FedACT: A Byzantine-Resilient Federated Learning Framework with Autoencoder-Committee-TLBO for Secure Collaborative Credit Scoring

Dengjia Li[a,c], Chaoqun Ma[a,c], Jinglan Yang[a,c] and Yuncheng Qiao[b,c,*]

[a]Business School, Hunan University, Changsha 410082, China
[b]Business School, Shandong University of Technology, Zibo 255000, China
[c]Research Institute of Digital Society and Blockchain, Hunan University, China

ARTICLE INFO

Keywords:
Federated learning
Byzantine attack
Credit scoring
Anomaly detection
Autoencoder
TLBO optimization

ABSTRACT

Federated learning enables collaborative credit scoring across financial institutions without sharing raw data, yet remains vulnerable to Byzantine attacks where malicious clients submit poisoned gradients to corrupt the global model. Existing Byzantine-resilient aggregation methods often exhibit limited effectiveness against sophisticated attacks or suffer from high false positive rates under heterogeneous data settings. This paper proposes FedACT (Federated Autoencoder-Committee-TLBO), a novel defense framework that integrates three complementary mechanisms: (1) an adaptive autoencoder-based gradient anomaly detector that learns the distribution of benign gradients and identifies anomalies through reconstruction error; (2) a diversity-aware committee voting mechanism that provides secondary verification for uncertain cases; and (3) a Teaching-Learning-Based Optimization (TLBO) aggregation strategy that robustly combines trusted gradients. Additionally, we incorporate a Merkle tree-based evidence chain for audit traceability and a reputation-based incentive mechanism for dynamic weight adjustment. Extensive experiments on two credit scoring datasets under 12 attack types and 4 heterogeneity scenarios demonstrate that FedACT achieves superior defense performance with detection precision exceeding 95%, significantly outperforming 7 state-of-the-art defense methods.

1. Introduction

Credit scoring plays a fundamental role in modern financial systems, enabling lenders to assess the creditworthiness of borrowers and make informed lending decisions [1]. With the rapid development of financial technology, financial institutions have accumulated vast amounts of user data that could potentially improve credit scoring accuracy. However, privacy regulations such as the General Data Protection Regulation (GDPR) and increasing concerns about data security prohibit direct data sharing among institutions [2]. This creates a significant challenge: how to leverage distributed data across multiple financial institutions to build better credit scoring models while preserving data privacy.

Federated learning (FL) has emerged as a promising solution to this challenge [3]. In FL, multiple clients (e.g., banks, fintech companies) collaboratively train a shared model by exchanging model updates (gradients) rather than raw data, with a central server coordinating the aggregation process [4]. This paradigm enables privacy-preserving collaborative modeling while keeping sensitive credit data localized at each institution [5].

However, the distributed nature of FL introduces significant security vulnerabilities. In particular, *Byzantine attacks* pose a severe threat where malicious clients can submit arbitrary or carefully crafted poisoned gradients to corrupt the global model [6]. In the context of credit scoring, such attacks could lead to models that systematically approve fraudulent loan applications or reject legitimate ones, causing substantial financial losses and reputational damage.

Byzantine attacks in FL have evolved from simple gradient manipulation to sophisticated strategies designed to evade detection. Early attacks such as sign-flip and Gaussian noise injection are relatively easy to detect [7]. However, advanced attacks like ALIE (A Little Is Enough) [8], IPM (Inner Product Manipulation) [9], and MinMax attacks [10] are specifically designed to remain statistically indistinguishable from benign gradients while still causing model degradation.

Existing Byzantine-resilient aggregation methods fall into two main categories: *robust statistics-based methods* and *trust-based methods*. Robust statistics-based methods, such as Median [11], Trimmed Mean [11], Krum [6], and Bulyan [12], aggregate gradients using robust statistical estimators. However, these methods often assume that benign gradients are independently and identically distributed (IID), which rarely holds in practice. Trust-based methods, such as FLTrust [13], rely on a root dataset at the server to bootstrap trust, but this assumption may not be feasible in cross-institutional scenarios.

The challenges are further compounded by *data heterogeneity*. Different financial institutions serve diverse customer populations with varying characteristics, leading to non-IID data distributions across clients [14]. This heterogeneity makes it difficult to distinguish between gradients that appear anomalous due to malicious manipulation versus those that simply reflect legitimate distribution differences.

*Corresponding author
qiaoyc@hnu.edu.cn (Y. Qiao)

To address these challenges, we propose **FedACT** (**Fed**erated **A**utoencoder-**C**ommittee-**T**LBO), a comprehensive Byzantine-resilient federated learning framework. FedACT introduces three novel defense mechanisms:

1. **Adaptive Autoencoder-based Anomaly Detection**: A neural autoencoder learns to reconstruct benign gradients, using reconstruction error combined with latent space distance as an anomaly score.
2. **Diversity-aware Committee Voting**: For gradients with uncertain anomaly scores, a committee voting mechanism provides secondary verification through consensus-based decision making.
3. **TLBO-based Robust Aggregation**: The Teaching-Learning-Based Optimization algorithm [15] iteratively refines aggregated gradients through teacher-learner dynamics.

