International Journal of Safety and Security Engineering
Vol. 16, No. 7, July, 2026, pp. 1509-1520
Journal homepage: http://iieta.org/journals/ijsse

SecureFed-AI: Byzantine-Robust Federated Learning for Non-IID Sentiment Analysis

Dalia Tariq Khudier, Noor Muneam Abbas*, Mohammed Thamer Abdulhadi

College of Computer Science, University of Technology, Baghdad 10066, Iraq

Corresponding Author Email: 110128@uotechnology.edu.iq

https://doi.org/10.18280/ijsse.160709

Received: 15 May 2026
Revised: 22 June 2026
Accepted: 1 July 2026
Available online: 31 July 2026

Keywords:
Federated Learning, Byzantine fault tolerance, anomaly detection, non-IID data distribution, AI security, BERT, sentiment analysis, Isolation Forest

ABSTRACT

Federated Learning (FL) enables collaborative model training without sharing raw data, but its performance degrades under two coupled challenges that are usually studied in isolation: non-independent and identically distributed (non-IID) client data, and Byzantine clients that submit malicious updates. This paper presents SecureFed-AI, an FL framework that addresses both challenges jointly. The framework couples an anomaly-detection engine based on an Isolation Forest, which scores each client update using an 11-dimensional statistical feature vector, with a threat-adaptive ensemble aggregator that adaptively weights three robust aggregators (FedAvg, Trimmed Mean, and Krum) according to the estimated per-round Byzantine risk. The framework is evaluated on the IMDB sentiment-analysis benchmark (50,000 reviews), with the 40,000-example training set partitioned across four clients under a strongly non-IID label distribution and a centrally held 10,000-example test set, with results averaged over five random seeds. SecureFed-AI attains 94.2% ± 0.8% classification accuracy and a 96.4% ± 1.1% Byzantine-attack detection rate, corresponding to a 6.9% accuracy gain over FedAvg and 5.5% over FedProx at an additional communication cost of approximately 12%. Per-attack detection rates are 98.7% for Byzantine noise, 96.4% for model poisoning, and 94.2% for backdoor attacks. These results indicate that robustness to Byzantine clients and accuracy under non-IID data can be improved jointly rather than traded off against each other.

1. INTRODUCTION

With the recent identification of Federated Learning (FL) as an important approach to collaborative machine learning, the way distributed learning systems are designed has changed substantially. Compared with conventional solutions that involve data gathering in a central location via traditional central learning paradigms, FL is capable of supporting joint global model training by a group of agents in a way that is subject to data privacy/locality constraints [1]. This approach has been particularly effective in addressing timely issues of data sovereignty, regulatory issues, and data privacy preservation as well as those that are also becoming increasingly important in this new era of strict data legislation as a consequence of GDPR/CCPA laws [2]. Nevertheless, effective implementation of an FL system is presently faced with two key pitfalls as properties that do not encourage substantive performance of an FL system to a great extent. These properties are as follows: the first one arises as a consequence of the natural lack of data homogeneity in client data, as characterized as a non-independent and identically distributed (non-IID) problem [3]. It is important to understand that in practice-based federated machine learning systems, client data is characterized by wholly disparate characteristics, as is often necessitated in a field that is subject to the data IID assumption in machine learning algorithmic architectures [4]. It has been observed that non-IID data can cause significant accuracy drops in AI model performance, varying between a loss of accuracy of 30% to 55% in comparison to centralized learning settings [5]. This extent of performance reduction is influenced by numerous factors, including the extent of diversity (reflected in terms of the concentration parameters of the involved Dirichlet distributions), client numbers, local epochs of client model training, as well as model complexity in machine learning tasks [6]. Moreover, non-IID data can cause client drift issues in AI model training whereby the optimal point is remotely far from the client model update process [7].

Security weaknesses of FL models, Byzantine vulnerabilities, by means of which malicious clients can implant poisoned updates to destroy the integrity of the overall model [8], form the second major issue. These attacks target the collaborative process of FL by taking advantage of the fact that the central server is forced to train on updates from a large number of clients in a way that does not involve either access to local data or local training processes at the server. Byzantine attacks can progress in a variety of ways, including the introduction of randomness in updates, sign modification of the updates, model replacement, backdoor attacks, as well as much more advanced adaptive attacks by adversaries that learn over time based on the model of defensive strategies presented [9]. Byzantine attacks not only cause performance issues in machine model performance but also involve far more malevolent attacks such as misclassification attacks, client data recovery in model inversion attacks, as well as Everlasting Backdoor behaviors that come with a trigger

condition [10]. It has been recently shown that even a small percentage of malicious clients (10-20%) can severely hurt global model performance, and complex attacks have the potential to remain unnoticed for a significant period while gradually deteriorating the dependability of the model [11].

