Supplementary Material

A. Supplementary Details for Method

A.1. Module Structures

In this section, we provide the model structure diagrams of the Belief Model (Fig. 1), Communication-based Reasoning Module (Fig. 2), and History-based Reasoning Module (Fig. 3).

A.2. Pre-training Method

This section outlines the pre-training procedure for FLToM, which encompasses two primary stages: data preparation and model training.

A.2.1. Data Preparation

To expedite the training of FLToM, belief data is collected during a pre-training phase, simulating the collaborative learning process. This is achieved using a variety of collaboration-oriented FL models (i.e., FedAvg, FedKNN and the robust NNM aggregator) that solely implement the belief models without the subsequent reasoning modules (i.e., the reasoning modules and the intention model). This setup allows for the simulation of belief and parameter patterns from all types of malicious clients without any filtering mechanisms interfering with their training. Notably, the model-agnostic and data-agnostic design of the BRM facilitates cross-model and cross-dataset simulations and data collection, thereby significantly enhancing the generalizability of the pre-trained FLToM model.

Different simulation strategies are employed for each primary attack type:

- **Byzantine Poisoning Attack**: Noise is directly injected into the model parameters, while the original local beliefs are maintained. The corrupted parameters then generate modified global beliefs through the aggregation process.
- **Belief Poisoning Attack**: Noise perturbation is applied specifically to local belief states. Global beliefs remain unchanged, as they are derived from the original, unmodified model parameters.
- **Data Poisoning Attack**: This involves a multi-step process:
  1. Obtain local models from the current training round.
  2. Perform aggregation using standard FedAvg, FedKNN or NNM.
  3. Execute inference on the poisoned client data using the newly aggregated model.
  4. Store the resulting local and global beliefs.
- **Blend Attack**: Outputs from the Byzantine and belief poisoning simulation strategies are combined. This involves using corrupted local beliefs from the belief poisoning simulation and modified global beliefs from the Byzantine poisoning simulation, selected through randomized sampling.

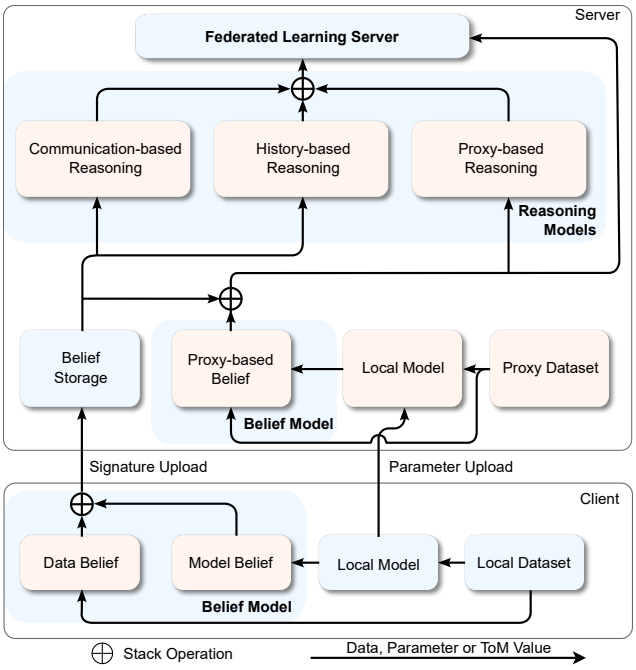


Figure 1. Illustration of the FLToM Model.

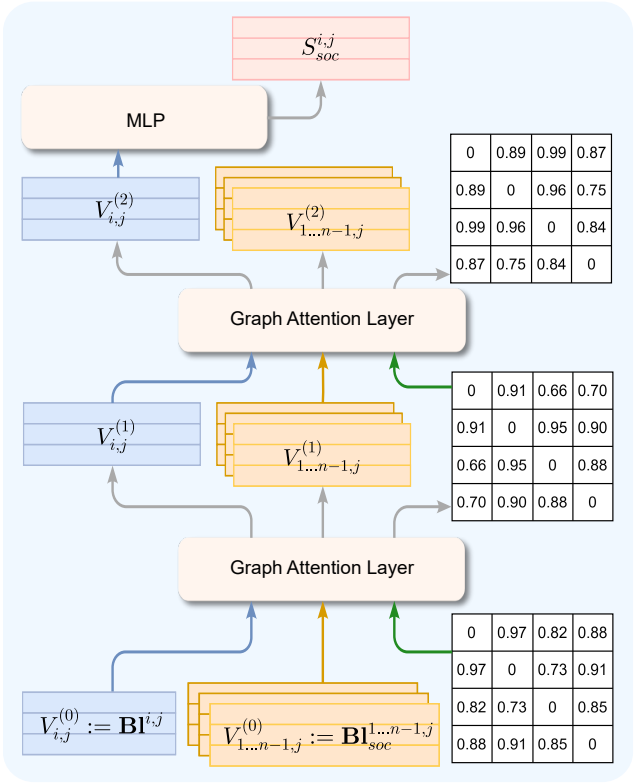


Figure 2. Illustration of Communication-based Reasoning Module.

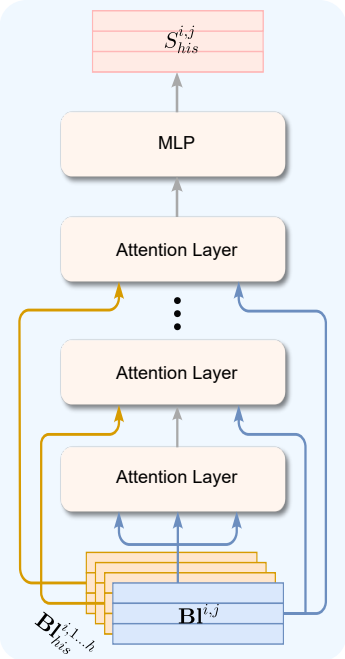


Figure 3. Illustration of the History-based Reasoning Module.

This framework's design allows for flexible extension to new attack types by applying appropriate data augmentation methods from the established categories, ensuring both practical effectiveness and methodological generality.

Finally, we collected a total of 80,000 data entries across the CIFAR-10 and VDZS datasets. Each data entry consists of the current client's belief, 5 to 10 historical beliefs, the global belief, and the latest beliefs of other clients. The collected belief dataset is partitioned into training and validation sets using a 3:1 ratio.

A.2.2. Model Pre-training

We pre-train FLToM using the dataset above. Its performance is evaluated on various validation sets. The hyperparameter configuration that demonstrates the best generalization across different datasets is selected for integration into subsequent FL tasks.

A key advantage is that since all beliefs are mapped to a unified feature space, the pre-trained model remains effective without requiring fine-tuning on individual datasets, significantly reducing deployment cost. The Adam optimizer is employed for training, and the prediction loss is computed using Smooth L1 loss:

$$\mathcal{L}(\hat{y}, y) = \begin{cases} 0.5(\hat{y} - y)^2 & \text{if } |\hat{y} - y| < 1, \\ |\hat{y} - y| - 0.5 & \text{otherwise.} \end{cases} \quad (1)$$

where $\hat{y}$ represents the predicted value and y denotes the ground truth label.