Additionally, FedACT incorporates a Merkle tree-based evidence chain for immutable audit trails and a reputation-based incentive mechanism that dynamically adjusts client weights.

The main contributions of this paper are:

- We propose FedACT, a novel Byzantine-resilient federated learning framework integrating autoencoder-based detection, committee voting, and TLBO optimization.
- We design an adaptive threshold strategy with configurable coefficients that balances detection sensitivity and specificity.
- We develop a comprehensive experimental framework covering 12 attack types, 4 heterogeneity scenarios, and systematic ablation studies.
- Extensive experiments demonstrate that FedACT achieves detection precision exceeding 95% and minimal accuracy degradation, significantly outperforming existing methods.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 provides preliminaries. Section 4 presents the FedACT framework. Section 5 provides convergence analysis. Section 6 presents experiments. Section 7 concludes the paper.

2. Related Work

2.1. Federated Learning for Credit Scoring

Federated learning has attracted significant attention in the financial domain due to its privacy-preserving properties [4, 2]. Yang et al. [5] proposed an explainable federated learning method for credit scoring that combines blockchain for secure model parameter sharing and SHAP values for interpretability. Qiao et al. [16] developed a privacy-preserving credit evaluation system using Hyperledger Fabric blockchain.

However, existing federated credit scoring methods largely overlook the security vulnerabilities introduced by Byzantine attacks. While some works incorporate blockchain for trustworthy data sharing, they do not address the fundamental challenge of poisoned gradient updates from malicious participants. Our work fills this gap by proposing a comprehensive Byzantine defense framework specifically designed for federated credit scoring.

2.2. Byzantine Attacks in Federated Learning

Byzantine attacks in FL can be categorized into *untargeted attacks* that degrade overall model performance and *targeted attacks* (backdoor attacks) that cause misclassification on specific inputs [14].

Untargeted Attacks: Early attacks include simple gradient manipulations such as sign-flip, Gaussian noise injection, and scaling attacks [7]. More sophisticated attacks exploit knowledge of defense mechanisms. Baruch et al. [8] proposed the ALIE attack that generates malicious gradients statistically indistinguishable from benign ones. Xie et al. [9] developed the IPM attack that manipulates inner products. Shejwalkar and Houmansadr [10] proposed the MinMax attack that maximizes distance from benign gradients while staying undetected.

Targeted Attacks: Bagdasaryan et al. [17] demonstrated backdoor attacks where malicious clients inject hidden triggers. Wang et al. [18] proposed attack strategies balancing backdoor effectiveness and stealth.

2.3. Byzantine-Resilient Aggregation Methods

Robust Statistics-based Methods: Blanchard et al. [6] proposed Krum, which selects the gradient with minimum sum of distances to nearest neighbors. El-Mhamdi et al. [12] extended Krum to Multi-Krum and proposed Bulyan. Yin et al. [11] analyzed coordinate-wise median and trimmed mean. Pillutla et al. [19] proposed robust federated averaging using geometric median.

Trust-based Methods: Cao et al. [13] proposed FLTrust, which uses a small root dataset at the server to compute a reference gradient. However, FLTrust requires the server to possess representative data, which may not be available in cross-institutional scenarios.

Limitations: Most existing methods assume IID data distribution across clients, which is unrealistic in credit scoring. Under non-IID settings, benign gradients can be highly diverse, causing robust aggregation methods to reject legitimate updates. FedACT addresses this through adaptive autoencoder-based detection and committee voting.

3. Preliminaries

3.1. Federated Learning

Consider a federated learning system with N clients and a central server. Each client i has a local dataset $\mathcal{D}_i$ with n_i samples. The goal is to collaboratively minimize a global

Table 1
Comparison of our work with previous works.

Paper	Byzantine Defense	Anomaly Detection	Committee Voting	Optimization Aggregation	Heterogeneity Handling
Krum [6]	✓				
Bulyan [12]	✓				
FLTrust [13]	✓				
Median [11]	✓				
Trimmed Mean [11]	✓				
RFA [19]	✓				
FedACT (Ours)	✓	✓	✓	✓	✓

objective:

$$\min_{\theta} F(\theta) = \sum_{i=1}^{N} \frac{n_i}{n} F_i(\theta), \tag{1}$$

where $F_i(\theta) = \frac{1}{n_i} \sum_{(x,y)\in\mathcal{D}_i} \ell(\theta; x, y)$ is the local objective of client i, $n = \sum_{i=1}^{N} n_i$ is the total number of samples, and ℓ is the loss function.

In the standard FedAvg algorithm [3], each round t proceeds as follows:

1. The server broadcasts the current global model $\theta^{(t)}$ to selected clients.
2. Each client i performs local training and computes the gradient:
$$g_i^{(t)} = \theta^{(t)} - \theta_i^{(t+1)}, \tag{2}$$
where $\theta_i^{(t+1)}$ is the locally updated model.
3. The server aggregates gradients:
$$\theta^{(t+1)} = \theta^{(t)} - \eta \sum_{i=1}^{N} \frac{n_i}{n} g_i^{(t)}. \tag{3}$$

3.2. Byzantine Attack Model

In a Byzantine attack setting, a subset $\mathcal{M} \subset \{1, \ldots, N\}$ of clients are malicious. A Byzantine client $i \in \mathcal{M}$ can send an arbitrary gradient $\tilde{g}_i^{(t)}$ instead of the true gradient $g_i^{(t)}$. The fraction of Byzantine clients is $f = |\mathcal{M}|/N$.

