Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.075457

ARTICLE

Gradient Feature-Based Collaborative Filtering in Verification Federated Learning with Privacy-Preserving

Chen Yu, Jingjing Tan, Wenwu Zhao and Ke Gu*

School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha, China

*Corresponding Author: Ke Gu. Email: gk4572@163.com

Received: 01 November 2025; Accepted: 22 January 2026; Published: 12 March 2026

ABSTRACT: Although federated learning (FL) improves privacy-preserving by updating parameters without collecting original user data, their shared gradients still leak sensitive user information. Existing differential privacy and encryption techniques typically focus on whether the aggregated gradient is correctly processed and verified only, rather than whether each user is honestly trained locally. To address these above issues, we propose a gradient feature-based collaborative filtering scheme in verification federated learning, where the authenticity of user training is verified using the collaborative filtering (CF) method based on gradient features. Compared with single user gradient detection (such as similarity detection of gradient median), our collaborative filtering scheme can provide more comprehensive and efficient user gradient detection by gradient dimensionality reduction. Also, user gradient security is protected by dynamically generating a mask matrix, and the verifiability of the aggregation result is realized by combining dynamic masks. Finally, we perform comprehensive comparisons and experiments by using CNN models on some classical datasets. Experimental results and analysis show that our scheme outperforms other state-of-the-art schemes, demonstrating the effectiveness of our scheme.

KEYWORDS: Federated learning; free-riders client detection; machine learning

1 Introduction

1.1 Background

Although one of the key advantages of FL is the ability to train models without exchanging original data, thereby preserving user privacy, gradients themselves can also leak sensitive information. Gradients contain information about the training data, and attackers might use this information to infer the original data, meaning that even without directly exchanging data, the transmission of gradients could potentially reveal specific user information [1]. In addition to direct data leakage, attackers can also launch model attacks using gradients, such as membership inference attacks or attribute inference attacks [2,3]. To address the issues mentioned above, some techniques like differential privacy [4,5] and homomorphic encryption [6] have been introduced into federated learning to prevent the leakage of user information through model gradient parameters.

On the other hand, some clients involved in the federated learning process may not be entirely trustworthy. Certain clients might adopt a "free-riding" strategy, benefiting from the model improvements achieved through the data and computational resources contributed by other participants, while failing to make any meaningful contributions to the collaborative training process. Specifically, such free-riding clients might submit inaccurate, irrelevant, or fabricated model gradients, yet still utilize the final trained

model for prediction or decision-making once it is publicly released [7,8]. Some malicious clients could deliberately submit erroneous or tampered gradients to disrupt the training process, potentially resulting in a degradation of model performance or even complete failure [9,10]. Additionally, in some scenarios, malicious servers might return incorrect results to users. For instance, "lazy" servers could compress the original model with a simplified and less accurate alternative to minimize their computational overhead, or they might maliciously alter the aggregated results before transmitting them to users [11]. To mitigate the aforementioned issues, various techniques have been employed to verify the correctness of aggregated gradients, including Lagrange interpolation [12,13], cryptographic signatures [14], commitments [15], and zero-knowledge proofs [16,17]. However, these methods only focus on whether the aggregated gradient is correctly processed and verified, rather than whether each user has honestly trained locally, and it is still possible for users to adopt a "free-riding" strategy or engage in malicious behaviors by submitting forged gradients without being directly detected. To further clarify the distinctive contributions of our proposed scheme, we provide a structured comparison with verifiable FL frameworks: FlexibleFL [18], SVeriFL [19] and SVFLC [20] in Table 1. As summarized in the table, while existing works primarily focus on ensuring that the server performs honest aggregation, our work shifts the emphasis toward verifying the integrity of the local training process itself. By employing JL-projection for dimensionality reduction and collaborative filtering for similarity-based detection, our scheme achieves more granular detection of malicious behaviors that are often overlooked by purely cryptographic verification methods.

Table 1: Comparison of the proposed method.

Scheme	Detection Dimension	Verification Objective	Global Optimality
SVeriFL	Aggregated gradient	Aggregation correctness	Summation
SVFLC	Chain-based audit	Result integrity	Chain-based
FlexibleFL	Individual updates	Poisoning defense	Heuristic filtering
Ours	Individual features	Local authenticity	Adaptive filtering

1.2 Contribution

To address the above issues, we propose a gradient feature-based collaborative filtering (CF) [21,22] scheme in verificatiCon federated learning to ensure the effectiveness and security of distributed model training. In our scheme, the storage-assisted server (SAS) is used to monitor whether the user and aggregation server (AS) are honestly implementing the training protocol to ensure that the entities participating in model training are complying with their protocol during training. Our contribution can be summarized as follows:

- We introduce a collaborative filtering (CF) mechanism based on gradient features for verifiable federated learning. Unlike prior approaches relying solely on direct gradient similarity, our method applies JL-type gradient dimensionality reduction to construct a unified, structure-preserving behavior space, enabling CF to capture multi-user behavioral patterns more effectively. This combination of CF with projected gradient representations provides a richer and more robust basis for detecting abnormal or dishonest updates.
- To detect potentially malicious behavior from users or the aggregation server, we design a monitoring mechanism centered on the SAS, which evaluates the consistency between masked gradients and aggregated gradients. The SAS leverages the projected behavioral features uploaded by clients to monitor the execution of the protocol and allows honest entities to verify the correctness of aggregation without exposing raw gradients.

- To demonstrate the effectiveness of our scheme, we perform a detailed security analysis showing resilience against integrity-related threats. We further evaluate its practical utility through comprehensive experiments on two real-world datasets, confirming the robustness and efficiency of our framework under diverse settings.

1.3 Organization

The remainder of the paper is organized as follows. In Section 2, we provide an overview of our proposed system model. In Section 3, we elaborate on our gradient feature-based collaborative filtering scheme. Experimental comparisons and analysis are conducted in Section 4. Section 5 discusses limitations and future research directions. Finally, Section 6 summarizes this paper.

2 System Overview

2.1 System Model

The proposed system model is including three key entities: trusted authority, users involved in the training and servers. The servers are further categorized into two types: the aggregation server and the storage-assisted server. The detailed functions of these three entities are defined as follows:

Trusted Authority (TA): TA is responsible for generating and distributing the parameters required for each training round. In the training process, TA handles the removal of masks from the global gradient after unmasking, and it then generates and distributes the updated global gradient along with the corresponding mask matrix for the next training round. By calculating the sum of all users' masks, TA is able to remove the mask from the aggregated result, thereby ensuring the accurate aggregation of these gradients. TA is fully trusted and is responsible for generating the random projection matrix R, per-round dynamic mask matrices, and cryptographic parameters. TA does not access any gradients, masked gradients, or verification materials.

AS and SAS: AS is responsible for receiving the local gradients from each user and aggregating the local gradients. Also, SAS is responsible for supervising the above aggregation process, including verifying whether the AS is honest about the aggregation results and whether the users are honest about their training. The SAS also informs those entities of its detection results. AS may be malicious and can alter, drop, or forge user updates, but it receives only masked gradients and never obtains R or mask components. SAS is honest-but-curious; it only receives projected gradients and projected masks for verification but does not participate in aggregation or hold any secret keys.

Users: Users are training participants in the federated learning process, that can possess local training data. They actively request federated learning training tasks, train their local models for each round using their local data, and send their masked local models to the AS for model aggregation.

We assume that no two entities collude, and neither AS nor SAS receives a complete mask vector. This non-collusion model ensures that no single server can recover user gradients or simultaneously manipulate both aggregation and verification paths.

2.2 Security Concerns and System Objective

Under the proposed system model, we assume that a TA is completely trustworthy and does not collude with any user or server (similar settings can be found in the paper [23]). The AS is considered to be malicious and endeavors to collude with other users to gain access to their original models, thereby inferring their original data and compromising their private information. To alleviate the workload, the AS may selectively aggregate only a subset of models or maliciously forge the aggregation results. Subsequently, these incorrect aggregated outputs are provided to the trainers. Also, the SAS is characterized by both honesty and curiosity:

it adheres to executing all protocol steps meticulously, while also being inquisitive about the data provided by other users. Further, users may employ the "free-riding" tactics, leveraging historical local gradients to alleviate their computational burden, but we assume the overall training system is composed of more users who engage in honest training than those who resort to free-riding. Additionally, some users might endeavor to deduce the private information of others. Therefore, our security objective is to maintain the effective oversight of user training throughout the federated learning process and ensure the stringent protection of user security. We acknowledge that the assumption of a fully trusted TA and an honest-but-curious SAS may not always hold in all real-world deployments. However, these assumptions are widely adopted in verifiable and privacy-preserving FL frameworks, and enable our scheme to focus on robustness and integrity without introducing additional cryptographic interaction overhead.

2.3 Design Goal

To clearly separate the protection goals addressed by our framework, we distinguish three security objectives: robustness, integrity, and privacy, and map each objective to the system component responsible for achieving it.

(1) Robustness: SAS is responsible for robustness-related verification. It receives only projected gradient features and projected masks, and performs collaborative-filtering–based analysis to determine whether a user behaves as a free-rider or uploads forged updates. The verification relies on checking temporal consistency of gradients, similarity among peers in the projected space, and the coherence between each projected gradient and its corresponding projected dynamic mask. These operations jointly enable robustness against dishonest users without exposing any raw gradient information.

(2) Integrity: AS aggregates masked gradients but does not participate in verification. Users and SAS independently generate verification materials based on the dynamic mask scheme. This separation ensures that a malicious AS cannot forge or manipulate aggregation results without being detected by the SAS, thereby guaranteeing the integrity of the global update.

