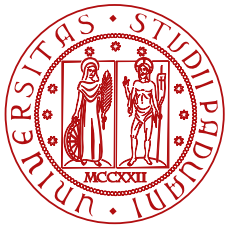

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

UNIVERSITY OF PADUA
DEPARTMENT OF INFORMATION ENGINEERING

Master's Degree in ICT for Internet and Multimedia

Trust-Aware Client Scoring in Federated Learning via Contrastive Attention and Representation Clustering

Candidate:

Reza Sarkhosh

Student ID: 2109663

Supervisor:

Prof. Federica Battisti

University of Padua

Co-supervisor:

Prof. Sadok Ben Yahia

Tallinn University of Technology (TalTech)

Academic Year: 2025 – 2026

Graduation Date: July 7, 2026

Acknowledgments

I would like to express my deepest gratitude to my supervisors, Prof. Sadok Ben Yahia from Tallinn University of Technology (TalTech) and Prof. Federica Battisti from the University of Padua, for their continuous guidance and support throughout the course of this thesis. Prof. Ben Yahia has provided exceptional mentorship, insightful feedback, and constant encouragement during my Erasmus research period at TalTech. His expertise and enthusiasm have greatly shaped both the technical and conceptual aspects of this work. I am equally thankful to Prof. Battisti for her kind supervision, timely feedback, and continuous academic support from the University of Padua, where I will be defending this thesis.

Finally, I would like to express my heartfelt appreciation to my family and friends for their unwavering encouragement and understanding during this journey. Their presence and support have been a constant source of motivation and strength throughout my studies.

Abstract

Federated machine learning is a crucial topic nowadays, playing a critical role in ensuring robustness against adversarial attacks and noisy participants and enhancing data privacy. However, most current approaches rely on static assumptions or prior information about client behavior. In this thesis, we introduce a novel trust scoring mechanism based on attention, clustering, and a dominant-cluster contrastive learning loss, where trust is dynamically assigned to clients based on their similarity to the dominant clusters. We partition the CIFAR-10 and GTSRB datasets across clients and extract their embeddings. In addition, we generate adversarial-style clients by poisoning their training dataset with strong visual perturbations, many-to-one label flipping, and backdoor-style triggers to simulate malicious and low-quality clients. First, each client performs an intra-client self-attention locally (on the client-side) to compress its D×N embeddings set into a single representation vector r_k and send that representation vector to the server instead of sharing raw embeddings, which would pose a significant privacy risk. Then, on the server side, we use a learnable inter-client attention module that compares all the representation vectors coming from all clients to produce trust scores. Also, there is a hybrid clustering method (K-means + DBSCAN) applied on the same set of representation vectors to detect major clusters and outlier clients. Our contrastive loss encourages higher trust among clients in large and coherent clusters while penalizing high trust in smaller clusters. This approach models a "rich get richer" dynamic that suppresses noisy or adversarial clients. Experiments on CIFAR-10 and GTSRB demonstrate that the proposed method effectively isolates adversarial behavior, generalizes to unseen clients from the same data distribution, and improves global model performance under diverse poisoning-based adversarial scenarios. This zero-trust, self-organizing framework provides a resilient and practical solution for trust-aware aggregation in federated learning.

Sommario

L'apprendimento federato è oggi un tema di grande rilievo, poiché svolge un ruolo cruciale nel garantire la robustezza contro attacchi avversari e partecipanti rumorosi, oltre a tutelare la privacy dei dati. Tuttavia, la maggior parte degli approcci attualmente disponibili si basa su assunzioni statiche o su informazioni a priori riguardo al comportamento dei client. In questa tesi, viene proposto un nuovo meccanismo di assegnazione del punteggio di affidabilità basato su attenzione, clustering e una funzione di perdita contrastiva orientata al cluster dominante, in cui l'affidabilità viene assegnata dinamicamente ai client in base alla loro somiglianza con i cluster dominanti. I dataset CIFAR-10 e GTSRB vengono partizionati tra i client e da essi vengono estratti gli embedding. Inoltre, sono stati generati client di tipo avversario contaminandone il dataset di addestramento mediante forti perturbazioni visive, label flipping di tipo many-to-one e trigger di tipo backdoor, al fine di simulare client malevoli e di bassa qualità. Innanzitutto, ciascun client esegue localmente (lato client) una self-attention intra-client per comprimere il proprio insieme di embedding di dimensione N×D in un unico vettore di rappresentazione r_k, che viene poi inviato al server al posto degli embedding grezzi, la cui condivisione comporterebbe un significativo rischio per la privacy. Successivamente, sul lato server, viene utilizzato un modulo di attenzione inter-client apprendibile che confronta tutti i vettori di rappresentazione provenienti dai client per produrre i punteggi di affidabilità. Inoltre, sullo stesso insieme di vettori di rappresentazione viene applicato un metodo di clustering ibrido (K-means + DBSCAN) per individuare i cluster principali e i client outlier. La funzione di perdita contrastiva proposta favorisce un'affidabilità più elevata tra i client appartenenti a cluster ampi e coerenti, penalizzando al contempo l'attribuzione di un'affidabilità elevata ai client di cluster più piccoli. Questo approccio modella una dinamica di tipo "il ricco diventa più ricco", che consente di sopprimere i client rumorosi o avversari. I risultati sperimentali evidenziano che il modello generalizza a client mai osservati in precedenza ed è in grado di isolare con successo il comportamento avversario senza supervisione. Questo meccanismo di tipo zero-trust e auto-organizzante fornisce una soluzione scalabile e resiliente per l'aggregazione basata sull'affidabilità nell'apprendimento federato.

Contents

1

Introduction

Federated learning (FL) is an emerging approach that allows multiple organizations or devices to collaboratively train their machine learning models without sharing their raw data [8]. This makes it especially valuable in sensitive fields like healthcare, where data privacy and security are critical concerns. However, this decentralized nature of the FL introduces fundamental trust challenges: in this system, participants may deploy low-quality or even malicious updates to decrease the global model's performance. Detecting such adversarial or unreliable clients is still a compelling and thriving challenge [7, 16], particularly in the absence of the base model as ground truth. The trust Scores are of utmost importance in FL because of the following reasons:

1. **Addressing Heterogeneity**: FL systems involve participants with varying data, system capabilities, and behavioral patterns, which can degrade performance and introduce vulnerabilities.

2. **Mitigating Adversarial Attacks**: Malicious participants can attempt to introduce poisoned model updates or exploit system vulnerabilities. Trust scores help identify and reduce the impact of these threats.

3. **Ensuring Fairness and Robustness**: Trust scores provide a basis for fair reward allocation and help select reliable participants, thereby improving the overall robustness and fairness of the FL process.

Existing trust evaluation methods in federated learning often rely on direct access to client data, or they need prior information about clients, which contradicts the core privacy-preserving principle of FL. Moreover, many approaches lack robustness when facing unseen or stealthy adversarial behaviors, which makes them generate unreliable aggregation results.

In this thesis, we introduce a novel attention-based trust scoring mechanism that dynamically estimates the reliability of the clients who want to participate

in the aggregation step of the global model, under the assumption of client-zero-trusted. Our approach relies solely on representation-level information extracted from each client's models, without requiring having access to the private data, explicit attack labels, or any prior trust history about clients. By analyzing the structure of the coming representations based on their similarity and inconsistencies with other client embeddings, the system dynamically assigns trust scores. By doing so, it also suppresses the influence of the noisy or adversarial clients during the aggregation.

To better reflect real-world FL conditions, we divide the datasets *CIFAR-10* [10] and *GTSRB* [15] into disjoint subsets, each allocated to a separate client. Furthermore, adversarial clients are simulated through input perturbations and intentional label flipping to mimic malicious behavior, with the role in degrading the models' overall performance. Our approach combines an attention-based trust module with a hybrid clustering mechanism (KMeans + DBSCAN). It is trained via a contrastive loss that encourages self-organizing behavior based on inter-client consistency.

Experimental results show that our method is capable of generalizing to previously unseen client instances under similar data modalities. The proposed mechanism serves as a scalable, interpretable, and robust defense strategy for real-world federated systems.

Contributions

The main contributions of this work are summarized as follows:

- We propose a zero-trust, representation-level trust scoring framework for FL that does not rely on trusted data or prior assumptions about clients.
- We introduce a contrastive, cluster-aware regularization strategy that aligns learned trust scores with inter-client structural consistency.
- We design an attention-based trust scoring mechanism combined with unsupervised clustering to identify and suppress adversarial clients during aggregation.
- We empirically validate the robustness of the proposed framework under diverse poisoning-based adversarial scenarios on two benchmark datasets, including CIFAR-10 and GTSRB.

Part of the work presented in this thesis has been accepted for publication as: R. Sarkhosh, S. Ben Yahia, C. Esposito, A. Faiz, "Trust-Aware Client Scoring in Federated Learning via Contrastive Attention and Representation Clustering," *IEEE COMPSAC 2026* (full paper).

2

Literature Review

2.1 Introduction

The trust score in FL is a metric that quantifies the reliability of a participant (client or server) by assessing factors like behavioral consistency, model update integrity, past security events, and network activity. This score helps in selecting trustworthy participants for the FL process, improving model accuracy, robustness, and security against malicious actors by filtering out unreliable or compromised nodes [20, 16]. In this respect, a client with a history of malicious activity or consistently poor model updates would receive a low trust score and be excluded from contributing to the model.

2.2 Trust Mechanisms in Federated Learning

The goal of Trust mechanisms in federated learning is to quantify the reliability of participating clients based on their behavior, model updates, and historical performance. All the approaches that exist, can be categorized into statistical, reputation-based, and model-update–driven strategies. All the statistical trust mechanisms assign trust scores based on the consistency of the client's updates in each round, and often by using similarity measurements to detect abnormal behaviors [9].

for the Reputation-based category, models accumulate long-term indicators of client reliability, rewarding stable contributors and penalizing those with unstable patterns. Another line of work evaluates trust directly from model updates by analyzing gradient norms, cosine similarity, or the direction of updates relative to the global model [16]. Finally, all of the introduced mechanisms are trying to identify participants whose behavior aligns with the global objective and to down-weight or exclude those whose updates may harm training

stability.

2.3 Defense Strategies Against Adversarial Clients

There are several forms of adversarial behaviors in federated learning such as data poisoning, model poisoning, and Byzantine attacks , where malicious clients intentionally manipulate their updates to decrease global model performance [2]. Existing defense strategies try to find and mitigate all these threats through robust aggregation rules, anomaly detection, and sometimes using adversarial training techniques.