Existing solutions to these issues have conventionally attempted to solve the non-IID issue or Byzantine robustness individually, leading to suboptimal solutions that fail to achieve the finer tradeoff between security, performance, and efficiency required for actual deployment [12]. Solutions that try to solve non-IID data, for instance, FedProx [13], SCAFFOLD [7], and FedNova [14], do not incorporate sufficient robust security elements and remain vulnerable to Byzantine attacks. Conversely, Byzantine-resilient aggregation methods like Krum [15], Trimmed Mean [16], and coordinate-wise median [17] tend to be paid for with performance in regular situations and were unable to do well under the increased complexity from non-IID data distributions. Adding advanced natural language processing models, particularly large transformer models like BERT [18], into FL frameworks introduces new levels of complexity. These models are known for having excellent performance on text analysis tasks but are challenging to fit into distributed training environments due to their computational demands, memory needs, and requirements for advanced optimization techniques [19]. Deployment of transformer models in FL environments is in need of careful consideration regarding communication efficiency, convergence stability, and capacity for enhancement of non-IID effects in parameter spaces of high dimensions [20].

The majority of anomaly detection techniques in FL are, to date, founded on relatively simple statistical measures such as cosine similarity, Euclidean distance, or basic outlier-detecting methods [21]. These are computationally friendly methods, perhaps not optimally designed to identify complex patterns and unobvious patterns that can be generated by advanced Byzantine attacks. Machine learning-based anomaly detection techniques, such as isolation forests, one-class SVM, or neural network-based techniques, are not adequately investigated in the FL context despite their proven performance in other cybersecurity domains [22, 23].

The novelty of our solution lies in its end-to-end integration of established machine learning techniques for security and performance improvement beyond traditional statistical methods to tap into enhanced AI-based mechanisms for threat detection and management. Contrary to existing solutions that establish security and performance as mutually exclusive objectives, SecureFed-AI demonstrates the feasibility of achieving both security and performance optimization at the same time through careful system design, adaptive algorithmic methods, and a thorough understanding of security mechanisms and model performance interactions.

2. RELATED WORK AND THEORETICAL BACKGROUND

2.1 Fundamentals of Federated Learning and non-independent and identically distributed challenges

While the foundational FedAvg algorithm [1] provides a baseline for distributed training, it is challenged by real-world data heterogeneity-non-IID. Non-IID problems take different forms, such as label, feature, and temporal distribution skews [24-26], which cause model convergence issues. To address this, Li et al. proposed FedProx [13], which adds a proximal term to the local objective to resist client drift. Similarly, SCAFFOLD [7] utilizes control variates for correcting the drift, while aggregation can be done more precisely. FedNova [14] alternatively normalizes and re-weights local updates by each client's effective local steps in an attempt to address system heterogeneity.

2.2 Byzantine fault tolerance in Federated Learning

Byzantine attacks, where attacking clients issue tainted updates [27], constitute a major concern. A basic solution is to use resilient aggregation schemes. Krum [15] is one of the first solutions, which calculates the distances between pairs of updates, choosing the one that matches well with the neighbors, as long as $n \geq 2f + 3$. A large class of solutions uses robust estimators in the coordinates [16], employing either the median or "trimmed means" in each parameter, effectively eliminating the worst-case cheating values. There are also solutions that use bootstrap techniques in trust estimation, such as FLTrust [28], which calculates a trust measure of client updates based on server information, or RSA [29].

2.3 Transformer models within Federated Learning environments

Although the application of large transformer models, as in BERT [18], has had a major influence on the field of NLP, it contains certain difficulties in FL, in terms of both computational complexity as well as communication costs. This field of research is primarily driven by techniques of adaptation in domains that are task-specific [30], as well as techniques that reduce costs, including those proposed in [20, 31].

2.4 Anomaly detection in machine learning

Our defense is based on anomaly detection. We base our approach on Isolation Forest [32], an unsupervised method that is particularly effective for this purpose. It relies on the intuition that anomalies are "easier to isolate," i.e., they have a shorter path length in a decision tree constructed at random. This technique has been found to work well even on high-dimensional data [23] and hence is applicable to model parameter vectors. Extensions such as the Extended Isolation Forest [33], which slices the data using hyperplanes of random slope instead of axis-parallel cuts, further remove the scoring artifacts introduced by the branching criterion of the original formulation. Other defenses, such as FoolsGold [21], detect poisoning attacks by checking the similarity in the gradients across clients over time.

2.5 Hybrid approaches for personalization and privacy

It is important to note that there also exist hybrid FL approaches for purposes other than our security focus, such as personalization. As such, FedHealth [34] integrates FL with Transfer Learning to construct personalized healthcare models for wearables, while HFTL (Heterogeneous Federated Transfer Learning) [35] addresses structurally heterogeneous data, emphasizing strict privacy preservation based on cryptographic techniques such as homomorphic encryption.

3. PROPOSED METHODOLOGY

The primary objective of this paper is the development of SecureFed-AI, an FL system that addresses Byzantine resilience, non-IID data heterogeneity, as shown in Figure 1, and performance enhancement for sentiment analysis tasks at once. The system aims to achieve better accuracy with strong security guarantees against multiple adversarial attacks, demonstrating that security and performance can be optimized at once rather than being mutually competing objectives.