(3) Privacy: TA distributes per-round dynamic mask matrices that hide user gradients from both AS and SAS. Random projection reduces the dimensionality of gradient representations and limits the amount of detailed structure exposed to the verifier, while approximately preserving distance relationships useful for similarity-based detection. We stress that projection is not a cryptographic privacy mechanism. Together, the masking and projection components ensure that user gradients remain unlinkable and unrecoverable across rounds.

Clarification on the role of random projection. We emphasize that random projection in our scheme is used for dimensionality reduction and to preserve pairwise similarity structure in a compact form for efficient collaborative-filtering based detection. It is not intended as a cryptographic privacy mechanism nor does it, by itself, provide formal privacy guarantees. The principal privacy protection is achieved by the per-round dynamic masking scheme; projection only reduces the exposed feature dimensionality and may empirically limit fine-grained leakage.

Scope clarification. We note that differential privacy (DP) is not included in our threat model. Our scheme operates under a masking-based privacy protection setting and focuses on verifying the correctness of client updates and detecting dishonest users. DP provides a separate layer of formal privacy guarantees through calibrated noise injection, which is orthogonal to the verification objective addressed in this work. Integrating DP as an additional privacy mechanism is possible, but is beyond the scope of this paper and is left for future work.

3 Proposed Scheme

As shown in Fig. 1, our gradient feature-based collaborative filtering scheme is divided into five phases, which includes system initialization, model training and upload, user gradient anomaly detection, gradient aggregation and aggregation result verification. Based on the proposed system model, our gradient feature-based collaborative filtering scheme with privacy-preserving is divided into five phases, which includes system initialization, model training and upload, user gradient anomaly detection, gradient aggregation and aggregation result verification. The detailed collaborative filtering process is shown in Fig. 2.

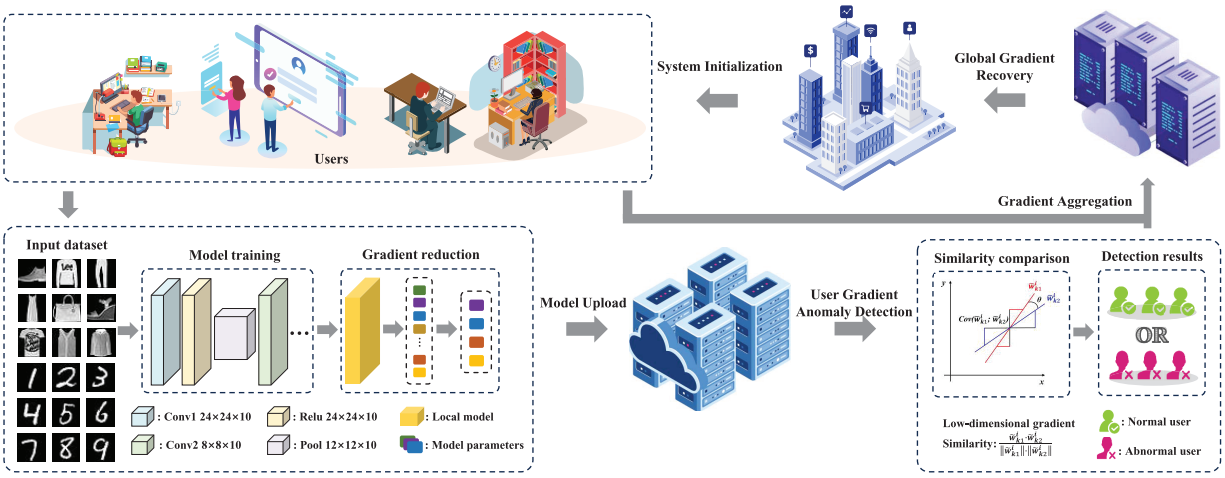


Figure 1: System model.

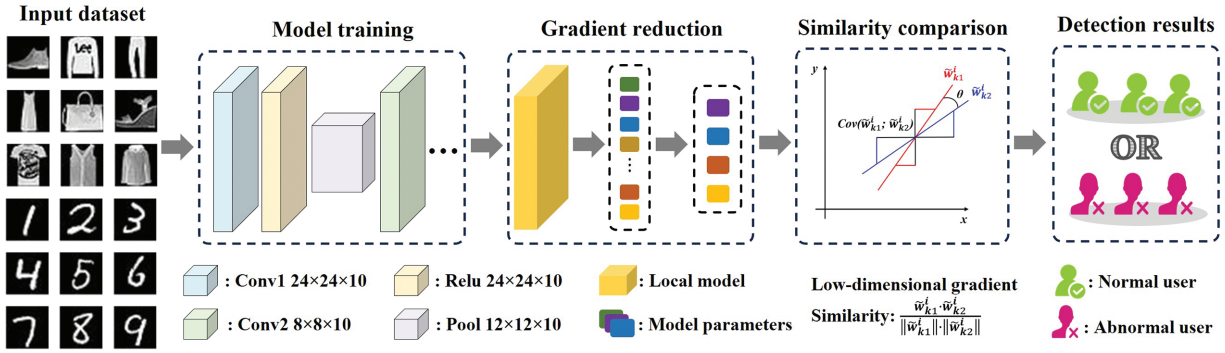


Figure 2: Collaborative filtering process.

3.1 System Initialization

In the system initialization phase, TA executes parameter configurations. The TA receives task requests and registration details from all participating users (assuming a total of n users), thereby forming a set of training users denoted as U. Subsequently, the TA randomly selects two servers with adequate computational resources to act as the aggregation server (AS) and the secure aggregation server (SAS) for the aggregation task, where we assume the TA can construct secure communication channels with all entities involved. Following this, the TA generates n random masks $(m_1^i, m_2^i, m_3^i, \ldots, m_n^i)$ and computes the total mask value locally as $M^i = \frac{1}{n}\sum_{k=1}^{n} m_k^i$. To ensure security, m_k^i and M^i are dynamically regenerated in each training round. Additionally, the TA generates a random projection matrix to map high-dimensional gradients into a

lower-dimensional feature space. This projection approximately preserves pairwise distances in expectation and reduces representation dimensionality for downstream detection. The TA also sets initial global model parameters w_g^i, along with other training parameters such as batch size B, learning rate η, local iteration count E, and global iteration count δ. Furthermore, the TA selects cyclic groups G_1 and G_2, where g and h are used as the bilinear pairing generators, and q represents the prime order of the finite field. The bilinear pairing map is defined as $e : G_1 \times G_2 \to G_I$. After initializing the parameters, the TA sends these initial parameters to each entity.

3.2 Model Training and Upload

After the TA completes the system initialization process, each user performs it local training using the global model parameters w_g^i received from the TA. The users calculate the loss function based on their own dataset and update their local model parameters to w_k^i:

$$w_k^{i+1} \leftarrow w_g^i - \eta \nabla L_f(D_k, w_g^i). \tag{1}$$

To ensure data security during the upload process, the users mask their local gradients $\hat{w}_k^i$ by adding the mask m_k^i provided by the TA:

$$\hat{w}_k^i = w_k^i + m_k^i. \tag{2}$$

Subsequently, each user computes the local verification value $verify_k^i$ based on the masked gradient and the mask:

$$verify_k^i = e(g^{w_k^i}, h^{m_k^i}) \mod q. \tag{3}$$

This verification value will be used to validate the honest operation of the AS. Additionally, each user maps (or reduces) its local gradients to a low-dimensional feature vector $\tilde{w}_k^i$ using the random matrix provided by the SAS for subsequent anomaly detection. And each user also computes the low-dimensional mask vector $\tilde{m}_k^i$ for the honesty upload detection phase:

$$\tilde{w}_k^i = R^T w_k^i. \tag{4}$$

Although SAS receives both the projected gradients $R^T(w_k^i + m_k^i)$ and the masks $R^T m_k^i$ gradient privacy remains preserved for two reasons. First, the JL-type projection with $r \ll d$ is intrinsically non-invertible; infinitely many original gradients correspond to the same low-dimensional projection. Second, the dynamic masks are refreshed every round, making projected values unlinkable across iterations. Consequently, SAS cannot reconstruct or approximate the original gradient structure beyond the coarse similarity information intentionally retained for collaborative filtering–based detection. We explicitly note that projection alone does not provide formal privacy guarantees, and such formal guarantees are outside the scope of this work. After completing these data calculations, each user sends the low-dimensional feature gradient $\tilde{w}_k^i$, the low-dimensional mask vector $\tilde{m}_k^i$, and the verification value $verify_k^i$ to the SAS. Additionally, each user sends the masked gradient $\hat{w}_k^i$ to the AS. The specific details of this phase are outlined in Algorithm 1. Inspired by the Johnson-Lindenstrauss Lemma [24], the following lemma holds.