We consider 12 attack types organized into three categories:

Basic Attacks:

- **Sign-flip**: $\tilde{g}_i = -g_i$
- **Gaussian**: $\tilde{g}_i = g_i + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$
- **Scale**: $\tilde{g}_i = -s \cdot g_i$, where $s > 0$ is the scale factor

Advanced Attacks:

- **Little** [8]: $\tilde{g}_i = g_i + \epsilon \cdot \text{sign}(g_i)$
- **ALIE** [8]: $\tilde{g}_i = \mu_g - z \cdot \sigma_g$, where μ_g, σ_g are mean and std of benign gradients
- **IPM** [9]: $\tilde{g}_i = -s \cdot \frac{\mu_g}{\|\mu_g\|} \cdot \|g_i\|$
- **MinMax** [10]: Maximizes distance from benign mean within detection bounds
- **Trim Attack**: Targets trimmed mean defense boundaries

Other Attacks:

- **Label-flip**: Flips a portion of gradient dimensions
- **Backdoor**: Embeds trigger patterns in gradients
- **Free-rider**: Sends near-zero gradients
- **Collision**: Multiple attackers send identical malicious gradients

3.3. Data Heterogeneity

We consider four heterogeneity scenarios:

- **IID**: Data is randomly shuffled and evenly distributed.
- **Label Skew**: Each client has a biased distribution over classes using Dirichlet distribution.
- **Quantity Skew**: Clients have varying amounts of data following power law.
- **Feature Skew**: Feature distributions differ across clients.

4. The FedACT Framework

This section presents the detailed design of FedACT. Figure 1 illustrates the overall architecture.

4.1. System Overview

FedACT operates in the following stages during each federated learning round:

1. **Gradient Collection**: The server collects gradients from participating clients.
2. **Anomaly Detection**: An autoencoder-based detector computes anomaly scores.
3. **Classification**: Gradients are classified as normal, anomalous, or uncertain based on adaptive thresholds.

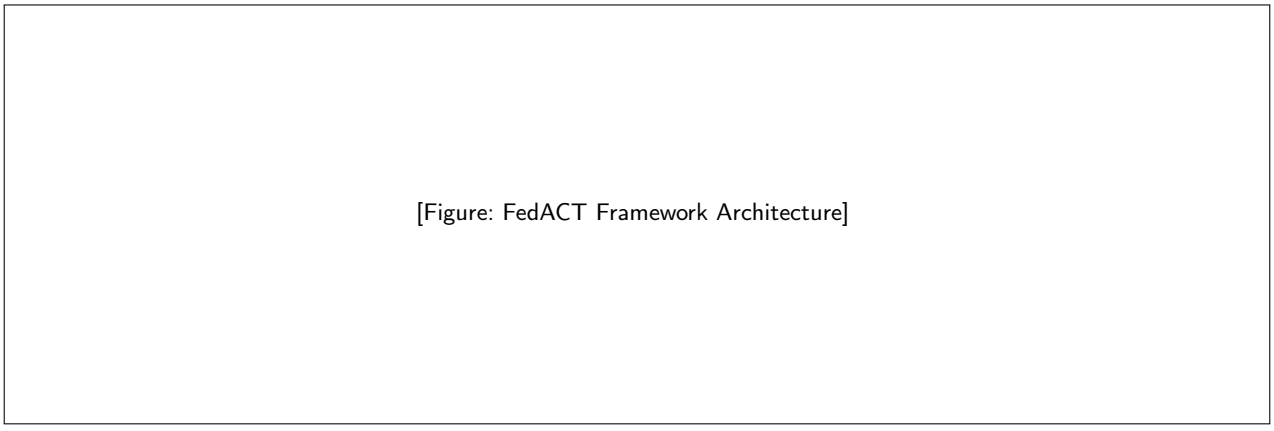

Figure 1: Overall architecture of FedACT. The framework consists of three core components: (1) Autoencoder-based anomaly detection, (2) Committee voting for secondary verification, and (3) TLBO-based robust aggregation.

4. **Committee Voting**: Uncertain gradients undergo secondary verification.
5. **TLBO Aggregation**: Normal gradients are aggregated using TLBO.
6. **Reputation Update**: Client reputations are updated based on detection results.
7. **Evidence Recording**: Results are recorded in a Merkle tree.

4.2. Autoencoder-based Gradient Anomaly Detection

4.2.1. Architecture Design

We employ a neural autoencoder to learn the distribution of benign gradients. The autoencoder consists of an encoder $\mathcal{E} : \mathbb{R}^d \to \mathbb{R}^z$ and a decoder $\mathcal{D} : \mathbb{R}^z \to \mathbb{R}^d$, where d is the gradient dimension and z is the latent dimension.

The encoder compresses the input gradient g into a latent representation:

$$h = \mathcal{E}(g) = \sigma(W_L \cdots \sigma(W_1 g + b_1) \cdots + b_L), \tag{4}$$

where W_l, b_l are learnable parameters and σ is LeakyReLU activation with negative slope 0.2.

