EasyChair Preprint

№ 13968

Stake-Driven Rewards and Log-Based Free Rider Detection in Federated Learning

Huong Nguyen, Hong-Tri Nguyen, Lauri Loven and Susanna Pirttikangas

July 13, 2024

Stake-Driven Rewards and Log-Based Free Rider Detection in Federated Learning

Huong Nguyen[a], Hong-Tri Nguyen[b], Lauri Lovén[a], Susanna Pirttikangas[a]
[a]*Center for Ubiquitous Computing, University of Oulu, Finland*. Email: {firstname.lastname}@oulu.fi
[b]*School of Computer Science, Aalto University, Finland*. Email: {firstname.lastname}@aalto.fi

***Abstract*—Federated learning has become increasingly popular due to its ability to bring together multiple learners, enhance model generalizability, and promote knowledge exchange. Such systems inherently rely on the bedrock of security, trust, and fairness among training workers to ensure a conducive learning environment. However, this collaborative landscape has encountered the challenge of free riders, individuals who join the systems to gain benefits without making any substantial contributions. This can negatively impact learning outcomes, fairness, sustainability, and trust in a collaborative system. In this paper, we first present a novel stake-based incentive mechanism aimed at promoting active participation among contributors, and concurrently maximizing the reward for clients with consideration of free rider presence in the system. Second, we propose an efficient method for identifying free riders in federated learning based on log analysis. Our method delegates the detection of free riders to training workers and the identification to the aggregator, rather than relying solely on the aggregator. We simulate potential deceptive strategies employed by free riders and assess the extent of our method's coverage across these scenarios. The experimental results conducted on different free rider ratios demonstrate the versatility and applicability of our approach in detecting these clients within the federated learning paradigm.**

***Index Terms*—Federated learning, collaborative system, security, free rider attacks, trust, fairness.**

I. Introduction

Federated Learning (FL) [1], [2] is a cutting-edge approach for distributed learning, divergent from the centralized training paradigm. It prioritizes data privacy by executing computations locally on devices, obviating the need for centralized data transfers. In the FL framework, training workers execute training, while a central aggregator merges these individual models into a global one and then distributes it back to the participating devices, ensuring data remains decentralized and secure.

The success of this collaborative architecture is from the contributions of training workers; therefore, an effective incentive mechanism plays a pivotal role in attracting more contributors. In the context of open collaborations, incentives should come with the guarantee of trustworthiness, fairness, and appealing rewards. This is vital because the distribution of rewards, divided equally among all training workers, could be considered inequitable while there are some lazy workers. Regarding this, the current FL implementations encounter challenges related to the system's sustainability, specifically concerns of fairness and trust among training workers (as a crucial requirement - **R1**), primarily due to the presence of free riders. These cheating people seek to obtain the global model like other training workers but contribute minimally to the learning process, causing an imbalance within the system.

In free-rider attacks [3], [4] to FL systems, the free riders act as training workers; instead of contributing to the learning process, they aim to benefit from the system through global updates while submitting false weights to the aggregator. These free riders can produce fake weights in multiple ways, such as generating random local weights or creating sophisticated random local weights by adding Gaussian noise and modifying standard deviation. This deceptive behavior compromises the fairness and efficiency of the FL system, as it enables some clients to gain advantages without making meaningful contributions to the learning process.

As another requirement **(R2)**, addressing the free riders issue is crucial for facilitating the successful growth and development of distributed learning and FL [5]. Many prior works have dedicated significant effort to proposing incentive methods [6], [7] and developing techniques to detect free riders [8], [9] in collaborative learning environments. However, these incentive methods often overlook free riders' presence and struggle with forming clear, quantifiable rewards for clients. On the other hand, recent mainstream approaches for detecting free riders did not explicitly provide information on creating the free rider weights, making it challenging to reproduce their results for comparison. The testing with a fixed noise value, even though across various datasets, also cannot reflect the reliable performance or coverage of their approach. Finally, the reliance on the processing of the aggregator in all these works can inhibit scalability and efficiency, causing bottlenecks and stagnant workloads.

In this paper, we detail two primary contributions of our work. Firstly, we introduce a novel incentive mechanism based on the concept of staking, wherein training workers demonstrating greater dedication are rewarded with proportionately valuable incentives. Secondly, our approach to free rider detection differs significantly from previous methods. Instead of relying solely on the aggregator for detection, we delegate this task to the training workers, empowering them to identify abnormalities and trigger investigations autonomously. By adopting this decentralized approach, we minimize the workload on the aggregator, ensuring that it only intervenes when necessary. Furthermore, our proposed solution undergoes rigorous evaluation through explicit and intensive experiments, encompassing diverse free rider strategies and ratios to validate

its efficacy and robustness.

The remainder of this work is organized as follows: Section II discusses related work. Our methodologies are presented in Section III. Section IV provides a detailed description and evaluation of the experiments. Finally, Section V discusses and concludes this work.

II. Related Works

A. Incentive systems

In the landscape of collaborative learning systems, incentive mechanisms assume a pivotal role in engendering contributor engagement. As posited by Tu et al. [10], the impetus for contributors often emanates from the enticements offered by these mechanisms, a facet particularly pertinent in economic and game theoretic paradigms, where the pursuit of maximum benefits holds sway. While these mechanisms encompass various strategies, they are mostly rooted in non-cooperative game theory, where entities optimize their actions based on Nash Equilibrium concepts [11]. For example, the work by Tang et al. [12] in the context of cross-silo FL focuses on incentive mechanisms that facilitate mutual agreement on computation and fees between the aggregator and trainers. This approach hinges on the historical profiles of trainers, enabling the aggregator to assess trainers' capabilities for maximal payoffs.

In close connection with our incentive viewpoint that requires the amount of deposit from training workers, Tahanian et al. [6] proposed a game-based robust federated averaging approach to accept or reject local models via Nash Equilibrium property. Particularly, the Nash Equilibrium property estimates the probability that the aggregator accepts local updates via the diversity of trainers' behavior in order to maximize their profits. Besides, the work of Weng et al. [7] introduces a novel federated prediction serving framework that employs an incentive mechanism rooted in Bayesian game theory to attract training workers for collaborative machine learning prediction. This mechanism also requires training workers to deposit funds to partake in the system before acquiring any rewards.

However, the crux of the matter is that: while existing incentive mechanisms hold significant potential, they are not a comprehensive solution due to the overlooked of free rider existence. Free riders, who exploit contributions without offering commensurate effort to gain rewards, can undermine the equilibrium these mechanisms endeavor to establish. Thus, incentive mechanisms aiming at fairness and engagement require a supplementary layer to account for the presence of free riders while still prioritizing reward maximization. This is where our proposed rewarding method comes into play.

