq p_m - p_j$, $p_m \leq p_j$.

Similarly, according to IC constraints, the following condition for the type-m edge servers must hold:

$$p_m - c_m s_m \geq p_j - c_m s_j \Leftrightarrow c_m(s_j - s_m) \geq p_j - p_m. \tag{A3}$$

Formula (A3) shows that if $s_m \geq s_j$, then $0 \geq c_m(s_j - s_m) \geq p_j - p_m$, $p_m \geq p_j$.

Based on the above analysis, we can know that $p_j \geq p_m$ if and only if $s_j \geq s_m$. By combining Formulas (A2) and (A3), we have

$$\begin{aligned} & c_m(s_j - s_m) \geq p_j - p_m \geq c_j(s_j - s_m) \\ \rightarrow & (c_j - c_m)(s_j - s_m) \leq 0. \end{aligned} \tag{A4}$$

According to Formula (A4), for a feasible contract, $s_j \geq s_m$ and $p_j \geq p_m$ if and only if $c_j \leq c_m$, $\forall j \neq m \in \{1, \cdots, J\}$. In other words, both p_j and s_j are negatively correlated with c_j. So, Condition (ii) holds.

(c) Condition (iii) $p_{j+1} + c_j(s_j - s_{j+1}) \leq p_j \leq p_{j+1} + c_{j+1}(s_j - s_{j+1})$: according to IC constraint, $p_j - c_j s_j \geq p_m - c_j s_m, \forall j, m \in \{1, \cdots, J\}$, for any two neighbor contract items, we have

$$\begin{cases} p_{j+1} - c_j s_{j+1} \leq p_j - c_j s_j, \\ p_j - c_{j+1} s_j \leq p_{j+1} - c_{j+1} s_{j+1}, \end{cases} \tag{A5}$$

which can be equivalent, as

$$p_{j+1} + c_j(s_j - s_{j+1}) \leq p_j \leq p_{j+1} + c_{j+1}(s_j - s_{j+1}). \tag{A6}$$

So, Condition (iii) holds.

(ii) Sufficiency:

We denote the subset of contract items that contain the last r data-reward contract items as $C(r)$, i.e., $C(r) = \{(s_j, p_j) \mid j = J - r + 1\}$, and use induction to complete the proof:

We first prove that $C(1)$ is feasible: according to condition (i), i.e., $p_J - c_J s_J \geq 0$, the IR constraint is satisfied. Since there is only one contract item, the IC constraint does not need to be considered. Therefore, $C(1)$ is feasible.

Then, we show that if $C(r)$ is feasible, then $C(r+1)$ is also feasible. This corresponds to two aspects:

(a) For the new type $(J - r)$, the IR and IC constraints are satisfied, i.e.,

$$\begin{cases} p_{(J-r)} - c_{(J-r)} s_{(J-r)} \geq 0, \\ p_{(J-r)} - c_{(J-r)} s_{(J-r)} \geq p_r - c_r s_r, \\ \qquad \forall r = J - r + 1, \cdots, J. \end{cases} \tag{A7}$$

(b) For the existing type $(J - r + 1, \cdots, J)$, the IR and IC constraints are still satisfied when the type $J - r$ exists, i.e.,

$$\begin{cases} p_r - c_r s_r \geq 0, \\ p_r - c_r s_r \geq p_{r'} - c_{r'} s_{r'}, \\ \quad \forall r, r' \in (J - r, J - r + 1, \cdots, J) \text{ and } r \neq r'. \end{cases} \tag{A8}$$

We first prove Formula (A7) as follows: since $C(r)$ is feasible, the IC constraint is satisfied for the type $(J - r + 1)$ client, i.e.,

$$p_{(J-r+1)} - c_{(J-r+1)} s_{(J-r+1)} \geq p_r - c_{(J-r+1)} s_r, \forall r = J - r + 1, \cdots, J. \tag{A9}$$

Moreover, according to Condition (iii), we have

$$p_{(J-r)} \geq p_{(J-r+1)} + c_{(J-r)}\left(s_{(J-r)} - s_{(J-r+1)}\right). \tag{A10}$$