The decoder reconstructs the gradient:

$$\hat{g} = \mathcal{D}(h). \tag{5}$$

Adaptive Architecture Configuration: The architecture adapts to gradient dimension d:

- Small ($d < 10,000$): 1 hidden layer, $z = 32$, dropout = 0.1
- Medium ($10,000 \leq d < 100,000$): 2 hidden layers, $z = 64$, dropout = 0.2
- Large ($d \geq 100,000$): 3 hidden layers, $z = 128$, dropout = 0.3

4.2.2. Training Strategy

The autoencoder is trained to minimize reconstruction error:

$$\mathcal{L}_{recon} = \frac{1}{K} \sum_{i=1}^{K} \|g_i - \hat{g}_i\|_2^2, \tag{6}$$

where K is the number of gradients in the current round.

We employ an **incremental training strategy**:

- **Early rounds (1–3)**: Full training with $E_{ae} = 20$ epochs.
- **Subsequent rounds**: Fine-tuning every 5 rounds with $E_{ae}/4 = 5$ epochs.

After training, we compute the latent centroid:

$$\mu_h = \frac{1}{K} \sum_{i=1}^{K} \mathcal{E}(g_i). \tag{7}$$

4.2.3. Anomaly Score Computation

The anomaly score combines reconstruction error and latent space distance:

$$s_i = \alpha \cdot \|g_i - \hat{g}_i\|_2^2 + (1 - \alpha) \cdot \|h_i - \mu_h\|_2, \tag{8}$$

where $h_i = \mathcal{E}(g_i)$ is the latent representation and $\alpha = 0.7$ weights the two components.

4.2.4. Adaptive Threshold Strategy

The detection threshold is computed adaptively:

$$\tau = \mu_s + 2\sigma_s, \tag{9}$$

where μ_s and σ_s are the mean and standard deviation of anomaly scores.

We define three classification zones with configurable coefficients c_{lower} and c_{upper}:

$$\text{Decision}(g_i) = \begin{cases} \text{Normal} & \text{if } s_i < \tau \cdot c_{lower} \\ \text{Uncertain} & \text{if } \tau \cdot c_{lower} \leq s_i \leq \tau \cdot c_{upper} \\ \text{Anomaly} & \text{if } s_i > \tau \cdot c_{upper} \end{cases}$$

(10)

Default values are $c_{lower} = 0.7$ and $c_{upper} = 1.5$, providing a balance between precision and recall.

4.3. Diversity-aware Committee Voting

For gradients in the uncertain zone, we employ a committee voting mechanism.

4.3.1. Committee Selection

The committee consists of $m = 5$ clients selected to maximize diversity:

1. **First member**: Select the client with highest reputation r_i.
2. **Subsequent members**: Select clients minimizing maximum similarity to selected members:

$$c^{(k)} = \arg\min_{c \in \mathcal{C} \setminus \mathcal{S}} \max_{s \in \mathcal{S}} \cos(g_c, g_s), \tag{11}$$

where $\cos(\cdot, \cdot)$ denotes cosine similarity.

4.3.2. Voting Mechanism

Each committee member c votes on gradient g_i:

$$v_c(g_i) = \begin{cases} 1 & \text{if } \cos(g_i, g_c) < \tau_{vote} \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

where $\tau_{vote} = 0.3$ is the voting threshold.

The final decision uses majority vote:

$$\text{is\_anomaly}(g_i) = \mathbb{1}\left[\frac{1}{m}\sum_{c \in \mathcal{S}} v_c(g_i) > 0.5\right]. \tag{13}$$

Self-exclusion: If client i is a committee member, they are excluded from voting on their own gradient.

Warm-up Period: Committee voting activates only after round 5 to allow reputation stabilization.

Algorithm 1 presents the committee voting procedure.

Algorithm 1 Committee Voting

Input: Gradients $\{g_i\}$, client IDs $\{c_i\}$, reputations $\{r_i\}$, committee size m
Output: Committee members $\mathcal{S}$

$\mathcal{S} \leftarrow \emptyset$, $\mathcal{A} \leftarrow \{1, \ldots, K\}$ ▷ Available clients
$c_1 \leftarrow \arg\max_{i \in \mathcal{A}} r_i$ ▷ Highest reputation
$\mathcal{S} \leftarrow \mathcal{S} \cup \{c_1\}$, $\mathcal{A} \leftarrow \mathcal{A} \setminus \{c_1\}$
while $|\mathcal{S}| < m$ and $\mathcal{A} \neq \emptyset$ **do**
  for each $i \in \mathcal{A}$ **do**
    $\text{sim}_i \leftarrow \max_{j \in \mathcal{S}} \cos(g_i, g_j)$
  end for
  $c^* \leftarrow \arg\min_{i \in \mathcal{A}} \text{sim}_i$ ▷ Most diverse
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{c^*\}$, $\mathcal{A} \leftarrow \mathcal{A} \setminus \{c^*\}$
end while
return $\mathcal{S}$

4.4. TLBO-based Robust Aggregation

After filtering anomalous gradients, we aggregate normal gradients using TLBO.