B. Free rider detection

Regarding free rider detection, there are diverse solutions coming from different proposed works. Deep learning-based methods, like those used in [13] and [3], may improve detection accuracy with Gaussian Mixture Model and Deep Neural Network. However, they also come with the cost of increased computation complexity and potentially longer detecting times. On the other hand, the statistical correlation approach proposed by Xu and Lyu in [14] may not be reliable in all free rider detection scenarios, as it is sensitive to outliers or noise in the data. Additionally, their work assumes a majority of honest clients rather than free riders, which may not hold in certain real-world scenarios. Lastly, requiring significant communication and computation overhead to update user reputation scores is another potential weakness of their method.

In terms of efficiency evaluation, prior works have predominantly overlooked the verification of their methods across diverse strategies and setups but often focused solely on a single scenario. Specifically, [13] exclusively tests on weights with Gaussian noise added, while [14] only focuses on random weights scenarios, believing that the addition of Gaussian noise to aggregated weights poses no significant challenge for detection.

To distinguish our work from other mainstream solutions, we provide more in-depth comparisons with a couple of recently highlighted papers within the research community [8], [9]. Wang et al. [8] conducted extensive experiments using various datasets and aggregation methods. However, they did not disclose the hyperparameters for setting up the noise used by free riders. This omission is crucial when variations in the σ value of the noise can significantly impact the generated fake weights, thereby affecting the reliability of their assessment. Lately, Chen et al. [9] introduced the WEF-defense method and conducted thorough evaluations across different experiments. Unlike Wang et al., they did not conceal the way of setting the noise, albeit using only a fixed value, which may not fully capture the coverage of their method across different noises. Additionally, the requirement for each client to upload the WEF-Matrix along with their model weights can be a downside of this method, being inefficient in large networks with thousands of training workers. Finally, all the mentioned approaches only run the detection task at the aggregator side, potentially causing a bottleneck and single failure.

In response, our approach delegates part of the task to the clients, ensuring that the aggregator only needs to work when necessary. Our assessment includes two main cases of free riders (random weights and Gaussian noise) with explicit setup information on each case. We also conduct our experiments with different free rider ratios, which are also considered in [13], [15], and [3]. Notably, our approach is a more lightweight and fast method to detect free riders since we enable the group skipping policy based on the linkage clustering.

III. Methods

This section presents separate components of our work, each designed to fulfill requirements R1 and R2.

A. Collaborative learning with stakes

1) General idea: We propose a novel monetary incentive mechanism for a fair collaboration (R1) where each client sets a stake, which is returned later with an additional reward if

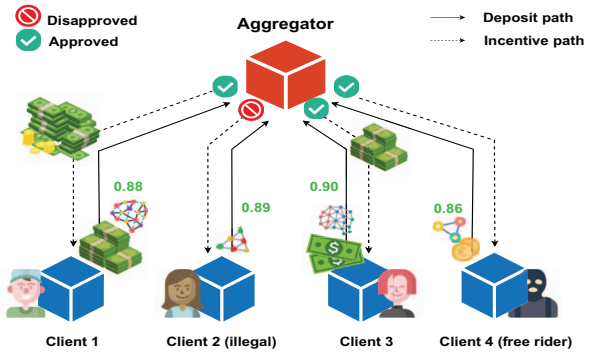


Fig. 1: The incentive proposed mechanism, with high stake leading to high returns

there is no indication of cheating (see, e.g., [16], [17] for studies considering game-theoretic modeling of stakes). Our idea is influenced by the underlying principle "Skin in the game" [18], put forth by Nassim Nicholas Taleb, who argues that individuals will exhibit greater responsibility, and less recklessly behave when they have a personal stake or risks involved in the outcomes of their actions. By introducing deposits into the collaborative learning process, we aim to create a gambling-like setting where learners cannot participate without an investment.

In more detail, training workers are required to deposit an amount of money and report their model's accuracy with the aggregator, in addition to submitting trained weights. In accordance with the principle of "high stake, high return", training workers with higher deposit amounts are considered dedicated learners, who are less likely to break the system, and thus, will receive more valuable rewards, while those with lower deposits will obtain less. This approach ensures that training workers receive a commensurate reward based on their deposit amount, regardless of slight differences in the models' quality, as long as their behavior does not violate the rules or involve cheating. To clarify, the origin of rewards comes from the allocated funds that regional aggregators received from the task owners before the commencement of the training process. This total reward will then be distributed equally among each round. Figure 1 illustrates how staking is applied in the joining process.

With regard to the distribution of rewards, let us consider the case where a free rider (client 4) and a no-stake client (client 2) are present in the system (Figure 1). At each update round, a deposit from clients serves as a joining fee and a guarantee for their model validity. This ensures that only committed clients can join the model update process and without it, the contribution of the client will not be counted towards the regional model. Additionally, the clients with no stake will also not receive the aggregated model, as is the case with client 2.

Upon receiving stake deposits from eligible clients, the aggregator combines all local model updates, and distributes

Algorithm 1: Stake-based reward mechanism

Input : K IDs of eligible clients, deposit amount d_i (i = client ID), total reward amount R_t
Output: Amount each client gets

1 $D \leftarrow 0$ // total deposits
2 **for** i *in* K **do**
3 $\quad D \leftarrow D + d_i$
4 **end**
5 **if** *conflict* **then**
6 $\quad L_{CHEAT} \leftarrow$ Execute Algorithm 2
7 $\quad$ **if** *len*(L_{CHEAT}) **then**
8 $\quad\quad D_{FR} \leftarrow 0$ // free riders' deposits
9 $\quad\quad$ **for** j *in* $K \cap L_{CHEAT}$ **do**
10 $\quad\quad\quad D_{FR} \leftarrow D_{FR} + d_j$
11 $\quad\quad$ **end**
12 $\quad\quad$ **if** $R_t == D$ **then**
13 $\quad\quad\quad$ **for** j *in* $K \setminus L_{CHEAT}$ **do**
14 $\quad\quad\quad\quad r_j \leftarrow R_t \times d_j / D$
15 $\quad\quad\quad$ **end**
16 $\quad\quad\quad R_{t+1} \leftarrow R_{t+1} + D_{FR}$
17 $\quad\quad$ **else**
18 $\quad\quad\quad \xi \leftarrow D_{FR} / len(K \setminus L_{CHEAT})$
19 $\quad\quad\quad$ **if** $R_t > D$ **then**
20 $\quad\quad\quad\quad$ **for** j *in* $K \setminus L_{CHEAT}$ **do**
21 $\quad\quad\quad\quad\quad r_j \leftarrow R_t \times (d_j + \xi) / D$
22 $\quad\quad\quad\quad$ **end**
23 $\quad\quad\quad$ **else**
24 $\quad\quad\quad\quad$ **for** j *in* $K \setminus L_{CHEAT}$ **do**
25 $\quad\quad\quad\quad\quad r_j \leftarrow R_t \times d_j / D + \xi$
26 $\quad\quad\quad\quad$ **end**
27 $\quad$ **end**
28 **else**
29 $\quad$ **for** i *in* K **do**
30 $\quad\quad r_i \leftarrow R_t \times d_i / D$
31 $\quad$ **end**

the results to eligible clients, regardless of whether they are free-riding clients or not. In case no conflict is raised, the distribution of rewards among clients is then based on their respective deposit amounts, calculated by the ratio (%) between that amount and the total deposits of all clients (Algorithm 1, lines 1-4, 29-31). However, even if just a single client makes a complaint, the aggregator must conduct an investigation, referred to Algorithm 2. The aggregator will then reaggregate the model and recalculate the reward for each client based on Algorithm 1.

