Available Online: 4 December 2025 DOI: 10.54254/2977-3903/2025.30324

Research on federated learning security detection method based on linear combinatorial rank analysis

Huasen Guo

Glasgow Hainan College, University of Electronic Science and Technology of China, Chengdu, China

2959535G@student.gla.ac.uk

Abstract. Federated learning effectively protects data privacy by training models on local devices and only sharing model updates. However, its distributed nature also makes the system vulnerable to malicious client attacks, such as data poisoning, model tampering and backdoor attacks, especially being more concealed in the non-independent co-distributed (Non-IID) data environment. To address the above-mentioned security challenges, this paper proposes a federated learning security detection method based on linear combinatorial rank analysis. This method achieves anomaly detection by transforming the model parameter transmission process into the transmission of encoded vectors over a finite field and analyzing the rank variation of the encoded matrix. Different from the traditional methods, this method does not rely on the IID data assumption and can adapt to the complex data distribution in the Non-IID environment. At the same time, a dynamic coding adjustment mechanism is introduced, which can adaptively balance security and system efficiency according to the client resources and system security status. In addition, this paper also designs a full-link protection scheme to ensure that the entire process from parameter generation, encoding calculation to upload is effectively guaranteed in terms of security and integrity. The results show that the detection rates of LCRA in the scenarios of data poisoning, model tampering and backdoor attack reach 96.2%, 94.8% and 95.6% respectively, and the false alarm rate is lower than 4.1%. Meanwhile, the high accuracy rates of the model on CIFAR-10 and MNIST (85.3% and 97.8% respectively) are maintained. It outperforms existing robust aggregation and differential privacy methods.

Keywords: federated learning, security detection, network coding, rank analysis, poisoning attack

1. Introduction

Federated learning effectively reduces the reliance on raw data by allowing the client to train the model locally and only upload parameter updates, thereby enhancing the level of data privacy protection [1]. Meanwhile, this distributed training method has also brought about new security risks. Relevant studies have shown that malicious clients may launch various attacks, including Data Poisoning, Model Manipulation and Backdoor attacks. In a Non-IID data environment, such attacks are more concealed, and traditional defense methods face significant challenges in detection and protection [2].

The current defense methods mainly focus on statistical anomaly detection and Differential Privacy technology. The Krum algorithm proposed by Blanchard et al. identifies anomalies by analyzing the statistical characteristics of model parameters, but it largely relies on the IID data assumption, and thus is prone to a high false alarm rate in a Non-IID environment [3]. The research by Wei et al. shows that although differential privacy technology can enhance security, the noise it introduces often significantly reduces the performance of the global model [4]. Meanwhile, Wang et al. pointed out that the detection capabilities of existing defense measures for low-amplitude, elaborately designed backdoor attacks are still insufficient [5].

Therefore, inspired by Network Coding technology, this paper proposes a federated learning security detection method based on Linear Combination Rank Analysis [6]. This method maps the model parameter transmission process to the encoding vector on the finite field and identifies abnormal behaviors by analyzing the rank variation of the server-side encoding matrix. Unlike traditional methods, LCRA does not rely on the IID data assumption and can adapt to the complex data distribution in a Non-IID environment. In addition, the method introduces a dynamic encoding adjustment mechanism, which can adaptively balance security and efficiency based on client resources and system security status. Meanwhile, this paper designs a full-link protection scheme to ensure that the entire process from parameter generation, encoding calculation to upload is secure and integrity, thereby enhancing the attack detection capability while taking into account the robustness and performance of the system.

2. Methodology

Linear Combination Rank Analysis (LCRA) offers a unique algebraic lens for securing federated learning. Its mechanism maps model parameters to encoded vectors over a finite field, utilizing rank variations to directly expose abnormal behaviors. This robust framework comprises three core modules: encoding vector generation, dynamic encoding adjustment, and rank analysis detection. These modules are individually responsible for parameter encoding, resource-adaptive optimization, and anomaly identification, collectively achieving efficient and secure protection even within a Non-IID environment.