Lemma 1: *For any two gradient vectors $w_{k1}^i, w_{k2}^i \in \mathbb{R}^d$, a random projection matrix $R \in \mathbb{R}^{d \times r}$ can approximately preserve their distance. The columns of R are random vectors with elements $R_{ij} \sim \mathcal{N}\left(0, \frac{1}{r}\right)$, independently and*

identically distributed. When R satisfies the independence and randomness conditions, the following holds:

$$(1-\varepsilon)\|w_{k1}^i - w_{k2}^i\|^2 \le \|R^T(w_{k1}^i - w_{k2}^i)\|^2 \le (1+\varepsilon)\|w_{k1}^i - w_{k2}^i\|^2 \tag{5}$$

where $\varepsilon \in (0,1)$ *depends on the projection dimension r and the number of points n, ensuring the projected distances are approximately preserved.*

Proof: For the gradient difference between users u_{k1} and u_{k2}:

$$x = w_{k1}^i - w_{k2}^i, \tag{6}$$

we need to analyze the properties of its projection $R^T x$. Since the columns of R are independent and identically distributed, with $R_{ij} \sim \mathcal{N}(0, \frac{1}{r})$, the squared distance after projection can be written as:

$$\|R^T x\|^2 = \sum_{k=1}^{r}\left(\sum_{j=1}^{d} R_{jk}x_j\right)^2. \tag{7}$$

For the expectation of $\|R^T x\|^2$, since $R_{ij} \sim \mathcal{N}(0, \frac{1}{r})$, a normal random variable with mean 0, its variance is:

$$\mathrm{Var}\left(\sum_{j=1}^{d} R_{jk}x_j\right) = \sum_{j=1}^{d} x_j^2 \cdot \frac{1}{r} = \frac{\|x\|^2}{r}. \tag{8}$$

Therefore, $\sum_{j=1}^{d} R_{jk}x_j \sim \mathcal{N}\left(0, \frac{\|x\|^2}{r}\right)$. Its square $\left(\sum_{j=1}^{d} R_{jk}x_j\right)^2$ is a scaled chi-square random variable. Specifically:

$$\left(\sum_{j=1}^{d} R_{jk}x_j\right)^2 \sim \frac{\|x\|^2}{r} \cdot \chi^2(1), \tag{9}$$

where $\chi^2(1)$ is a chi-square distribution with 1 degree of freedom, having expectation 1 and variance 2. Thus:

$$E\left[\left(\sum_{j=1}^{d} R_{jk}x_j\right)^2\right] = \frac{\|x\|^2}{r} \cdot 1 = \frac{\|x\|^2}{r}. \tag{10}$$

Summing over all r dimensions, we get:

$$E\left[\|R^T x\|^2\right] = \sum_{k=1}^{r} E\left[\left(\sum_{j=1}^{d} R_{jk}x_j\right)^2\right] = \sum_{k=1}^{r} \frac{\|x\|^2}{r} = \|x\|^2. \tag{11}$$

This indicates that the expectation of the squared distance after projection equals the squared original distance, meaning that random projection preserves the distance in expectation. Next, we calculate the variance of $\|R^T x\|^2$. Let $Z_k = \sum_{j=1}^{d} R_{jk}x_j$, then $\|R^T x\|^2 = \sum_{k=1}^{r} Z_k^2$. Since the columns of R are independent, the Z_k are also independent. Each $Z_k \sim \mathcal{N}\left(0, \frac{\|x\|^2}{r}\right)$, and its square $Z_k^2 \sim \frac{\|x\|^2}{r} \cdot \chi^2(1)$. The variance of the chi-square distribution $\chi^2(1)$ is 2, so:

$$\mathrm{Var}(Z_k^2) = \left(\frac{\|x\|^2}{r}\right)^2 \cdot 2. \tag{12}$$

The total variance is:

$$\mathrm{Var}\left(\|R^T x\|^2\right) = \sum_{k=1}^{r} \mathrm{Var}(Z_k^2) = r \cdot 2\left(\frac{\|x\|^2}{r}\right)^2 = \frac{2\|x\|^4}{r}. \tag{13}$$

The variance is inversely proportional to the projection dimension r, indicating that a larger r results in a smaller variance and thus a more concentrated projected distance. To quantify the approximation error, we use Chebyshev's inequality. Let $Y = \|R^T x\|^2$, with expectation $E[Y] = \|x\|^2$ and variance $\mathrm{Var}(Y) = \frac{2\|x\|^4}{r}$. According to Chebyshev's inequality:

$$P\left(|Y - E[Y]| \geq \epsilon\, E[Y]\right) \leq \frac{\mathrm{Var}(Y)}{(\epsilon\, E[Y])^2}. \tag{14}$$

Substituting the values:

$$P\left(|Y - \|x\|^2| \geq \epsilon\, \|x\|^2\right) \leq \frac{\frac{2\|x\|^4}{r}}{(\epsilon\, \|x\|^2)^2} = \frac{2}{r\,\epsilon^2}. \tag{15}$$

This shows that the probability that the projected distance deviates from the original distance by more than $\epsilon\ \|x\|^2$ is at most $\frac{2}{r\epsilon^2}$, i.e.,

$$P\left((1-\ \epsilon)\|x\|^2 \leq \|R^T x\|^2 \leq (1+\epsilon)\|x\|^2\right) \geq 1 - \frac{2}{r\ \epsilon^2}. \tag{16}$$

Therefore, with a sufficiently large projection dimension r, making $\frac{2}{r\,\epsilon^2}$ small, the projected distance will lie within $(1 \pm\ \epsilon)\|x\|^2$ with high probability. If we require this probability to be less than a small error probability δ, then:

$$\frac{2}{r\,\epsilon^2} \leq \delta \implies r \geq \frac{2}{\delta\,\epsilon^2}. \tag{17}$$

This indicates that to ensure the projected distance lies within $(1 \pm\ \epsilon)\|x\|^2$ with probability at least $1 - \delta$, the projection dimension r needs to satisfy $r = \Omega\left(\frac{1}{\delta\,\epsilon^2}\right)$. □

3.3 User Gradient Anomaly Detection

After the SAS receives the low-dimensional feature vectors $\tilde{w}_k^i$ uploaded by all the users, it calculates a collaborative filtering embedding feature vector E_k^i for each user:

$$E_k^i = \alpha \cdot \left(\tilde{w}_k^i - \tilde{w}_k^{i-1}\right) + \beta \cdot \left(\tilde{w}_k^i - \tilde{w}_{j^*}^i\right).$$

This vector encapsulates the changes in the user's local feature gradient relative to the historical gradients and the global gradient. Using these user collaborative filtering embedding feature vectors, the SAS constructs a user gradient behavior matrix H^i. Once the construction is complete, the SAS employs a random sampling method to compare each user's feature vector with randomly selected feature vectors from other users, calculating similarity scores to detect anomalies in user gradients. To identify whether users or the AS are fabricating gradient information or utilizing outdated historical gradient data for continued training, we propose a user gradient anomaly detection method based on the Johnson-Lindenstrauss Lemma and

collaborative filtering (CF) mechanism. In addition, we employ the Top-q Sign Fingerprint (TSF). For each user u_k, let $T_q(\tilde{w}_k^i)$ denote the index set of the top-q coordinates of $\tilde{w}_k^i$ ranked by absolute value:

$$T_q(\tilde{w}_k^i) = \{p_1, p_2, \ldots, p_q\}, \tag{18}$$

where each p_t is a coordinate index. The sign function for coordinate p is defined as

$$\text{sign}(\tilde{w}_{k,p}^i) = \begin{cases} +1, & \tilde{w}_{k,p}^i \geq 0, \\ -1, & \tilde{w}_{k,p}^i < 0. \end{cases} \tag{19}$$

The TSF similarity between users u_k and u_j is then given by:

$$\text{TSF}(\tilde{w}_k^i, \tilde{w}_j^i) = \frac{\left|\{p \in T_q(\tilde{w}_k^i) \cap T_q(\tilde{w}_j^i) : \text{sign}(\tilde{w}_{k,p}^i) = \text{sign}(\tilde{w}_{j,p}^i)\}\right|}{q}. \tag{20}$$

If $T_q(\tilde{w}_k^i) \cap T_q(\tilde{w}_j^i) = \varnothing$, then TSF = 0. When the intersection $T_q(\tilde{w}_k^i) \cap T_q(\tilde{w}_j^i)$ is empty or very small, TSF is defined as 0 to avoid undefined values. This value lies in $[0, 1]$, where higher values indicate a greater degree of sign consistency on the most important coordinates between two users. The most similar user of u_k is determined as:

$$j^* = \arg\max_{j \neq k} \text{TSF}(\tilde{w}_k^i, \tilde{w}_j^i). \tag{21}$$

The specific details of this phase are summarized in Algorithm 1. The schematic process of user abnormal behavior detection based on collaborative filtering is shown in Fig. 3. Based on this process, the following theorems hold:

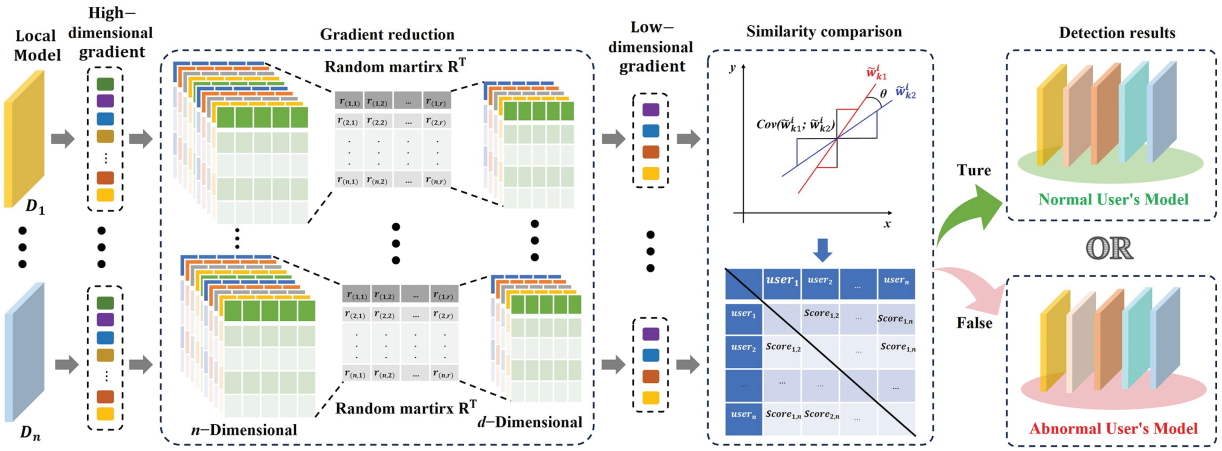


Figure 3: Johnson-Lindenstrauss Lemma schematic.