Through this pre-training arrangement, FLToM is effectively initialized for multiple tasks, providing a robust foundation for subsequent FL training. It is notable that, on the mixed validation set, FLToM achieved **94%** multi-class malicious client detection accuracy and **96%** overall malicious client detection accuracy. On the VDD validation set, it achieved **89%** multi-class malicious client detection accuracy and **90%** overall malicious client detection accuracy, demonstrating its strong generalization capability.

A.3. On the ρ-Robustness of Mean Aggregator under Constrained Attacks

This appendix provides a theoretical analysis to support the observation that the standard mean aggregator (i.e., FedAvg) exhibits a degree of robustness against a non-trivial fraction of attackers (ρ-robustness), particularly under constrained attack models such as Label Shift (LS) and Mimic (MiP) attacks, as evaluated in our main paper. We draw upon the theoretical framework established by Peng et al. [12], which investigates the performance of aggregators under label poisoning attacks.

A.3.1. Attack Models and Key Assumptions

Classical Byzantine attacks assume adversaries can send arbitrary messages. However, many practical attacks, including those in our work, are more constrained.

Definition 1 (*Constrained Malicious Gradients*). *We define attacks like Label Poisoning Attacks (i.e. Label Shift, Label Random Editing), and Mimic Parameter (MiP) as constrained attacks. In such scenarios, malicious workers still follow the algorithmic protocol, but their computed gradients are derived from manipulated local data or strategic imitation of other clients. Consequently, the malicious gradients are not arbitrary but are functionally dependent on the data distribution and the states of other clients. This model is a specific instance of the Label Poisoning Attacks defined in [12].*

We denote $\mathcal{W}$ as the set of all W workers and $\mathcal{R} \subset \mathcal{W}$ as the set of R benign workers. The global objective is defined as the average loss over benign workers, i.e., $f(x) = \frac{1}{R}\sum_{w\in\mathcal{R}} f_w(x)$. We first assume the smoothness of this objective:

Assumption 2 (*Bounded Disturbances*). *For any model parameter $x \in \mathbb{R}^D$, the maximum distance between the poisoned local gradient of a malicious worker $w \in \mathcal{W} \setminus \mathcal{R}$, denoted as $\tilde{\nabla} f_w(x)$, and the true global gradient $\nabla f(x)$, is upper-bounded by a constant A:*

$$\max_{w\in\mathcal{W}\setminus\mathcal{R}} \|\tilde{\nabla} f_w(x) - \nabla f(x)\| \leq A. \quad (2)$$

This assumption is adapted from Assumption 4 in [12]. It holds for our constrained attacks because: (1) For LS/LRE

attacks, the gradient's magnitude is inherently bounded by the data features, as formally shown in Lemma 1 of [12]. (2) For MiP attacks, the malicious gradient is intentionally crafted to be close to a benign gradient, thus its distance to the true global gradient is naturally bounded.

A.3.2. Convergence and Robustness Bound of FedAvg

Under the Bounded Disturbances assumption, Peng et al. [12] proved that the mean aggregator converges to a neighborhood of a stationary point, with an error bound explicitly characterized by the fraction of attackers ρ.

Theorem 3 (*Convergence of Mean Aggregator under Constrained Attacks, adapted from Theorem 2 in [12]*). *Consider a distributed learning process using the mean aggregator (FedAvg). Under label poisoning attacks (or more generally, constrained attacks satisfying Assumption 2) where the fraction of malicious workers is $\rho \in [0, 1)$, the final learning error of the algorithm is bounded. Specifically, the squared norm of the gradient at the output model $\hat{x}$ satisfies:*

$$\|\nabla f(\hat{x})\|^2 \leq \frac{8(f(x_0) - f^*)}{\eta T} + 10\rho^2 A^2, \tag{3}$$

where f^* is the minimum value of the global cost, x_0 is the initial model, T is the total number of iterations, and η is the step size.

Remark 1 (*Implication for ρ-Robustness*). *Theorem 3 provides the theoretical foundation for the ρ-robustness of FedAvg against constrained attacks. The critical insight is that the error bound holds for any fraction of attackers $\rho \in [0, 1)$, without requiring the common condition $\rho < 0.5$. This implies that even when malicious clients constitute the majority, the learning process does not diverge uncontrollably but converges to a neighborhood whose size is determined by ρ^2. The performance degradation is graceful, not a catastrophic breakdown.*

Remark 2 (*Comparison with Robust Aggregators*). *In contrast, many well-known robust aggregators, such as Trimmed Mean possess a "breakdown point" at $\rho = 0.5$. As shown in Lemma 3 and Theorem 1 of [12], their error bounds often contain terms like $1/(1 - 2\rho)$, which explode as ρ approaches 0.5. This explains why, in scenarios with a high fraction of constrained attackers, the seemingly non-robust FedAvg can outperform "robust" aggregators. This observation motivates our work's approach of first diagnosing the attack type and fraction, rather than indiscriminately applying a potentially unsuitable robust aggregator.*

B. Details of Experiment Settings

B.1. Hardware and Software

The model was implemented using `Python 3.11` and `PyTorch 2.1.0` with `CUDA 12.1`. Data preprocessing was performed using `OpenCV` and `NumPy` was utilized for model management. All experiments were conducted on a server equipped with 2 Intel XEON GOLD 6438V processors (48 cores) and 8 NVIDIA A800 GPUs.

B.2. Dataset

In this paper, we use CIFAR-10, VDZS, and VDD to thoroughly test the performance of ToM-structured FL for real-world collaborations. CIFAR-10 contains 60,000 samples with 32×32 resolution and 10 categories. VDZS contains vertical-view drone aerial images with 800×800 resolution, with multiple objects per image across 16 categories. VDD includes oblique 45° drone aerial images, with resolutions ranging from 560×720 to 1080×1920, containing multiple objects per image across 10 classes. We employed a tiled cropping strategy with overlaps as our primary data augmentation technique. By generating overlapping patches from the original images, we significantly increased the sample diversity while maintaining the original input resolution. Consequently, the dataset size was expanded, with VDZS and VDD containing 30,000 and 32,300 samples, respectively. All datasets were split into training, test, and pre-training sets in a 5:1:1 ratio.

Following SDEA, we use a task-related unlabeled proxy dataset, which can be sampled from public dataset or generated. In our experiments, we randomly sampled 1200 instances ($\leq$ **5% of the global dataset**). In non-IID settings, such a sampling method effectively simulates the distribution shift between the real-world proxy and local data, thus proving the practical effectiveness of FLToM. Unlike methods like FLTrust that require large, labeled, client-contributed datasets ($\geq 50\%$), our approach requires only a small, unlabeled proxy dataset, making it more practical and data-efficient.

B.3. Malicious Attack Models and Settings

FLToM was evaluated against a variety of malicious model behaviors, categorized as single or blend attacks:

B.3.1. Single Attacks

- **Random Parameter Attack (RNP)**: Randomizes all uploaded model parameters.
- **Mimic Parameter Attack (MiP)**: This attack selectively replicates the benign update exhibiting the maximum variance within the system. By collectively mimicking this target, malicious clients artificially amplify statistical heterogeneity to skew the global aggregation while evading detection.
- **Label Shift Attack (LS)**: Increments the class index of all client samples by 1 (samples with the largest index are assigned to class 0).
- **Label Random Editing (LRE)**: Randomizes the class indices of all client samples.