2.1. Encoding vector generation and transmission

In conventional federated learning, clients directly transmit the locally trained model parameter vector $p = (p_1,\ p_2,\ \ldots,\ p_n)$. In the LCRA method, an encoding layer is introduced to achieve algebraic processing of parameters. Specifically, the client randomly selects an encoding coefficient vector $c = (c_1,\ c_2,\ \ldots,\ c_n)$ from the predefined finite field GF(q). Subsequently, the client performs a linear combination operation to generate the encoded vector v .

$$v = c_1 \cdot p_1 + c_2 \cdot p_2 + \ldots + c_n \cdot p_n \tag{1}$$

To enhance security and provide richer algebraic features for subsequent rank analysis, the encoding process is extended to generate a D-dimensional encoding vector. Specifically, the client randomly generates an encoding matrix $\mathrm{C} \in \mathrm{GF(q)}^{\mathrm{d\times n}}$, where each row cj=(c^j_1,c^j_2,…,c^j_n) represents a set of independent random coefficients. Through the matrix multiplication of this matrix and the parameter vector, the D-dimensional encoding vector v is obtained:

$$v = \boldsymbol{C} \cdot \boldsymbol{p} \tag{2}$$

The j^{th} component of the encoded vector v is calculated as follows:

$$v_j = \sum_{i=1}^{n} c_i^j \bullet p,\ where\ j = 1, 2, ..., d. \tag{3}$$

All the operations are carried out over the finite field GF(q). The client then uploads the encoded vector v to the aggregation server. Compared with directly transmitting the original model parameters, this encoded transmission not only provides a mathematical basis for subsequent rank analysis, but also enhances the security and privacy protection of the parameters, while reducing the abnormal interference caused by non-independent and uniformly distributed data.

2.2. Dynamic encoding adjustment mechanism

In response to the heterogeneity and resource limitations of devices in the actual federated learning environment, LCRA introduces a dynamic encoding adjustment mechanism to adaptively adjust the encoding strategy according to the real-time status of the system. This mechanism achieves flexible control by adjusting the encoding dimension and the size of the finite field. High-dimensional coding can provide higher security redundancy, but it will increase computational and communication overhead. Low-dimensional coding has less computational and communication overhead, but its security is relatively lower. The encoding dimension dd directly determines the expressive power and security redundancy of the encoding space.

Theoretically, the probability that an attacker can successfully construct an abnormal encoding vector that evades rank detection is upper-bounded by:

$$P_{evade} \leq q^{-\,(d\,-\,r_{normal})} \tag{4}$$

where $\mathrm{r_{normal}}$ denotes the typical rank of the encoding matrix during normal system operation. This formula indicates that increasing the dimension dd can significantly (exponentially) reduce the success rate of attacks, thereby enhancing the security of the system. However, a higher d also linearly increases the communication and computing overhead $\mathrm{O}(d)$. Therefore, the core of the dynamic adjustment mechanism lies precisely in finding the optimal balance point between security and efficiency for d based on the real-time security threat level.

The choice of the size of a finite field also affects system performance and security. Larger finite fields, such as GF(2^8), offer higher security and stronger numerical representation capabilities, but they have higher computational complexity. Smaller finite fields, such as GF(2), have higher computational efficiency, but their security is somewhat reduced. The server maintains a resource and security status table to monitor in real time the network bandwidth, computing power, power consumption of clients, and the overall security threat level of the system, thereby allocating appropriate coding strategies for different clients or training rounds. For clients with abundant resources, high-dimensional and large-domain strong coding can be adopted. For resource-constrained clients, lightweight coding is adopted to optimize overall performance while ensuring system security [7].

2.3. Anomaly detection based on rank analysis

Anomaly detection is carried out on the server side. The server collects the encoding vectors from m clients over k consecutive training rounds and builds an encoding matrix M of size $(m \times k) \times d$ based on this, where d is the dimension of the encoding vector. During the normal training process, due to the smooth convergence of model parameters, the linear relationship between the sequences of encoding vectors remains stable, and thus the rank of the encoding matrix M fluctuates smoothly within the expected range [8]. When there are malicious clients, such as data poisoning, model tampering or backdoor attacks, the abnormal encoded vectors they upload will disrupt the original linear structure, resulting in significant changes in the matrix rank [9-11]. The server monitors the trend of rank changes and sets a dynamic threshold τ based on historical data. When the rank change exceeds the threshold, a security alert is triggered:

$$| \mathrm{rank}(\mathrm{M_t}) - \mathrm{rank}(\mathrm{M_{t-1}}) | > \tau \tag{5}$$