Algorithm 1: User gradient anomaly detection based on collaborative filtering and Johnson-Lindenstrauss lemma

Require: Low-dimensional feature vector $\tilde{w}_k^i$ of user u_k, global low-dimensional feature vector $\tilde{w}_g^i$ for the i-th round.

Ensure: User's similarity score S_{uk}.

1: SAS initializes a collaborative filtering embedding feature vector E_k^i for each user u_k:

$$E_k^i = \alpha \cdot \left(\tilde{w}_k^i - \tilde{w}_k^{i-1}\right) + \beta \cdot \left(\tilde{w}_k^i - \tilde{w}_{j^*}^i\right), \tag{22}$$

where $j^* = \arg\max_{j \neq k} \mathrm{TSF}(\tilde{w}_k^i, \tilde{w}_j^i)$ denotes the user most similar to u_k based on the Top-q Sign Fingerprint (TSF) similarity.

2: Combine all users' collaborative filtering embedding feature vectors into a user gradient behavior matrix H^i:

$$H^i = [E_1^i, E_2^i, E_3^i, \ldots, E_n^i]. \tag{23}$$

3: Randomly select the feature vectors of r other users to form a random sampling set:

$$Set_r = \text{Random Sampling } Set(H_{k \neq u}^i). \tag{24}$$

4: Compare the feature vector E_k^i of user u_k with the feature vectors of r randomly selected other users:

$$S_{uk} = \mathrm{similar}(E_{uk}^i, E_{ur}^i) = \frac{1}{r} \cdot \sum_{l=1}^{r} \mathrm{Cosine_Similarity}(E_{uk}^i, E_{ul}^i). \tag{25}$$

5: Determine user behavior according to the similarity score S_{uk} of a user u_k:

If $S_{uk} < S_{u_{other}}$, then user behavior is abnormal;

Otherwise, then user behavior is normal;

Theorem 1: *(Rationality analysis of embedded features and similarity score): In our scheme, the embedded feature vector E_k^i is expressed as follows:*

$$E_k^i = \alpha \cdot \left(\tilde{w}_k^i - \tilde{w}_k^{i-1}\right) + \beta \cdot \left(\tilde{w}_k^i - \tilde{w}_{j^*}^i\right),$$

which combines low-dimensional local gradient variation and peer difference derived from the most similar user, effectively characterizing user behavioral features. Additionally, the similarity score S_{uk} accurately captures anomalous behaviors, enabling user anomaly detection.

Proof: The embedded feature vector E_k^i consists of two components, each capturing a different behavioral dimension. The first component, the local gradient variation $\tilde{w}_k^i - \tilde{w}_k^{i-1}$, reflects the change in local gradients between consecutive rounds. For honest users, this difference tends to be smooth and consistent, with its distribution concentrated away from zero. However, if a user uploads outdated gradients, the term becomes zero, indicating stagnation. Conversely, fabricated gradients may cause sudden and irregular changes, leading to significant deviations. This component thus serves as an indicator of temporal continuity and behavioral stability across rounds. The second component, the peer difference $\tilde{w}_k^i - \tilde{w}_{j^*}^i$, measures how well a user's local update aligns with that of its most similar peer identified by the TSF similarity. Honest users typically share similar update directions with their close peers, resulting in small deviations. In contrast, malicious

or extremely deviating users tend to produce updates that diverge strongly from their peer's direction, even when their local change appears normal. This dual design ensures that any anomaly triggers detectable deviations in at least one component. The combined feature E_k^i thus simultaneously monitors temporal consistency and peer-level behavioral conformity. Cosine similarity is then used to evaluate directional alignment between feature vectors:

$$S_{uk} = \text{similar}(E_{uk}^i, E_{ur}^i) = \frac{1}{r} \cdot \sum_{l=1}^{r} \text{Cosine}_\text{S}\text{imilarity}(E_{uk}^i, E_{ul}^i) < S_{u_{other}}$$

where $S_{u_{other}}$ is the median similarity score of all users, the detection rule is defined as: if $S_{uk} < S_{u_{other}}$, u_k exhibits abnormal behavior. During the aggregation phase, gradients from user u_k are explicitly excluded based on our anomaly detection criteria. □

Theorem 2: *(Effectiveness of collaborative filtering in detecting user behavior matrix similarity differences): In our scheme, a user's honest training behavior can be monitored and evaluated based on its behavior score and the user gradient behavior matrix H^i:*

$$H^i = [E_1^i, E_2^i, E_3^i, \ldots, E_n^i],$$

where each embedded feature vector E_k^i integrates both local gradient variation and peer difference derived from the most similar user. By converting user behavior into numerical vectors and constructing the collaborative filtering matrix, the SAS can effectively monitor and analyze user training behavior patterns.

Proof: The behavior score S_{uk} of user u_k is computed by randomly selecting k users and comparing the cosine similarity of their low-dimensional embedded feature vectors. Let U_n denote the number of normal users and U_a the number of abnormal users, where $U_n + U_a = N$, and N is the total number of users. To ensure that the behavior scores of normal users remain higher than those of abnormal users, the number of normal users must exceed the number of abnormal users. Specifically, for a normal user, the number of its comparisons with other normal users should be greater than that with abnormal users. Thus, when at least $(\frac{k}{2}+1)$ normal users are compared, the minimum probability that a normal user's score is positive can be expressed as:

$$Probability_n = \left(\frac{U_n - 1}{N - 1}\right) \cdot \left(\frac{U_n - 2}{N - 2}\right) \cdots \left(\frac{U_n - \left(\frac{k}{2}+1\right)}{N - \left(\frac{k}{2}+1\right)}\right) = \prod_{i=0}^{\frac{k}{2}+1} \frac{U_n - i}{N - i}. \tag{26}$$

For an abnormal user whose score is positive, the probability is:

$$Probability_a = \left(\frac{U_a - 1}{N - 1}\right) \cdot \left(\frac{U_a - 2}{N - 2}\right) \cdots \left(\frac{U_a - \left(\frac{k}{2}-1\right)}{N - \left(\frac{k}{2}-1\right)}\right) = \prod_{i=0}^{\frac{k}{2}-1} \frac{U_a - i}{N - i}. \tag{27}$$

When $Probability_n > Probability_a$, the overall scoring mechanism can effectively distinguish between normal and abnormal users. Therefore, the collaborative filtering mechanism, by leveraging the similarity among user gradient behavior features derived from E_k^i, can reliably perform user anomaly detection. □

3.4 Gradient Aggregation

After the SAS completes the anomaly detection of user gradients, it needs to verify whether the user masking gradient received by the AS is consistent with the gradient source of the low-dimensional feature

gradient uploaded by each user and verified by the SAS. To achieve this, the SAS utilizes the previously uploaded low-dimensional feature gradient vector $\tilde{w}_k^i$ and the user low-dimensional feature mask vector $\tilde{m}_k^i$. Upon receiving the user masking gradient $\hat{m}_k^i$, the AS must calculate the low-dimensional feature of the masking gradient $R^T\tilde{w}_k^i$ and upload it to the SAS. Subsequently, the SAS calculates the sum of $\tilde{w}_k^i$ and $\tilde{m}_k^i$ and determines whether the $R^T\hat{w}_k^i$ and $(\tilde{w}_k^i + \tilde{m}_k^i)$ are equal:

$$R^T\hat{w}_k^i \stackrel{?}{=} \tilde{w}_k^i + \tilde{m}_k^i. \tag{28}$$

If the verification is passed, then it indicates that the user has honestly uploaded the masked gradient. Next, the AS collects and aggregates the masked local gradients $\hat{w}_k^i$ from each user:

$$\hat{w}_g^i = \frac{1}{n}\sum_{k=1}^{n}\hat{w}_k^i. \tag{29}$$

To ensure the honest execution of the protocol and successful completion of the aggregation operation, the AS must correctly process the data uploaded by users. Upon receiving the masked gradients calculated by users during local training, the AS performs the gradient aggregation operation to generate the masked global gradient $\hat{w}_g^i$. After the global gradient aggregation is completed, the AS needs to send the aggregated masked global gradient to the SAS. Since verification materials are independently generated by each user and transmitted directly to SAS, the AS cannot alter or suppress them. SAS verifies both the per-user update behavior and the final aggregated gradient using these user-provided commitments. Therefore, even if AS is malicious, it cannot forge a valid aggregation result without being detected by SAS, as the computation paths of aggregation and verification are completely separated.

Theorem 3: *(Detection correctness of user honesty upload): In our scheme, if the user honestly uploads its masking gradient and authentication information and the AS can honestly calculate the value of* $R^T\hat{w}_k^i$*:*

$$R^T\hat{w}_k^i \stackrel{?}{=} \tilde{w}_k^i + \tilde{m}_k^i,$$

then the correctness of the user uploading the masking gradient can be verified.