Classical Byzantine-resilient methods such as Krum, Multi-Krum, Trimmed Mean, and coordinate-wise Median try to filter out abnormal updates by keeping only those that are statistically similar to the majority [18]. Other defense strategies are made with clustering model updates to isolate all of the outliers, leveraging gradient similarity metrics to detect suspicious contributions, or using trust-based weighting such as FLTrust, which requires the server side to hold a small root dataset to validate updates [3]. More of the recent approaches also are considering defenses against Sybil attacks, where adversaries spawn multiple fake clients; methods like FoolsGold reduce the influence of highly correlated gradients to prevent collusion [7]. Despite their effectiveness, the majority of these methods struggle when the adversary is stealthy, when data heterogeneity is high, or when reference models are unavailable, highlighting the need for more adaptive and data-free defense mechanisms.

2.4 Trust Score Computation and Evaluation Methods

Worthy of mention, trust scores are often calculated based on a combination of factors:

1. Participation Frequency and Consistency: How often a client participates in training and the consistency of their submitted updates are indicators of reliability.
2. Contribution Effectiveness: The quality of a client's model updates and their actual contribution to the global model's accuracy are evaluated.
3. Model Consistency and Accuracy: The similarity of a client's updates to those from other reputable clients or the actual global model can influence their trust score.
4. Network Connectivity: Reliable network connectivity is critical for timely communication of model updates, and its stability can impact a client's trust score.

2.5 Confidence-Based Filtering Methods

Confidence-based filtering is one of the important categories of defense mechanisms in federated learning. In this kind of approach, there is a score for each client that shows how trustworthy or reliable the contribution is expected to be. Instead of treating every single update equally, the server evaluates all these scores from clients and selectively down-weights and discards updates deemed suspicious. This approach allows the aggregation phase to focus more on clients that are stable, consistent, and norm-aligned model updates, while it reduces the impact of malicious and low quality participants. Furthermore, the important part of the confidence-based scoring is that it does not require access to raw client data; instead, it relies solely on update-level signals such as cosine similarity, deviation from expected norms, or cross client consistency. In situations where privacy restrictions prohibit a more thorough examination of local datasets, these techniques thus offer a simple yet powerful defense strategy. However, confidence-based filtering may struggle in highly heterogeneous environments, where benign updates naturally are different from other clients, which leads to the unintended penalization of honest participants.

2.6 Multidimensional and Hierarchical Trust Models

Multidimensional and hierarchical trust models increase their perspective and go beyond single-metric evaluations by incorporating several complementary indicators of client behavior. In this kind of approach, instead of relying only on one signal - such as gradient similarity or update magnitudes - these models try to integrate multiple dimensions (factors), including historical reliability, contribution quality, responsiveness, and statistical consistency across training rounds. We make the system understand the behavior of each user by using trust in layered components, such as short-term versus long-term trust and behavioral trust versus data-related trust. This approach reduces the risk of misjudging clients based on noisy or one-off deviations and enables being more informed in aggregation decisions. However, while these models provide richer representations of trust, they often require additional metadata or long-term monitoring, which may not always be feasible in privacy-preserving or highly dynamic federated environments.

2.7 Attention and Representation Learning in Federated Settings

Attention mechanisms and representation learning have gained traction in Federated learning as one of the effective tools for better understanding the structure and quality of client updates without requiring them to share raw data. Instead of working with model updates as flat numerical vectors, representation learning approach extracts richer embeddings that capture high level patterns such as feature, consistency, inter-class separation, or latent model characteristics.

On the other hand, attention modules conduct an identification process to find coherent behavior with respect to the broader population. By assigning higher weight to updates that align with global patterns—and diminishing the influence of isolated or inconsistent ones among all — attention makes it feasible to make a more adaptive and context-aware form of aggregation. This process is particularly valuable and important in zero-trust and heterogeneous environments where client updates may vary substantially due to local data differences or malicious intent. Compared to conventional aggregation schemes, attention-based models may add more computational overhead and rely on the quality of the learned representations, despite their demonstrated strong potential to improve robustness and interpretability.

2.8 Comparison of Existing Approaches

Among multidimensional trust models, Fed-DTB [1] is one of the most current methods, which computes a comprehensive trust score from seven contextual indicators (such as network condition, past behavior, IDS alerts, and contribution history), with an adaptive weighting mechanism that prioritizes different factors based on current conditions. Compared to one-dimensional similarity-based defenses, this design offers a more comprehensive evaluation of credibility. However, its main limitation is the reliance on domain-specific metrics and external monitoring infrastructure (e.g., vehicular data in IoV scenarios), which restricts its applicability in more general FL settings such as healthcare.

Another approach among the multidimensional trust models is Multi-Level Trust [12], which proposes a hierarchical trust evaluation framework that simultaneously considers behavioral trust, model-based trust (e.g., local accuracy and gradient consistency), and historical trust (long-term reliability over rounds). In order to systematically down-weight the clients who exhibit suspicious or inconsistent behavior, these multi-level trust scores are then incorporated into a hierarchical aggregation mechanism. This method captures both the static and temporal aspects of client reliability, in contrast to one-dimensional similarity-based defenses like FoolsGold or FLTrust. Its primary drawback, though, is the

extra work involved in monitoring and combining several metrics over time, which might limit its scalability to extensive federated learning implementations.

In the category of robust aggregation methods, *Krum* and *Multi-Krum* [2] select the client updates that are closest to their neighbors and remove outliers under the Byzantine assumptions. This works for a small number of malicious clients but throws away useful information in general, especially in non-IID settings where honest clients can be very different. The Coordinate-wise Median [18] computes the median of each parameter rather than the average, which is robust to some corrupted updates. However, the analysis assumes i.i.d. data and its performance deteriorates with increasing heterogeneity of data.

Our framework can be considered a hybrid approach, which combines elements of both confidence-based filtering and multidimensional trust modeling and offers a more flexible and data-driven alternative compared to FLTrust. Unlike FLTrust, which requires a root data set and uses cosine similarity to filter out malicious clients, our method does not assume access to any clean data. Compared to FoolsGold, which heuristically down-weights colluding clients by monitoring cosine similarities of their gradient directions over time, our framework takes a learning-based route. In the initial stage, we apply K-Means and DBSCAN clustering to capture structural patterns and isolate potential anomalies. We use a contrastive cluster-aware loss to power up the separation between honest and adversarial clients. In the last phase, we deploy a cross-client attention mechanism to dynamically assign trust scores during aggregation.

This three-stage design enables our system to adaptively suppress poisoned or colluding clients and, at the same time, maintain strong performance, where traditional rule-based defenses such as FLTrust and FoolsGold often face limitations.

2.9 Limitations of Current Methods

Most of the defense mechanisms currently used in federated learning still face significant limitations. Many approaches rely on assumptions such as access to clean reference data, stable client behavior, or domain-specific signals, which are not always available in real-world problems. Confidence-based filters often struggle under high data heterogeneity, while multidimensional trust models introduce additional monitoring and computational overhead. All these challenges highlight the need for more adaptive, data-free, and robust trust mechanisms that also work in zero-trust federated environments.

Table 2.1: Comparison of related defense methods in federated learning.

Method	Category	Core Idea	Strengths	Limitations
Federated Trust ([14])	Trustworthy FL Framework	Modular secure aggregation, anomaly detection, incentive-aware trust scoring, and reliable communication.	Holistic design; captures long-term trust.	High-level; no specific aggregation defenses.
FLTrust ([3])	Confidence-based filtering	Server uses trusted data and cosine similarity to evaluate client updates.	Simple and effective for detecting poisoned updates.	Needs trusted data; weak under non-IID.
FoolsGold ([6])	Confidence-based filtering	Down-weights clients with similar updates to counter Sybil and collusion attacks.	Good against Sybil attacks; no trusted data needed.	Vulnerable to adaptive adversaries; heuristic-based.
Fed-DTB ([1])	Multi dimensional trust modeling	Adaptive trust scoring using multiple contextual indicators in IoV-FL.	Strong early robustness; multi-indicator trust.	Requires domain-specific signals; increased complexity.
Krum / Multi-Krum ([2])	Robust aggregation rule	Selects the client update(s) with minimal summed distance to nearest neighbors, excluding Byzantine outliers during aggregation.	Strong theoretical guarantees against bounded adversaries; no trusted data required.	Discards useful updates; degrades under non-IID data.
Coord. Median ([18])	Robust aggregation rule	Replaces mean aggregation with coordinate-wise median to suppress corrupted updates.	Simple to implement; robust to a minority of corruptions; no reference dataset.	Assumes IID data; degrades significantly under heterogeneity.
Multi-Level Trust ([12])	Multi dimensional trust modeling	Behavioral + model-based + historical hierarchical trust.	Captures long-term reliability; good for dynamic environments.	Requires tracking multiple indicators; higher computational cost.
Ours	Hybrid	KMeans + DBSCAN clustering + contrastive loss + attention-based trust scoring.	Strong adversary separation; robust under heterogeneity.	More complex; requires tuning.

3 Methodology

3.1 Overview of the Proposed Framework

FL is one of the popular topics nowadays. It enables multiple clients to collaboratively train machine learning models without sharing their raw data [8, 17]. However, this decentralized nature of the FL raises important trust issues: How can we ensure all the participants are collaborating honestly and contributing meaningfully to the global model?

In many real world applications, such as healthcare and other applications dealing with sensitive data, a central server may not know in advance which clients are trustworthy [11]. Furthermore, adversarial or malicious clients could attempt to poison the model by providing outlier representations or deceptive updates. This means a *zero-trust* environment: the client is not trusted by default.

To address this issue, we introduce a trust scoring system provided in Figure 3.1 based on representations. In this case, all clients train locally and perform an intra-client self-attention in the client-side and compress their N×D into a single representation vector r_k and send these vectors to the server. On the server-side, our framework assigns a trust score to all of the clients only based on their representations, without relying on explicit attack labels, trusted reference clients, or prior trust history. Our goal is to develop a self-organizing mechanism that:

- Rewards clients that are consistent and aligned with dominant representation patterns;
- Penalizes clients that are isolated, inconsistent, or possibly adversarial;
- Generalizes to unseen clients without requiring retraining.