By summing up Formulas (A9) and (A10), we have

$$p_{(J'-r+1)} + p_{(J-r)} - c_{(J-r+1)}s_{(J-r+1)} \geq p_r - c_{(J-r+1)}s_r + p_{(J-r+1)} + c_{(J-r)}\left(s_{(J-r)} - s_{(J-r+1)}\right) \tag{A11}$$

which can be rewritten as

$$\begin{aligned} p_{(J-r)} - c_{(J-r)}s_{(J-r)} &\geq p_r - c_{(J-r)}s_{(J-r+1)} + c_{(J-r+1)}\left(s_{(J-r+1)} - s_r\right) \\ &\geq p_r - c_{(J-r)}s_{(J-r+1)} \\ &\geq p_r - c_{(J-r)}s_r, \forall r \in \{J-r+1, \cdots, J\}. \end{aligned} \tag{A12}$$

where Formula (A12) is based on condition (ii), i.e., $s_{(J-r+1)} \geq s_r, \forall r = J-r+1, \cdots, J$. Therefore, the IC constraints are satisfied for type $(J-r)$ edge servers. Additionally, we can prove that $p_r - c_r s_r \geq p_r - c_{(J-r)}s_r, \forall r \in \{J-r+1, \cdots, J\}$. To sum up, the IR constraint for type $(J-r)$ edge servers is satisfied.

We then prove Formula (A8): obviously, proving Formula (A8) is equivalent to demonstrating the following:

$$p_r - c_r s_r \geq p_{(J-r)} - c_r s_{(J-r)}, \forall r \in \{J-r+1, \cdots, J\}. \tag{A13}$$

According to condition (iii), we have

$$p_{(J-r)} \leq p_{(J-r+1)} + c_{(J-r+1)}\left(s_{(J-r)} - s_{(J-r+1)}\right). \tag{A14}$$

Moreover, since the IC constraints are satisfied for type-r edge servers ($\forall r \in \{J-r+1, \cdots, J\}$), we have

$$\begin{aligned} p_{(J-r)} + p_{(J-r+1)} - c_r s_{(J-r+1)} \leq\ & p_{(J-r+1)} \\ & + c_{(J-r+1)}\left(s_{(J-r)} - s_{(J-r+1)}\right) + p_r - c_r s_r, \end{aligned} \tag{A15}$$

which can be rewritten as

$$\begin{aligned} p_r - c_r s_r &\geq p_{(J-r)} - c_r s_{(J-r+1)} - c_{(J-r+1)}\left(s_{(J-r)} - s_{(J-r+1)}\right) \\ &\geq p_{(J-r)} - c_r s_{(J-r+1)} - c_r\left(s_{(J-r)} - s_{(J-r+1)}\right) \\ &= p_{(J-r)} - c_r s_{(J-r)}. \end{aligned} \tag{A16}$$

which is equivalent to Formula (A13), so Formula (A8) is proven.

Appendix B

In the following, we prove that the optimal payments to edge servers satisfies

$$(p_j)^* = \begin{cases} c_j s_j, & \text{if } j = J, \\ c_j s_j + \sum_{i=j+1}^{J}(c_i - c_{i-1})s_i, & \\ \qquad \text{if } j \neq J \text{ and } j = \{1, \cdots, (J-1)\}. & \end{cases}$$

We use contradiction to demonstrate the optimality of Lemma 2, which asserts that the optimal payments specified in the theorem can realize the minimum payment costs for the cloud platform. We assume that for a given data size, there exists at least one payment $\tilde{p}_j$

such that $\tilde{p}_j < (p_j)^*$. In order to ensure the feasibility of the contract, based on Condition (iii) of Lemma 1, $\tilde{p}_j$ must satisfy

$$\tilde{p}_j \geq \tilde{p}_{j+1} + c_j(s_j - s_{j+1}),$$

which can be rewritten as

$$\tilde{p}_{j+1} \leq \tilde{p}_j - c_j\Big(s_j - s_{j+1}\Big) < \Big(p_j\Big)^* - c_j\Big(s_j - s_{j+1}\Big) = \Big(p_{j+1}\Big)^*.$$

By repeating the aforementioned process, we can obtain

$$\tilde{p}_J < \left(\tilde{p}_J\right)^* = c_J s_J. \tag{A17}$$

Apparently, Formula (A17) does not satisfy Condition (i) of Lemma 1, thus violating the IR constraints. Therefore, the payments in Lemma 2 are optimal.