Proof: In the aggregation phase, it is necessary to verify whether the masked gradient $\hat{w}_k^i$ uploaded to the Aggregation Server (AS) and the low-dimensional feature gradient vector $\tilde{w}_k^i$ uploaded to the Secondary Authentication Server (SAS) originate from the same source gradient, namely the raw training gradient w_k^i. Since SAS stores both $\tilde{w}_k^i$ and $\tilde{m}_k^i$, which are computed by projecting the user's original gradient vector and the masking vector through a random matrix R, these values have previously been used to verify the user's honest participation in training. Simultaneously, users are required to upload the masked gradient $\hat{w}_k^i$ to AS for aggregation. To verify the consistency, AS can use the same random matrix R to project the masked local gradient $\hat{w}_k^i$ and compare it with the verification information uploaded to SAS. The consistency condition is:

$$R^T\hat{w}_k^i = R^T\left(w_k^i + m_k^i\right) \stackrel{?}{=} \tilde{w}_k^i + \tilde{m}_k^i,$$

where $\tilde{w}_k^i = R^T w_k^i$ and $\tilde{m}_k^i = R^T m_k^i$. If the equation holds, it proves that the user has honestly uploaded the correct masked gradient based on their genuine training process. However, if the computed value $R^T\hat{w}_k^i$ does not match $\tilde{w}_k^i + \tilde{m}_k^i$, it indicates a mismatch between the masked gradient uploaded to AS and the verification information uploaded to SAS. This inconsistency implies dishonest behavior by the user, and consequently, the masked gradient $\hat{w}_k^i$ will be excluded from the aggregation phase. □

3.5 Aggregation Result Verification Phase

To verify whether the AS has honestly executed the protocol and completed the aggregation operation, the SAS can verify the correctness of the aggregation performed by the AS using only the global masked gradient and the masked gradient verification values uploaded by each user. When the SAS receives the verification value $verify_k^i$ calculated by each user during local training and the masked global gradient aggregated by the AS, it proceeds with the verification of the aggregation result:

$$verify_{agg}^i = \prod_{k=1}^{n} e(g^{w_k^i}, h^{m_k^i}) \mod q. \tag{30}$$

Specifically, the SAS receives the masked global gradient $\hat{w}_g^i$ sent by the AS, SAS calculates the verification value $verify_{calculate}^i$ on the masked global gradient:

$$verify_{calculate}^i = e(g^{\hat{w}_g^i}, h) \mod q. \tag{31}$$

Then, SAS only needs to compare whether $verify_{agg}^i$ and $verify_{calculate}^i$ are the same to determine whether AS is integrity aggregation.

Theorem 4: *(Correctness of Verifying the Global Gradient Aggregation Result): In our proposed scheme, if the AS honestly executes the global aggregation operation and correctly computes the masked global gradient $\hat{w}_g^i$, then the verification value $verify_{calculate}^i$ computed by the SAS will match the aggregated verification value $verify_{agg}^i$ derived from all users:*

$$verify_{calculate}^i = verify_{agg}^i = \prod_{k=1}^{n} e(g^{w_k^i}, h^{m_k^i}) \bmod q. \tag{32}$$

Therefore, SAS can correctly determine whether the AS has honestly performed the aggregation. If the AS deviates from the protocol or returns an incorrect aggregation result, the above equality does not hold, and the verification consequently fails.

Proof: In the verification phase, our objective is to verify whether the following equation holds:

$$verify_{calculate}^i \stackrel{?}{=} verify_{agg}^i. \tag{33}$$

So, when the SAS calculates the verification value $verify_{calculate}^i$ on the masked global gradient and the aggregated verification value $verify_{agg}^i$ from all the users, we can obtain

$$\begin{aligned} verify_{agg}^i &= \prod_{k=1}^{n} verify_k^i \mod q \\ &= \prod_{k=1}^{n} e(g^{w_k^i}, h^{m_k^i}) \mod q. \end{aligned} \tag{34}$$

And $verify_{calculate}^i$ can be obtained through the following calculation:

$$\begin{aligned} verify_{calculate}^i &= e(g^{\hat{w}_g^i, h}) \mod q \\ &= e(g^{(\sum_{k=1}^{n} \hat{w}_k^i)}, h) \mod q \\ &= e(g^{(\sum_{k=1}^{n} (w_k^i + m_k^i)}, h) \mod q \end{aligned}$$

$$= \prod_{k=1}^{n} e(g^{w_k^i}, h^{m_k^i}) \mod q. \tag{35}$$

Therefore, the verification equation can be expressed in the following:

$$verify^i_{calculate} = verify^i_{agg} = \prod_{k=1}^{n} e(g^{w_k^i}, h^{m_k^i}) \mod q. \tag{36}$$

Then the SAS can use this equation to verify whether the AS has honestly completed the gradient aggregation; when the AS fails to aggregate the gradients, the equation does not hold. □

4 Experiment and Analysis

In this section, we perform a comprehensive performance evaluation and comparative analysis for our proposed scheme. The experiments are conducted on the computer running Windows 10, with the following hardware specifications: Intel(R) Core(TM) i7-9750H CPU @ 2.60 GHz, 20 GB RAM, and an NVIDIA GTX 1660 Ti with Max-Q Design (6 GB VRAM). The software environment includes Python 3.6.13 and PyTorch 0.4.1.

4.1 Experimental Setup

We show a detailed overview of our experimental setup, including the descriptions of test datasets, the types of data distribution and federated learning settings. The relevant details are described as follows:

1) Test Datasets: We conduct experiments by using two widely recognized datasets: MNIST and Fashion-MNIST.

- MNIST [25]: The MNIST dataset consists of 60,000 samples as the training set, each dataset being a 28 × 28 pixel grayscale image, and 10,000 samples as the test set, also 28 × 28 pixel grayscale images. Unlike the others, the MNIST consists of 70,000 handwritten digits from 0 to 9 written by different people.
- Fashion-MNIST [26]: This is a more challenging dataset as an alternative to MNIST, provided by Zalando Research. It contains 10 categories of fashion product images, totaling 70,000 grayscale images of 28 × 28 pixels.

2) Types of Data Distribution: For these evaluation experiments, we simulate two types of data distributions: Independent and Identically Distributed (IID) and Non-Independent and Identically Distributed (Non-IID). To evaluate the robustness of the algorithm under Non-IID scenarios, we adopt a shard-based user data partitioning method, which follows the commonly used label distribution skew simulation strategy in federated learning literature. Specifically, the dataset is divided into shards and randomly allocated to construct highly heterogeneous user data distributions. The partitioning process is as follows: for a dataset containing k classes, we first sort the data by labels and divide it into multiple shards, each containing a fixed number of samples. Since the data are sorted by labels, each shard tends to have a highly concentrated label distribution. Subsequently, multiple shards are randomly assigned to each client to ensure sufficient local data. This allocation strategy results in highly skewed data distributions on clients, typically covering only a small subset of classes, thereby simulating challenging Non-IID conditions in federated learning.

3) Federated Learning Settings: In the federated learning setting, we configure 100 participating trainers, with a local iteration count of 5, a batch size of 64, and a learning rate of 0.005. The cross-entropy loss function is employed, and the neural network architecture used is a CNN. During the training process,

30% free-riding abnormal users are introduced to evaluate the effectiveness of our proposed scheme, where abnormal users evade the training process by uploading untrained local gradients.

4) Parameter Settings: In our implementation, k denotes the number of important coordinates selected in TSF computation, r is the number of randomly sampled peers used for behavioral comparison, α and β are weighting coefficients for local and peer differences in E_k^i, and T is the decision threshold for anomaly detection. Unless otherwise specified, we set $k = 100$, $r = 5$, $\alpha = 0.5$, $\beta = 0.5$. The similarity threshold $S_{u_{other}}$ is adaptively set as the median of S_{uk} values. q denotes the number of top coordinates for TSF computation, $q = 0.1d$.

5) Attack Settings: To evaluate the effectiveness of the proposed anomaly detection mechanism, we simulate a free-riding attack scenario in which a portion of users never participate in real local training. Instead of uploading freshly computed gradients, these dishonest users continuously reuse and upload the initial global model arameters distributed by the trusted authority in the first round. Such behavior leads to static or near-zero local updates, disrupting the aggregation process and degrading the global model performance. The proportion of free-riding users is varied in the experiments to analyze the detection capability of the proposed scheme under different levels of dishonest articipation.

4.2 Experimental Results

We present the experimental results obtained by testing our scheme based on various environments. The details of these comparative experimental analysis are listed as follows:

1) Accuracy comparative analysis in IID and Non-IID scenarios: In this experiment, we utilize two public available datasets to evaluate the performance of various federated learning schemes in two application scenarios. The accuracy of our scheme is compared with those of FedAvg [27], FlexibleFL [18], FedProx [28], SVeriFL [19], and SVFLC [20]. Table 2 presents the comparative results of test accuracy. In the IID scenario, our scheme achieves an accuracy exceeding 75% on the Fashion-MNIST dataset and approximately 95% on the MNIST dataset. In the Non-IID scenario, as the Non-IID setup simulates real-world conditions where users possess their own datasets, the training conditions for each user vary across rounds. As a result, the test accuracy curves exhibit greater fluctuations and converges more slowly. This behavior can be attributed to significant data heterogeneity across users, leading to inconsistent update directions for the local models. Consequently, the global model must adapt to these variations during aggregation, resulting in large fluctuations in test results and inconsistent accuracy across rounds. Despite the inconsistency in training effects due to varying datasets and models, our scheme demonstrates strong performance under both IID and Non-IID conditions. In terms of model accuracy comparison, we can see that our scheme is better than other schemes, which can prove that our scheme is effective in excluding malicious users.

Table 2: Comparison of accuracy and other indicators.