As Figure 3.1 depicts it, the overall pipeline of the process is as follows. Stage (1) includes client-side operations, where each client extracts local embeddings from its private data and applies an intra-client self-attention pooling

module to obtain a fixed-length representation vector r_k. All these operations are performed entirely on the client side to limit information sharing. Then, clients participate in the standard FL procedure by sharing model updates for aggregation and transmitting their representation vectors r_k to the server for the trust scoring process. In stage (2), on the server-side, our learnable inter-client attention module compares all the representation vectors coming from all clients and calculates the initial trust scores for each client. In stage (3), our framework applies K-Means and DBSCAN to assign the best describing cluster to each one of the clients, and also detect outliers. After determining the clusters and outliers, in stage (4), a contrastive cluster-aware loss is applied to align trust scores with client similarities and cluster assignments. It encourages consistent trust within large clusters, penalizes over-similarity in small clusters, and softly reduces trust for isolated clients. In stages (5) and (6), the resulting trust scores are used to weight or filter out client contributions during the aggregation process, and in stage (7), the aggregated parameters are sent back to all clients to complete the federated round. By this, after a certain number of steps, the model can detect adversarial clients even if they collude. For completeness, hyperparameter selection for the contrastive loss was performed offline during the experimental phase using an LLM-assisted exploration strategy, and is not part of the deployed FL pipeline; we describe it separately in Section 3.6. These steps are thoroughly described in the following.

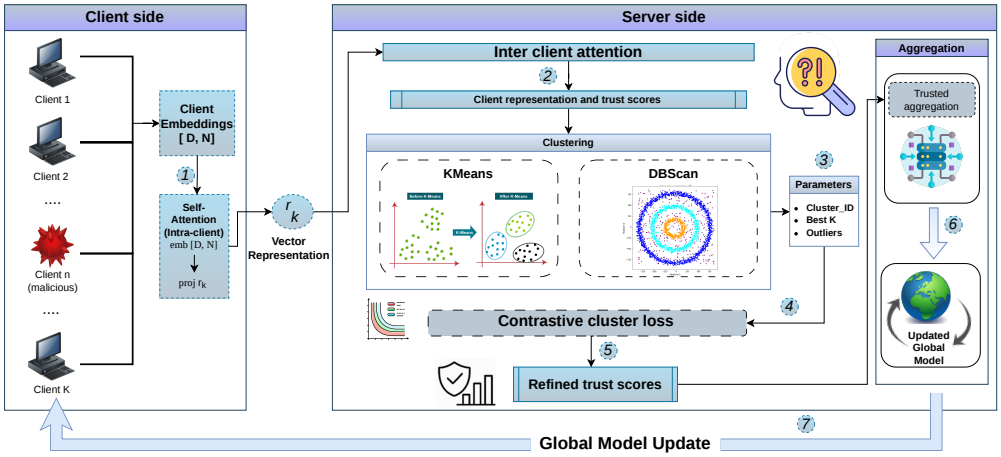

Figure 3.1: Overview of the proposed trust scoring and regularization pipeline. The illustrated pipeline represents the deployed system. Offline hyperparameter selection (Offline-LLM-Guided Tuning) is not part of the runtime architecture and is described separately in the experimental setup.

3.2 Client Embedding Generation

We partition the global training dataset into several disjoint local subsets, where each one of the local subsets corresponds to a single federated client,

which enables us to create client-level representations without training fully independent models from scratch. By this strategy, the underlying data distribution across clients is preserved while the computational and communication overheads are significantly reduced.

For each dataset, we create a set of **benign clients** and a smaller subset of **adversarial clients**. Benign clients are assigned clean, non-overlapping portions of the original training data and strictly follow the prescribed training protocol. In contrast, adversarial clients operate on locally poisoned datasets, where a controlled fraction of samples is intentionally modified prior to local training.

Adversarial samples are generated directly from each malicious client's local subset using a *class-conditional many-to-one poisoning strategy*. Specifically, samples belonging to a predefined set of source classes are selected, visually perturbed, and relabeled to a single target class. The poisoning process consists of the following steps:

- **Selective Sample Targeting:** Only samples from designated source classes are considered for poisoning, ensuring that the attack remains targeted and stealthy.

- **Strong Visual Perturbations:** A sequence of realistic image transformations—including affine distortions, color jittering, perspective warping, Gaussian blurring, random erasing, and trigger-based poisoning patterns—is applied to alter visual appearance while preserving overall plausibility.

- **Many-to-One Label Mapping:** The perturbed samples are relabeled to a single target class, inducing a systematic bias in the learned decision boundaries.

As a result, each adversarial client holds a mixture of clean and poisoned samples while maintaining the same dataset size as benign clients. This setup reflects a realistic threat model in which malicious participants comply with the training protocol but manipulate their local data distribution to influence the global model.

Figure 3.2: Clean and poisoned samples from CIFAR-10 and GTSRB under different attack strategies. Left: clean image, Right: poisoned image with visual distortions, label changes, and backdoor-style perturbations.

Each client trains a lightweight classification head on top of a shared feature extractor using its local dataset. From this process, we extract fixed-dimensional embedding vectors (of size 2,048) that summarize the semantic characteristics of each client's data. The raw per-sample embeddings from each client never leave

the client device; only a single representation vector per client is transmitted to the server. These embeddings serve as the primary input to the proposed trust scoring and clustering framework, enabling the identification and isolation of adversarial clients based on representation-level discrepancies rather than raw performance metrics.

3.3 Attention-Based Trust Scoring Module

We design a two-stage attention-based module that operates directly on the tensor $X \in \mathbb{R}^{K \times D \times N}$ to compute trust scores for a pool of K clients, with each client represented by a set of N embeddings of dimension D. Our goal is to assign a scalar trust score t_k to each client k, based on the semantic consistency and inter-client similarity of their embeddings.

Stage 1: Intra-client Embedding Pooling. For each client k, we apply a multihead self-attention mechanism over their local embedding set $X_k \in \mathbb{R}^{D \times N}$ to obtain a fixed-length client representation $r_k \in \mathbb{R}^D$. This stage captures intra-client semantic coherence and highlights the dominant latent patterns across the client's samples. The architecture includes a Transformer-style residual structure with layer normalization and feedforward sublayers. The architecture also employs mean pooling across the sequence dimension:

$$r_k = \texttt{SelfAttentionPooling}(X_k) \quad \text{for } k = 1, \ldots, K.$$

Stage 2: Inter-client Attention. Once we obtain all client representations $R = [r_1, \ldots, r_K] \in \mathbb{R}^{K \times D}$, we project them into query, key, and value spaces using learnable linear layers.

$$Q = RW_Q, \quad K_{\text{att}} = RW_K, \quad V = RW_V.$$

We compute pairwise attention weights using scaled dot-product attention:

$$A = \texttt{softmax}\left(\frac{QK_{\text{att}}^\top}{\sqrt{D}}\right),$$

which aggregates information across clients. The attended features $Z = AV$ are then passed through a final projection to produce scalar trust scores.

$$T = \texttt{sigmoid}(ZW_T) \in \mathbb{R}^K.$$

These scores are finally min-max normalized:

$$t_k = \frac{T_k - \min(T)}{\max(T) - \min(T)},$$

ensuring all trust values lie within the unit interval $[0, 1]$.

Interpretation. This architecture implements a hybrid trust mechanism: In the first stage, it ensures that the embedding pool of each client is internally consistent (through self-attention), while the second stage learns to assign trust scores by comparing each of the clients with all the others (through inter-client attention). This self-organizing structure is especially designed to facilitate identifying outlier or adversarial clients, without requiring any supervision or ground-truth labels.

3.4 Clustering-Based Trust Regularization

We further regularize attention-based trust scoring with a discrete structural signal based on unsupervised clustering, while it provides a continuous and learnable mechanism. The intuition is as follows: trustworthy clients tend to form coherent clusters in the representation space, while noisy or adversarial clients are often outliers or are in small clusters. Our approach combines two clustering techniques:

1. **Step 1: Adaptive K-Means Clustering**: We begin by applying K-Means clustering over the client representations $R \in \mathbb{R}^{K \times D}$. Instead of predefining the number of clusters, we search for the best value of $C \in \{2, 3, 4, 5\}$ by maximizing the silhouette score. This dynamic approach allows the model to adapt to different data distributions during training. The initial cluster assignments form our first client community.

2. **Step 2: Outlier detection via DBSCAN**: Next, we use DBSCAN to detect potential outlier clients—those that do not fit well into any dense cluster. DBSCAN is density-based and does not require a predefined number of clusters. If the number of detected outliers is reasonable (i.e., not zero and not too many), we mark them as a separate cluster; otherwise, we fall back to the KMeans output alone without considering the results coming from DBSCAN.

3. **Final Assignment**: We combine two clustering strategies to decide how clients are grouped: we use K-Means to assign clients to main clusters based on their similarity, while DBSCAN acts as a final protection for safety. If DBSCAN strongly flags a client as an outlier, we override its K-Means label and place it into a separate cluster.

This hybrid approach lets us build a structural map of trust: large, well-populated clusters typically represent coherent client groups — like those sharing similar data distributions — while tiny or isolated clusters often signal noisy or adversarial behavior. The contrastive loss then uses this structure to reinforce trust in the majority and penalize inconsistencies.

For example, with diverse client groups (e.g., CIFAR-10), K-Means separates them into clusters. However, in the cases where most of the clients belong to the same group, and only one adversarial client is present, DBSCAN helps by marking the outlier instead of forcing it into a normal cluster.

3.5 Contrastive Cluster-Aware Loss

To guide the training of our attention-based trust scoring, we introduce a new custom loss function that integrates both representation similarity and clustering structure into a contrastive learning objective. The core idea is straightforward: **clients belonging to large and well-structured clusters are considered more trustworthy and thus receive higher rewards**. In contrast, **clients from smaller or poorly formed clusters are penalized to discourage overestimation of their trustworthiness**. This mechanism aligns with a merit-based aggregation strategy. The full implementation of the proposed loss is provided in Algorithm 1. The latter shows a step-by-step calculation of our contrastive cluster-aware trust loss. For each pair of clients (c.f., Lines 8 – 32), the loss considers their similarity, trust score difference, and cluster sizes. Pairs from similar and large clusters are rewarded, while overly similar clients from small clusters are penalized. A soft penalty also discourages high trust scores for clients in small clusters (c.f., Lines 32 – 34). All components are normalized and combined to form the final loss (Line 36).