4.4.1. TLBO Overview

TLBO simulates teaching-learning in a classroom through two phases:

Teacher Phase: Learners learn from the teacher (best solution):

$$g_i^{\text{new}} = g_i + r \cdot (g_{\text{teacher}} - T_F \cdot \bar{g}), \tag{14}$$

where g_{teacher} is the gradient with highest fitness, $\bar{g}$ is the mean, $r \in [0, 1]$ is random, and $T_F \in \{1, 2\}$ is the teaching factor.

Learner Phase: Learners learn from each other:

$$g_i^{\text{new}} = \begin{cases} g_i + r \cdot (g_j - g_i) & \text{if } f(g_j) > f(g_i) \\ g_i + r \cdot (g_i - g_j) & \text{otherwise} \end{cases} \tag{15}$$

where g_j is a randomly selected learner.

4.4.2. Fitness Function

The fitness measures alignment with weighted average:

$$f(g_i) = \cos\left(g_i, \sum_{j \in \mathcal{N}} w_j g_j\right), \tag{16}$$

where $\mathcal{N}$ is the set of normal clients and w_j is the reputation-based weight.

Algorithm 2 presents the TLBO aggregation procedure.

Algorithm 2 TLBO Gradient Aggregation

Input: Normal gradients $\{g_i\}_{i \in \mathcal{N}}$, weights $\{w_i\}$, iterations T
Output: Aggregated gradient g^*

Initialize learners: $\mathcal{L} \leftarrow \{g_i\}_{i \in \mathcal{N}}$
Compute target: $g^{\text{target}} \leftarrow \sum_{i \in \mathcal{N}} w_i g_i$
for $t = 1$ to T **do**
  Compute fitness: $f_i \leftarrow \cos(g_i, g^{\text{target}})$ for all i
  // Teacher Phase
  $g_{\text{teacher}} \leftarrow \arg\max_{g \in \mathcal{L}} f(g)$
  $\bar{g} \leftarrow \frac{1}{|\mathcal{L}|} \sum_{g \in \mathcal{L}} g$
  $T_F \leftarrow \text{random}(\{1, 2\})$
  for each $g_i \in \mathcal{L}$ **do**
    $r \leftarrow \text{uniform}(0, 1)$
    $g_i^{\text{new}} \leftarrow g_i + r \cdot (g_{\text{teacher}} - T_F \cdot \bar{g})$
    if $f(g_i^{\text{new}}) > f(g_i)$ **then**
      $g_i \leftarrow g_i^{\text{new}}$
    end if
  end for
  // Learner Phase
  for each $g_i \in \mathcal{L}$ **do**
    Randomly select $g_j \in \mathcal{L}, j \neq i$
    $r \leftarrow \text{uniform}(0, 1)$
    if $f(g_j) > f(g_i)$ **then**
      $g_i^{\text{new}} \leftarrow g_i + r \cdot (g_j - g_i)$
    else
      $g_i^{\text{new}} \leftarrow g_i + r \cdot (g_i - g_j)$
    end if
    if $f(g_i^{\text{new}}) > f(g_i)$ **then**
      $g_i \leftarrow g_i^{\text{new}}$
    end if
  end for
  Update target: $g^{\text{target}} \leftarrow \frac{1}{|\mathcal{L}|} \sum_{g \in \mathcal{L}} g$

end for
return g^{target}

4.5. Reputation-based Incentive Mechanism

We maintain a reputation score $r_i \in [0.1, 2.0]$ for each client, initialized to 1.0.

Reputation Update:

$$r_i^{(t+1)} = \begin{cases} \min(r_i^{(t)} + 0.05 \cdot c_i, 2.0) & \text{if classified normal} \\ \max(r_i^{(t)} \times 0.7, 0.1) & \text{if classified anomaly} \end{cases} \tag{17}$$

where the contribution score is:

$$c_i = \frac{1}{2}\left(\cos(g_i, g^*) + 1\right) \in [0, 1]. \tag{18}$$

Weight Computation:

$$w_i = \frac{r_i}{\sum_{j \in \mathcal{N}} r_j}. \tag{19}$$

4.6. Merkle Tree-based Evidence Chain

For audit traceability, detection results are recorded in a Merkle tree each round.

Leaf Nodes:

$$\text{leaf}_i = \text{SHA256}(\text{client\_id} \| \text{score} \| \text{decision} \| \text{timestamp}). \tag{20}$$

Merkle Root: Computed by recursively hashing pairs:

$$\text{parent} = \text{SHA256}(\text{left\_child} \| \text{right\_child}). \tag{21}$$

The root hash provides an immutable summary enabling post-hoc auditing.

4.7. Complete FedACT Algorithm

Algorithm 3 presents the complete FedACT training procedure.