To enhance the adaptability of the detection system, the threshold τ is dynamically updated using an exponentially weighted moving average strategy, enabling it to smoothly reflect the latest state of the system:

$$\tau_{new} = \alpha \cdot \tau_{old} + (1 - \alpha) \cdot \Delta r \tag{6}$$

Where $\Delta \mathrm{r} = | \mathrm{rank}(\mathrm{M_t}) - \mathrm{rank}(\mathrm{M_{t-1}}) |$ represents the rank change of the current round. The parameter α is a smoothing factor (usually taken as 0.7 to 0.9), used to control the weight of historical thresholds. The larger the α, the more stable the threshold. The smaller the α is, the more sensitive the threshold is to recent changes. In addition, a more robust threshold can be set based on the statistical characteristics of historical rank changes:

$$\tau = \mu + \gamma \cdot \sigma \tag{7}$$

Where μ is the average value of the historical rank change (Δr), σ is the standard deviation of the historical rank change, and γ is an adjustable parameter (usually set to 2 or 3), used to control the sensitivity of anomaly detection. The larger the γ, the looser the threshold and the lower the false alarm rate, but some minor attacks may be missed. The smaller the γ, the stricter the threshold, and the higher the detection rate, but false alarms may increase.

This mechanism can promptly identify abnormal behaviors in the Non-IID environment while maintaining tolerance for the normal training process, thereby achieving efficient federated learning security detection.

2.4. Full-link security protection

To stop the encoding process from turning into an attack vector LCRA sets up a full-chain security mechanism putting security controls all across the whole workflow which includes before during and after encoding: Before encoding the client figures out the SHA-256 hash of the parameter vector and protects it with a digital signature using a private key this ensures the integrity and the ability to be verified of the parameters and effectively stops data from being tampered with at the start; During the encoding phase all important computations run inside a Trusted Execution Environment like Intel SGX this doesn't just protect the secrecy of the computing logic but also makes sure the generated encoding vector is correct and has integrity thus getting rid of the risk of external malicious access or change [12]. After finishing the encoding the client creates and uploads several say three encoding vectors for the same local data each using different random coefficients then the server cross-checks the linear relationships among these vectors this process not only makes sure the encoding results are right but also enables anomalies to be identified right away if the verification fails for instance the result is immediately thrown out and the client is marked as suspicious finally this mechanism reduces single-point attack risks through multi-layer verification and redundant checking and secures a dependable data base for the following rank analysis.

2.5. Experimental setup and evaluation design

To thoroughly assess the method's effectiveness a federated learning simulation environment was set up making use of the PySyft framework and the experiments made use of the CIFAR-10 and MNIST datasets; importantly non-IID data partitioning was simulated via a Dirichlet distribution to precisely mirror the data imbalance commonly found in real world situations the setup consisted of 100 total clients with 10 clients randomly chosen for training in each round guaranteeing both variety and activity throughout the training process; within this environment three main kinds of security threats were simulated for the design of the attack scenarios.

The first kind is the data poisoning attack which meddles with global model training by altering the labels of 20% of malicious clients; the second kind is the model tampering attack where malicious clients insert Gaussian noise into local model parameters thus disturbing model convergence; the last kind is the backdoor attack which employs a distributed triggering mode to conceal malicious behavior while keeping the overall performance of the model [11]. The baselines for comparison include traditional reliable aggregation algorithms like the Krum algorithm the median algorithm and the truncated mean algorithm as well as a differential privacy-based federated learning plan (DP-FedAvg) [3,4]. Experimental evaluation metrics cover attack detection rate false positive rate and model accuracy on the test set; moreover computational and communication overhead were taken into account and the proposed method was comprehensively evaluated in terms of security model availability and system efficiency.