Let K denote the number of clients. For each pair of clients (i, j), we compute the squared trust difference $(T_i - T_j)^2$ and weigh it based on the cosine similarity of their embeddings. If two clients are highly similar ($\text{sim}_{i,j} > \tau$), then we penalize large trust differences to promote alignment (Line 26). However, if they are dissimilar, we apply a margin-based contrastive penalty to avoid blindly trusting divergent clients.

Furthermore, we separate large clusters (e.g., the one that represents the main consensus) from small or singleton clusters (which can be caused by noise or adversarial influence) to define cluster significance. In large clusters, high trust variation is less aggressively penalized as some variation is expected to occur naturally. But when clients are very similar in small clusters, another separate penalty is induced, indicating potential collusion or embedded manipulation. We also add a **soft trust penalty** that penalizes high trust for clients in small or isolated clusters. This prevents us from being too confident in clients that are not well supported, and is in line with our zero-trust design: trust needs to be earned by aligning with larger and more stable communities.

Finally, we subtract a **cooperation reward**: when two clients in strong clusters exhibit high mutual trust, the loss function rewards the model for capturing this consensus.

The total loss balances four components:

- A **contrastive alignment term** across all client pairs;
- A **penalty** for over-similar clients in minor clusters;
- A **reward** for consistent trust in major clusters;
- A **soft trust penalty** to down-weight trust in small groups.

Algorithm 1 Contrastive Cluster-Aware Trust Loss

Input: Trust scores $T = [T_1, \ldots, T_K]$, similarity matrix $S \in \mathbb{R}^{K \times K}$, cluster labels $C = [c_1, \ldots, c_K]$
Hyperparameters: Margin γ, thresholds τ, τ_{high}
Weights: λ_{pen}, λ_{reward}, λ_{trust}

Initialize losses: $L_{\text{contrastive}} = 0$, $L_{\text{penalty}} = 0$, $L_{\text{reward}} = 0$, $L_{\text{trust}} = 0$
Compute cluster sizes $size_c \leftarrow |\{i : c_i = c\}|$
$s_{\max} \leftarrow \max(size_c)$; $\quad s_{\text{thresh}} \leftarrow \lfloor s_{\max}/2 \rfloor$
for $i = 1$ to K **do**
    for $j = 1$ to K **do**
        if $i = j$ **then**
            continue
        end if
        $sim \leftarrow S_{ij}$; $\quad diff \leftarrow (T_i - T_j)^2$
        $large_i \leftarrow (size_{c_i} > s_{\text{thresh}})$; $\quad large_j \leftarrow (size_{c_j} > s_{\text{thresh}})$
        $anySmall \leftarrow (\neg large_i) \vee (\neg large_j)$
        if $large_i \wedge large_j$ **then**
            $w \leftarrow 0.5$
        else
            $w \leftarrow 1.0$
        end if
        if $sim > \tau$ **then**
            $L_{\text{contrastive}} \leftarrow L_{\text{contrastive}} + w \cdot diff$
            if $large_i \wedge large_j$ **then**
                $L_{\text{reward}} \leftarrow L_{\text{reward}} + \lambda_{\text{reward}} \cdot T_i T_j$
            end if
        else
            $L_{\text{contrastive}} \leftarrow L_{\text{contrastive}} + w \cdot \max(0, \gamma - diff)^2$
        end if
        if $sim > \tau_{\text{high}}$ **and** $anySmall$ **then**
            $L_{\text{penalty}} \leftarrow L_{\text{penalty}} + (T_i + T_j)(sim - \tau)$
        end if
    end for
    if $size_{c_i} \leq s_{\text{thresh}}$ **then**
        $L_{\text{trust}} \leftarrow L_{\text{trust}} + T_i^2$
    end if
end for

$$L_{\text{final}} \leftarrow \frac{L_{\text{contrastive}}}{K(K-1)} + \lambda_{\text{pen}} \frac{L_{\text{penalty}}}{K(K-1)} - \frac{L_{\text{reward}}}{K(K-1)} + \lambda_{\text{trust}} \frac{L_{\text{trust}}}{K}$$

return L_{final}

This formulation allows the model to autonomously learn who to trust and who to doubt autonomously, even in the presence of unseen or adversarial clients, without requiring labeled attacks or centralized verification. The functionality of this loss is provided in Figure 3.1, which takes the inputs from K-Means, DBSCAN, and the attention method.

3.6 LLM-Guided Hyperparameter Optimization

Our cluster-aware contrastive loss consists of three coefficients: λ_{pen}, λ_{reward}, and λ_{trust} that control the balance of the penalty, reward and trust regularization. We find that choosing appropriate hyperparameter values is essential to maintain stable training dynamics and avoid trust-score collapse, while still allowing for a meaningful separation between honest and adversarial clients.

We perform hyperparameter selection *offline*, using an LLM-guided exploration strategy as a lightweight alternative to brute-force grid search or expensive Bayesian optimization [19]. This selection procedure is not part of the deployed FL pipeline. We use a large language model (GPT-5, queried via the OpenAI API) as an auxiliary proposer of candidate configurations within the ranges $\lambda_{\text{pen}} \in [0.2, 0.7]$, $\lambda_{\text{reward}} \in [0.05, 0.6]$, and $\lambda_{\text{trust}} \in [0.1, 0.4]$. In each round, the model receives the history of previously evaluated configurations along with summary statistics from the previous round, including the loss value, trust variance, average trust scores of honest and adversarial clients, and the separation between them, and based on this information it returns candidate configurations in JSON format. The prompt template used to query the model is shown in Figure 3.3.

We focus on the following three key hyperparameters of our trust loss:

- `penalty_weight`: penalizes trust between semantically dissimilar or inter-cluster clients.
- `reward_weight`: promotes trust alignment between semantically similar clients within large clusters.
- `trust_penalty_weight`: penalizes clients in small or noisy clusters to suppress anomalous trust propagation.

The exploration uses a fixed budget of 5 iterations with 5 candidate configurations proposed per iteration, for a total of 25 evaluated configurations. Each candidate is evaluated by training the trust-scoring module for 50 epochs with Adam (learning rate 10^{-3}). The configuration with the largest benign-vs-adversarial separation and non-degenerate trust variance is used in all experiments.

This approach enables:

- a faster convergence toward optimal regions of the trust loss landscape,

```
You are optimizing three hyperparameters of a contrastive cluster-aware
trust loss:
  penalty_weight    in [0.2, 0.7]
  reward_weight    in [0.05, 0.6]
  trust_penalty_weight in [0.1, 0.4]
The evaluation uses 24 federated clients with 4 adversarial clients.
A good configuration should minimize the final loss, keep trust-score
variance non-degenerate, and maximize the separation between the mean
trust of benign and adversarial clients.
Given the history of previously evaluated configurations (each with
final loss, trust variance, mean benign trust, mean adversarial trust,
and separation), propose N new candidate configurations.  Return
strictly a JSON list of {"penalty_weight":  x, "reward_weight":  y,
"trust_penalty_weight":  z} with no additional text.
```

Figure 3.3: Prompt template used to query GPT-5 during offline hyperparameter selection.

- avoidance of trust collapse due to suboptimal configurations,
- and semantic feedback-driven tuning rather than purely statistical methods.

We emphasize that this LLM-guided procedure was executed *offline* once before the experiments and is not part of the deployed FL framework. The trust scoring mechanism is fully unsupervised at deployment with no prior knowledge of adversarial clients.

3.7 Integration into Federated Learning Pipeline

The proposed trust scoring module is designed to integrate easily with a standard federated learning (FL) workflow without needing any changes in the client-side training loop. Each client computes its own representation vector locally, while the server is responsible for similarity computation, clustering, and trust score updates.

The integration proceeds in three stages:

1. **Client Representation Collection:** At each round of communication, the server receives the local computation of the representation vector r_k of each client through intra-client self-attention pooling. The raw per-sample embeddings never leave the client device, and only the single fixed-length representation vector is transmitted. This provides a compact summary of each client's behavior.
2. **Trust Score Computation:** The Contrastive Cluster-Aware Loss (Algorithm 1) is applied to produce trust scores using cluster structure, similarity patterns, and historical trust dynamics. The resulting vector of trust scores is normalized and mapped to the range $[0, 1]$.

3. **Trust-Aware Aggregation:** The normalized trust vector is used to filter and weight client updates during aggregation. In our experiments, clients with trust scores below a fixed threshold are excluded before aggregation, and FedAvg is applied to the remaining trusted clients. The framework is also flexible and can equivalently be deployed with weighted aggregation based directly on the trust scores, e.g.

$$w_i = \frac{T_i}{\sum_j T_j},$$

which assigns higher contribution weights to clients with more reliable behavior.

Because of its design, the entire pipeline is modular and works with any FL optimizer, including FedAvg, FedProx, and FedOpt. Above all, the trust module serves as a *defense layer* that is not dependent on the model architecture, allowing it to be applied to different datasets, modalities, and client behaviors.

3.8 Summary

In this chapter, we presented the main components of our proposed trust scoring framework for federated learning. We introduced the client embedding extraction module, similarity computation based on attention, and the clustering-driven trust regularization mechanism. The Contrastive Cluster-Aware Loss was described in detail, illustrating that how exactly it works and encourages coherent trust assignment within meaningful clusters while suppressing anomalous or adversarial behaviors.

Then we introduced an offline LLM-guided hyperparameter optimization strategy, which replaces the traditional search phase with a semantic and feedback-driven tuning process performed prior to deployment, and finally, we explained how these modules integrate into a standard FL pipeline to produce trust-aware aggregation weights.

Overall, the methodology establishes a unified, modular, and adaptive trust scoring system capable of operating under heterogeneous data conditions and in the presence of strong adversaries.

4

Threat Model

We assume a standard cross-device FL scenario where there is central server that coordinates model training among K clients. At each communication round, clients receive the current global model, perform local training on their private data, and finally return the model parameters to the server, where server conducts a weighted aggregation on these update to achieve a new global model. In this work, we consider that a majority of clients behave honestly, follow the determined training protocol by using their clean local datasets that are representative of the underlying data distribution. However, there is a small subset of clients that are controlled by an adversary.

4.1 Adversary Capabilities

These malicious clients can arbitrarily alter their local training data and labels before executing local updates, and they participate in aggregation protocol. In fact, the adversary perform a data poisoning attack on the local datasets of malicious clients, while still using the same model architecture, optimizer, and number of local epochs as honest clients. The features manipulated by the adversary originate from the local training data of each compromised client. These features are transformed using data poisoning techniques, thereby influencing the model representations transmitted to the server.