- **Random Belief Attack (RNB)**: Randomizes all uploaded beliefs.
- **Mimic Belief Attack (MiB)**: Replicates the beliefs uploaded by other clients.
- **A Little is Enough (LIE)[3]**: By computing a malicious gradient that makes the mean of the malicious client closer to the global mean than that of normal clients, all existing defense mechanisms will remove the normal clients and select an aggregation in the direction of the malicious clients, thereby shifting the mean.
- **Minimize Sum of Distances Attack (MS)[14]**: Ensure that the upper limit of the sum of squared distances between malicious gradients and all benign gradients is equal to the sum of squared distances between any benign gradient and other benign gradients, thereby maintaining that all malicious gradients have the same maximum attack impact.

B.3.2. Blend Attacks

- **RNP mixed with RNB (+RN)**: Randomizes both uploaded parameters and beliefs.
- **RNP mixed with MiB (+Mi)**: Randomizes parameters while mimicking the beliefs of other clients.
- **RNP mixed with K-Means Mimic Attack (+Km)**: Randomizes parameters while mimicking beliefs as the average belief of a "good" client cluster. In +Km, K-means clustering partitions good clients, with the number of clusters equaling the number of +Km adversarial clients, and each +Km client is assigned to a distinct cluster.

B.4. Baseline Methods

Our method was compared against FedAvg (a non-robust baseline) and 11 state-of-the-art robust FL algorithms. These robust baselines are categorized as:

The following provides brief descriptions of the operational mechanisms for the baseline methods evaluated:

- **MultiKrum** [4]: Krum selects one gradient from its input set that is closest to its $(n-m-2)$ neighboring gradients (in terms of squared Euclidean distance), where m is the upper bound on the number of malicious clients and n is the total number of clients. MultiKrum iteratively applies Krum: it selects a gradient using Krum from the set of remaining gradients, adds it to a selection set, and removes it from the remaining set. This process is repeated, and the selected gradients are then averaged.
- **AFA** (Adaptive Federated Averaging) [10]: This method aims to remove malicious gradients based on the cosine similarity of benign gradients. In each FL round, AFA computes a weighted average of the collected gradients. It then calculates the cosine similarity between this weighted average and each individual collected gradient. Gradients whose similarity scores fall outside a defined range (a function of the mean, median, and standard deviation of similarities) are discarded.
- **Bucketing** [8]: Clients are randomly partitioned into k buckets (e.g., k=10), each containing m clients (e.g., m=3). The clients within each bucket compute a mean of their updates. Finally, each bucket's aggregated update is averaged with those of its n nearest neighboring buckets (e.g., n=2, based on Euclidean distance) to derive the parameters for the clients in that bucket. This is an asynchronous two-stage aggregation.
- **NNM** (Nearest-Neighbor-Median) [1]: A synchronous two-stage aggregation. Randomly select k clients (e.g., k=10) and compute their mean update. Then, calculate the distance of all client updates to this mean. Discard the s clients whose updates are furthest from the mean (e.g., s=8, providing a margin if 6 clients are malicious).
- **SDEA** (Self-Driven Ensemble Aggregation) [7]: Initializes a proxy dataset on the server (consistent with FLToM's proxy dataset). The optimizer's goal is to find optimal aggregation weights for each client. Each client's update is evaluated on the proxy dataset, generating predictions. Two losses, corresponding to sharpness and confidence change, are calculated from these predictions. By minimizing these losses, the aggregation weights are optimized. Finally, these weights are clustered, and the largest cluster is considered the set of benign clients. Their updates are averaged (unweighted) to produce the global model.
- **FedKNN** [16]: Uses a K-Nearest Neighbors approach for aggregation in a decentralized setting. Each client aggregates its model with those of its m closest neighbors (e.g., m=5), where closeness can be based on belief similarity.
- **Random Init** [13]: The model is first trained to convergence using FedAvg. Then, the model parameters are frozen, and the classifier head is re-initialized randomly. The classifier head is then unfrozen and trained locally on each client's data to produce the final model.
- **InferGuard** [15]: Uses the coordinate-wise median as a reference point. It calculates the Euclidean distance of all client updates to this median. Updates whose distances exceed a predefined threshold are discarded. The threshold is typically adjusted based on the specific attack scenario to optimize performance. For our belief-based variant, this was applied to beliefs as well.
- **FLTrust [5]**: FLTrust obtains the globally descending gradient through a root dataset, then uses ReLU to clip the cosine similarity score, and finally normalizes each local model update. The new aggregation rule takes into account both the direction and magnitude of local model updates and server model updates to calculate the global model update.
- **SIREN [6]**: A proactive Byzantine-robust framework that orchestrates defense mechanisms across distinct client and server levels. This system employs a distributed alarming process where clients actively verify the global

model integrity using local datasets, enabling the server to execute a joint detection strategy based on client alarms, model accuracies, and gradient discrepancies to filter malicious updates.

- **ARC [2]**: A principle-based adaptive clipping strategy that dynamically adjusts the clipping threshold according to the gradients sent by the working nodes and the tolerable proportion of adversarial working nodes.

B.5. Hyperparameter

Hyperparameter settings are: **N=30** clients, 20% malicious (6 clients), **H=80** rounds (VDZS, VDD) or **200** (CIFAR-10), lr=0.01 (CIFAR-10) or 3e-4 (VDZS, VDD), and local iterations h=5.

C. Additional Experiment

C.1. Complexity and Overhead Analysis

To comprehensively assess the practicality of FLToM, we evaluate its computational complexity and empirical overhead under a moderate client scale ($n = 30$ with 6 malicious clients), consistent with the main experimental setup in Sec. 4.

C.1.1. Computational Complexity

The high efficiency of FLToM stems from its low-dimensional reasoning space. Let N be the number of clients, M the number of model parameters, D the dimension of the belief vector, and H the historical window length. Standard distance-based baselines (e.g., Krum, FedKNN, NNM) rely on pairwise statistics of client parameters, yielding a complexity of $\mathcal{O}_{Baseline} = O(N^2M)$. In contrast, FLToM shifts the filtering reference from the high-dimensional parameter space to the low-dimensional belief space. Employing social and temporal reasoning (via Graph and Self-Attention), its complexity is bounded by $\mathcal{O}_{FLToM} = O(N^2D + NH^2D)$. Since the belief dimension D (e.g., hundreds) is strictly smaller than the model size M (e.g., millions), i.e., $D \ll M$, the computational complexity for client classification and filtering is drastically reduced. This theoretical efficiency translates directly to the empirical time savings discussed below.

C.1.2. Time Overhead

Table 1 details the average time consumed per communication round during FL training.