2) Rewarding mechanism: We consider three ways (A1-A3) of rewarding when the investigation is triggered. With K is the list of eligible clients, L_{CHEAT} is the list of detected free riders, $D = sum(d_1, d_2, \ldots, d_n)$ is the total deposit amount of all clients, D_{FR} is the total deposit amount of detected free riders ($D_{FR} \leq D$), and the distributed reward at round t is R_t, we analyze the gain of each client in corresponding approaches, summarized in Table I.

TABLE I: The reward amount client i get besides the returned deposit, set $r = \frac{R_t d_i}{D}$ and $\xi = \frac{D_{FR}}{len(K \backslash L_{CHEAT})}$

Case		No conflict	Conflict		
Strategy		-	A1	A2	A3
Apply condition		-	$R_t = D$	$R_t > D$	$R_t < D$
Gain amount	Client	r	r	$r + \xi R_t / D$	$r + \xi$
	R_{t+1}	0	D_{FR}	0	0

- A1 (Algorithm 1 lines 12-16): The deposit amount from detected free riders, D_{FR} will be added to the next round (t+1)'s reward R_{t+1}. Incentive amounts are calculated based on the original deposits of clients D by ratio.
- A2 (Algorithm 1 lines 18-22): An extra from free riders' deposits (divided by number of benign clients) $\xi = \frac{D_{FR}}{len(K \backslash L_{CHEAT})}$ is added to each client's original deposit amount d_i before calculating the incentive amount for each client by ratio.
- A3 (Algorithm 1 lines 24-26): Incentive amounts are first calculated based on the original deposits D by ratio, then an extra amount ξ will be added to each client reward.

From Table I, we can see that the incentive gained for each client at round t is varied with different reward values R_t and the total deposit amount D. To maximize the benefit in each case, the aggregator with the information of those amounts will choose the proper approach to calculate.

By using A2 and A3, each client will gain an extra amount from the deposits of detected free riders. Specifically, with A2, the reward amount for each client is calculated as $\frac{R_t(d_i+\xi)}{D}$, which can be further interpreted as $\frac{R_t d_i}{D} + \frac{R_t \xi}{D}$. On the other hand, with A3, the reward is given as $\frac{R_t d_i}{D} + \xi$. The distinction between these two approaches lies in the fraction $\frac{R_t}{D}$, which determines the additional reward allocated to each client. If this fraction is greater than one (indicating $R_t > D$), A2 offers a higher reward to the clients, while A3 would maximize this extra amount when the fraction is less than one. In contrast, $A1$ maintains the same reward amount as in non-conflict cases ($\frac{R_t d_i}{D}$), while increasing the reward in the next round. By using A1, unnecessary computation steps for calculating the extra reward can be minimized, while ensuring clients' rewards correspond appropriately to their deposits.

B. Log-based Free Rider Detection

1) Free rider presence detection: As previously discussed, training workers are assigned the responsibility of detecting the potential presence of free riders (R2) and triggering an investigation on the aggregator's end. This delegation aims to reduce the aggregator's workload by avoiding an unnecessary investigation in each round of aggregation.

The collaborative learning process, as observed in the diagram in Figure 2, begins with clients receiving the model weights from the aggregator and subsequently conducting local training. Upon completion of the local training, clients join the collaborative learning process with a deposit as the required fee.

The aggregator then performs a **weight copying check** on all submissions received to ensure that none of the submitted

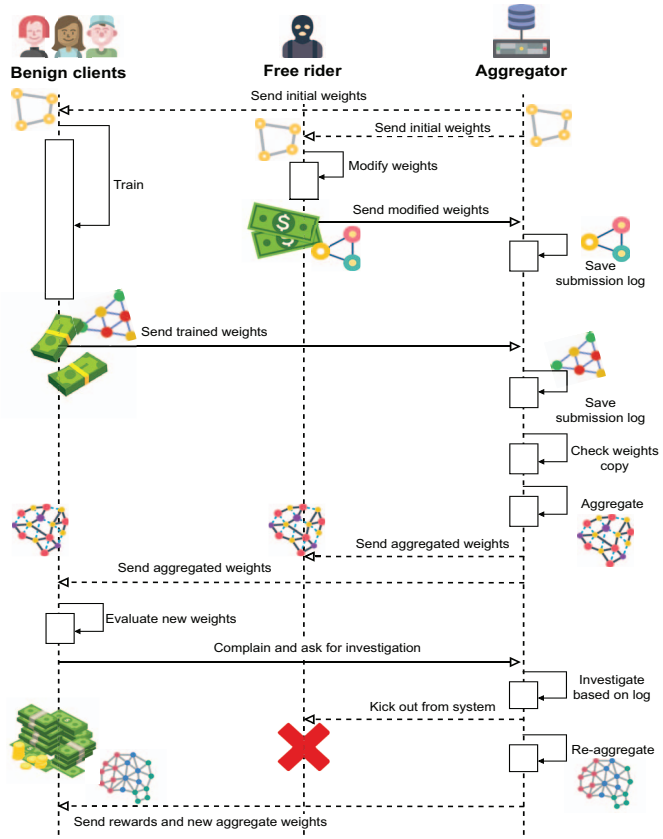


Fig. 2: Training process in our learning system, with investigation from the aggregator after the client's trigger

models are copied from the previous round. This step helps to identify and remove any free riders who directly use the aggregated weights without making any additional adjustments. Once the copying check is completed, the aggregator conducts the aggregation and distributes the aggregated weights back to all clients. It should be noted that at this stage, only the aggregated model is returned, not the rewards.

Upon receipt of the aggregated model, clients will perform a re-evaluation using the same validation dataset previously used in their local training. The accuracy of the current aggregated model is then compared to (i) the aggregated accuracy of the previous round and (ii) the client's own model at the current round.

(i) If a decrease in the aggregated model's accuracy is detected, clients will request the aggregator to conduct an investigation to identify any abnormalities, e.g. instances of free riding. Our idea of this simple consideration derives from the widely recognized paper of McMahan et al. [1], which validated the improvement in the global model accuracy when increasing rounds of communication. Similar studies, [19] and [20], support this dependency.