Test Items	Methods	Fashion-MNIST		MNIST	
		IID	NonIID	IID	NonIID
Accuracy	FedAvg	65%	43%	86%	63%
	FlexibleFL	70%	52%	94%	80%
	SVeriFL	57%	46%	81%	53%
	FedProx	50%	40%	84%	63%
	SVFL	56%	41%	81%	61%
	Our Scheme	75% ↑	61% ↑	95% ↑	83% ↑

(Continued)

Table 2 (continued)

Test Items	Methods	Fashion-MNIST		MNIST	
		IID	NonIID	IID	NonIID
Recall	FedAvg	0.652	0.437	0.862	0.638
	FlexibleFL	0.7090	0.527	0.941	0.804
	SVeriFL	0.575	0.463	0.817	0.530
	FedProx	0.510	0.401	0.848	0.635
	SVFL	0.564	0.416	0.817	0.520
	Our Scheme	0.756 ↑	0.614 ↑	0.957 ↑	0.839 ↑
F1	FedAvg	0.621	0.350	0.860	0.563
	FlexibleFL	0.712	0.521	0.952	0.811
	SVeriFL	0.521	0.413	0.809	0.509
	FedProx	0.440	0.312	0.845	0.604
	SVFL	0.486	0.396	0.813	0.504
	Our Scheme	0.743 ↑	0.556 ↑	0.957 ↑	0.834 ↑

2) Loss comparative analysis in IID and Non-IID scenarios: In this experiment, we compare the loss value of our scheme with the FedAvg, FlexibleFL, FedProx, SVeriFL and SVFLC. Fig. 4 presents the experimental results of the loss value variation. As shown in Fig. 4, our scheme exhibits significant advantages under the IID scenario. Specifically, after 100 training rounds, the loss value of our scheme on the MNIST dataset is reduced to 0.46, while on the FashionMNIST dataset it decreases to 0.83. These results indicate that our scheme not only effectively reduces the initial loss but also maintains a consistent downward trend throughout the training process. Notably, our scheme is the only one that exhibits a continuous decrease in loss value across all 100 epochs and reaches a desirable convergence state over a longer timescale. Furthermore, the overall experimental results show that, whether under IID or Non-IID conditions, our scheme consistently outperforms several other federated learning schemes. The non-IID configuration yields lower training loss, the corresponding test accuracy is lower than that of the IID setting. This indicates a potential mismatch between the training and testing distributions. Specifically, due to the skewed label distribution in non-IID data, users tend to overfit to a limited set of labels seen during training. As a result, the global model may fail to generalize to the full label space present in the test set, which contains uniformly distributed classes.

3) Training of other test indicators: We use a table to specifically show the final training of the above-mentioned indicators and some other indicators. Table 2 shows the comprehensive results for each indicator. From the table, it is evident that our scheme has good performance across all evaluation metrics, particularly in key indicators such as Recall Score and F1 Score, outperforming other existing schemes. These results demonstrate that our scheme not only enhances model accuracy but also maintains strong performance under complex data distribution conditions. Although the fluctuations occur in the Non-IID scenario for all schemes, our scheme consistently outperforms other schemes throughout the training process and in the final stage. This further validates the effectiveness of our scheme. Additionally, by filtering out some users who upload abnormal gradients during aggregation, our scheme can optimize both the model training speed and the final performance.

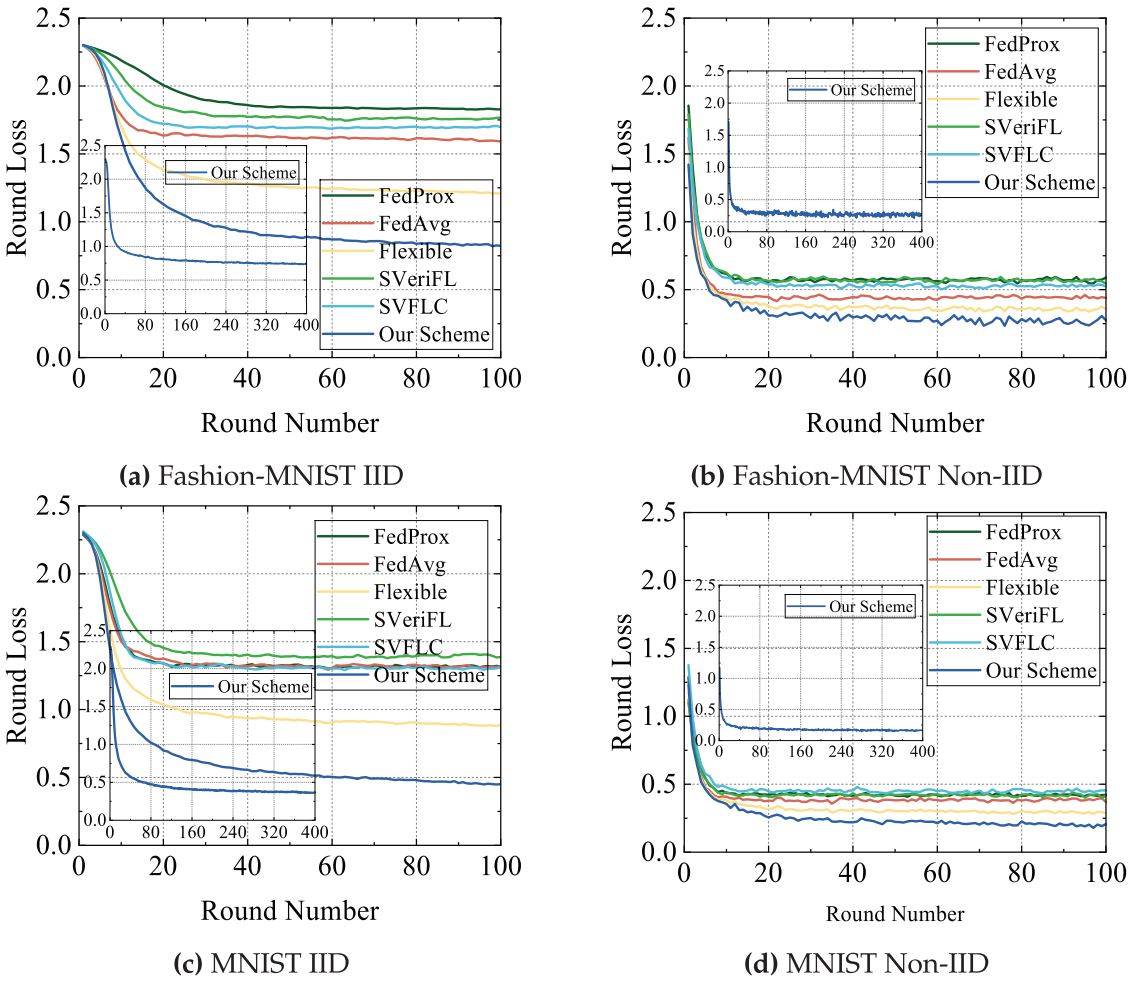


Figure 4: Comparison and analysis of loss on two datasets.

4) Performance comparison of different proportions of abnormal users: In this experiment, we compare the final model performance after 100 training rounds with 100 participants, varying the fraction of malicious users to 20%, 30%, and 40%. As shown in Figs. 5 and 6, with an increase in the number of abnormal users, all the schemes exhibit decreased test accuracy with slightly elevated final loss values. Notably, our scheme exhibits the least susceptibility to these adverse effects, underscoring its enhanced robustness in the exclusion of abnormal users. In contrast, other schemes suffer significant performance degradation due to their inability to effectively detect and manage anomalous user gradients. Benefiting from integrated anomaly detection mechanisms, our scheme is able to maintain superior performance metrics even as the proportion of anomalous users increases. Conversely, the performance of the other schemes regresses progressively, with the detrimental impact becoming increasingly severe as the proportion of abnormal users rises.

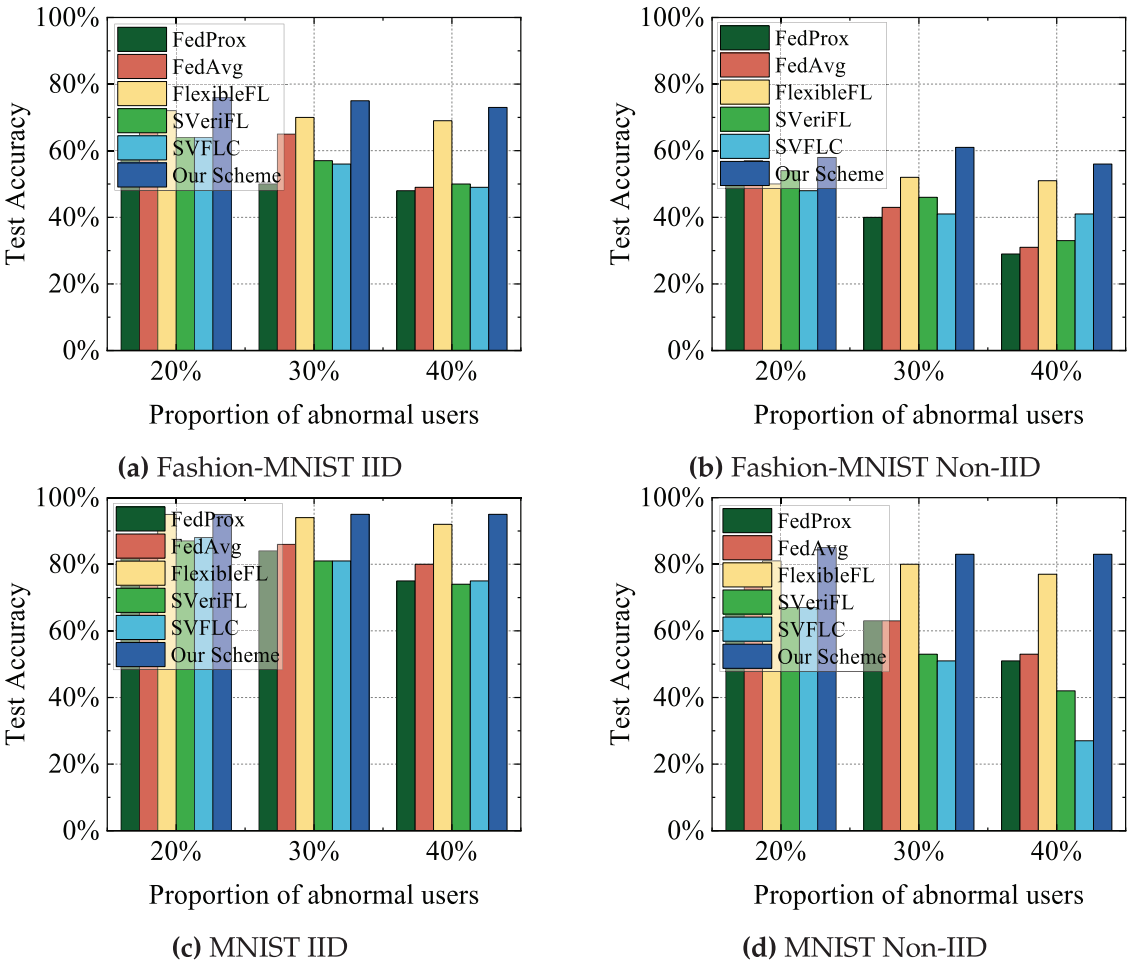


Figure 5: Comparison of accuracy under different proportions of abnormal users.