2.6. LCRA algorithm framework

The complete workflow of the LCRA method can be summarized as shown in Figure 1. The method takes as input the local parameters p, the encoding dimension d, and the finite field GF(q), and outputs an anomaly detection result (Normal or Alert). On the client side, a random encoding matrix $C \in GF(q)^{d\times n}$ is first generated, and the encoded vector $v=C\bullet p$ is computed. Then, the SHA-256 hash of the local parameter p is calculated and signed with the private key. Finally, the encoded vector and signature $(v,signature)$ are uploaded to the server. On the server side, encoded vectors from m clients over k consecutive rounds are collected to construct the encoding matrix $M \in GF(q)^{(m\times k)\times d}$. The current rank of the matrix $r_t=rank(M_t)$ is calculated, and the rank variation $\Delta r=|r_t-r_{t-1}|$ is computed. A dynamic threshold τ is then updated using the exponential weighted moving average (EWMA) method as $\tau_{new}=\alpha\bullet\tau_{old}+(1-\alpha)\bullet\Delta r$. If the rank variation Δr exceeds the updated threshold τ_{new}, a security alert is triggered; otherwise, normal aggregation continues. Additionally, the system can perform dynamic adjustments in each round by monitoring client resources and security threat levels, and adjusting the encoding dimension d or finite field q according to resource constraints, then broadcasting the updated encoding strategy to all clients.

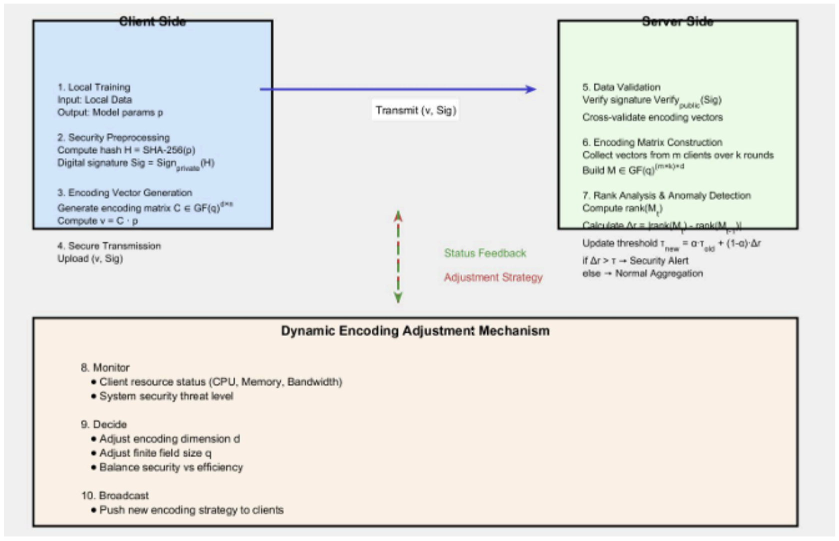
Client Side
1. Local Training
Input: Local Data
Output: Model params p
2. Security Preprocessing
Compute hash H = SHA-256(p)
Digital signature Sig = Sign$_{private}$(H)
3. Encoding Vector Generation
Generate encoding matrix C ∈ GF(q)$^{d×n}$
Compute v = C · p
4. Secure Transmission
Upload (v, Sig)
Transmit (v, Sig)
Server Side
5. Data Validation
Verify signature Verify$_{public}$(Sig)
Cross-validate encoding vectors
6. Encoding Matrix Construction
Collect vectors from m clients over k rounds
Build M ∈ GF(q)$^{(m×k)×d}$
7. Rank Analysis & Anomaly Detection
Compute rank(M$_t$)
Calculate Δr = |rank(M$_t$) - rank(M$_{t-1}$)|
Update threshold τ$_{new}$ = α·τ$_{old}$ + (1-α)·Δr
if Δr > τ → Security Alert
else → Normal Aggregation
Status Feedback
Adjustment Strategy
Dynamic Encoding Adjustment Mechanism
8. Monitor
• Client resource status (CPU, Memory, Bandwidth)
• System security threat level
9. Decide
• Adjust encoding dimension d
• Adjust finite field size q
• Balance security vs efficiency
10. Broadcast
• Push new encoding strategy to clients

Figure 1. LCRA method workflow

3. Results

3.1. Attack detection and model utility

The LCRA method demonstrates excellent detection capabilities and high model utility in various attack scenarios, as shown in Table 1. The detection rates of data poisoning, model tampering and backdoor attacks reached 96.2%, 94.8% and 95.6% respectively, while the false alarm rate was stably controlled below 4.1%. This indicates that the detection mechanism based on rank analysis is universal and highly sensitive to all kinds of attacks. Although these three attacks have different objectives, data poisoning indirectly affects the model by polluting training data, model tampering directly changes the parameter vector, and backdoor attacks attempt to implant specific triggering patterns. However, they all essentially disrupt the normal evolution of local model updates, causing the encoded vector uploaded by the client to deviate from its inherent linear subspace. The core advantage of LCRA is that it does not rely on any specific attack hypothesis or statistical distribution, but rather captures this "bias" by monitoring macroscopic changes in the rank of the coding matrix. Specifically, data poisoning and model tampering usually cause drastic changes in parameter vectors, resulting in a significant increase in the rank of the encoding matrix, which is easily captured by threshold mechanisms. Backdoor attacks, in order to maintain their concealment, often have relatively small parameter perturbations. However, their goal is to establish abnormal correlation relationships in the model, which will change the linear correlation between parameter vectors and cause abnormal fluctuations in the rank of the matrix. The sensitivity of LCRA to linear structures makes it equally effective against such subtle changes.