The adversary does not control the central server where aggregation is conducting and can not tamper with the global aggregation rule itself. Also, we assume that communication channels are not subject to man-in-the-middle manipulation; The attacker only operates on their data and labels on the compromised devices. The attacker also is not able to influence benign clients and cannot perform gradient manipulation beyond what results naturally from training on poisoned data. Within its local dataset, the adversary is able to:

- selectively identify samples belonging to chosen source classes;

- apply arbitrary image transformations, distortions, or backdoor-style perturbations to these samples;
- relabel a configurable fraction of these samples to a chosen target class (many-to-one label-flip);
- keep all other clean samples untouched to maintain protocol compliance.

These settings illustrate the real scenario where a limited number of clients are manipulated, but server is trusted. Our defense mechanism is designed under this threat model: The goal is straightforward, the framework assigns a trust score to each client based on their representative vector, down-weight and isolate adversarial clients during the aggregation.

4.2 Adversary Goals

The primary goal of the adversary is to bias the global model by injecting systematically mislabeled and visually altered samples into the training dataset of each client. Specifically, the adversary performs *many-to-one* poisoning attack which images from several source classes are relabeled into a single target class. The main goal of the attack is to shift the representation of these chosen classes in the embedding space, causing the global model to misclassify source-class samples as the target class in inference. We consider two different versions of this objective:

1. **Targeted poisoning on a small subset of classes.** A limited set of semantically related classes (e.g., deer, horse in CIFAR) are mapped to a chosen target class (e.g., bird) with a moderate poisoning fraction. The goal of this attack is to intentionally modify the target class's decision boundary.
2. **High-intensity many-to-one attack on multiple classes.** A larger set of source classes is aggressively relabeled to a single target class using a high poisoning fraction and strong image perturbations, such as backdoor triggers, spatial distortions, or occlusion patterns. This setting enables an accurate evaluation of the proposed trust-based defense, leading to a more pronounced change in the embedding space and exerting strong pressure on the aggregation mechanism.

5 Experiments

5.1 Experimental Setup

This section presents the empirical results obtained from our proposed trust scoring framework for clients in a zero-trust environment in federated learning. We gauge the trust scoring module under various experimental settings, including offline LLM-guided exploration of the contrastive-loss hyperparameters and the effectiveness of the attention-based mechanism, which can be developed for the aggregation phase following trust scoring.

5.2 Datasets

To evaluate our trust score framework, our experiments are conducted on 2 different image classification benchmarks: *CIFAR-10* and the *German Traffic Sign Recognition Benchmark* (GTSRB). These two datasets are different in visual complexity, semantic structure, and the number of classes, which allows us to assess the robustness and generality of our proposed framework.

- The **CIFAR-10** dataset [10]: A widely used dataset of 32×32 color images across 10 object categories, such as airplane, dog, and truck. Each image contains a single object centered in the frame.

- The **GTSRB** [15] dataset: The German Traffic Sign Recognition Benchmark is a real-world dataset of traffic sign images captured from different viewpoints and backgrounds, with 43 classes.

5.3 Federated Setup

We simulate FL by distributing data to clients, who train locally without sharing raw samples with the other clients or with the central server. The FL training process is implemented completely manually to have full control over local training. All clients use the same model architecture. The central server aggregates updates using FedAvg as the base aggregation rule. In our framework, clients with trust scores below a fixed threshold are excluded before aggregation, and FedAvg is applied to the remaining trusted clients. However, our method is flexible and can be deployed equivalently with weighted aggregation based directly on the trust scores. The specific number of clients, adversarial ratio, and poisoning fraction per scenario are detailed in Table 6.1.

5.4 Hyperparameter Selection

Our contrastive cluster-aware loss involves three coefficients, λ_{pen}, λ_{reward}, and λ_{trust}, which control the balance between penalization, reward, and trust regularization. Finding the most appropriate values for these hyperparameters is essential to ensure stable training dynamics, prevent trust-score collapse, and achieve meaningful separation between honest and malicious clients.

We perform hyperparameter selection *offline*, using an LLM-guided exploration strategy as a lightweight alternative to grid search or Bayesian optimization [19]. This selection procedure is not part of the deployed FL pipeline. As described in Section 3.6, we use a large language model (GPT-5, queried via the OpenAI API) as an auxiliary proposer of candidate configurations within the ranges $\lambda_{\text{pen}} \in [0.2, 0.7]$, $\lambda_{\text{reward}} \in [0.05, 0.6]$, and $\lambda_{\text{trust}} \in [0.1, 0.4]$. The exploration uses a fixed budget of 5 iterations with 5 candidate configurations proposed per iteration, for a total of 25 evaluated configurations. Each candidate is evaluated by training the trust-scoring module for 50 epochs with Adam (learning rate 10^{-3}). The configuration with the largest benign-vs-adversarial separation and non-degenerate trust variance is used in all experiments reported in this chapter.

We emphasize that this LLM-guided procedure was performed only once *offline* prior to the experiments and is not part of the deployed FL framework. During deployment, the trust scoring mechanism operates fully unsupervised with no prior knowledge of adversarial clients.

5.5 Attack Setup

We simulate data poisoning attacks in a controlled experimental setting to evaluate the robustness of the proposed trust-aware federated learning framework. Following the threat model described in Chapter 4, we assign a small

subset of clients as adversarial clients and the remaining clients behave honestly and train on clean local datasets.

Adversarial clients manipulate their local training data before local model updates, but follow the same training protocol as benign clients. In particular, all clients share the same model architecture, optimizer setup, and number of local training epochs. This design choice guarantees that the malicious behavior can only be due to data manipulation, and not due to protocol deviation or explicit gradient tampering.

We investigate targeted many-to-one data poisoning attacks where samples from selected source classes are systematically relabeled to a single target class. The attack is applied independently on each adversarial client and there is no coordination between the malicious clients other than the same poisoning strategy. This setup is realistic in federated learning scenarios where only limited control over client devices is assumed.

This setup reflects realistic federated learning scenarios in which only limited control over client devices is assumed.

We evaluate two versions of data poisoning attacks, each with different characteristics, depending on the dataset.

5.5.1 Poisoning Strategy

Within each adversarial client, data poisoning is performed by selectively modifying a fraction of the local training samples. Specifically, samples belonging to predefined source classes are identified, and a configurable proportion of these samples is poisoned. The poisoning fraction controls the strength of the attack and allows us to simulate both moderate and high-intensity adversarial behavior.

Simultaneously, the poisoned samples are modified in two ways: the input image is visually perturbed, and it is relabeled to a specific target class. The remaining samples in the client's dataset are unchanged, so that the data distribution of the overall data is partially corrupted. Such a selective poisoning strategy allows the adversary to keep stealth while having sufficient influence to distort the learned representations.

For CIFAR-10, the first attack variant applies strong visual transformations and relabels samples to a specific target class — a high-intensity many-to-one label-flipping attack that distorts the local distribution and biases updates. We also evaluate a stronger, destructive poisoning variant on CIFAR-10. In this setting, different fractions of each adversarial client's local samples are poisoned regardless of the original class, and severe visual distortions are applied to collapse feature representations and assign incorrect labels. This disrupts local training and generates highly noisy, protocol-compliant adversarial updates.

For the second poisoning variant on the GTSRB dataset, we use a hybrid attack of label manipulation and trigger-based perturbations. Some of the poisoned samples are triggered by a random visible patch and re-labelled to the

target class, while others are just label-corrupted. This design allows for simultaneous evaluation of backdoor-style attacks and label corruption on a single adversarial client.

5.5.2 Label Mapping

The core mechanism of the poisoning attack is many-to-one label manipulation. In this setting, samples from multiple source classes are relabeled to one target class before local training. This mapping brings a systematic bias to the local objective of adversarial clients, compelling the model to associate heterogeneous visual patterns with the same semantic label.

Formally, let $\mathcal{S}$ denote the set of source classes and y_t the chosen target class. For a selected subset of training samples with labels $y \in \mathcal{S}$, the adversary replaces the original label with y_t. This relabeling is applied only to poisoned samples and does not affect the labels of clean data.

By collapsing multiple classes into a single target class, the adversary aims to distort the geometry of the learned embedding space. As a result, representations of distinct source classes are drawn closer together and aligned with the target class, increasing the likelihood of misclassification during inference.

5.5.3 Visual Transformations

To further amplify the effect of label poisoning, adversarial samples are subjected to strong visual transformations prior to training. These transformations are designed to preserve overall image realism while introducing significant appearance variations that challenge feature consistency.

The applied transformations include random affine distortions (rotation, translation, scaling, and shearing), color jittering, perspective warping, Gaussian blurring, random erasing, and — in the GTSRB hybrid attack — visible patch triggers. Together, these operations alter spatial structure, texture, and color statistics, producing visually plausible yet heavily perturbed samples.

Importantly, these transformations are only applied to poisoned samples and sampled stochastically so as not to introduce easily detectable patterns. This design forces the model to learn unstable and misleading feature association for the target class, making it harder to distinguish the adversarial behavior based on only local training performance.

5.5.4 Attack Scenarios

We organize our evaluation into six attack scenarios spanning both datasets, three attack types, and varying numbers of adversarial clients and poisoning fractions. The scenarios are summarized together with their detection results in Table 6.1 (Chapter 6). Scenarios A1–A4 target CIFAR-10 with two adversarial clients (A1, A2 with moderate label-flipping; A3 with strong visual + label

collapse) and four adversarial clients (A4 with strong visual + label collapse). Scenarios A5 and A6 target GTSRB with the hybrid backdoor-driven poisoning attack at moderate (A5) and high (A6) intensity. Together they allow us to assess the sensitivity of aggregation and trust mechanisms under different attack intensities and dataset characteristics.

5.6 Evaluation Metrics

We apply both *trust-scoring* metrics and *detection* metrics to quantitatively assess the performance of our trust assignment mechanism. The trust-scoring metrics measure how well our framework separates honest from adversarial clients in trust space, while the detection metrics evaluate the resulting hard decisions used to filter clients during aggregation.

1. Mean Trust. We define the average trust score for honest and adversarial clients separately as follows:

$$\mu_h = \frac{1}{N_h} \sum_{i=1}^{N_h} T_i^{(h)}, \quad \mu_a = \frac{1}{N_a} \sum_{j=1}^{N_a} T_j^{(a)}, \tag{5.1}$$

where $T_i^{(h)}$ and $T_j^{(a)}$ denote the trust scores of honest and adversarial clients, and N_h, N_a represent their respective counts. Ideally, $\mu_h \approx 1.0$ and $\mu_a \approx 0.0$.