(ii) In order to ascertain the deviation from their own model, each client calculates the cosine distance between the aggregated model and their individual model. If this disparity exceeds the average distances observed in prior

rounds, clients may duly submit a request to the aggregator, signaling the need for further investigation.

Finally, after the investigation is closed, identified cheaters are removed from the system and the aggregated weights will be recalculated, along with rewards, will be returned to all benign clients.

Algorithm 2: Log-based Investigation

Input : Round R, Aggregated accuracy A_R, previous aggregated accuracy A_{R-1}, latest safe gap A_{gap} (if R!=0), last submitted weights W, deposits D, reported accuracy A_{re}

Output: $L_{CHEAT} \leftarrow$ IDs of cheaters

1 **Initialize:** $L_{CHEAT} \leftarrow$ empty list
2 **Investigation process:**
3 1. Calculate the distance among submitted weights
4 **for** W_i *in* W **do**
5 **for** W_j *in* W **do**
6 $D_{mat}^{ij} \leftarrow$ calculate cosine distance (W_i, W_j)
7 **end**
8 **end**
9 2. Groups $G \leftarrow$ Apply Linkage Clustering on D_{mat}
10 3. Check weights in each group
11 **for** G_i *in* G **do**
12 **for** j *in* $G_i \setminus L_{CHEAT}$ **do**
13 $A_j \leftarrow$ evaluate model accuracy (W_j, test set)
14 **if** *not IsSafe*$(A_j, A_{R-1}, A_R, A_{gap})$ **then**
15 $L_{CHEAT} \leftarrow L_{CHEAT} + j$
16 **else**
17 break `// skip group`
18 **end**
19 **end**
20 **if** *len*(L_{CHEAT}) **then**
21 recalculate A_{gap}
22 **return** L_{CHEAT}
23 **Function** `IsSafe`$(A, A_{R-1}, A_R, A_{gap})$**:**
24 $max_A \leftarrow max(A_{R-1}, A_R)$
25 $gap \leftarrow \mid max_A - A \mid$
26 **if** $(A < max_A$ *and* $gap > A_{gap})$
27 *or* $A < A_{R-1} \leq A_R$ **then**
28 **return** *False*
29 **return** *True*

2) Log-based investigation: Our investigation starts with calculating all pairwise cosine distances between client i and j and forming the distance matrix D_{mat} (Algorithm 2, line 4-8), using it to construct a linkage clustering tree (Algorithm 2, line 9) and spot the cheaters. In our work, linkage clustering is used as an effective method for grouping similar objects (here clients) into the same cluster based on their pairwise values and calculating the distance between different clusters. At this step, the aggregator will then go through each cluster and re-evaluate the accuracy of each submitted weight (Algorithm 2, line 13) and then assess the honesty of clients with the safe checking function - IsSafe (Algorithm 2, line 23-29). It should be noted that the aggregator will first perform the re-evaluation on a single member of each group and subsequently determine whether to proceed with checking all clients within the group or move on to another group for time and computation optimization. Finally, upon going through all groups, the aggregator will check if any cheater is detected and remove them from the system, reaggregate the model, and calculate the new A_{gap} before proceeding to a new communication round (Algorithm 2, line 20-22).

Considerations about metrics and functions used in our investigation are detailed below:

Safe gap calculation: One of the important metrics in our analysis is the safe accuracy gap A_{gap}, which serves as a threshold to identify free riders. This value is calculated based on the average gap between the accuracy of the aggregated model and clients in all prior rounds. For example, if the investigation is triggered at iterator 8, the average gap value is referred to the one computed up to round 7. Of note, for the first aggregation round, all submitted weights will be re-evaluated to obtain the correct accuracy value since there is no previous round to reference. We formulate this gap value under Equation 1, where M is the number of clients, R is the number of rounds before the investigation, A_{agg}^t and A_{re}^{it} are the aggregated model's accuracy and reported value from client i at round t, respectively.

$$A_{gap} = \frac{1}{M}\frac{1}{R}\sum_{i=1}^{M}\sum_{t=1}^{R} \mid A_{agg}^t - A_{re}^{it} \mid \tag{1}$$

In preparation for any potential future investigations, this gap value is calculated and updated at every round. However, if cheating is detected, the aggregator will need to calculate the new A_{gap} after reaggregating and excluding the violated weights (Algorithm 2, line 21).

IsSafe function: The purpose of this function is to determine whether a client is safe enough and not engaged in any cheating behavior by reevaluating their submitted weight's accuracy. During each aggregation round, aggregators receive submissions from participating clients, containing details like submitted weights W, timing information, deposit amounts D, and reported accuracy of the client's model A_{re}. At this stage, free riders may fabricate the accuracy to deceive the aggregator about their low-quality models, whereas benign clients will provide their actual accuracy scores obtained from their models. However, it is important to note that free riders may also provide accurate reports of their underperformance. Aside from this, certain strategies can be employed by different types of free riders. Some may promptly send their randomly generated or noise-added weights along with a minimal deposit, while more dodgy cheaters might delay their submission, subsequently depositing a larger amount to appear more legit. Given these intricacies, individually checking clients based on submission time and deposit size becomes time-wasting and ineffective in identifying all free riders. As a result, we formulate this function by analyzing the

typical learning process and setting conditions that potential free riders are likely to breach.

The first condition involves checking for a decrease in accuracy during the investigation round as compared to the previous round. While this condition is rooted in the client-side detection mechanism, it needs to be coupled with another condition in this identification process: the absolute difference between the maximum aggregated accuracy and the re-evaluated accuracy of the client (Algorithm 2 line 25) not exceeding the safe accuracy gap A_{gap}. To clarify, the maximum aggregated accuracy is determined by selecting the higher value between the two aggregated accuracies - one from the investigation round and the other from the previous round (Algorithm 2, line 24). With input from submission logs, the aggregator reevaluates the corresponding weights of each client on its test data and makes the comparison. If this absolute difference is within the bounds of A_{gap}, the client is considered benign and passes the checking. However, if this value surpasses A_{gap}, it does not necessarily indicate a free rider. Instead, the client is appended to the list of potentially cheating clients, L_{CHEAT}, if their re-evaluated accuracy is lower than the maximum aggregated accuracy (Algorithm 2, line 26). We can see that when the gap value exceeds both A_{gap} and the maximum aggregated accuracy, it suggests that the client is benign. This interpretation is valid because their accuracy is consistently increasing over communication rounds, which is in stark contrast to free riders. This combined condition is of particular importance in scenarios where free riders dominate the system, resulting in the aggregated model accuracy closely aligning with the free riders' values rather than those of legitimate clients. In such cases, solely relying on the gap value concerning the current aggregated accuracy (not with the maximum value) could lead to misclassifying benign clients as free riders.