Appendix C

By restricting $\overline{\theta}$ to 1, the optimization problem is transformed into a vanilla contract incentive problem, which aligns with the upper-bound cost that we aim to optimize. Therefore, by setting $\overline{\theta} = 1$, it yields

$$\begin{aligned} \min\ & U_S = \frac{1}{\sqrt{\sum_{j\in\mathcal{J}} I_j s_j}} + \xi \sum_{j\in\mathcal{J}} I_j p_j, \\ \text{s.t.}\ & \begin{cases} 0 < s_j \leq s_{\max}, \\ p_J - c_J s_J \geq 0; \\ p_1 \geq \cdots \geq p_J \geq 0 \text{ and } s_1 \geq \cdots \geq s_J \geq 0; \\ p_{j+1} + c_j(s_j - s_{j+1}) \leq p_j \leq p_{j+1} + \\ \quad c_{j+1}(s_j - s_{j+1}), \forall j \in \mathcal{J}. \end{cases} \end{aligned} \tag{A18}$$

Based on Lemma 2, we can transform the above problem into a single variable optimization problem, which is only correlated with s_j. By using calculus, we have

$$\begin{cases} \frac{\partial U_S}{\partial s_j} = -\frac{1}{2\sqrt{I_j s_j{}^3}} + \xi I_j c_j, \\ \frac{\partial^2 U_S}{\partial (s_j)^2} = \frac{3}{4\sqrt{I_j s_j{}^5}} > 0. \end{cases} \tag{A19}$$

The fact that the second-order derivative is greater than 0 indicates that Formula (A18) is convex and $s_j^* = \frac{1}{I_j(2\xi c_j)^{\frac{2}{3}}}, \forall j \in \mathcal{J}$ is the minimum point. Therefore, by substituting $r_{max} = 1$ and s_j^* into Formula (A18), we have

$$\overline{U_S} = \frac{1}{\sqrt{\sum_{j\in\mathcal{J}} \frac{1}{(2\xi c_j)^{\frac{2}{3}}}}} + \xi|\mathcal{J}| \sum_{j\in\mathcal{J}} \frac{c_j}{(2\xi c_j)^{\frac{2}{3}}} + \xi \sum_{j=1}^{\mathcal{J}-1} I_j \cdot \sum_{i=j+1}^{\mathcal{J}} (c_i - c_{i-1}) \frac{1}{I_i (2\xi c_i)^{\frac{2}{3}}}.$$

Appendix D

In the following, we prove the establishment of *(i)* and *(ii)* in Proposition 1.

(i) For a given required data size, $s_j, \forall j \in \mathcal{J}$, according to Theorem 1, a solution $U_S'(s_j, \mathcal{J}')$ is feasible when it is lower than $\overline{U_S}$, i.e.,

$$\begin{aligned} & U_S' < \overline{U_S} \\ \Leftrightarrow\ & \frac{1}{\sqrt{\sum_{j\in\mathcal{J}'} r_j I_j s_j}} + \xi \sum_{j\in\mathcal{J}'} r_j I_j p_j' < \frac{1}{\sqrt{\sum_{j\in\mathcal{J}} I_j s_j}} + \xi \sum_{j\in\mathcal{J}} I_j p_j. \end{aligned}$$

In order to ensure the satisfaction of the above condition, we need to prove sufficient conditions, i.e.,

$$\Rightarrow \begin{cases} \frac{1}{\sqrt{\sum_{j\in\mathcal{J}'} r_j I_j s_j}} < \frac{1}{\sqrt{\sum_{j\in\mathcal{J}} I_j s_j}}, & \xi \sum_{j\in\mathcal{J}'} r_j I_j p'_j < \xi \sum_{j\in\mathcal{J}} I_j p_j, \end{cases}$$
$$\Leftrightarrow \begin{cases} \sum_{j\in\mathcal{J}'} r_j I_j s_j > \sum_{j\in\mathcal{J}} I_j\, s_j & \sum_{j\in\mathcal{J}'} r_j I_j p'_j < \sum_{j\in\mathcal{J}} I_j p_j, \end{cases}$$