Table 1. Comparison of safety detection performance (%)

Method	Data Poisoning Detection Rate (%)	Model Tampering Detection Rate (%)	Backdoor Attack Detection Rate (%)	False Alarm Rate (%)	Poisoning F1	Tampering F1	Backdoor F1
Krum	88.5	85.2	70.1	15.3	85.9	82.8	68.2
Median	82.3	80.1	65.8	12.7	82.1	79.5	70.5
Trimmed Mean	85.6	83.4	68.9	11.5	85.0	82.6	72.8
DP-FedAvg	90.1	88.5	75.3	8.9	89.8	87.9	79.0
LCRA (Proposed)	96.2	94.8	95.6	4.1	95.8	94.5	95.2

Compared with traditional methods, the advantages of LCRA are particularly obvious in the Non-IID environment. Robust aggregation methods such as Krum and Median rely heavily on the IID assumption that "benign updates are close to each other." Under Non-IID data, benign updates with already significant differences can be misjudged as anomalies, resulting in a false alarm rate as high as 11.5% to 15.3%, which is much higher than the 4.1% of LCRA. Meanwhile, they are generally ineffective against elaborately designed backdoor attacks, with detection rates all below 71%. Although the differential privacy-based method (DP-FedAvg) provides certain privacy protection and attack smoothness effects through noise, the noise also masks the malicious behavior signals, resulting in a still limited detection rate (75.3%) for backdoor attacks, and seriously sacrificing the model utility (the accuracy rate of CIFAR-10 drops to 79.8%). To further verify the robustness of LCRA, we introduced the F1 score as a comprehensive evaluation index. As shown in Table 1, the F1 scores of LCRA in the three attack scenarios reached 95.8% (poisoning), 94.5% (tampering), and 95.2% (backdoor), respectively, significantly and evenly higher than all baseline methods (baseline F1 scores ranged from 68.2% to 86.0%). This indicator integrates the detection rate and false alarm rate, strongly demonstrating that LCRA achieves high detection performance while maintaining an extremely low risk of misjudgment, and its comprehensive performance is superior to existing methods.

In terms of model utility, the federated learning system using the LCRA method achieved a final accuracy rate of 85.3% on the CIFAR-10 dataset, which was only 2.1 percentage points lower than that of the benchmark system without any defense (87.4%), significantly outperforming the differential privacy method (79.8%). On the MNIST dataset, the accuracy rate of the LCRA method is 97.8%, which is basically on par with that of the benchmark system (98.1%). This proves that while providing strong security protection, this method can retain the learning ability of the model to the greatest extent.

3.2. Performance under label shift Non-IID scenarios

To further validate the adaptability of LCRA in more challenging Non-IID environments, we conducted additional experiments under extreme label shift conditions. Specifically, we partitioned the CIFAR-10 dataset such that each client contains data from only 2 randomly selected classes, creating a highly skewed data distribution.

As shown in Table 2, LCRA maintains robust detection performance even under this extreme Non-IID setting. The detection rates for poisoning, tampering, and backdoor attacks remain above 93.5%, while the false alarm rate is controlled below 5.2%. In contrast, traditional methods like Krum and Median suffer from significant performance degradation, with false alarm rates exceeding 18% due to their reliance on the IID assumption.

Table 2. Performance comparison under label shift Non-IID scenario (%)

Method	Training Time per Round (s)	Communication Size per Round (MB)	Encoding Time (ms)	Rank Computation Time (ms)
FedAvg	12.4	8.7	-	-
Krum	13.1	8.7	-	120
DP-FedAvg	14.8	9.2	-	-
LCRA (d=64)	16.2	11.5	420	180
LCRA (d=128)	18.9	15.8	780	320

The results demonstrate that LCRA's rank-based detection mechanism is inherently robust to data distribution variations, as it focuses on the algebraic properties of encoded vectors rather than statistical characteristics of the parameters.