The second criterion for detecting a client as a free rider occurs when their re-evaluated accuracy is lower than the accuracy in the previous round, despite the fact that the average accuracy of this round continues to gradually increase and surpasses that of the previous round (Algorithm 2, line 27). This condition is grounded in the consistent incremental improvement observed in both the global accuracy and the accuracy of individual client models across communication rounds. Regarding this, we expect both the global model and a benign client to have higher accuracy compared to the last aggregation round, not just higher in the global model.

Distance metric: We measure the distances between client weight matrices, as flattened vectors, by utilizing cosine distance as opposed to other metrics such as Euclidean, Manhattan, or any Lp distances. This can be explained by a couple of reasons. First, cosine distance can be directly calculated based on the angle between vectors, without depending on the magnitude. This is particularly important in the case where a free rider generates a random weight matrix with a different size and shape compared to a properly fitted weight matrix of the model. In such a case, Lp distance may need to pad one vector with zeroes to equalize the dimensions between the two before calculating the distance. This requires additional steps and time in the investigation process. Second, the range of cosine distance is within $[-1, 1]$, which makes it a simple and effective metric compared to the other two, which have no specific range.

Linkage clustering: Linkage clustering is used due to its standing out in hierarchical clustering for its adaptability and depth in revealing complex data relationships through a dendrogram. Unlike flat clustering methods like K-means, which require prior knowledge of cluster numbers and are sensitive to outliers, this one does not necessitate a predefined number of clusters and effectively handles different data and densities. Besides, its intuitive dendrogram presentation also helps in visualizing and interpreting the clustering process, providing clear insights into the hierarchical structure of data groupings.

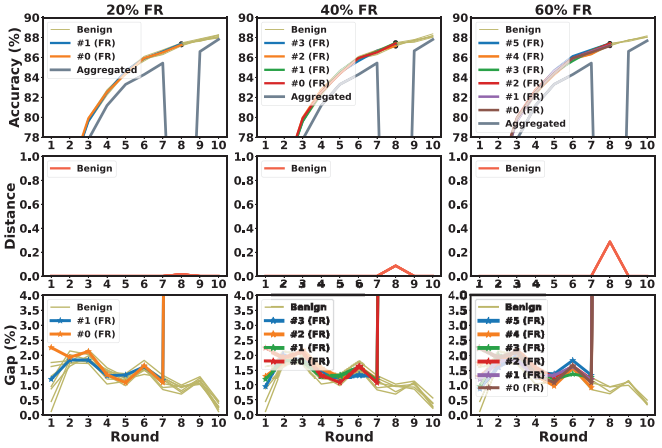
IV. Experiment and Evaluation

A. Experiment setup

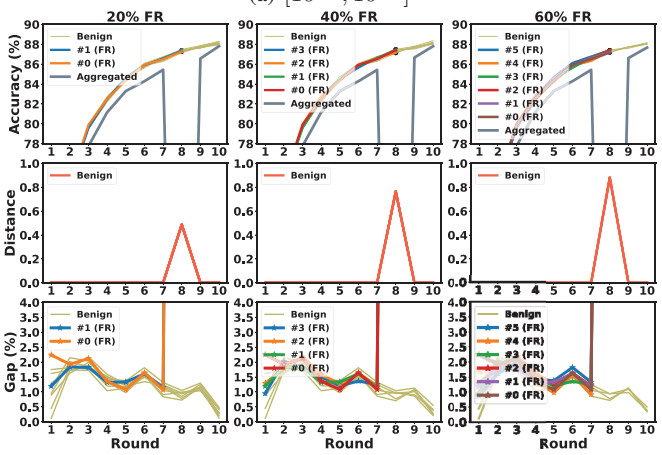
Data preparation: We used the MNIST [21] dataset to evaluate our proposed approach. The training set was split by a ratio of $0.85 : 0.15$ for the training and validation set, resulting in 51000 images for training and 9000 images for validation. The training data was then equally divided among 10 clients in the network, with each receiving 5100 images for their local training and 900 images for validation. We also shuffled the data with the same random seed value before dividing and distributing it to clients. Beyond the foregoing, the entire test dataset of 10000 images was used with a random of 5000 images for each testing time, making sure the test set is smaller than the training set of each client to provide unbiased evaluation.

Model training: Before the training, all images need to be transformed into tensors and normalized. Next, on each round, we trained the local model for 10 epochs using the Adadelta optimizer and a learning rate of 0.001. All experiment setups were conducted with 10 clients and 1 aggregator, running with 10 FedAvg aggregation rounds. The model is a simple convolutional neural network, containing over 62K parameters with 6 layers, including 2 convolutional layers, 2 dropouts, and 2 fully connected layers with softmax activation at the end.

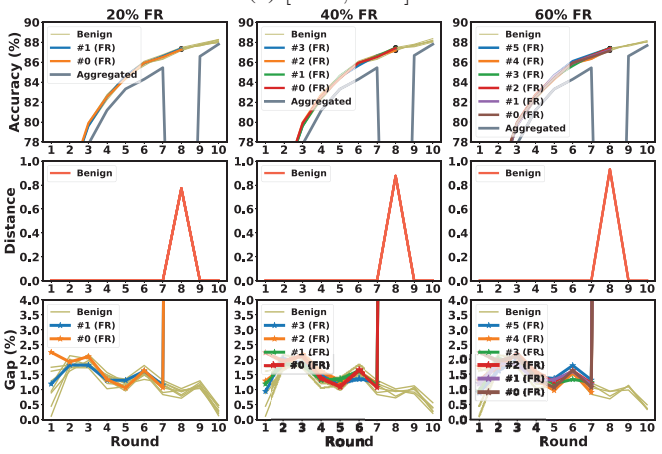
Experiment scenarios: To evaluate the versatility and effectiveness of our approach, we conducted experiments under two different scenarios of fake weights: random weights (w_{f_1}) and noise-added weights (w_{f_2}). Regarding w_{f_1}, we run a random function on free rider clients to randomize weight values drawn in the range of $[a, b]$ and send it to the aggregator. In most cases, a and b values are decided based on the previous weights to have better random results [3]. These random weights are formulated as $w_{f_1} = a + (b - a) \times rand()$. About noise-added cases (w_{f_2}), Fraboni et al. [4] showed that free riders aim to utilize the weights sent from the aggregator at each iteration and to evade detection, free riders may generate new weight values by adding noise $\mathcal{N}(\mu, \sigma^2)$ to the ones they obtained. Here μ denotes the mean (usually set to 0) and σ denotes the standard deviation in the Gaussian noise