While FLToM requires more computation time than the vanilla FedAvg due to belief extraction and server-side reasoning, its overhead remains strictly comparable to, or even lower than, optimization-based robust aggregators such as SDEA (134s/round vs. FLToM's ∼20s/round) and NNM. Interestingly, when FLToM is deployed as a pre-filtering plugin for complex aggregators like NNM (i.e., FLToM+NNM), it effectively reduces the overall training runtime compared to the standalone baseline. By identifying and discarding malicious clients prior to aggregation, FLToM spares the underlying aggregator from executing time-consuming, distance-based filtering on the entire client pool.

Table 1. Comparison of time overhead for different methods.

Algorithm	Time Overhead (s)
FedAvg[9]	1.37
Multi-krum[4]	13.80
AFA[10]	6.40
Bucketing[8]	10.39
NNM[1]	66.02
SDEA[7]	134.70
KNN[16]	49.98
InferGuard[15]	16.62
FLTrust[5]	26.72
SIREN[6]	41.39
ARC[2]	9.64
FLToM+Avg	20.63
FLToM+NNM	52.54
FLToM+KNN	60.88

C.1.3. Communication Overhead

Beyond computation, the additional communication footprint introduced by FLToM consists of two minor components.

Lightweight Belief Sharing: The belief payload is extremely small (merely 4KB per client). Transmitting this belief alongside a ResNet-34 update on CIFAR-10 (∼83MB) or a YOLOv5++ update on VDZS/VDD (∼153MB) adds an imperceptible overhead of approximately **0.005%** and **0.0026%**, respectively. The recurring communication cost remains entirely dominated by standard model parameter exchanges.

One-time Proxy Dataset Setup: The server-side proxy dataset, essential for robust belief calibration, incurs only a one-time transmission cost before training commences. Sampling roughly 1,200 instances for CIFAR-10 or 120 instances for drone datasets results in a total data transfer of approximately 10MB across all clients. Importantly, the proxy dataset size is fixed and decoupled from the total number of clients, ensuring scalability.

C.2. Additional accuracy under IID setting

In this section, we provide additional experimental results under IID data settings in Tab. 2. These findings support our initial discussion in Section 1 regarding the challenges faced by specialized robust aggregation methods.

Our method, FLToM, not having been specifically optimized for IID scenarios, demonstrated notable resilience.

Table 2. Accuracy (%) of different methods under various attack scenarios.

Method	NA	RNP	LIE	RNB	LS
FedAvg[9]	**71.17**	**64.53**	35.53	—	50.78
Multi-krum[4]	68.21	58.83	66.12	—	53.32
FedKNN[16]	60.48	54.11	35.73	56.78	53.37
Inferguard[15]	70.82	59.58	65.68	58.97	52.34
FLToM	70.06	42.43	**67.80**	**59.58**	**53.39**

As seen in Tab. 2, FLToM achieved the highest accuracy against LIE (67.80%), RNB (59.58%), and LS (53.39%) attacks in this IID setting. This underscores the robustness of its core mechanisms even when data heterogeneity, a factor FLToM is designed to handle, is absent.

In this experiment, while FLToM excelled in several targeted attacks, FedAvg (mean aggregation) demonstrated the optimal performance against RNP with 64.53% accuracy. Our results in the IID setting reinforce this: when faced with simpler, untargeted noise on parameters in a homogeneous data environment, the straightforward averaging approach of FedAvg proved most effective among the tested methods.

Despite the IID setting not being the primary design focus for FLToM, its strong performance against LIE, RNB, and LS attacks demonstrates its continued effectiveness in mitigating specific malicious manipulations.

C.3. Dynamic Attack

To evaluate the robustness of our method against non-stationary adversarial behaviors, we designed a dynamic attack scenario where the identities of malicious clients change during training. In this setup, we maintain a constant 20% malicious client fraction. A "dynamic group" of clients is established, comprising 20% initially malicious and 20% initially benign clients. Each malicious client is paired with a benign one. At a randomly selected communication round for each pair, their roles are permanently swapped: the malicious client becomes benign, and the benign one turns malicious. This setup directly challenges defenses that rely on long-term reputation by creating a churn of adversarial identities.

The performance under this challenging scenario is detailed in Table 3. The results highlight FLToM's superior adaptability. Under the RNP attack, FLToM achieves an accuracy of **62.08%**, outperforming all baselines. This suggests that its reasoning mechanism can rapidly detect the abrupt change in behavior from a newly malicious client. For the more subtle LIE and LS data poisoning attacks, FLToM also demonstrates the best performance with **58.11%** and **56.25%** accuracy, respectively. Notably, in the LS attack, FLToM surpasses the next-best competitor (SDEA at 46.40%) by a significant margin of nearly 10 percentage points. These results validate that by continuously inferring client intent from recent actions, FLToM effectively adapts to a dynamic threat landscape where trust can be subverted, maintaining more stable and robust performance than methods reliant on static historical assessments.

Table 3. Accuracy (%) of different methods under dynamic attack scenarios.

Method	RNP	LIE	LS
FLTrust[5]	46.47	57.85	54.00
SIREN[6]	48.08	50.21	45.99
SDEA[7]	61.09	57.63	46.40
FLToM	**62.08**	**58.11**	**56.25**

C.4. Detailed results of Post-hoc interpretability

In this section, we extend the post-hoc interpretability analysis conducted in Section 4. For each adversarial type discussed, the accompanying interpretability visualizations (e.g., belief contribution maps) focus on the 9 most relevant samples, showcasing patterns consistent with the methodology detailed in Section 4. The key observations from this analysis include:

1. **Interpreting Parameter-Based Attacks (e.g., RNP):** For attacks primarily targeting model parameters, as in Fig. 6 and Fig. 7, the interpretability analysis reveals that FLToM's belief mechanism largely relies on contrasting an individual client's update characteristics against a consensus derived from the majority of other participating clients.
2. **Interpreting Label-Based Attacks (e.g., LRE, LS):** In scenarios involving label corruption, as in Fig. 8 and Fig. 9, FLToM's detection strategy, as illuminated by interpretability, often hinges on identifying anomalous belief features originating from specific, often smaller subsets of clients. The system effectively leverages the server's proxy data and the resulting inference discrepancies to pinpoint these deviations, making the impact of label poisoning on client beliefs more discernible.
3. **Interpreting Blend Attacks:** When addressing belief-based or blend attacks, as shown in Fig. 10-14, where each malicious client typically may employ a targeted or composite strategy, FLToM's hierarchical mechanism leverages a varying combination of belief components and cross-client comparisons, reflecting its capacity to flexibly respond to complex and evolving threat profiles rather than relying on a fixed set of indicators.

The figures visually presenting these interpretability analyses are provided at the end of this material.

C.5. Further Empirical Evaluations and Ablations

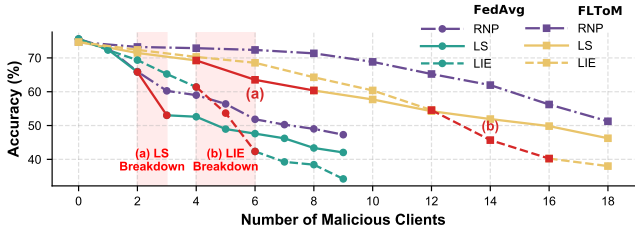
To thoroughly assess the generalization, scalability, and structural advantages of the proposed FLToM framework,

we extend our evaluations to encompass robustness thresholds, unseen attacks, architectural variations, and task formulations.