2. Trust Separation. To evaluate how well the trust mechanism distinguishes honest clients from adversarial ones, we compute the *trust separation score* as follows:

$$S = \mu_h - \mu_a. \tag{5.2}$$

A higher separation score S indicates a stronger distinction between trustworthy and malicious participants. A perfect separation would correspond to $S = 1$.

3. Cluster Consistency. We also assess the consistency of the clustering results obtained from DBSCAN in the embedding space. Let C_i denote the assigned cluster label for client i, and C_h and C_a represent the sets of clusters containing honest and adversarial clients, respectively. We compute the *cluster purity* as:

$$P = \frac{1}{N_h} \sum_{i=1}^{N_h} \mathbb{I}(C_i \in C_h), \tag{5.3}$$

where $\mathbb{I}(\cdot)$ is the indicator function. Higher P reflects that honest clients form dense and coherent clusters, while adversarial ones remain isolated.

4. Detection Metrics. To evaluate adversarial-client detection at the threshold level, we additionally report the area under the ROC curve (*ROC-AUC*), which assesses the ranking quality of trust scores; the F1 score at the threshold $\tau = 0.5$ ($F1@\tau$), which measures hard detection performance; and the false positive rate (*FPR*) on benign clients, which quantifies the rate at which honest clients are incorrectly excluded from aggregation. Higher is better, except for FPR where lower is better.

5. Global Test Accuracy. Finally, we report the global test accuracy of the aggregated model under each aggregation rule, which directly measures the end-to-end robustness of the FL system to adversarial clients.

Discussion. These metrics collectively capture trust-assignment quality (via μ_h, μ_a, and S), structural integrity of the clustering (P), detection performance (ROC-AUC, F1@τ, FPR), and end-to-end robustness (global accuracy). In our experiments, we report these values across datasets and attack scenarios to analyze the sensitivity and robustness of the proposed trust model.

5.7 Baselines

We evaluate our framework with four widely used aggregation strategies in the literature of robust FL: FedAvg as the undefended baseline, FoolsGold [6], a confidence-based filtering technique that down-weights clients with highly correlated gradients, the coordinate-wise Median [18], a robust aggregation rule that replaces averaging with per-parameter medians, and Multi-Krum [2], selecting client updates that are closest to their nearest neighbors and discarding Byzantine outliers . We evaluate all baselines over the same client partitions, attack configurations, and training hyperparameters, such that the only difference between baselines is the aggregation rule.

5.8 Implementation Details

All components of our proposed trust scoring framework is implemented in Python using PyTorch [13] as the primary deep learning backend. To simulate parallel training and heterogeneous behaviors, the federated learning environment was replicated on a single machine, with each client running as a separate process. Below we summarize the most relevant implementation aspects.

5.8.1 Environment

All of the experiments were conducted in Google Colab Pro+, using an NVIDIA A100 GPU with 40 GB of VRAM and High-RAM mode enabled. The

software environment was based on Python 3.10, PyTorch 2.2, and CUDA 12 support. Additional packages included `scikit-learn` for clustering, `matplotlib` for visualization, and the OpenAI API client for offline LLM-guided hyperparameter tuning.

5.8.2 Client Training

In FL, we have 2 different sides - client side and server side - that are cooperating to achieve satisfying results. Each client is trained independently on its local dataset without sharing raw samples with the other clients or server. To maintain the architecture flexibility, we adopt a back-bone model design: the back-bone acts as a shared feature extractor across all clients, while the classification head is kept lightweight and tailored to specific domain and dataset.

By this, during each one of communication rounds, each client performs local training for a fixed number of epochs with its own data. After local optimization, the model weights are transmitted to the server and the back-bone outputs also are also used to extract intermediate embeddings that serve as inputs to the similarity, clustering, and trust computation modules. This idea of separation between backbone and head allows us to (i) separate representation learning from classification, (ii) extract consistent embedding spaces across clients, and (iii) perform trust-aware aggregation without disrupting the standard FL training loop.

6

Results

6.1 Overview

We evaluated our trust-aware training objective in a simulated federated learning setup comprising honest and adversarial clients. As we discussed in Section 3.5, honesty comes from the majority in federated learning, and our results confirm that our proposed contrastive cluster-aware trust loss leads to high calibration and clear separation between honest and adversarial clients who want to disrupt the master model's final weights by sharing unrelated weights.

The goal that we are following in these experiments is twofold: (i) to assess how different poisoning strategies affect the global model in the absence of any defense, and (ii) to analyze the effectiveness of the proposed trust-based framework in mitigating these attacks across the six attack scenarios A1–A6 introduced in Section 5.5.4.

6.2 Attack Scenarios and Detection Performance

The adversarial-client detection performance for all six scenarios is shown in Table 6.1, including ROC-AUC, F1 score at $\tau = 0.5$, false positive rate, and the separation of trust-score between benign and adversarial clients. Adversarial clients are assigned lower trust score in consecutive rounds and are excluded from the aggregation. In all cases, our trust-scoring framework achieves perfect ranking (ROC-AUC = 1.00) and zero false positives. The hard-detection F1 is 1.00 in all but the two CIFAR-10 label-flip scenarios (A1, A2), where the moderate poisoning fraction makes a small subset of adversarial clients harder to separate at the fixed threshold $\tau = 0.5$.

Table 6.1: Attack configuration and adversarial-client detection performance across experimental scenarios.

Dataset	ID	Attack Type	# Adv.	Poison Ratio	ROC-AUC	F1@τ	FPR	Sep (b-a)
CIFAR-10	A1	Many-to-One Label Flip	2	0.45	1.00	0.67	0.00	0.51
CIFAR-10	A2	Many-to-One Label Flip	2	0.60	1.00	0.67	0.00	0.46
CIFAR-10	A3	Strong Visual + Label Collapse	2	0.90	1.00	1.00	0.00	0.64
CIFAR-10	A4	Strong Visual + Label Collapse	4	0.90	1.00	1.00	0.00	0.99
GTSRB	A5	Hybrid Backdoor-Driven Data Poisoning	2	0.60	1.00	1.00	0.00	0.50
GTSRB	A6	Hybrid Backdoor-Driven Data Poisoning	4	0.90	1.00	1.00	0.00	0.97

6.3 Trust Score Distribution and Separation Analysis

Trust Separation. Among the top-performing configurations, the mean trust score assigned to honest clients is above 0.99 and even 1.00, while for adversarial clients, the trust score is close to zero (e.g., 0.003). The model's ability to accurately distinguish between normal and abnormal clients is shown by trust separation, which is more than 0.99 in some configurations.

Trust Distribution. In addition to the strong separation that the model achieved, the variance of the trust scores among honest clients remained moderate, which is more based on hyperparameter selection to set the sensitivity of the model.

Failure Cases. In addition, one of the lowest-performing configurations had poor results, including a separation score with `-0.4533`, meaning that adversarial clients received trust scores up to 0.8007 and honest clients as low as 0.3473. These results demonstrate the importance of proper calibration and cluster-awareness in the trust mechanism.

6.4 Impact of the Contrastive Cluster-Aware Loss

The contrastive cluster-aware loss played an important role in stabilizing trust estimation and improving the separation between honest and adversarial clients. This loss function penalized clients with anomalous or conflicting similarity patterns while encouraging clients with consistent representations to converge toward similar trust scores by taking into account inter-client similarity, cluster structure, and trust regularization all at once.

During training, the loss slowly increased the difference between benign and adversarial behavior while adversarial clients maintained more variance and continuously pushed toward lower trust scores, honest clients created compact

clusters with high internal similarity and stable trust trajectories. Both datasets used for our evaluation showed this behavior, proving the method's generalizability.

6.5 Effect of Hybrid Clustering (K-Means + DBSCAN)

We utilize a hybrid clustering method by integrating K-Means and DBSCAN to efficiently identify adversarial actions under various attack levels. The reasoning for this design is that we want to benefit from the complementary strengths of both methods: K-Means provides a global partitioning of clients based on their embedding similarity, whereas DBSCAN enables a density-based identification of outliers that do not belong to any dominant cluster structure.

In practice, Using just K-Means tends to force every client into a cluster, even when some of the clients exhibit highly irregular or isolated representations due to poisoning. This behavior can lead to adversarial clients being absorbed into benign clusters, especially under strong or coordinated attacks.With integrating DBSCAN, we allow for the existence of low-density regions in the embedding space, enabling the detection and isolation of anomalous clients that deviate significantly from the dominant client populations.Our experiments show that DBSCAN consistently identifies a small subset of clients as outliers during early training epochs, particularly when adversarial clients introduce strong perturbations.

Moreover, the hybrid approach improves the robustness of trust estimation by resisting adversarial clients' inflation of their trust scores via partial alignment with benign clusters. DBSCAN is a defense which preserves the separation at density level even in worst-case attack scenarios where the adversarial embeddings are starting to become similar to the honest ones. In the long run, this translates into more stable trust trajectories and clearer cluster boundaries.

In conclusion, the combination of K-Means and DBSCAN is able to provide an effective and flexible clustering mechanism that improves the robustness and interpretability of our trust scoring framework, especially in the presence of severe data poisoning attacks.

6.6 Aggregated Model Accuracy and Robustness

Table 6.2 reports global test accuracy under different aggregation rules across the six attack scenarios. As shown, increasing either the poisoning fraction or the number of adversarial clients leads to severe degradation under standard FedAvg, with accuracy dropping to 26.86% in scenario A4. Similarity-based defenses such as FoolsGold (20.47% in A4) fail to recover under this destructive attack, and the coordinate-wise Median collapses to nearly random behavior. Multi-Krum performs competitively across most scenarios, but our proposed

Table 6.2: Global test accuracy (%) under different aggregation rules across scenarios A1–A6. Higher is better. Bold indicates the best result per scenario.

ID	FedAvg	FoolsGold	Median	Multi-Krum	Ours
A1	75.41	74.51	52.85	**79.02**	78.88
A2	74.02	73.99	52.28	77.33	**78.48**
A3	67.97	66.66	23.48	78.25	**78.29**
A4	26.86	20.47	10.00	76.17	**77.12**
A5	96.40	97.81	85.12	97.60	**98.07**
A6	89.46	89.27	57.81	95.00	**96.52**

framework matches or outperforms it in every scenario, with the largest margin appearing under the most destructive attacks (A4: 77.12% vs FedAvg's 26.86%).