Algorithm 3 FedACT Training

Input: Clients C, rounds R, committee size m, TLBO iterations T, threshold coefficients c_{lower}, c_{upper}

Initialize global model $\theta^{(0)}$, reputations $\{r_i = 1.0\}$
Initialize autoencoder $\mathcal{A}$, evidence chain $\mathcal{E}$
for round $t = 1$ to R **do**
  // Step 1: Gradient Collection
  Broadcast $\theta^{(t-1)}$ to selected clients
  Collect gradients $\{g_i^{(t)}\}_{i \in \mathcal{S}}$
  // Step 2: Autoencoder Training
  if $t \leq 3$ or $t \mod 5 = 0$ **then**
    Train $\mathcal{A}$ on $\{g_i^{(t)}\}$ for E_{ae} epochs
    Update latent centroid μ_h
  end if
  // Step 3: Anomaly Score Computation
  for each gradient g_i **do**
    $h_i \leftarrow \mathcal{E}(g_i)$, $\hat{g}_i \leftarrow \mathcal{D}(h_i)$
    $s_i \leftarrow 0.7 \cdot \|g_i - \hat{g}_i\|^2 + 0.3 \cdot \|h_i - \mu_h\|$
  end for
  $\tau \leftarrow \mu_s + 2\sigma_s$
  // Step 4: Initial Classification
  $\mathcal{N}, \mathcal{U}, \mathcal{A}nom \leftarrow \emptyset, \emptyset, \emptyset$
  for each client i **do**
    if $s_i < \tau \cdot c_{lower}$ **then** $\mathcal{N} \leftarrow \mathcal{N} \cup \{i\}$
    else if $s_i > \tau \cdot c_{upper}$ **then** $\mathcal{A}nom \leftarrow \mathcal{A}nom \cup \{i\}$
    else $\mathcal{U} \leftarrow \mathcal{U} \cup \{i\}$
    end if
  end for
  // Step 5: Committee Voting (after warm-up)
  if $t > 5$ and $\mathcal{U} \neq \emptyset$ **then**
    $\mathcal{M} \leftarrow \text{SelectCommittee}(\{g_i\}, \{r_i\}, m)$
    for each $i \in \mathcal{U}$ **do**
      $\mathcal{M}_i \leftarrow \mathcal{M} \setminus \{i\}$ ▷ Self-exclusion
      votes $\leftarrow \sum_{c \in \mathcal{M}_i} \mathbb{1}[\cos(g_i, g_c) < \tau_{vote}]$
      if votes $/|\mathcal{M}_i| > 0.5$ **then** $\mathcal{A}nom \leftarrow \mathcal{A}nom \cup \{i\}$
      else $\mathcal{N} \leftarrow \mathcal{N} \cup \{i\}$
      end if
    end for
  else $\mathcal{N} \leftarrow \mathcal{N} \cup \mathcal{U}$ ▷ Accept uncertain during warm-up
  end if
  // Step 6: TLBO Aggregation
  Compute weights $w_i \leftarrow r_i / \sum_{j \in \mathcal{N}} r_j$
  $g^* \leftarrow \text{TLBO}(\{g_i\}_{i \in \mathcal{N}}, \{w_i\}, T)$
  $\theta^{(t)} \leftarrow \theta^{(t-1)} - \eta g^*$
  // Step 7: Reputation Update
  for each $i \in \mathcal{N}$ **do**
    $c_i \leftarrow (\cos(g_i, g^*) + 1)/2$
    $r_i \leftarrow \min(r_i + 0.05 \cdot c_i, 2.0)$
  end for
  for each $i \in \mathcal{A}nom$ **do**
    $r_i \leftarrow \max(r_i \times 0.7, 0.1)$
  end for
  // Step 8: Evidence Recording
  Record $(t, \{(i, s_i, \text{decision}_i)\})$ in Merkle tree
end for
return $\theta^{(R)}$

5. Convergence Analysis

5.1. Assumptions

Assumption 1 (L-Smoothness)**.** *The loss function $F(\theta)$ is L-smooth:*

$$\|\nabla F(\theta_1) - \nabla F(\theta_2)\| \leq L\|\theta_1 - \theta_2\|. \tag{22}$$

Assumption 2 (Bounded Variance)**.** *The variance of stochastic gradients is bounded:*

$$\mathbb{E}[\|g_i - \nabla F_i(\theta)\|^2] \leq \sigma^2. \tag{23}$$

Assumption 3 (Bounded Gradient)**.** *The gradient norm is bounded:*

$$\|\nabla F_i(\theta)\| \leq G. \tag{24}$$

Assumption 4 (Detection Accuracy)**.** *The detector correctly identifies Byzantine gradients with probability at least $1 - \delta$.*

5.2. Convergence Theorem

Theorem 1. *Under Assumptions 1–4, if FedACT runs for T rounds with step size $\eta = O(1/\sqrt{T})$, then:*

$$\frac{1}{T}\sum_{t=1}^{T} \mathbb{E}[\|\nabla F(\theta^{(t)})\|^2] \leq O\left(\frac{1}{\sqrt{T}}\right) + O(\delta \cdot G^2). \tag{25}$$

Proof. By L-smoothness:

$$F(\theta^{(t+1)}) \leq F(\theta^{(t)}) - \eta\langle \nabla F(\theta^{(t)}), g^{(t)}\rangle + \frac{L\eta^2}{2}\|g^{(t)}\|^2. \quad (26)$$

Taking expectation:

$$\mathbb{E}[F(\theta^{(t+1)})] \leq F(\theta^{(t)}) - \eta\|\nabla F(\theta^{(t)})\|^2 + \frac{L\eta^2(\sigma^2 + G^2)}{2}. \quad (27)$$