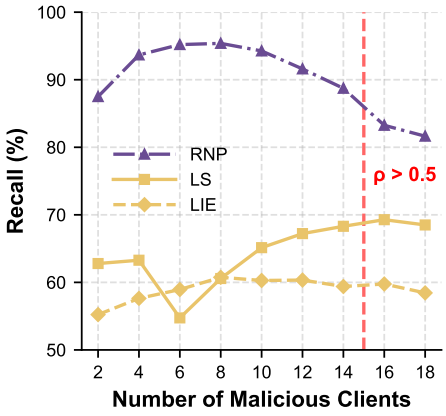
C.5.1. Mitigation of Breakdown Thresholds

Theoretical analysis (Theorem 3) suggests that conventional robust aggregators fail when the attacker fraction ρ exceeds a certain breakdown point. FLToM provides robustness by shifting from passive statistical filtering to active intention inference. By identifying and removing "Malicious Intentions" via social/temporal consistency *before* aggregation, FLToM explicitly reduces the effective attacker fraction ρ from high levels (e.g., 35%) to safe levels ($< 5\%$).

As shown in Figure 4(i) and Figure 4(ii), FLToM maintains a stable filtering ratio as the attack size grows, elongating the breakdown interval and mitigating explicit attack thresholds. Specifically, for RNP attacks, FLToM consistently maintains a robust filtering rate above 80% even when $\rho > 0.5$, effectively exhibiting no breakdown while mitigating thresholds for LS/LIE attacks.



(i) T. Acc. w/ increasing attackers. (RNP, LS, LIE Attack + CIFAR-10)



(ii) Filtering Efficienty of FLToM.

Figure 4. **Threshold mitigation analysis. (i)** Accuracy breakdown mitigation. **(ii)** Robust filtering after threshold $\rho > 0.5$ (> 15 Attackers).

C.5.2. Generalization to Unseen Attacks

FLToM is designed to detect *consistency violations* rather than memorizing specific attack patterns, enabling zero-shot generalization. In our main evaluation, FLToM achieves robust defense against LIE and Min-Sum (MS) attacks, even though these specific strategies were deliberately excluded from the pre-training set.

To push this further, we evaluated FLToM against unseen stealthy Backdoor Attacks, including traditional BadNets, pattern-based Blended attacks, and decentralized DBA. To provide a comprehensive baseline comparison, we additionally compared our approach with the Robust Learning Rate (RLR) method[11]. RLR mitigates backdoor injections by dynamically adjusting the aggregation server's learning rate per dimension and per round based on the sign information of the clients' updates, effectively penalizing anomalous parameter updates. As reported in Table 4, utilizing a bi-classification pre-training formulation, FLToM successfully detects the belief anomaly caused by trigger injections. By utilizing server-side verification with the proxy dataset, it accurately identifies and discards backdoor-implanted updates, neutralizing the malicious objectives.

Table 4. **Unseen backdoor defense using bi-classification FLToM, reported in Training Accuracy (%).**

Method	BadNets	Blended	DBA
FedAvg [9]	58.35	53.00	56.93
R. Init. [13]	62.93	57.66	61.35
RLR [11]	**64.21**	58.60	63.25
FLToM	63.02	**59.84**	**63.93**

C.5.3. Scalability to ViT and Proxy Sensitivity

The reasoning capabilities of FLToM are fundamentally built upon high-level abstraction behaviors (e.g., output distributions) rather than parameter-specific geometric distances. Evaluated on the CIFAR-10 dataset using a ViT architecture (Table 5), FLToM maintains superior robustness against diverse attacks compared to baselines, validating its broad applicability.

Table 5. **ViT on CIFAR-10.** We report Training Accuracy (%).

Method	NA	RNP	LS	LIE
FedAvg[9]	**82.25**	67.32	63.39	71.48
FLToM	81.61	**78.64**	**71.25**	**75.76**

Furthermore, Table 6 illustrates that FLToM is highly resilient to proxy dataset scaling and external noise. This stability is crucial for practical deployment, permitting the proxy data to be flexibly sourced from public domains or synthetically generated without compromising defense efficacy.

C.5.4. Ablation on Task Formulation

The formulation of the ToM reasoning task directly impacts filtering granularity. Removing detailed identifica-

Table 6. **Proxy Dataset Sensitivity.** Training Accuracy (%) for varying proxy settings, compared with the Main setting (1.2k).

Attack	1.2k (Main)	0.3k	4.8k	Noisy	Shift
RNP	72.52	↓ 0.56	↑ 0.29	↓ 4.34	↓ 2.76
RNB	67.35	↓ 0.16	↓ 0.04	↓ 0.86	↓ 0.53

tion for coarse-grained benign/malicious classification (binary) leads to less interpretability. As shown in Table 7, transitioning to a multi-classification formulation for attacker profiling significantly enhances reasoning performance, leading to notably higher training accuracy under complex and belief-targeted attacks (e.g., MS and RNB).

Table 7. **Bi-classification vs. Multi-classification.** We list Training Accuracy (T. Acc., %) across binary and multi-classification of attackers.

Task	Metric	RNP	LIE	MS	RNB
Binary	**T. Acc.**	72.52	68.71	66.37	67.35
Multi	**T. Acc.**	72.36	68.56	67.58	73.32

D. Detailed Proof of Theorem 3

We provide a detailed proof for the convergence of the mean aggregator under constrained attacks. The proof follows the standard descent lemma analysis, with the key part being the careful bounding of the error term introduced by the malicious clients. For simplicity and clarity, we analyze the case of distributed gradient descent, which corresponds to the analysis in [12] by setting the momentum coefficient $\alpha = 1$ and the stochastic gradient variance $\sigma^2 = 0$.

1. Preliminaries and Notation Let $\mathcal{W}$ be the set of all clients with $|\mathcal{W}| = W$, $\mathcal{R}$ be the set of benign clients with $|\mathcal{R}| = R$, and $\mathcal{W} \setminus \mathcal{R}$ be the set of malicious clients. The fraction of malicious clients is $\rho = (W - R)/W$. The update rule at iteration t is $x_{t+1} = x_t - \eta m_t$, where m_t is the aggregated gradient. For the mean aggregator (FedAvg), $m_t = \frac{1}{W}\sum_{w\in\mathcal{W}} \hat{\nabla}_w^t$.

- Benign clients ($w \in \mathcal{R}$) compute $\hat{\nabla}_w^t = \nabla f_w(x_t)$.
- Malicious clients ($w \in \mathcal{W}\setminus\mathcal{R}$) compute $\hat{\nabla}_w^t = \tilde{\nabla} f_w(x_t)$.

The true global gradient is defined as the average over benign clients: $\nabla f(x_t) = \frac{1}{R}\sum_{w\in\mathcal{R}} \nabla f_w(x_t)$. We assume the global objective function $f(x)$ is L-smooth, meaning $f(y) \le f(x) + \langle \nabla f(x), y - x\rangle + \frac{L}{2}\|y - x\|^2$.