Figures 6.1–6.3 show the global test accuracy versus federated rounds for scenarios A1, A3 and A4, with all aggregation methods tested over rounds. During the training process, the proposed framework achieves the highest final accuracy with stable learning dynamics. Median degrades sharply under strong poisoning, in contrast, FedAvg and FoolsGold show unstable or severely degraded convergence under attack. Multi-Krum performs well but is slightly outperformed by our approach.

Similarity-based aggregation, such as FoolsGold, fails to improve over the undefended FedAvg baseline in most scenarios. In earlier experiments with colluding adversaries and uniform attacks, poisoned clients formed compact clusters; under these conditions, the proposed framework suppressed the malicious clients using its cluster-aware trust penalization mechanism.

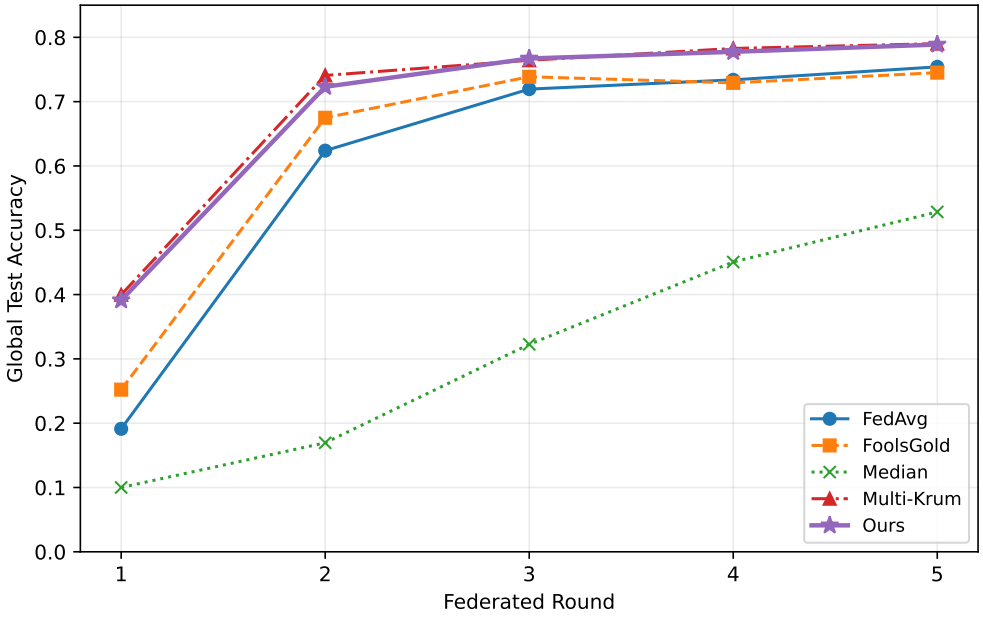

Figure 6.1: Global test accuracy over federated rounds in scenario A1, comparing FedAvg, FoolsGold, Median, Multi-Krum, and the proposed framework.

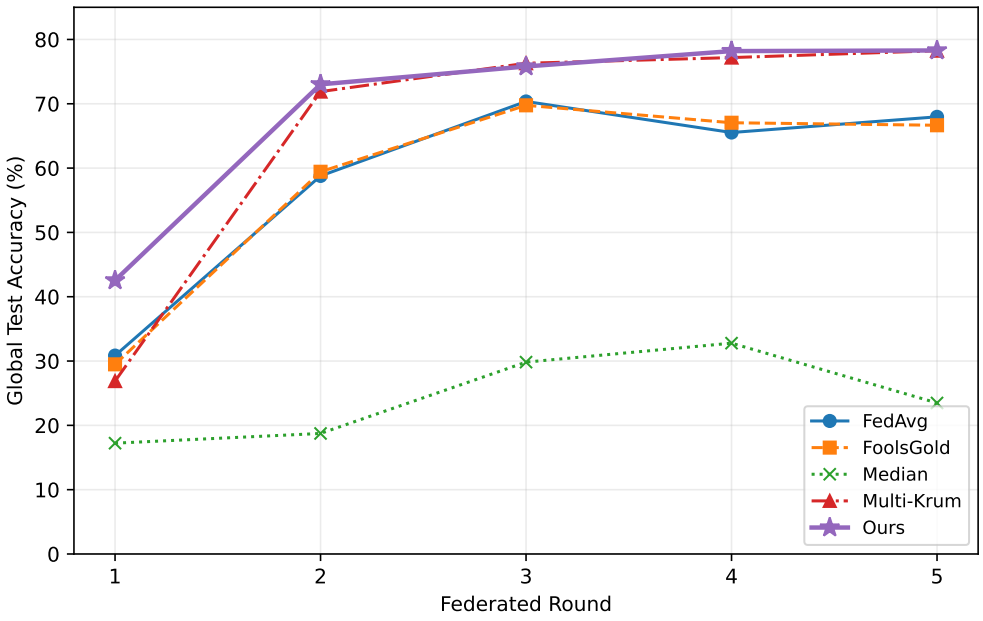

Figure 6.2: Global test accuracy over federated rounds in scenario A3, comparing FedAvg, FoolsGold, Median, Multi-Krum, and the proposed framework.

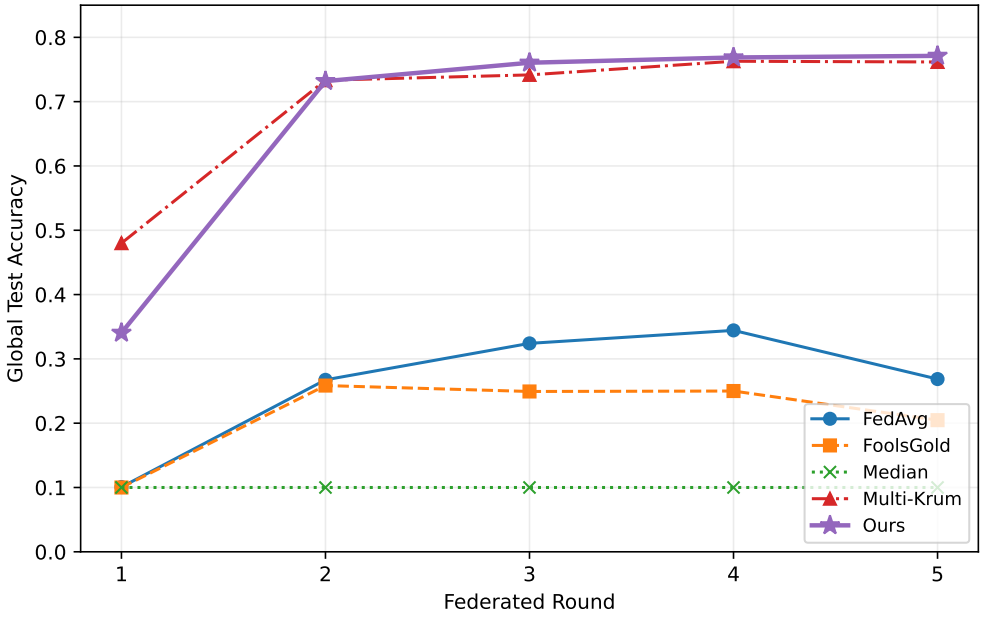

Figure 6.3: Global test accuracy over federated rounds in scenario A4, comparing FedAvg, FoolsGold, Median, Multi-Krum, and the proposed framework.

6.7 Ablation Studies

We start this subsection by putting the focus on gauging the impact of our contrastive cluster-aware trust loss.

6.7.1 LLM-Guided Hyperparameter Selection for Trust Calibration

To evaluate the impact of our contrastive cluster-aware trust loss, we conducted an ablation study using the LLM-Guided tuning mentioned in Section 3.6. The best configuration achieved by the LLM using different hyperparameters:

- `penalty_weight` = 0.397
- `reward_weight` = 0.587
- `trust_penalty_weight` = 0.180
- Loss: `-0.1576`
- Separation: `0.9887`
- Mean Adversarial Trust: `0.0105`
- Mean Honest Trust: `0.9992`

In contrast, one of the worst-performing configurations was:

- `penalty_weight` = 0.5
- `reward_weight` = 0.001
- `trust_penalty_weight` = 0.132
- Loss: `0.0190`
- Separation: `0.1256`
- Mean Adversarial Trust: `0.7045`
- Mean Honest Trust: `0.8300`

These results highlight how important hyperparameters are in achieving low loss and strong separation between adversarial and honest clients. Specifically, we can see the impact of the `reward_weight` in these results. Poor configurations not only fail in minimizing the loss, but can also lead to inverted trust patterns, assigning high trust to adversaries and low trust to honest clients.

Table 6.3 summarizes the final trust scores and cluster assignments. Higher reward led to a more polarized and confident trust separation.

6.7.2 Clustering Hyperparameter: Selection of ϵ for DBSCAN

We carried out a hybrid clustering strategy to combine both K-Means and DBSCAN, which you see in the procedure in section 3.4. To identify outliers, we employed the DBSCAN algorithm. A critical parameter in DBSCAN is ϵ, which defines the neighborhood radius for clustering. Instead of selecting ϵ randomly, we used a k-distance plot (2nd nearest neighbor distances) to empirically estimate a suitable value for this parameter.

In particular, we computed the distance of each client to its second closest neighbor after scaling the embedding space and plotted these distances in ascending order. We found a noticeable inflection point (elbow) around $\epsilon = 7.5$, which we used as our final clustering threshold.

Overall, choosing the right magnitude for ϵ is critical to detecting the exact number of outliers, which can be achieved through various methods, including k-distance and normalization.

6.8 Limitations of Current Evaluation

We conducted multi-round FL experiments in a simulated, synchronized environment. Our framework employs inter-client attention and clustering, whose computational and communication costs increase with the number of clients. We assume reliable client–server communication, not accounting for network latency, bandwidth, or client dropouts. in practice, such factors may

Table 6.3: Trust scores and cluster assignments under a strong adversarial attack. Adversarial clients are highlighted.