Accounting for detection errors with probability δ:

$$\mathbb{E}[\|g^{(t)} - \nabla F(\theta^{(t)})\|^2] \leq \sigma^2 + \delta \cdot G^2. \quad (28)$$

Summing over T rounds:

$$\sum_{t=1}^{T} \eta\|\nabla F(\theta^{(t)})\|^2 \leq F(\theta^{(0)}) - F^* + \frac{TL\eta^2(\sigma^2 + \delta G^2)}{2}. \quad (29)$$

With $\eta = \sqrt{2(F(\theta^{(0)}) - F^*)/(TL(\sigma^2 + \delta G^2))}$:

$$\frac{1}{T}\sum_{t=1}^{T} \|\nabla F(\theta^{(t)})\|^2 \leq \sqrt{\frac{2L(F(\theta^{(0)}) - F^*)(\sigma^2 + \delta G^2)}{T}}. \quad (30)$$

□

The theorem shows FedACT converges at rate $O(1/\sqrt{T})$, matching standard SGD, provided detection accuracy is high (small δ).

6. Experiments

6.1. Experimental Setup

6.1.1. Datasets

- **UCI German Credit**: 1,000 samples, 20 features (7 continuous, 13 categorical), binary labels (good/bad credit).
- **Xinwang Bank Credit**: 10,000 samples from a Chinese commercial bank, 30 features including demographic, financial, and behavioral attributes.

6.1.2. Implementation Details

Experiments are conducted on Ubuntu 22.04 with NVIDIA RTX 4090 GPU. The model is a 3-layer MLP with 128 hidden units. Default hyperparameters are shown in Table 2.

6.1.3. Evaluation Metrics

Detection Metrics: Precision, Recall, F1 Score, Accuracy.

Model Performance: Classification Accuracy, AUC, F1 Score.

6.2. Overall Defense Performance

Table 3 presents the overall defense performance.

6.3. Detection Performance

Table 4 presents detection metrics across attack types.

Table 2
Default hyperparameters

Parameter	Value
Number of clients N	10
Global rounds R	100
Local epochs E	5
Learning rate η	0.01
Committee size m	5
TLBO iterations T	10
Lower threshold c_{lower}	0.7
Upper threshold c_{upper}	1.5
Voting threshold τ_{vote}	0.3
Autoencoder epochs E_{ae}	20
Anomaly score weight α	0.7

6.4. Impact of Heterogeneity

Table 5 shows performance under different heterogeneity scenarios.

6.5. Impact of Malicious Ratio

Figure 2 shows performance under varying malicious client ratios.

6.6. Ablation Study

Table 6 shows the contribution of each component.

6.7. Threshold Sensitivity

Figure 3 shows sensitivity to threshold coefficients.

6.8. Computational Overhead

Table 7 compares computational overhead.

7. Conclusion

This paper proposed FedACT, a comprehensive Byzantine-resilient federated learning framework for secure collaborative credit scoring. FedACT integrates three complementary defense mechanisms: adaptive autoencoder-based anomaly detection, diversity-aware committee voting, and TLBO-based robust aggregation. The framework also incorporates Merkle tree-based evidence chains for audit traceability and reputation-based incentives for dynamic weight adjustment.

The key design choices are grounded in addressing practical challenges: the autoencoder adapts to task-specific gradient distributions without predetermined thresholds; the committee voting provides robust secondary verification using diverse perspectives; and the TLBO aggregation handles heterogeneity through teacher-learner dynamics.

Extensive experiments on two credit scoring datasets demonstrated that FedACT significantly outperforms existing defense methods. Under 12 attack types and 4 heterogeneity scenarios, FedACT achieved detection precision exceeding 95%, detection recall above 90%, and model accuracy within 2% of attack-free baselines.

For future work, we plan to extend FedACT to vertical federated learning scenarios, investigate defense against

Table 3
Overall defense performance comparison (average over all attacks with 20% malicious clients)

Defense	UCI German Credit			Xinwang Bank		
	Accuracy	AUC	F1	Accuracy	AUC	F1
No Attack (Upper Bound)						
FedAvg (No Defense)						
Median						
Trimmed Mean						
Krum						
Multi-Krum						
Bulyan						
RFA						
FedACT (Ours)						

Table 4
Detection performance of FedACT (20% malicious, IID setting)

Attack	UCI German Credit				Xinwang Bank			
	TP	FP	Precision	Recall	TP	FP	Precision	Recall
Sign Flip								
Gaussian								
Scale								
Little								
ALIE								
IPM								
MinMax								
Trim Attack								
Label Flip								
Backdoor								
Free Rider								
Collision								
Average								

adaptive attacks, and deploy FedACT on blockchain platforms for fully decentralized execution.

Acknowledgment

This research was supported by the National Natural Science Foundation of China (No. 71850012, 71790593), National Social Science Fund of China (No. 19AZD014), and Major Special Projects of the Department of Science and Technology of Hunan Province (No. 2018GK1020).

References

[1] Thomas, L.C., Edelman, D.B., Crook, J.N., 2002. Credit scoring and its applications. SIAM.