If the above sufficient conditions hold, then the platform's incentivized type set is $\mathcal{J} = \{1, 2, \cdots, J'\}$. Correspondingly, since the edge servers with an AMEC larger than $\tilde{e}_{x'}$ are not included in the incentivized type set $\mathcal{J}'$, the optimal contract rewards of the platform under such a feasible type x' are

$$p'_j = \begin{cases} c_j s_j, & \text{if } j = J', \\ c_j s_j + \sum_{i=j+1}^{J'} (c_i - c_{i-1}) s_i, \\ \qquad \text{if } j \neq x'_\epsilon \text{ and } j = \{1, \cdots, (J'-1)\}. \end{cases}$$

It is worth noting that type-x' edge servers have the largest AMEC in the score order $\{\tilde{e}_j\}_{j\in\mathcal{J}}$, but they may not necessarily have the largest training costs, c_j, in the incentivized set $\mathcal{J}'$. So, *(i)* is proven.

(ii) Since the cost optimization problem of the cloud server involves two variables (s_j, r_{max}), if a threshold type x'_ϵ exists under the current $r_{max} = \epsilon$, then the cloud server can always search for a new solution that is not inferior (to such a feasible solution) by adjusting its data requirement, s_j. Therefore, under a feasible $r_{max} = \epsilon$, the cloud server's optimal incentivized type set can be formalized as $\mathcal{J}^*_\epsilon \triangleq \{1, 2, \cdots, x^*_\epsilon\}$, which ensures that $U^*_{S_\epsilon}(s^*_j, r_{max}) \leq U'_{S_\epsilon}(s_j, r_{max})$ holds. Hence, the validity of *(ii)* is proven.

References

1. Pantelopoulos, A.; Bourbakis, N.G. A survey on wearable sensor-based systems for health monitoring and prognosis. *IEEE Trans. Syst. Man, Cybern. Part C Appl. Rev.* **2009**, *40*, 1–12. [CrossRef]
2. Anguita, D.; Ghio, A.; Oneto, L.; Parra, X.; Reyes-Ortiz, J.L. A public domain dataset for human activity recognition using smartphones. In Proceedings of the Esann, Bruges, Belgium, 24–26 April 2013; Volume 3, p. 3.
3. Zhu, G.; Liu, D.; Du, Y.; You, C.; Zhang, J.; Huang, K. Toward an intelligent edge: Wireless communication meets machine learning. *IEEE Commun. Mag.* **2020**, *58*, 19–25. [CrossRef]
4. Ma, Z.; Xu, Y.; Xu, H.; Meng, Z.; Huang, L.; Xue, Y. Adaptive batch size for federated learning in resource-constrained edge computing. *IEEE Trans. Mob. Comput.* **2023**, *22*, 37–53. . [CrossRef]
5. Lu, Y.; Huang, X.; Dai, Y.; Maharjan, S.; Zhang, Y. Federated learning for data privacy preservation in vehicular cyber-physical systems. *IEEE Network* **2020**, *34*, 50–56. [CrossRef]
6. Xiao, Y.; Zhang, X.; Li, Y.; Shi, G.; Krunz, M.; Nguyen, D.N.; Hoang, D.T. Time-sensitive learning for heterogeneous federated edge intelligence. *IEEE Trans. Mob. Comput.* 2023, *early access*. [CrossRef]
7. Luo, S.; Chen, X.; Wu, Q.; Zhou, Z.; Yu, S. Hfel: Joint edge association and resource allocation for cost-efficient hierarchical federated edge learning. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 6535–6548. [CrossRef]
8. Lyu, L.; Chen, C. A novel attribute reconstruction attack in federated learning. *arXiv* **2021**, arXiv:2108.06910.
9. Varshney, P.; Simmhan, Y. Characterizing application scheduling on edge, fog, and cloud computing resources. *Softw. Pract. Exp.* **2020**, *50*, 558–595. [CrossRef]
10. Stich, S.U. Local sgd converges fast and communicates little. *arXiv* **2018**, arXiv:1805.09767.
11. Liu, B.; Shen, W.; Li, P.; Zhu, X. Accelerate mini-batch machine learning training with dynamic batch size fitting. In Proceedings of the 2019 International Joint Conference on Neural Networks (IJCNN), Budapest, Hungary, 14–19 July 2019; pp. 1–8.
12. Ho, Q.; Cipar, J.; Cui, H.; Lee, S.; Kim, J.K.; Gibbons, P.B.; Gibson, G.A.; Ganger, G.; Xing, E.P. More effective distributed ml via a stale synchronous parallel parameter server. *Adv. Neural Inf. Process. Syst.* **2013**, *26*, 1223–1231.
13. Zhang, J.; Tu, H.; Ren, Y.; Wan, J.; Zhou, L.; Li, M.; Wang, J. An adaptive synchronous parallel strategy for distributed machine learning. *IEEE Access* **2018**, *6*, 19222–19230. [CrossRef]
14. Chen, C.; Wang, W.; Li, B. Round-robin synchronization: Mitigating communication bottlenecks in parameter servers. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 532–540.
15. Wang, S.; Tuor, T.; Salonidis, T.; Leung, K.K.; Makaya, C.; He, T.; Chan, K. Adaptive federated learning in resource constrained edge computing systems. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1205–1221. [CrossRef]