2. Bounding the Aggregation Error The core of the proof is to bound the distance between the aggregated gradient m_t and the true global gradient $\nabla f(x_t)$.

First, we decompose the aggregated gradient m_t:

$$m_t = \frac{1}{W}\left(\sum_{w\in\mathcal{R}} \nabla f_w(x_t) + \sum_{w\in\mathcal{W}\setminus\mathcal{R}} \tilde{\nabla} f_w(x_t)\right). \quad (4)$$

Using the fact that $\sum_{w\in\mathcal{R}} \nabla f_w(x_t) = R \cdot \nabla f(x_t)$, we can rewrite m_t as:

$$m_t = \frac{R}{W}\nabla f(x_t) + \frac{1}{W}\sum_{w\in\mathcal{W}\setminus\mathcal{R}} \tilde{\nabla} f_w(x_t). \quad (5)$$

Now, we can express the aggregation error vector $m_t - \nabla f(x_t)$:

$$\begin{aligned} & m_t - \nabla f(x_t) && (6)\\ &= \left(\frac{R}{W} - 1\right)\nabla f(x_t) + \frac{1}{W}\sum_{w\in\mathcal{W}\setminus\mathcal{R}} \tilde{\nabla} f_w(x_t) && (7)\\ &= -\frac{W-R}{W}\nabla f(x_t) + \frac{1}{W}\sum_{w\in\mathcal{W}\setminus\mathcal{R}} \tilde{\nabla} f_w(x_t) && (8)\\ &= \frac{1}{W}\left(\sum_{w\in\mathcal{W}\setminus\mathcal{R}} \tilde{\nabla} f_w(x_t) - (W-R)\nabla f(x_t)\right) && (9)\\ &= \frac{1}{W}\sum_{w\in\mathcal{W}\setminus\mathcal{R}} \left(\tilde{\nabla} f_w(x_t) - \nabla f(x_t)\right). && (10) \end{aligned}$$

This elegant decomposition shows that the aggregation error is precisely the average of the "disturbance vectors" from all malicious clients, scaled by $1/W$. We can now bound the squared norm of this error vector.

Lemma 1 (*Aggregation Error Bound*). *The squared norm of the aggregation error is bounded by:*

$$\|m_t - \nabla f(x_t)\|^2 \le \rho^2 A^2. \quad (11)$$

Proof. Using the result from the decomposition above and applying Jensen's inequality for norms ($\|\frac{1}{n}\sum x_i\|^2 \le \frac{1}{n}\sum\|x_i\|^2$):

$$\begin{aligned} & \|m_t - \nabla f(x_t)\|^2 && (12)\\ &= \left\|\frac{W-R}{W}\frac{1}{W-R}\sum_{w\in\mathcal{W}\setminus\mathcal{R}} \left(\tilde{\nabla} f_w(x_t) - \nabla f(x_t)\right)\right\|^2 && (13)\\ &\le \rho^2\frac{1}{W-R}\sum_{w\in\mathcal{W}\setminus\mathcal{R}} \left\|\tilde{\nabla} f_w(x_t) - \nabla f(x_t)\right\|^2. && (14) \end{aligned}$$

By Assumption 2, we have $\|\tilde{\nabla} f_w(x_t) - \nabla f(x_t)\|^2 \le A^2$ for all malicious workers $w \in \mathcal{W}\setminus\mathcal{R}$. Substituting this into

the inequality:

$$\|m_t - \nabla f(x_t)\|^2 \leq \rho^2 \frac{1}{W-R} \sum_{w \in \mathcal{W} \setminus \mathcal{R}} A^2 \tag{15}$$

$$= \rho^2 \frac{1}{W-R}(W-R)A^2 = \rho^2 A^2. \tag{16}$$

This completes the proof of the lemma. □

3. Convergence Analysis We start with the descent lemma derived from the L-smoothness of $f(x)$:

$$f(x_{t+1}) \leq f(x_t) + \langle \nabla f(x_t), x_{t+1} - x_t \rangle + \frac{L}{2}\|x_{t+1} - x_t\|^2. \tag{17}$$

Substituting the update rule $x_{t+1} - x_t = -\eta m_t$:

$$f(x_{t+1}) \leq f(x_t) - \eta \langle \nabla f(x_t), m_t \rangle + \frac{L\eta^2}{2}\|m_t\|^2. \tag{18}$$

We use the identity $2\langle a, b \rangle = \|a\|^2 + \|b\|^2 - \|a-b\|^2$ to rewrite the inner product term:

$$-\langle \nabla f(x_t), m_t \rangle = -\frac{1}{2}\left(\|\nabla f(x_t)\|^2 + \|m_t\|^2 - \|\nabla f(x_t) - m_t\|^2\right). \tag{19}$$

Substituting this back into the descent inequality gives:

$$f(x_{t+1}) \leq f(x_t) - \frac{\eta}{2}\|\nabla f(x_t)\|^2 - \frac{\eta}{2}\|m_t\|^2 + \frac{\eta}{2}\|\nabla f(x_t) - m_t\|^2 + \frac{L\eta^2}{2}\|m_t\|^2 \tag{20}$$

$$= f(x_t) - \frac{\eta}{2}\|\nabla f(x_t)\|^2 + \frac{\eta}{2}\|\nabla f(x_t) - m_t\|^2 - \frac{\eta(1-L\eta)}{2}\|m_t\|^2. \tag{21}$$

By choosing a step size $\eta \leq 1/L$, the term $(1 - L\eta)$ is non-negative. We can therefore drop the final negative term to get a looser but simpler bound:

$$f(x_{t+1}) \leq f(x_t) - \frac{\eta}{2}\|\nabla f(x_t)\|^2 + \frac{\eta}{2}\|\nabla f(x_t) - m_t\|^2. \tag{22}$$

Now, we apply our key result from Lemma 1:

$$f(x_{t+1}) \leq f(x_t) - \frac{\eta}{2}\|\nabla f(x_t)\|^2 + \frac{\eta}{2}(\rho^2 A^2). \tag{23}$$

Rearranging the terms to isolate the gradient norm:

$$\frac{\eta}{2}\|\nabla f(x_t)\|^2 \leq f(x_t) - f(x_{t+1}) + \frac{\eta}{2}\rho^2 A^2. \tag{24}$$

Multiplying by $2/\eta$, and sum it over all iterations from $t = 0$ to $T-1$ gives

$$\sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2 \leq \sum_{t=0}^{T-1} \frac{2}{\eta}(f(x_t) - f(x_{t+1})) + \sum_{t=0}^{T-1} \rho^2 A^2. \tag{25}$$

The first term on the right is a telescoping sum: $\sum_{t=0}^{T-1}(f(x_t) - f(x_{t+1})) = f(x_0) - f(x_T)$. Since $f(x)$ is lower-bounded by f^*, we have $f(x_0) - f(x_T) \leq f(x_0) - f^*$. The second term is simply $T\rho^2 A^2$. Substituting these back, we get:

$$\sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2 \leq \frac{2}{\eta}(f(x_0) - f^*) + T\rho^2 A^2. \tag{26}$$

Dividing by T to get the average squared gradient norm over the trajectory:

$$\frac{1}{T}\sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2 \leq \frac{2}{\eta T}(f(x_0) - f^*) + \rho^2 A^2. \tag{27}$$

Since

$$\min_{t \in \{0,\ldots,T-1\}} \|\nabla f(x_t)\|^2 \leq \frac{1}{T}\sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2, \tag{28}$$

we finally arrive at:

$$\min_{t \in \{0,\ldots,T-1\}} \|\nabla f(x_t)\|^2 \leq \frac{2(f(x_0) - f^*)}{\eta T} + \rho^2 A^2. \tag{29}$$

This completes the proof. The final result shows that the algorithm converges to a neighborhood of a stationary point, where the size of the neighborhood is characterized by the non-vanishing error term $O(\rho^2 A^2)$, which holds for any $\rho \in [0, 1)$.