5) Compared with the free-riders detection schemes: In this experiment, we compare the final training results after 100 rounds of training with 100 users under different proportions of abnormal users, specifically with 20%, 30%, and 40% abnormal users. We compare the test accuracy and loss of our scheme with those of Krum [29], Median [30] in terms of robustness to aggregation accuracy. Fig. 7 shows the related experimental results. In the IID scenario, each user is assigned similarly distributed data, and the variance among benign updates remains relatively small. Under these homogeneous conditions, both Krum and Median achieve high accuracy when the proportion of abnormal users is low. This is because distance-based (Krum) and coordinate-wise (Median) aggregation methods can effectively identify and aggregate the majority of benign updates. However, as the proportion of abnormal users increases, Krum suffers from a sharp drop in accuracy and a noticeable increase in loss, reflecting reduced robustness. In contrast, our scheme consistently maintains high accuracy and stable loss. In the Non-IID scenario, users hold significantly different data distributions, resulting in greater variability among updates. Under such heterogeneous conditions, the performance of Krum and Median declines markedly. While accuracy declines and loss increases notably with a higher fraction of adversarial users, our scheme consistently achieves superior test accuracy and maintain low loss, highlighting its robustness and adaptability in varied environments.

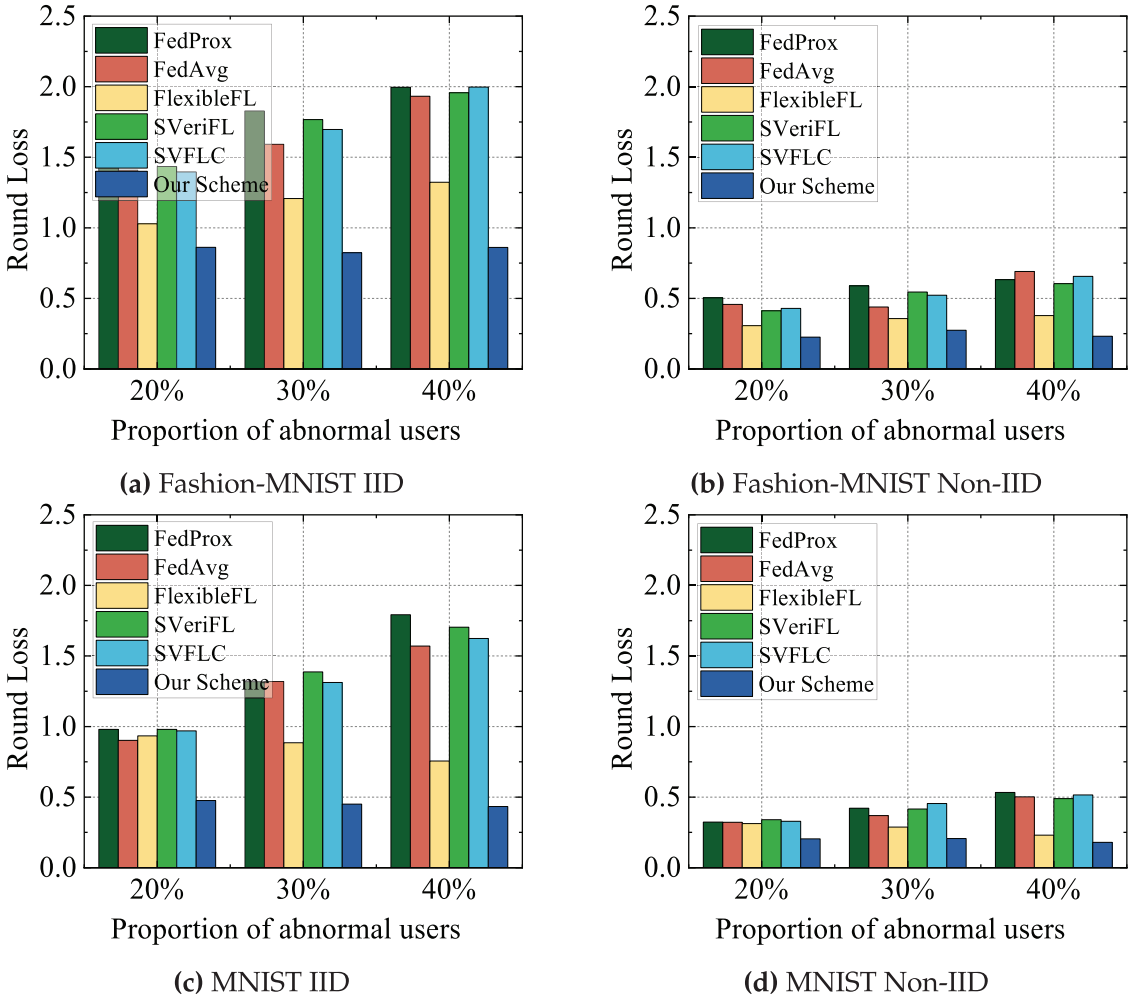


Figure 6: Comparison of loss under different proportions of abnormal users.

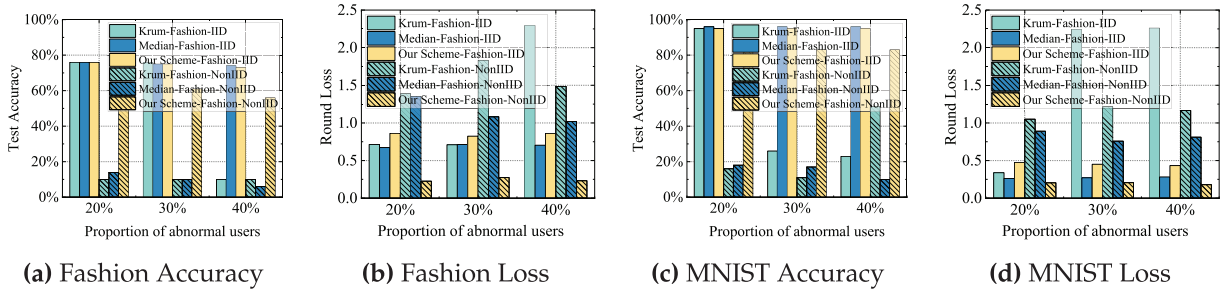


Figure 7: Comparison of test accuracy and loss among free-riders detection schemes across two datasets under both IID and Non-IID scenarios.

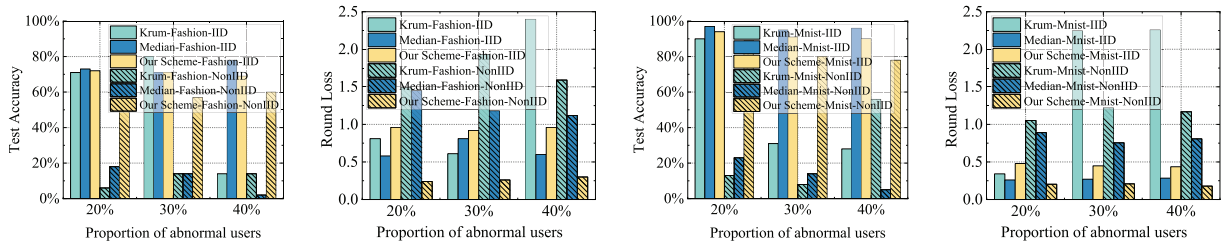
6) Ablation study results: The ablation results are shown in Table 3. The ablation experiments demonstrate that varying the weighting parameters α and β as well as the peer-sampling size r does not lead to substantial changes in performance; the resulting accuracy differences remain within a small range. This indicates that the proposed framework exhibits strong robustness and does not rely heavily on fine-grained

parameter tuning. Moreover, under all tested configurations, the detection accuracy and loss convergence remain within normal statistical variation, confirming that the method maintains stable behavior across different parameter settings.

Table 3: Comparison of accuracy under different parameter settings.

Test Items	Methods	Fashion-MNIST		MNIST	
		IID	NonIID	IID	NonIID
Accuracy	$\alpha = 0.5$ $\beta = 0.5$ $r = 0.1 \cdot d$	75%	61%	95%	83%
	$\alpha = 0.3$ $\beta = 0.7$ $r = 0.1 \cdot d$	70%	57%	91%	79%
	$\alpha = 0.7$ $\beta = 0.3$ $r = 0.1 \cdot d$	72%	58%	92%	80%
	$\alpha = 0.5$ $\beta = 0.5$ $r = 0.2 \cdot d$	71%	56%	90%	78%

7) Evaluation under label-flipping poisoning attacks: To examine whether the proposed CF-JL verification can distinguish poisoned updates from benign non-IID behavior, we further evaluate the system under the label-flipping attack, where malicious users invert labels during training. We compare our scheme with Median and Krum, two widely used Byzantine-robust aggregation baselines. The results show that all three methods are able to suppress the degradation caused by poisoned users. Our scheme achieves detection accuracy and global-model robustness comparable to these baselines. This indicates that the gradient-feature representation used in the collaborative-filtering verification still captures structural inconsistencies induced by poisoning, even when the attack is embedded in a seemingly "honest" training process. These findings also confirm that our approach can function as an effective verification component for mitigating harmful updates and can be combined with robust aggregation algorithms to further strengthen FL security under stronger poisoning threat models, as illustrated in Fig. 8.