16. Tran, N.H.; Bao, W.; Zomaya, A.; Nguyen, M.N.; Hong, C.S. Federated learning over wireless networks: Optimization model design and analysis. In Proceedings of the IEEE INFOCOM 2019—IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 1387–1395.
17. Zeng, R.; Zhang, S.; Wang, J.; Chu, X. Fmore: An incentive scheme of multi-dimensional auction for federated learning in mec. In Proceedings of the 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS), Singapore, 29 November–1 December 2020; pp. 278–288.
18. Kang, J.; Xiong, Z.; Niyato, D.; Xie, S.; Zhang, J. Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory. *IEEE Internet Things J.* **2019**, *6*, 10700–10714. [CrossRef]
19. Tang, M.; Wong, V.W. An incentive mechanism for cross-silo federated learning: A public goods perspective. In Proceedings of the IEEE INFOCOM 2021—IEEE Conference on Computer Communications, Vancouver, BC, Canada, 10–13 May 2021; pp. 1–10.
20. Wang, Y.; Su, Z.; Luan, T.H.; Li, R.; Zhang, K. Federated learning with fair incentives and robust aggregation for uav-aided crowdsensing. *IEEE Trans. Netw. Sci. Eng.* **2021**, *9*, 3179–3196. [CrossRef]
21. Xiao, Y.; Shi, G.; Li, Y.; Saad, W.; Poor, H.V. Toward self-learning edge intelligence in 6g. *IEEE Commun. Mag.* **2020**, *58*, 34–40. [CrossRef]
22. Cipar, J.; Ho, Q.; Kim, J.K.; Lee, S.; Ganger, G.R.; Gibson, G.; Keeton, K.; Xing, E. Solving the straggler problem with bounded staleness. In Proceedings of the 14th Workshop on Hot Topics in Operating Systems (HotOS XIV), Santa Ana Pueblo, NM, USA, 13–15 May 2013.
23. Tyagi, S.; Sharma, P. Taming resource heterogeneity in distributed ml training with dynamic batching. In Proceedings of the 2020 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS), Washington, DC, USA, 17–21 August 2020; pp. 188–194.
24. Ye, Q.; Zhou, Y.; Shi, M.; Sun, Y.; Lv, J. Dbs: Dynamic batch size for distributed deep neural network training. *arXiv* **2020**, arXiv:2007.11831.
25. Xia, W.; Quek, T.Q.; Guo, K.; Wen, W.; Yang, H.H.; Zhu, H. Multi-armed bandit-based client scheduling for federated learning. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 7108–7123. [CrossRef]
26. Shi, W.; Zhou, S.; Niu, Z. Device scheduling with fast convergence for wireless federated learning. In Proceedings of the ICC 2020—2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, 7–11 June 2020; pp. 1–6.
27. Chen, M.; Poor, H.V.; Saad, W.; Cui, S. Convergence time minimization of federated learning over wireless networks. In Proceedings of the ICC 2020—2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, 7–11 June 2020; pp. 1–6.
28. Feng, S.; Niyato, D.; Wang, P.; Kim, D.I.; Liang, Y.-C. Joint service pricing and cooperative relay communication for federated learning. In Proceedings of the 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Atlanta, GA, USA, 14–17 July 2019; pp. 815–820.
29. Sarikaya, Y.; Ercetin, O. Motivating workers in federated learning: A stackelberg game perspective. *IEEE Netw. Lett.* **2019**, *2*, 23–27. [CrossRef]