Client	Trust Score	Cluster ID
Client 1	0.9983	2
Client 2	0.9979	2
Client 3	0.9983	2
Client 4	0.9975	2
Client 5	0.9975	2
Client 6	0.0028	1
Client 7	0.0038	1
Client 8	0.9994	2
Client 9	0.9993	2
Client 10	0.9995	2
Client 11	0.9994	2
Client 12	0.9994	2
Client 13	**0.0000**	1
Client 14	0.0006	1
Client 15	0.9996	0
Client 16	0.9999	0
Client 17	1.0000	0
Client 18	1.0000	0
Client 19	0.9999	0
Client 20	0.0272	1
Client 21	0.0283	1
Client 22	0.9999	0
Client 23	0.9999	0

also influence the transmission of representation vectors. The network latency can cause stale or delayed representations, resulting in temporal misalignment of the client updates at the server. This can affect the effectiveness of inter-client attention and clustering which rely on consistent information. While the framework uses compact client-level representations rather than raw data, we do not claim that these representation vectors are completely privacy-free. Exploring privacy-preserving mechanisms, such as encryption or differential privacy, for representation exchange is left for future work.

6.9 Discussion of Findings

The experimental results in this thesis demonstrate that using trust scoring method based on representation-level can effectively distinguish honest clients from adversarial or low-quality participants in federated learning, even under strong data poisoning scenarios. Across both evaluated datasets, CIFAR-10

Table 6.4: Client trust scores and cluster assignments under weak and strong data poisoning attacks.

Client	**Weak Attack**		**Strong Attack**	
	Trust Score	Cluster	Trust Score	Cluster
Client 1	0.9990	1	0.9479	1
Client 2	1.0000	2	1.0000	1
Client 3	0.9991	1	0.9247	1
Client 4	0.9958	1	0.9555	1
Client 5	0.9994	1	0.9103	1
Client 6	0.9990	1	0.9801	1
Client 7	0.9961	1	0.7516	2
Client 8	**0.0000**	0	**0.0000**	0

and GTSRB, our proposed framework consistently assigns high trust scores to coherent client groups while suppressing the influence of suspicious and isolated clients.

One of the most important findings is strong separation between honest and adversarial clients. After performing LLM-Guided Hyper-parameter Selection, by using best performing configurations, malicious clients get trust scores close to zero, while normal clients receive near to one.

Another successful structural regularization technique is the hybrid clustering approach that combines K-Means and DBSCAN. DBSCAN assists K-Means by detecting low-density regions and distinguishing outlier clients that would otherwise be incorporated into benign clusters, while K-Means identifies global grouping patterns among clients with similar data distributions. This combination is especially helpful because during powerful poisoning attacks, adversarial and honest representations can be partially aligned. Results suggest that density-aware clustering is necessary for robust trust estimation in heterogeneous federated environments.

Another key finding concerns the interaction between trust scoring and aggregation performance. Combining the proposed trust scoring method with aggregation process leads to noticeable improvements in global model's accuracy under adversarial conditions, especially for high-intensity scenarios such as A4 (CIFAR-10 strong visual + label collapse with four adversarial clients) and A6 (GTSRB hybrid backdoor-driven poisoning with four adversarial clients). However, the experiments also show that unsuitable calibration of the trust loss hyperparameters can lead to inverted trust assignments, highlighting the sensitivity and importance of the system to loss balancing and the importance of careful tuning.

6.10 Summary

In this thesis, we introduced a novel trust scoring mechanism for federated learning that performs entirely on the representation level and does not rely on accessing to the client raw data, or prior knowledge of client behavior. Our proposed framework integrates all intra-client self-attention, inter-client attention-based trust estimation, hybrid clustering, and a contrastive cluster-aware loss to dynamically assign trust scores in a zero-trust federated settings.

The approach was created to address the primary limitations of the current defense and trust systems, especially their reliance on heuristic rules, static assumptions, or centralized validation data. Our proposed method allows for self-organizing trust dynamics that they are resistant to adversarial manipulation and heterogeneous data by modeling inter-client relationships using learned representations and structural clustering. Strong separation between honest and adversarial clients and enhanced aggregation robustness under data poisoning attacks have shown by experimental evaluations on CIFAR-10 and GTSRB. In conclusion, the results of this study point to attention-based and cluster-aware trust modeling as a viable approach to developing robust federated learning systems. The framework is modular and can be expanded to other data modalities, aggregation techniques, and threat models, even though the current study concentrates on image classification tasks and simulated poisoning attacks. Future research may examine fully automated hyperparameter calibration, deployment in cross-silo federated environments or real-world healthcare, and temporal trust dynamics across training rounds.

7

Conclusion

In this thesis, we introduced a zero-trust attention mechanism for robust federated learning. We started by training and extracting embeddings from CIFAR-10 and GTSRB datasets, to generate different clients, including normal ones, and also for diversity and resistance of the model, against poisoning attacks, we added adversarial ones generated by perturbing original images from specific clients.

To make the framework robust against such adversarial clients, our method is introduced in a two-stage trust scoring pipeline. First, we apply K-Means clustering (using the silhouette score) to detect clusters of clients, and then use DBSCAN with a data-driven choice of ϵ to identify outlier clients. Next, these clustering signals were used to guide the multi-part contrastive loss, which penalized trust between semantically dissimilar clients and rewarded alignment among large, similar clusters. They reduced trust in clients from anomalous or small clusters.

We apply a cross-client attention mechanism to learn global trust scores. Specifically, each client's embedding is processed through a self-attention layer followed by a learnable query-key-value attention mechanism among all clients. This enables the model to dynamically down-weight adversarial clients and emphasize reliable ones during the aggregation phase.

Our final results demonstrate a strong separation between adversarial and normal clients and consistent high-trust scoring for honest clients in various rounds and configurations. In experiments on CIFAR-10 and GTSRB across six poisoning scenarios, our framework consistently assigns lower trust to adversarial clients and higher trust to benign ones, allowing us to exclude abnormal clients from aggregation and improve global model performance compared to FedAvg, FoolsGold, the coordinate-wise Median, and Multi-Krum.

Future Work

There are a number of research directions that follow naturally from this work. One direction is exploring federated and edge learning for small language models, how FL paradigms can be customized to handle the particularities of small LMs, and still enable collaborative training under privacy constraints.

Another direction is to embed our trust score into Quantum Federated Learning (QFL) [5, 4], using quantum principles such as Quantum Key Distribution (QKD) for information-theoretic security or quantum-enhanced differential privacy. Quantum-assisted algorithms could also help with complex data structures or accelerate training.

Beyond these, future work may also examine fully automated hyperparameter calibration, deployment in cross-silo federated environments such as real-world healthcare, temporal trust dynamics across training rounds, and privacy-preserving mechanisms (e.g., differential privacy or encryption) for the exchange of representation vectors between clients and the server.

References

[1] Ahmed Alruwaili, Sardar Islam, and Iqbal Gondal. "Fed-DTB: A Dynamic Trust-Based Framework for Secure and Efficient Federated Learning in IoV Networks: Securing V2V/V2I Communication". In: *Journal of Cybersecurity and Privacy* 5.3 (2025), p. 48. DOI: 10.3390/jcp5030048. URL: https://doi.org/10.3390/jcp5030048.

[2] Peva Blanchard et al. "Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 30. 2017.

[3] Xiaoyu Cao et al. "FLTrust: Byzantine-robust Federated Learning via Trust Bootstrapping". In: *Proceedings of the 28th Network and Distributed System Security Symposium (NDSS)*. 2021.

[4] Siddhant Dutta et al. "Federated Learning with Quantum Computing and Fully Homomorphic Encryption: A Novel Computing Paradigm Shift in Privacy-Preserving ML". In: *CoRR* abs/2409.11430 (2024). DOI: 10.48550/ARXIV.2409.11430. arXiv: 2409.11430. URL: https://doi.org/10.48550/arXiv.2409.11430.

[5] Siddhant Dutta et al. "MQFL-FHE: Multimodal Quantum Federated Learning Framework with Fully Homomorphic Encryption". In: *CoRR* abs/2412.01858 (2024). DOI: 10.48550/ARXIV.2412.01858. arXiv: 2412.01858. URL: https://doi.org/10.48550/arXiv.2412.01858.

[6] Clement Fung, Chris J. M. Yoon, and Ivan Beschastnikh. "Mitigating Sybils in Federated Learning Poisoning". In: *arXiv preprint arXiv:1808.04866*. 2018.

[7] Clement Fung, Chris Joon Yoon, and Ivan Beschastnikh. "The Limitations of Federated Learning in Sybil Settings". In: *Proceedings of the 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID)*. USENIX. 2020, pp. 301–316.

[8] Peter Kairouz, Brendan McMahan, et al. "Advances and open problems in federated learning". In: *Foundations and Trends in Machine Learning* 14.1–2 (2021), pp. 1–210.

[9] Jiawen Kang et al. “Incentive Mechanism for Reliable Federated Learning: A Joint Optimization Approach to Combining Reputation and Contract Theory”. In: *IEEE Internet of Things Journal* 6.6 (2019), pp. 10700–10714.

[10] Alex Krizhevsky, Geoffrey Hinton, et al. “Learning multiple layers of features from tiny images”. In: *Technical Report, University of Toronto* (2009).

[11] Xiaoxiang Li, Yufeng Gu, Nicha Dvornek, et al. “Federated learning for healthcare informatics”. In: *Journal of Healthcare Informatics Research* 4.1 (2020), pp. 1–19.

[12] Lihua Ma et al. “A multi-level trust evaluation approach for secure and reliable federated learning”. In: *The Journal of Supercomputing* (2024). DOI: 10.1007/s11227-024-06873-5.

[13] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 32. 2019.

[14] José Antonio Sánchez, Zohra Riahi, and Raouf Boutaba. “FederatedTrust: A solution for trustworthy federated learning”. In: *arXiv preprint arXiv:2302.09844* (2023).

[15] Johannes Stallkamp et al. “The German Traffic Sign Recognition Benchmark: A multi-class classification competition”. In: *The 2011 International Joint Conference on Neural Networks*. IEEE. 2011, pp. 1453–1460.

[16] Yiyang Xie et al. “Federated Learning: A Survey on Attacks and Defenses”. In: *IEEE Transactions on Neural Networks and Learning Systems* 33.11 (2022), pp. 6245–6261.

[17] Qiang Yang et al. “Federated machine learning: Concept and applications”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 10.2 (2019), pp. 1–19.

[18] Dong Yin et al. “Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates”. In: *Proceedings of the 35th International Conference on Machine Learning (ICML)*. 2018, pp. 5650–5659.

[19] Michael R. Zhang et al. “Using Large Language Models for Hyperparameter Optimization”. In: *arXiv preprint arXiv:2312.04528* (2023).

[20] Yue Zhao et al. “Towards Trustworthy Federated Learning: A Survey on Trust Mechanisms, Threats, and Future Directions”. In: *IEEE Communications Surveys & Tutorials* (2022).