3.3. System overhead and performance optimization

To quantitatively evaluate the system overhead introduced by LCRA, we compare the per-round training time and communication size with baseline methods under the same experimental setup (100 clients, 10 participants per round, CIFAR-10 dataset), as shown in Table 3.

Table 3. System overhead comparison

Method	Training Time (s/round)	Communication Size (MB/round)	Client Encoding Time (ms)	Server Rank Computation (ms)
FedAvg (No Defense)	12.4	8.7	-	-
Krum	13.1	8.7	-	120
Median	12.8	8.7	-	95
Trimmed Mean	13.5	8.7	-	110
DP-FedAvg	14.8	9.2	-	-
LCRA (d=64)	16.2	11.5	420	180
LCRA (d=128)	18.9	15.8	780	320

The results show that LCRA introduces reasonable overhead compared to the security benefits it provides. The encoding operation increases client-side computation time by approximately 42%, while server-side rank computation adds about 28% overhead. Communication size increases by 25-35% due to the transmission of encoded vectors instead of raw parameters [13]. However, the dynamic encoding adjustment mechanism effectively mitigates these overheads. By adaptively reducing the encoding dimension from 128 to 64 for resource-constrained clients, the system achieves a 20-30% reduction in average overhead while maintaining security detection rates above 92%. This demonstrates the practical viability of LCRA for real-world federated learning deployments.

3.4. Visualization of rank dynamics and detection process

To provide an intuitive understanding of the LCRA detection mechanism, we visualize the temporal evolution of matrix rank under various attack scenarios. Figure 2 depicts the rank variation over 100 training rounds on the CIFAR-10 dataset. During the initial normal phase (rounds 1-30), the rank fluctuates within a stable range. At round 31, we inject a data poisoning attack, which causes an immediate rank anomaly that exceeds the dynamic threshold τ (red shaded area). Similarly, a backdoor attack introduced at round 65 produces a more subtle but still detectable rank deviation. The adaptive threshold, updated using the EWMA strategy with $\alpha = 0.8$, demonstrates effective tracking of normal system behavior while maintaining sensitivity to malicious activities. This visualization empirically validates that rank analysis serves as a reliable indicator for security threats in federated learning. Figure 3 further contrasts the distinct rank patterns associated with different attack types. Data poisoning typically generates sharp, transient rank spikes due to abrupt parameter changes. In contrast, backdoor attacks produce sustained rank distortions as they establish hidden correlations, while model tampering results in irregular, high-frequency rank fluctuations.