(a) Fashion Accuracy **(b)** Fashion Loss **(c)** MNIST Accuracy **(d)** MNIST Loss

Figure 8: Comparison of test accuracy and loss among poisoning detection schemes across two datasets under both IID and Non-IID scenarios.

8) Runtime and complexity analysis: The verification overhead remains small relative to the overall training time. As shown in Table 4, under 100 users and 100 training rounds, the combined cost of verification and similarity analysis accounts for only a minor portion of the total runtime, indicating that the proposed scheme is efficient and adds only limited latency. The results confirm that the collaborative-filtering-based verification remains time-effective and scales well with the number of users.

Table 4: Runtime overhead under 100 users and 100 training rounds.

Component	Total Runtime (s)
Training time (100 users × 100 rounds)	85.69
Verification time (bilinear-pairing based check)	7.34
Similarity computation (collaborative filtering)	27.06
Total additional overhead ratio	40.2% of training time

5 Future Work

Several promising directions remain for future investigation. While this paper focuses on representative free-riding behaviors and label-flipping poisoning attacks, the emergence of more sophisticated adaptive adversaries deserves further attention. In particular, attackers that deliberately mimic benign clients pose a greater challenge to existing similarity-based verification mechanisms. Extending the proposed framework to address such adaptive threats by incorporating richer temporal constraints or cross-round behavioral consistency analysis is a priority for our future research. And the present evaluation primarily considers CNN-based image classification tasks on MNIST and Fashion-MNIST. Although the proposed verification mechanism is model-agnostic and operates on gradient-level representations, further validation on larger-scale datasets (e.g., CIFAR-10/100), deeper models, and non-vision modalities such as text classification would provide a more comprehensive assessment of its scalability and generality. Investigating the performance-efficiency trade-offs in these complex settings remains a critical objective for future work.

6 Conclusion

In this paper, we propose a gradient feature-based collaborative filtering scheme in verification federated learning, where the authenticity of user training is verified using the CF method based on gradient features and user gradient security is protected by dynamically generating a mask matrix. Further, to show the effectiveness of our scheme, we conduct an in-depth security analysis to demonstrate its resilience against some malicious attacks. Subsequently, we assess its practicality by comparing its performance with several state-of-the-art schemes on the MNIST and Fashion-MNIST datasets.

Acknowledgement: Not applicable.

Funding Statement: This work was supported by the National Natural Science Foundation of China (No. 62572077), Open Research Fund of Hunan Provincial Key Laboratory of Intelligent Processing of Big Data on Transportation (No. B202403), and Hunan Provincial Social Science Achievement Evaluation Committee (No. XSP25YBC686).

Author Contributions: The authors confirm contribution to the paper as follows: Study conception and design: Chen Yu, Ke Gu; data collection: Chen Yu; analysis and interpretation of results: Chen Yu, Ke Gu, Jingjing Tan; draft manuscript preparation: Chen Yu, Ke Gu, Wenwu Zhao. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The datasets used in this paper are respectively in references [25,26].

Ethics Approval: Not applicable.

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

References

1. Zhu L, Han S. Deep leakage from gradients. In: Federated learning: Privacy and incentive. Cham, Switzerland: Springer; 2020. p. 17–31. doi:10.1007/978-3-030-63076-8_2.
2. Shokri R, Stronati M, Song C, Shmatikov V. Membership inference attacks against machine learning models. In: 2017 IEEE Symposium on Security and Privacy (SP). Piscataway, NJ, USA: IEEE; 2017. p. 3–18.
3. Wang Z, Huang Y, Song M, Wu L, Xue F, Ren K. Poisoning-assisted property inference attack against federated learning. IEEE Trans Dependable Secure Comput. 2023;20(4):3328–40. doi:10.1109/tdsc.2022.3196646.
4. Sun L, Lyu L. Federated model distillation with noise-free differential privacy. arXiv:2009.05537. 2020.
5. Wei K, Li J, Ding M, Ma C, Yang HH, Farokhi F, et al. Federated learning with differential privacy: algorithms and performance analysis. IEEE Trans Inf Forensics Secur. 2020;15:3454–69.
6. Phong LT, Aono Y, Hayashi T, Wang L, Moriai S. Privacy-preserving deep learning via additively homomorphic encryption. IEEE Trans Inf Forensics Secur. 2018;13(5):1333–45. doi:10.1109/tifs.2017.2787987.
7. Fraboni Y, Vidal R, Lorenzi M. Free-rider attacks on model aggregation in federated learning. In: International Conference on Artificial Intelligence and Statistics. London, UK: PMLR; 2021. p. 1846–54.
8. Lin J, Du M, Liu J. Free-riders in federated learning: attacks and defenses. arXiv:1911.12560. 2019.
9. Fang M, Cao X, Jia J, Gong N. Local model poisoning attacks to byzantine-robust federated learning. In: 29th USENIX Security Symposium (USENIX Security 20); 2020 Aug 12–14; Boston, MA, USA. p. 1605–22.
10. Zhang J, Chen B, Cheng X, Binh HTT, Yu S. PoisonGAN: Generative poisoning attacks against federated learning in edge computing systems. IEEE Internet Things J. 2021;8(5):3310–22. doi:10.1109/jiot.2020.3023126.
11. Xu G, Li H, Liu S, Yang K, Lin X. VerifyNet: secure and verifiable federated learning. IEEE Trans InfForensics Secur. 2020;15:911–26.
12. Wang G, Zhou L, Li Q, Yan X, Liu X, Wu Y. FVFL: a flexible and verifiable privacy-preserving federated learning scheme. IEEE Internet Things J. 2024;11(13):23268–81.
13. Fu A, Zhang X, Xiong N, Gao Y, Wang H, Zhang J. VFL: a verifiable federated learning with privacy-preserving for big data in industrial IoT. IEEE Trans Ind Inform. 2022;18(5):3316–26.
14. Xia Y, Liu Y, Dong S, Li M, Guo C. SVCA: secure and verifiable chained aggregation for privacy-preserving federated learning. IEEE Internet Things J. 2024;11(10):18351–65.
15. Guo X, Liu Z, Li J, Gao J, Hou B, Dong C, et al. VeriFL: communication-efficient and fast verifiable aggregation for federated learning. IEEE Trans Inf Forensics Secur. 2021;16:1736–51.
16. Wang H, Guo Y, Bie R, Jia X. Verifiable arbitrary queries with zero knowledge confidentiality in decentralized storage. IEEE Trans Inf Forensics Secur. 2024;19:1071–85. doi:10.1109/tifs.2023.3330305.
17. Du R, Li X, He D, Choo KKR. Toward secure and verifiable hybrid federated learning. IEEE Trans Inf Forensics Secur. 2024;19:2935–50. doi:10.1109/tifs.2024.3357288.
18. Zhao Y, Cao Y, Zhang J, Huang H, Liu Y. FlexibleFL: mitigating poisoning attacks with contributions in cloud-edge federated learning systems. Inf Sci. 2024;664:120350.
19. Gao H, He N, Gao T. SVeriFL: successive verifiable federated learning with privacy-preserving. Inf Sci. 2022;622:98–114.
20. Li N, Zhou M, Yu H, Chen Y, Yang Z. SVFLC: secure and verifiable federated learning with chain aggregation. IEEE Internet Things J. 2024;11(8):13125–36.
21. Ammad-ud-din M, Ivannikova E, Khan SA, Oyomno W, Fu Q, Tan KE, et al. Federated rollaborative filtering for privacy-preserving personalized recommendation system. arXiv:1901.09888. 2019.
22. Perifanis V, Efraimidis PS. Federated neural collaborative filtering. Knowl Based Syst. 2022;242:108441. doi:10.1016/j.knosys.2022.108441.
23. Gao S, Luo J, Zhu J, Dong X, Shi W. VCD-FL: verifiable, collusion-resistant, and dynamic federated learning. IEEE Trans Inf Forensics Secur. 2023;18:3760–73.

24. Johnson WB, Lindenstrauss J. Extensions of Lipschitz mappings into Hilbert space. Contemp Math. 1984;26:189–206. doi:10.1090/conm/026/737400.
25. LeCun Y, Cortes C. The mnist database of handwritten digits; 2005 [cited 2025 Dec 15]. Available from: https://api.semanticscholar.org/CorpusID:60282629.
26. Xiao H, Rasul K, Vollgraf R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. arXiv:1708.07747. 2017.
27. McMahan B, Moore E, Ramage D, Hampson S, Arcas BAY. Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS) 2017; 2017 Apr 20–22; Fort Lauderdale, FL, USA. 2017. p. 1273–82.
28. Li T, Sahu AK, Zaheer M, Sanjabi M, Talwalkar A, Smith V. Federated optimization in heterogeneous networks. Proc Mach Learn Syst. 2020;2:429–50.
29. Blanchard P, El Mhamdi EM, Guerraoui R, Stainer J. Machine learning with adversaries: byzantine tolerant gradient descent. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S et al., editors. Advances in neural information processing systems. Vol. 30. Red Hook, NY, USA: Curran Associates, Inc.; 2017. p. 1–11.
30. Yin D, Chen Y, Kannan R, Bartlett P. Byzantine-robust distributed learning: towards optimal statistical rates. In: Dy J, Krause A, editors. Proceedings of the 35th International Conference on Machine Learning. Vol. 80. London, UK: PMLR; 2018. p. 5650–9.