References

[1] Youssef Allouah, Sadegh Farhadkhani, Rachid Guerraoui, Nirupam Gupta, Rafaël Pinot, and John Stephan. Fixing by mixing: A recipe for optimal byzantine ml under heterogeneity. *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1232–1300, 2023. 4, 5

[2] Youssef Allouah, Rachid Guerraoui, Nirupam Gupta, Ahmed Jellouli, Geovani Rizk, and John Stephan. Adaptive gradient clipping for robust federated learning. In *Proceedings of the 13th International Conference on Learning Representations(ICLR)*, 2024. 5

[3] Moran Baruch, Gilad Baruch, and Yoav Goldberg. A little is enough: Circumventing defenses for distributed learning. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019 (NeurIPS)*, 2019. 4

[4] Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 119–129, 2017. 4, 5, 6

[5] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. Fltrust: Byzantine-robust federated learning via trust bootstrapping. In *28th Annual Network and Distributed System Security Symposium, NDSS 2021, virtually, February 21-25, 2021*. The Internet Society, 2021. 4, 5, 6

[6] Hanxi Guo, Hao Wang, Tao Song, Yang Hua, Zhangcheng Lv, Xiulang Jin, Zhengui Xue, Ruhui Ma, and Haibing Guan. Siren: Byzantine-robust federated learning via proactive alarming. *Proceedings of the ACM Symposium on Cloud Computing*, 2021. 4, 5, 6

[7] Wenke Huang, Zekun Shi, Mang Ye, He Li, and Bo Du. Self-driven entropy aggregation for byzantine-robust heterogeneous federated learning. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, 2024. 4, 5, 6

[8] Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Byzantine-robust learning on heterogeneous datasets via bucketing. In *Proceedings of the 10th International Conference on Learning Representations(ICLR)*, 2022. 4, 5

[9] H. B. McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016. 5, 6, 7

[10] Luis Muñoz-González, Kenneth T. Co, and Emil C. Lupu. Byzantine-robust federated machine learning through adaptive model averaging. 4, 5

[11] Mustafa Safa Ozdayi, Murat Kantarcioglu, and Yulia R. Gel. Defending against backdoors in federated learning with robust learning rate. *AAAI Conference on Artificial Intelligence (AAAI)*, 2021. 7

[12] Jie Peng, Weiyu Li, and Qing Ling. Mean aggregator is more robust than robust aggregators under label poisoning attacks. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4797–4805, 2024. 2, 3, 8

[13] Zeyu Qin, Liuyi Yao, Daoyuan Chen, Yaliang Li, Bolin Ding, and Minhao Cheng. Revisiting personalized federated learning: Robustness against backdoor attacks. *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (SigKDD)*, 2023. 4, 7

[14] Virat Shejwalkar and Amir Houmansadr. Manipulating the byzantine: Optimizing model poisoning attacks and defenses for federated learning. *Proceedings 2021 Network and Distributed System Security Symposium*, 2021. 4

[15] Yichang Xu, Ming Yin, Minghong Fang, and Neil Zhenqiang Gong. Robust federated learning mitigates client-side training data distribution inference attacks. *Companion Proceedings of the ACM on Web Conference 2024 (WWW)*, 2024. 4, 5, 6

[16] Xuyang Zhao, Huiyuan Wang, and Wei Lin. The aggregation-heterogeneity trade-off in federated learning. In *The Thirty Sixth Annual Conference on Learning Theory (COLT)*, pages 5478–5502, 2023. 4, 5, 6

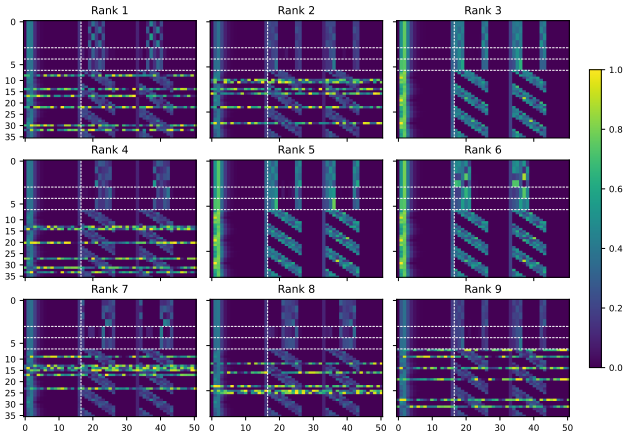


Figure 5. Non-Attack scenario.

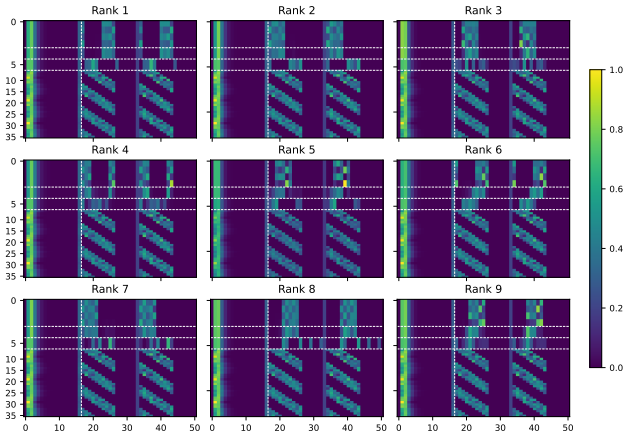


Figure 6. Random Parameter Attack (RNP).

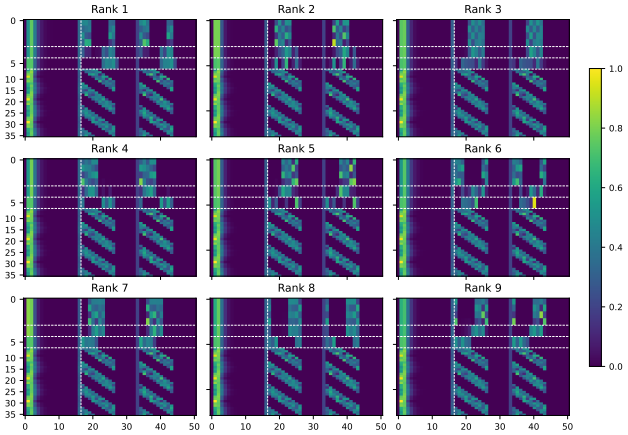


Figure 7. Mimic Parameter Attack (MiP).