(a) $[10^{-3}, 10^{-2}]$

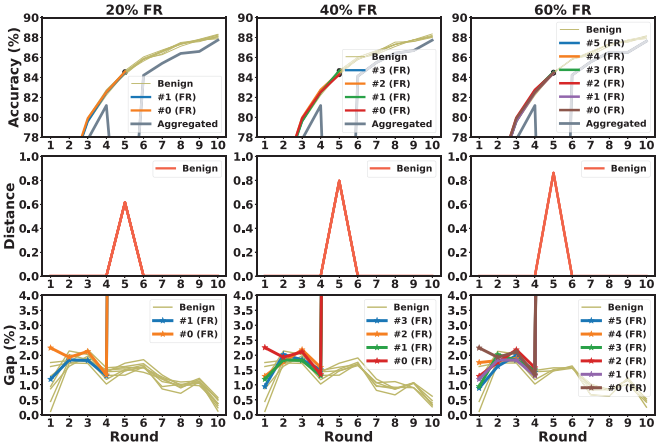


(b) $[10^{-2}, 10^{-1}]$

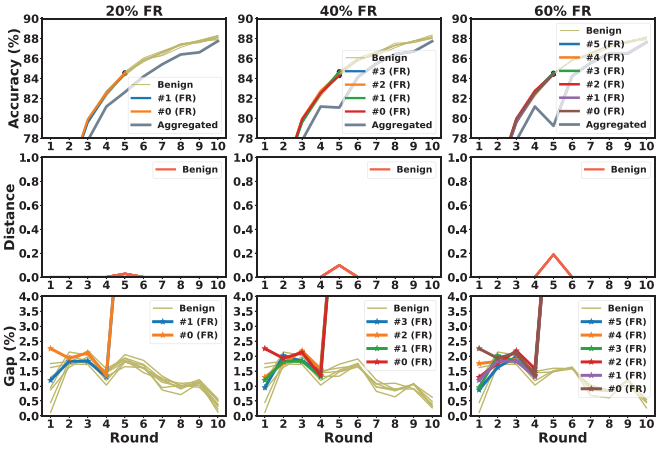


(c) $[min, max]$

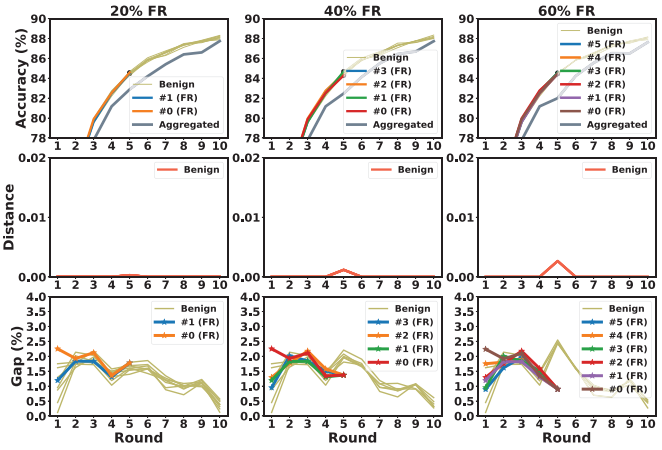
Fig. 3: Random scenario: accuracy reported from clients and aggregated model (upper), distance between clients' weights and aggregated weights (middle), and accuracy gap between aggregated model and clients' re-evaluated models (lower) across communication rounds



(a) $\sigma = 0.1$



(b) $\sigma = 0.01$



(c) $\sigma = 0.001$

Fig. 4: Noised scenario: accuracy reported from clients and aggregated model (upper), distance between clients' weights and aggregated weights (middle), and accuracy gap between aggregated model and clients' re-evaluated models (lower) across communication rounds

function. The modification process at iterator t is illustrated via formulated as $w_{f_2} = w_{agg_{(t-1)}} + \mathcal{N}(\mu, \sigma^2)$.

In more detail, we create noise with 4 different σ values (0, 0.001, 0.01, and 0.01) and $\mu = 0$ in Gaussian cases, and use different ranges of value to randomize the weight in the random cases ($[10^{-3}, 10^{-2}]$, $[10^{-2}, 10^{-1}]$, and [min, max]).

To clarify, in the [min, max] random scenario, we test our method with the fake weights generated in the range of the minimum and maximum values of the previous aggregated weights. We also varied the proportion of free riders in each scenario (20%, 40%, and 60% from the total number of training workers) to examine how our method performs when free riders dominate the system. To ensure the objectivity of our experiments in all setups, free riders initially pose as benign clients before revealing their cheating behavior at a randomly selected activation round. This is referred to the work of Xie et al. [22], who pointed out that a malicious party may pretend to join the FL system solely to obtain the distributed global model. The free rider mode was activated at round 8 for the random case and at round 5 for the two aggregated weight scenarios.

Evaluation metrics: For the performance assessment, we validate the efficacy of our method in abnormal detection and free rider identification ability, ensuring no missing cases or false alarms are called. In particular, we employ two checking conditions on the client side: the drop in the aggregated accuracy and the distance between the aggregated and client models. We present our findings through various visualizations and tables. Firstly, we plot the self-reported accuracy values from each client alongside the corresponding aggregated model's accuracy for each communication round (Figure 3 and 4). Additionally, we analyze the accuracy gap between the aggregated model and each client's re-evaluated model (Table IV), as well as the cosine distance between the client's model and the aggregated model during the investigation round (Table II). Next, we present Table III with detailed figures on the accuracy gap and the averaged gap values to provide a comprehensive analysis of the identification process. Finally, to emphasize the importance of client grouping in our investigation, we utilize a dendrogram to visualize the linkage clustering results based on clients' submission weights (Figure 5 and 6).

TABLE II: Cosine distance between benign clients' and the aggregated model

Weights	Params	Free riders ratio		
		20%	**40%**	**60%**
Randomized	$[10^{-3}, 10^{-2}]$	0.014559	0.08678	0.2873
	$[10^{-2}, 10^{-1}]$	0.4877	0.7648	0.88156
	[min, max]	0.7727	0.8765	0.9313
Noised	$\sigma = 0.1$	0.6154	0.7958	0.8620
	$\sigma = 0.01$	0.0277	0.0989	0.1893
	$\sigma = 0.001$	0.00031	0.0012	0.0027

B. Evaluations

1) Detecting the potential presence of free riders: In this evaluation, we examine the client's sensitivity to the received aggregated model during the learning process and rigorously assess the effectiveness of each checking condition employed by the clients to trigger an investigation.

As in Figure 3, we observe a significant drop in the accuracy of the aggregated model (upper plots) at round 8 across all three experimented ranges of the random scenario. Concurrently, the distance between clients' and aggregated weights (middle plots) also witnessed a surge at the activation round of cheating, surpassing the safe distance value by at least 1000 times (safe distance calculated from benign clients vs. aggregators is $1.434e^{-5}$ or 0.00001434). In particular, the smallest distance of 0.0146 is observed in the $[10^{-3}, 10^{-2}]$ range with 20% free riders, and this difference becomes more pronounced with higher percentages of free riders and across different ranges. For example, in the $[10^{-2}, 10^{-1}]$ and [min, max] ranges, the distances are 0.4877, 0.7648, 0.88156, and 0.7727, 0.8765, 0.9313 for 20%, 40%, and 60% free riders, respectively. However, it is worth noting that across all random cases in our experiments, the second condition of the cosine distance is not necessary, as the drop in the aggregated weights alone can capture the client's attention, effectively triggering an investigation.