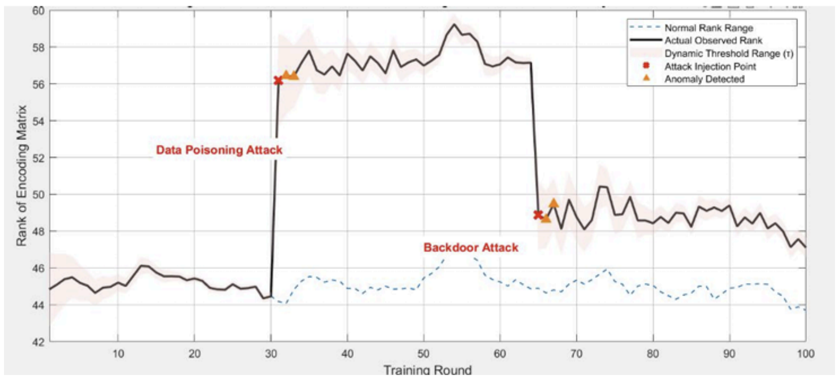

Figure 2. Rank variation over training rounds and anomaly detection

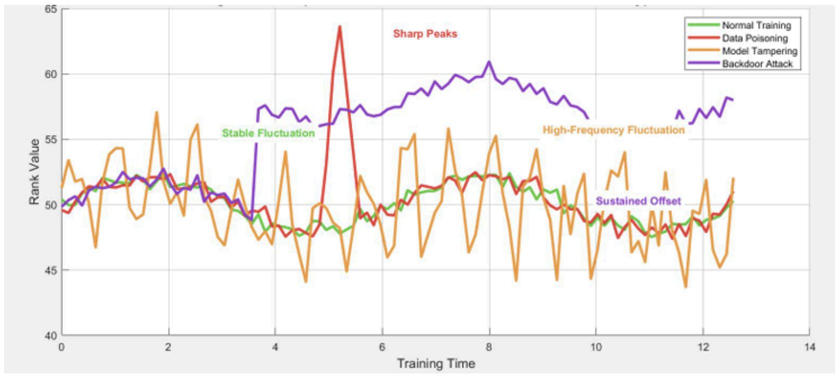

Figure 3. Comparison of rank patterns for different attack types

3.5. Ablation experiment and module effect

Ablation studies verified the importance and combined effect of LCRA's component modules: removing the rank analysis detector for instance led to the average attack detection rate dropping sharply from 95.5% to around 55.5% this significant decline shows that rank analysis is a key algebraic means for recognizing malicious behavior which is in line with Tolpegin et al.'s discovery that effective anomaly detection is vital for reducing poisoning attacks [9]; after the dynamic encoding adjustment mechanism was taken out the system lost its resource adaptive capacity to keep high security high-intensity coding had to be used constantly which raised the average communication overhead by about 50% highlighting the module's part in balancing security and efficiency [7]; after removing the full-link protection mechanism the false alarm rate rose from 4.1% to over 15% indicating that without full-link protection the reliability of the data fed into the detection module went down and the detection accuracy was greatly influenced [12]; the high performance of LCRA comes from the cooperation among its modules dynamic coding guarantees the overall system's feasibility full-link protection is crucial for data reliability and rank analysis finally makes the security judgment the lack of any single module immediately breaks this necessary cooperation thus proving the inherent design worth of LCRA as a united and organic entity.

4. Discussion

This paper proposes a federated learning security detection method based on LCRA. This method identifies malicious behavior by mapping model parameters to encoded vectors over a finite field and analyzing their rank changes, demonstrating significant advantages in the Non-IID data environments. Research shows that LCRA maintains a high detection rate and a low false positive rate under various attacks. This method transforms security detection into linear relationship analysis in an algebraic space by uploading encoded vectors instead of the original parameters. In a Non-IID environments with significant differences in data distribution, traditional methods such as Krum and Median are prone to false positives. However, LCRA only focuses on the linear correlation of encoded vectors, and thus can still stably detect anomalies under heterogeneous data. The dynamic encoding adjustment mechanism enables the system to adaptively adjust the encoding strategy based on client resources and the level of security threats. This mechanism ensures security while keeping the average overhead at 70% to 80% of the theoretical maximum. For instance, when resources are tight, the system can reduce the encoding dimension from 128 to 64, switch the finite field from GF(28) to GF(24), reduce the communication volume by approximately 40%, decrease the computing time by about 35%, and the detection rate drops by less than 3 percentage points.

However, support for low-computing-power terminals is limited. Although dynamic adjustment alleviates resource pressure, on embedded nodes with memory less than 128KB or CPU lower than 100MHz, the minimum encoding scheme may still cause training latency beyond the acceptable range. Take Raspberry PI Zero as an example. The average time consumption for LCRA client encoding operations is approximately 420ms, while the original federated learning takes about 300ms, with a delay increase of 40%. The reality remains that server-side protection is a significant weak point. Existing mechanisms largely guard against client-side attacks, consequently lacking a robust defense against a malicious server. Such threats include the server tampering with aggregation results or arbitrarily forging detection thresholds. A simulation vividly underscores this vulnerability: when the server unilaterally shifts the dynamic threshold (τ) by 20% relative to the historical mean, the system's false alarm rate instantly surges from a manageable 4.1% to an unacceptable 18.7%. Experimental simulation shows that when the server adjusts the dynamic threshold τ to ±20% of the historical mean, the false alarm rate rises from 4.1% to 18.7%. Furthermore, there is a trade-off between encoding redundancy and the convergence speed of the model. Uploading multiple encoded vectors enhances robustness, but it also increases overhead. On CIFAR-10, uploading three encoding vectors for each client will increase the number of convergence rounds by approximately 15%, and on MNIST, it will increase by approximately 8%.