30. Ding, N.; Fang, Z.; Huang, J. Optimal contract design for efficient federated learning with multi-dimensional private information. *IEEE J. Sel. Areas Commun.* **2020**, *39*, 186–200. [CrossRef]
31. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, FL, USA, 20–22 April 2017; pp. 1273–1282.
32. Carli, R.; Chiuso, A.; Schenato, L.; Zampieri, S. A pi consensus controller for networked clocks synchronization. *IFAC Proc. Vol.* **2008**, *41*, 10289–10294. [CrossRef]
33. Wang, D.; Ren, J.; Wang, Z.; Wang, Y.; Zhang, Y. Privaim: A dual-privacy preserving and quality-aware incentive mechanism for federated learning. *IEEE Trans. Comput.* **2022**, *72*, 1913–1927. [CrossRef]
34. Ding, N.; Gao, L.; Huang, J. Joint participation incentive and network pricing design for federated learning. In Proceedings of the IEEE INFOCOM 2023—IEEE Conference on Computer Communications, New York City, NY, USA, 17–20 May 2023; pp. 1–10.
35. Li, M.; Zhang, T.; Chen, Y.; Smola, A.J. Efficient mini-batch training for stochastic optimization. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 24–27 August 2014; pp. 661–670.
36. Dekel, O.; Gilad-Bachrach, R.; Shamir, O.; Xiao, L. Optimal distributed online prediction using mini-batches. *J. Mach. Learn. Res.* **2012**, *13*, 165–202.
37. Wang, X.; Zhao, Y.; Qiu, C.; Liu, Z.; Nie, J.; Leung, V.C. Infedge: A blockchain-based incentive mechanism in hierarchical federated learning for end-edge-cloud communications. *IEEE J. Sel. Areas Commun.* **2022**, *40*, 3325–33422. [CrossRef]
38. Lu, J.; Liu, H.; Jia, R.; Zhang, Z.; Wang, X.; Wang, J. Incentivizing proportional fairness for multi-task allocation in crowdsensing. *IEEE Trans. Serv. Comput.* **2023**. [CrossRef]
39. Lu, J.; Liu, H.; Jia, R.; Wang, J.; Sun, L.; Wan, S. Towards personalized federated learning via group collaboration in iiot. *IEEE Trans. Ind. Inform.* **2023**, *19*, 8923–8932. [CrossRef]
40. Lu, J.; Liu, H.; Zhang, Z.; Wang, J.; Goudos, S.K.; Wan, S. Toward fairness-aware time-sensitive asynchronous federated learning for critical energy infrastructure. *IEEE Trans. Ind. Inform.* **2022**, *18*, 3462–3472. [CrossRef]

41. Ying, C.; Jin, H.; Wang, X.; Luo, Y. Double insurance: Incentivized federated learning with differential privacy in mobile crowdsensing. In Proceedings of the 2020 International Symposium on Reliable Distributed Systems (SRDS), Shanghai, China, 21–24 September 2020; pp. 81–90.
42. Huang, T.; Lin, W.; Wu, W.; He, L.; Li, K.; Zomaya, A.Y. An efficiency-boosting client selection scheme for federated learning with fairness guarantee. *IEEE Trans. Parallel Distrib. Syst.* **2020**, *32*, 1552–1564. [CrossRef]
43. Ma, C.; Konečnỳ, J.; Jaggi, M.; Smith, V.; Jordan, M.I.; Richtárik, P.; Takáč, M. Distributed optimization with arbitrary local solvers. *Optim. Methods Softw.* **2017**, *32*, 813–848. [CrossRef]
44. Sultana, A.; Haque, M.M.; Chen, L.; Xu, F.; Yuan, X. Eiffel: Efficient and fair scheduling in adaptive federated learning. *IEEE Trans. Parallel Distrib. Syst.* **2022**, *33*, 4282–4294. [CrossRef]