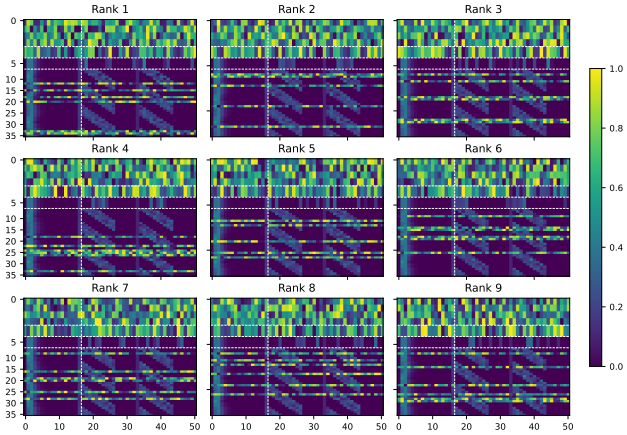


Figure 8. Label Shift Attack (LS).

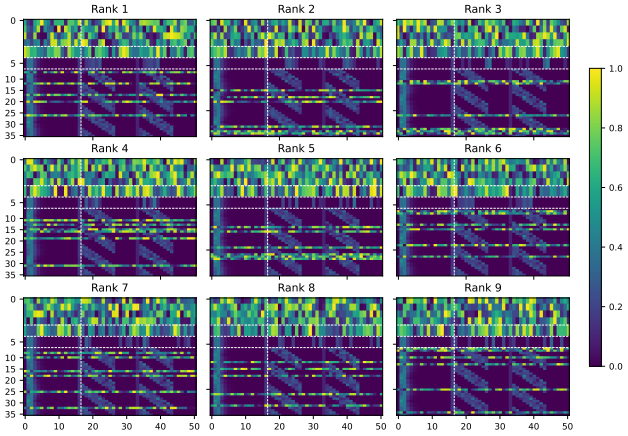


Figure 9. Label Random Editing (LRE).

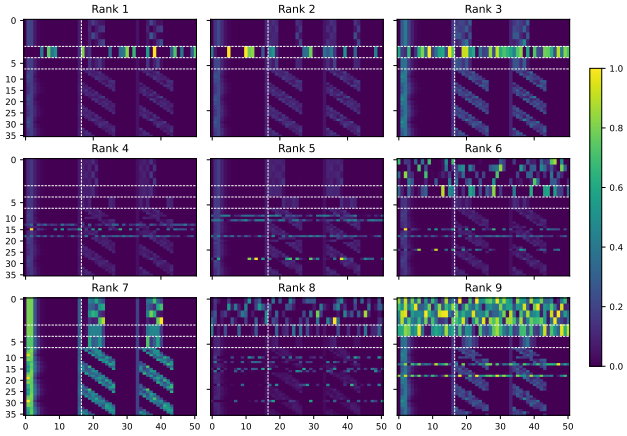


Figure 10. Random Belief Attack (RNB).

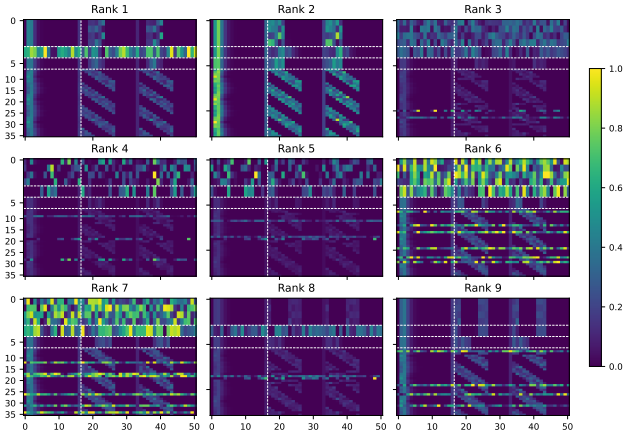


Figure 11. Mimic Belief Attack (MiB).

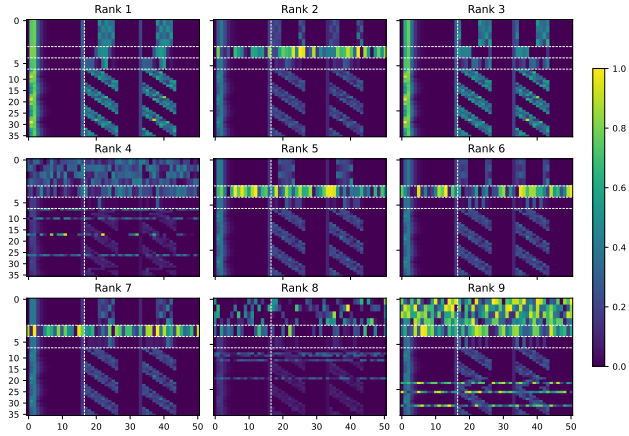


Figure 12. RNP mixed with RNB (+RNB).

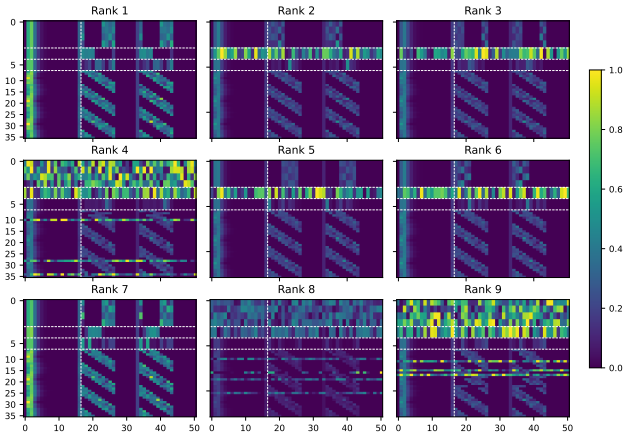


Figure 13. RNP mixed with MiB (+MiB).

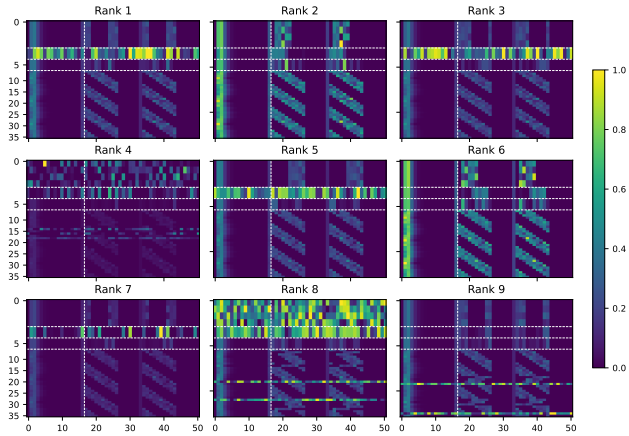


Figure 14. RNP mixed with K-Means Mimic Attack (+KmB).