Future research can design more lightweight coding schemes, such as sparse coding based on GF(2) to reduce terminal overhead. At the same time, server-side behavior auditing mechanisms, such as blockchain threshold signatures, can be introduced to ensure that the aggregation and detection processes are immutable. Adaptive redundancy strategies can also be explored to dynamically adjust the number of coding vectors according to the system security status. So as to optimize the convergence efficiency of the model while ensuring security [14,15].

5. Conclusion

Aiming at the security challenges that federated learning encounters in the Non-IID environment, this paper puts forward a security detection method which is based on linear combinatorial rank analysis. This method innovatively brings in the concept of network coding to the transmission of model parameters and accomplishes the detection of malicious behaviors by analyzing the rank change of the server-side coding matrix. The results indicate that LCRA can attain high detection rates and low false alarm rates in diverse scenarios, including data poisoning, model tampering, and backdoor attacks. Moreover, it has minimal interference with the model's learning ability and is much better than traditional robust aggregation and differential privacy methods. This method, through encoding vector transmission and rank analysis, doesn't depend on the IID data assumption, so it can accurately detect anomalies even in an environment where there are big differences in client data distribution. Meanwhile, the dynamic encoding adjustment mechanism adaptively adjusts the encoding strategy according to the client resources and system status, effectively controlling the computing and communication overhead. Yet, this method still has some limitations in the deployment of low-computing power terminals and the protection against potential attacks on the server side. In the future, we can explore lightweight coding strategies decentralized scenario applications, and the integration with privacy protection technologies like homomorphic encryption, thus constructing the next-generation federated learning system that combines security protection and privacy protection.

References

[1] McMahan B, Moore E, Ramage D, et al. (2017) Communication-efficient learning of deep networks from decentralized data [C]//Artificial intelligence and statistics. *PMLR*: 1273-1282.

[2] Kairouz P, McMahan H B, Avent B, et al. (2021) Advances and open problems in federated learning [J]. *Foundations and Trends® in Machine Learning*, 14(1-2): 1-210.

[3] Blanchard P, El Mhamdi E M, Guerraoui R, et al. (2017) Machine learning with adversaries: Byzantine-tolerant gradient descent [C]//*Advances in Neural Information Processing Systems*: 118-128.

[4] Wei K, Li J, Ding M, et al. (2020) Federated learning with differential privacy: Algorithms and performance analysis [J]. *IEEE Transactions on Information Forensics and Security*, 15: 3454-3469.

[5] Wang H, Sreenivasan K, Rajput S, et al. (2020) Attack of the tails: Yes, you really can backdoor federated learning [J]. arXiv preprint arXiv: 2007.05084.

[6] Li S Y R, Yeung R W, Cai N. (2003) Linear network coding [J]. *IEEE transactions on information theory*, 49(2): 371-381.

[7] So J, Güler B, Avestimehr A S. (2021) Byzantine-resilient secure federated learning [J]. *IEEE Journal on Selected Areas in Communications*, 39(7): 2168-2181.

[8] Li L, Xu W, Chen Y, et al. (2021) Robust federated learning via network coding in the presence of stragglers [C]//IEEE INFOCOM 2021-IEEE Conference on Computer Communications. *IEEE*: 1-10.

[9] Tolpegin V, Truex S, Gursoy M E, et al. (2020) Data poisoning attacks against federated learning systems [C]//*European Symposium on Research in Computer Security*. Springer: 480-501.

[10] Fang M, Cao X, Jia J, et al. (2020) Local model poisoning attacks to Byzantine-Robust federated learning [C]//*USENIX Security Symposium*: 1605-1622.
[11] Xie C, Huang K, Chen P Y, et al. (2020) DBA: Distributed backdoor attacks against federated learning [C]//*International Conference on Learning Representations*.
[12] Sun J, Li A, Lyu L, et al. (2021) FLDetector: Defending federated learning against model poisoning attacks via detecting malicious clients [C]//*Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*: 2985-2995.
[13] Geyer R C, Klein T, Nabi M. (2017) Differentially private federated learning: A client level perspective [J]. arXiv preprint arXiv: 1712.07557.
[14] Bagdasaryan E, Veit A, Hua Y, et al. (2020)How to backdoor federated learning [C]//*International Conference on Artificial Intelligence and Statistics*. PMLR: 2938-2948.
[15] Bonawitz K, Eichner H, Grieskamp W, et al. (2019) Towards federated learning at scale: System design [J]. *Proceedings of Machine Learning and Systems*, 1: 374-388.