TABLE III: Safe gap value - A_{gap} (%) and the averaged gap (%) of free riders model to the global one

Weights	Params	A_{gap}	Free riders ratio		
			20%	**40%**	**60%**
Randomized	$[10^{-3}, 10^{-2}]$	1.45	75.77	75.60	75.16
	$[10^{-2}, 10^{-1}]$	1.45	75.55	75.55	75.41
	[min, max]	1.45	76.09	75.62	75.28
Noised	$\sigma = 0.1$	1.57	67.86	67.86	67.86
	$\sigma = 0.01$	1.57	8.67	7.12	5.92
	$\sigma = 0.001$	1.57	1.78	1.37	0.9

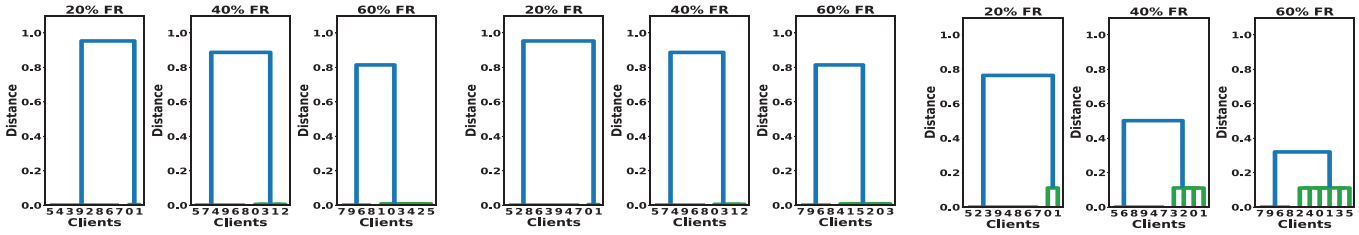
Regarding the Gaussian scenario, it is worth noting that the accuracy drop does not hold true for all cases. Specifically, while the first condition remains effective in noise cases with standard deviations of 0.01 and 0.1, we observe a challenge when using $\sigma = 0.001$, which generates sophisticated weights. In this case, the aggregated accuracy exhibits a gradual increase (see Figure 4), potentially misleading the client into considering it a normal training process. However, the analysis of distance values between the client and the returned aggregated weights can still spot the abnormality, especially in the case of 20% free riders, where the distance value was 0.00031, approximately 20 times higher than the safe distance. This difference also becomes more pronounced with higher ratios of free riders and larger standard deviations, as indicated in Table II. This can be explained by the higher standard deviation values creating more significant disparities between the disguised and original weights.

Finally, for the case of using direct weights, the initial check for weight copying conducted right after the aggregator receives the submission will help identify the occurrence of this cheating type at an early stage without requiring further investigation triggers, referred to Figure 2.

2) Identifying free riders: In the evaluation of the effectiveness of the log-based algorithm, our primary focus remains on the random and Gaussian scenarios, given that free riders who directly utilize aggregated weights have been eliminated through a straightforward check conducted at the aggregator before initiating any investigation. Our assessment involves a

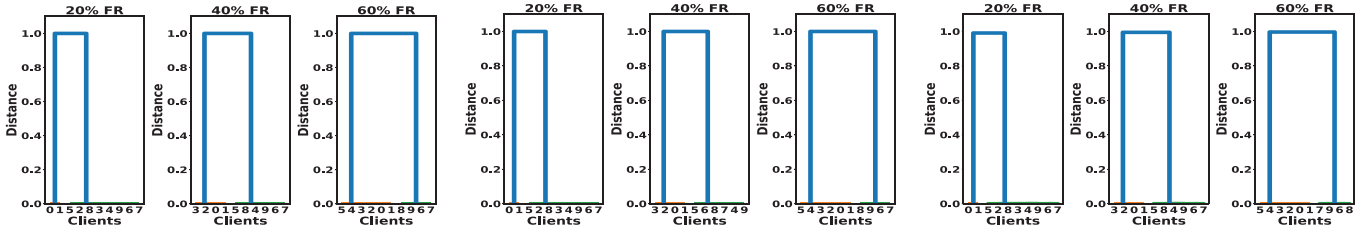
TABLE IV: Aggregated accuracy in previous round, current investigation round and the free rider's re-evaluated (real) accuracy

Weights	Params	Previous	Free riders ratio					
			20%		40%		60%	
			Current	Re-evaluated	Current	Re-evaluated	Current	Re-evaluated
Randomized	$[10^{-3}, 10^{-2}]$	0.8544	0.1079	0.0989	0.1008	0.0988	0.0977	0.1002
	$[10^{-2}, 10^{-1}]$	0.8544	0.4188	0.0966	0.1839	0.0984	0.1237	0.1027
	[min, max]	0.8544	0.1935	0.0936	0.1292	0.0982	0.101	0.1015
Noised	$\sigma = 0.1$	0.8118	0.5952	0.1332	0.3154	0.1332	0.1992	0.1332
	$\sigma = 0.01$	0.8118	0.8263	0.7396	0.8108	0.7396	0.7925	0.7396
	$\sigma = 0.001$	0.8118	0.8288	0.811	0.8247	0.811	0.82	0.811



(a) $[10^{-2}, 10^{-1}]$ (b) $[10^{-3}, 10^{-2}]$ (c) $[min, max]$

Fig. 5: Dendrogram of linkage clustering tree at investigation round - randomized weights. Orange and green lines represent intra-group distances among clients, while the blue line indicates the inter-group distance.



(a) $\mu = 10^{-1}$ (b) $\mu = 10^{-2}$ (c) $\mu = 10^{-3}$

Fig. 6: Dendrogram of linkage clustering tree at investigation round - noised weights. Orange and green lines represent intra-group distances among clients, while the blue line indicates the inter-group distance.

detailed analysis of the IsSafe function's condition, examining its performance and reliability in each specific case.

In our experiments of both random and Gaussian scenarios, the benign clients demonstrate a consistently low average gap between the aggregated and re-evaluated accuracy, which does not exceed 1.6% (refer to Table III). However, the measured gap in the two scenarios was observed with differences.