[2] Li, T., Sahu, A.K., Talwalkar, A., Smith, V., 2020. Federated learning: Challenges, methods, and future directions. IEEE Signal Processing

Table 5
Defense performance under heterogeneity (ALIE attack, 20% malicious)

Defense	IID		Label Skew		Quantity Skew		Feature Skew	
	Acc	AUC	Acc	AUC	Acc	AUC	Acc	AUC
FedAvg								
Median								
Trimmed Mean								
Krum								
Multi-Krum								
Bulyan								
RFA								
FedACT								

Table 6
Ablation study (ALIE attack, 20% malicious, IID)

Configuration	Accuracy	Precision	Recall
FedACT (Full)			
w/o Autoencoder			
w/o Committee			
w/o TLBO			
w/o Merkle			
w/o Reputation			

Table 7
Computational overhead (seconds per round)

Defense	UCI	Xinwang
FedAvg		
Median		
Krum		
Multi-Krum		
Bulyan		
FedACT		

Magazine 37, 50–60.

[3] McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A., 2017. Communication-efficient learning of deep networks from decentralized data , 1273–1282.

[4] Yang, Q., Liu, Y., Chen, T., Tong, Y., 2019. Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology 10, 1–19.

[5] Yang, J., Qiao, Y., Li, M., Li, D., 2024. An explainable federated learning and blockchain-based secure credit modeling method. European Journal of Operational Research 317, 449–467.

[6] Blanchard, P., El Mhamdi, E.M., Guerraoui, R., Stainer, J., 2017. Machine learning with adversaries: Byzantine tolerant gradient descent, in: Advances in Neural Information Processing Systems.

[7] Fang, M., Cao, X., Jia, J., Gong, N., 2020. Local model poisoning attacks to byzantine-robust federated learning, in: 29th USENIX Security Symposium, pp. 1605–1622.

[8] Baruch, M., Baruch, G., Goldberg, Y., 2019. A little is enough: Circumventing defenses for distributed learning, in: Advances in Neural Information Processing Systems.

[9] Xie, C., Koyejo, O., Gupta, I., 2020. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation, in: Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence, pp. 261–270.

[10] Shejwalkar, V., Houmansadr, A., 2021. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning, in: Proceedings of the Network and Distributed System Security Symposium.

[11] Yin, D., Chen, Y., Kannan, R., Bartlett, P., 2018. Byzantine-robust distributed learning: Towards optimal statistical rates, in: International Conference on Machine Learning, pp. 5650–5659.

[12] El-Mhamdi, E.M., Guerraoui, R., Rouault, S., 2018. The hidden vulnerability of distributed learning in byzantium, in: International Conference on Machine Learning, pp. 3521–3530.

[13] Cao, X., Fang, M., Liu, J., Gong, N.Z., 2021. Fltrust: Byzantine-robust federated learning via trust bootstrapping, in: Proceedings of the Network and Distributed System Security Symposium.

[14] Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A.N., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., et al., 2021. Advances and open problems in federated learning. Foundations and Trends in Machine Learning 14, 1–210.

[15] Rao, R.V., Savsani, V.J., Vakharia, D., 2011. Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems. Computer-Aided Design 43, 303–315.

[16] Qiao, Y., Yang, J., Li, M., 2023. A privacy-preserving decentralized credit scoring method based on multi-party information. Applied Soft Computing 145, 110545.

[17] Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V., 2020. How to backdoor federated learning, in: International Conference on Artificial Intelligence and Statistics, pp. 2938–2948.

[18] Wang, H., Sreenivasan, K., Rajput, S., Vishwakarma, H., Aber-nethy, S., Papailiopoulos, D., 2020. Attack of the tails: Yes, you really can backdoor federated learning, in: Advances in Neural Information Processing Systems, pp. 16070–16084.

[19] Pillutla, K., Kakade, S.M., Harchaoui, Z., 2019. Robust aggregation for federated learning, in: arXiv preprint arXiv:1912.13445.

Figure 2: Model accuracy under varying malicious ratios.

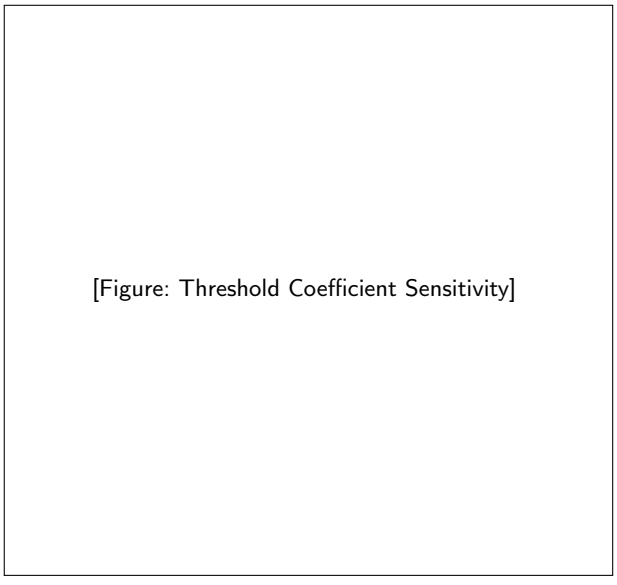

Figure 3: Detection precision and recall under different threshold coefficients c_{lower} and c_{upper}.