Specifically, Figure 3 illustrates that the sharp drop in accuracy of the aggregated model (upper) coincides with a rapid surge in the accuracy gap value (lower) for free riders, occurring between the round when the free-rider mode is activated and the previous round. These gap values reached over 75 or 76 in all conducted experiments of random scenarios. A similar trend is observed in the Gaussian cases with σ values of 0.1 and 0.001, yet with a less steep accuracy drop and smaller accuracy gap, as detailed in Table III. Remarkably, the noise weights with σ being 0.0001, as previously analyzed in the detection evaluation, exhibit an increase in the cheating activation round (Figure 4). Consequently, the gap values become extremely low (1.78 for 20% free riders) or even fall below the safe value (e.g., 1.37 and 0.9 for 40% and 60% free riders, respectively). As a result, we can see that the first gap condition in the IsSafe function appears to be effective only for detecting cases with 20% of free riders or less in the Gaussian case. However, by also incorporating the second condition, which considers the relationship between the aggregated accuracy of the previous and current rounds, along with the re-evaluated accuracy, our method was able to accurately identify free riders in the rest two cases with smaller accuracy compared to both previous and current rounds' values. Detailed figures are reported in Table IV. Finally, at round 8 in the random scenario and round 5 in the Gaussian scenario, the lines representing free riders in all plots come to an abrupt end, indicating that the free riders, who had fabricated their accuracy reports and were successfully detected and exposed through our investigation process.

The dendrograms in Figure 5 and Figure 6 provide a visual representation of the distances between clients using the distance matrix obtained from the second part of the investigation. Since the hierarchical clustering treats each data

point as a separate cluster and iteratively merges them based on their distances to get one cluster at the end, clients displayed with the same colors in the dendrogram will have a higher possibility of being in the same group. Additionally, the height of the line segment implies the possibility of merging these clusters, with higher values indicating larger distances between them and lower values indicating smaller distances. The results observed in Figure 5 and Figure 6 demonstrate the exactness of the clustering method in all scenarios by sharply grouping free riders and benign clients. We can observe the difference between clients in the same group (orange or green) with relatively small heights in the dendrogram, while the blue line indicates a significant distance between the benign and free rider groups. This enables us to confidently skip the entire group checking after the first positive re-evaluation without the risk of missing any free riders in that group. More intriguingly, the dendrograms reveal that while the randomized weights can be easily detected via our investigation with the accuracy gaps, the distance between client clusters in those cases is varied according to the change in the free riders ratio. Particularly, the higher the ratio, the smaller the distance, which means the clients become more similar to each other. In contrast, the Gaussian scenario causes challenges in detecting differences in accuracy, but a clear distinction between the groups of benign and cheating players can be seen with a distance of 1 for all experimental cases, showing that the weights are clearly different among the benign clients and free riders.

V. Conclusion

This study emphasizes ensuring fairness and trust in the FL system, dealing with the presence of free riders. In detail, our contribution encompasses a novel monetary incentive mechanism that ensures clients receive optimal rewards proportional to their contribution, thus fostering active participation. We detailed the joining rules with stakes and analyzed the rewards for various cases considering free rider presence. This provides a more explicit approach for calculating the incentive of each client, compared to other prior works. Concurrently, we propose a lightweight yet effective approach to detect and identify free riders in an FL system, where the detection task is delegated to FL training workers while the aggregator identifies the possible free riders. This approach aims to reduce the bottleneck and unnecessary tasks at the central agent, the aggregator. The effectiveness of our proposed method was demonstrated through empirical experiments involving different free rider ratios and settings in both randomized and noise-added scenarios.

Acknowledgment

This research is supported by the Research Council of Finland through the 6G Flagship program (Grant 318927), and by Business Finland through the Neural pub/sub research project (diary number 8754/31/2022).

References

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*, pp. 1273–1282, PMLR, 2017.

[2] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, jan 2019.

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

[4] Y. Fraboni, R. Vidal, and M. Lorenzi, "Free-rider attacks on model aggregation in federated learning," in *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics* (A. Banerjee and K. Fukumizu, eds.), vol. 130 of *Proceedings of Machine Learning Research*, pp. 1846–1854, PMLR, 13–15 Apr 2021.

[5] L. Lyu, H. Yu, J. Zhao, and Q. Yang, *Threats to Federated Learning*, pp. 3–16. Cham: Springer International Publishing, 2020.

[6] E. Tahanian, M. Amouei, H. Fateh, and M. Rezvani, "A game-theoretic approach for robust federated learning," *International Journal of Engineering*, vol. 34, no. 4, pp. 832–842, 2021.

[7] J. Weng, J. Weng, H. Huang, C. Cai, and C. Wang, "Fedserving: A federated prediction serving framework based on incentive mechanism," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pp. 1–10, 2021.

[8] J. Wang, X. Chang, R. J. Rodrìguez, and Y. Wang, "Assessing anonymous and selfish free-rider attacks in federated learning," in *2022 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–6, IEEE, 2022.

[9] J. Chen, M. Li, T. Liu, H. Zheng, H. Du, and Y. Cheng, "Rethinking the defense against free-rider attack from the perspective of model weight evolving frequency," *Information Sciences*, vol. 668, p. 120527, 2024.

[10] X. Tu, K. Zhu, N. C. Luong, D. Niyato, Y. Zhang, and J. Li, "Incentive mechanisms for federated learning: From economic and game theoretic perspective," *IEEE Transactions on Cognitive Communications and Networking*, vol. 8, no. 3, pp. 1566–1593, 2022.

[11] J. F. Nash, "Equilibrium points in n-person games," *Proceedings of the National Academy of Sciences*, vol. 36, no. 1, pp. 48–49, 1950.

[12] M. Tang and V. W. Wong, "An incentive mechanism for cross-silo federated learning: A public goods perspective," in *IEEE INFOCOM 2021 - IEEE Conference on Computer Communications*, pp. 1–10, 2021.

[13] H. Huang, B. Zhang, Y. Sun, C. Ma, and J. Qu, "Delta-dagmm: a free rider attack detection model in horizontal federated learning," *Security and Communication Networks*, vol. 2022, 2022.

[14] X. Xu and L. Lyu, "A reputation mechanism is all you need: Collaborative fairness and adversarial robustness in federated learning," *arXiv preprint arXiv:2011.10464*, pp. 1–13, 2020.

[15] H. Lv, Z. Zheng, T. Luo, F. Wu, S. Tang, L. Hua, R. Jia, and C. Lv, "Data-free evaluation of user contributions in federated learning," in *2021 19th International Symposium on Modeling and Optimization in Mobile, Ad hoc, and Wireless Networks (WiOpt)*, pp. 1–8, 2021.

[16] R. Jurca and B. Faltings, "Minimum payments that reward honest reputation feedback," in *Proceedings of the 7th ACM Conference on Electronic Commerce*, pp. 190–199, 2006.

[17] C. Aperjis and B. A. Huberman, "A market for unbiased private data: Paying individuals according to their privacy attitudes," *arXiv preprint arXiv:1205.0030*, 2012.

[18] N. N. Taleb, *Skin in the game: Hidden asymmetries in daily life*. Random House, 2018.

[19] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE signal processing magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[20] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[21] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE signal processing magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[22] X. Xie, C. Hu, H. Ren, and J. Deng, "A survey on vulnerability of federated learning: A learning algorithm perspective," *Neurocomputing*, p. 127225, 2024.