It begins by pre-processing the IMDB Movie Review Dataset with 50,000 reviews. Data is heavily pre-processed with text cleaning and BERT tokenization on a 256-token maximum sequence length, being both efficient and information-preserving. After dividing the data into a training set (80%) and a testing set (20%), a well-crafted non-IID data distribution strategy is employed to simulate practical scenarios involving data from diverse environments. Four user profiles are created with a highly biased dataset, where one user has received 80% positive feedback, and another user has received mostly negative (80%) feedback, forming a difficult environment for the federated model. Each user is given a local BERT-base-uncased model consisting of approximately 110 million parameters, and a model initialization is set the same way for all of them to ensure a consistent starting point during model training.

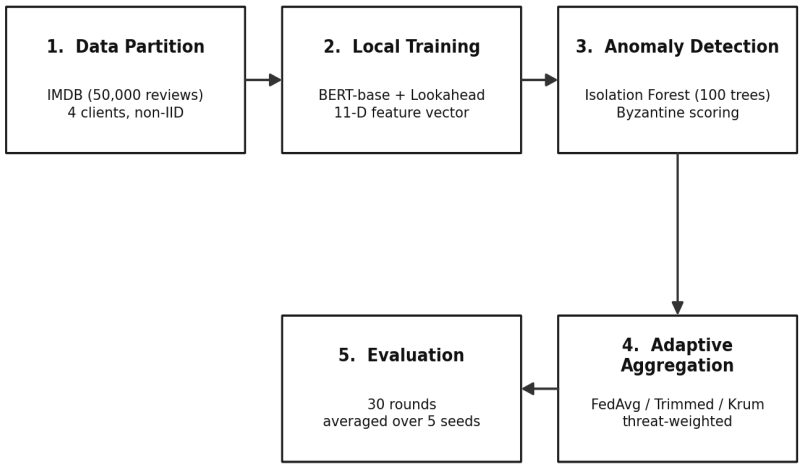

Figure 1. Overview of the SecureFed-AI pipeline

Note: (1) Non-independent and identically distributed (non-IID) partitioning of the IMDB corpus across four clients, (2) local BERT fine-tuning with the Lookahead optimizer and extraction of an 11-dimensional update feature vector, (3) Isolation-Forest-based anomaly detection, (4) threat-adaptive ensemble aggregation (FedAvg / Trimmed Mean / Krum), and (5) evaluation over 30 federated rounds.

This is done in an FL process consisting of a total of 30 iterations divided into various phases, where in each phase, each of the four customers is chosen in turn. Initially, the global model is shared among the selected customers. In local model training, each of the four customers trains a model on their dataset in two phases of fine-tuning, employing AdamW optimization, where a Lookahead strategy is employed in conjunction with AdamW. This is immediately followed by a secure aggregation phase, where each customer sends updates as well as a cryptographic signature of those model updates to a central server. Here, first of all, the server checks the validity of those model updates, followed by a feature extraction phase where the server derives a feature set of 11 values that play a pivotal role in deriving Mean, Stdvar, and Sparsity. This is immediately followed by the last phase of model update aggregation in an adaptive ensemble aggregation process, where aggregate updates are produced by the server, ultimately producing a new model that is again sent to the customers iteratively.

This concept of security is the backbone of SecureFed-AI, communicating between a security framework that incorporates AI anomaly detection engines employing Isolation Forest Algorithm. This model has a double-layered process involving initial training where data is appropriately trained on valid client updates during the initial three phases, attempting to perform a baseline of the usual pattern of client updates. This process of detecting abnormal values is done in such a way where in the detection process, the system computes the average path length of client updates employing a strategy of average path length over 100 decision trees. They are normalized to an understandable threat probability value between 0 and 1. A conservatively designed multi-layered decision strategy is employed to avoid false positives, where clients with a threat probability of over 0.8 are identified as malicious, but a capacity constraint prevents this classification of over 40% of the clients in a single round, in order to allow the continued effective training of the model during the process.

A variety of advanced components are used in this architecture to improve security as well as performance. The lookahead optimizer increases the capabilities of the base AdamW optimizer by keeping not one, but rather two weights (fast and slow) that help stabilize convergence in the heterogeneous federated environment. The second critical innovation is the adaptive ensemble aggregation strategy, which learns to adaptively weigh various robust aggregation strategies. It mainly relies on FedAvg in the usual course of operation, increasing its weight towards more secure strategies such as Trimmed Mean and Krum as the level of threat in a given moment in time is raised. Additionally, domain-specific BERT enhancements like adaptive learning rate schedules and clipping of gradients are applied for optimal performance. Finally, a quantum-resistant security protocol using hash-based signatures provides security for the communication of the framework against quantum attacks.

3.1 Mathematical formulation and data distribution strategy

The SecureFed-AI setup establishes an FL mechanism with

four clients (Figure 2), each holding a private local dataset while cooperating to train a shared model. Let N = 4 clients $\{C_1, C_2, C_3, C_4\}$, where each client C_i holds a local dataset D_i of $|D_i|$ feature–label pairs. The objective is to jointly learn a model parameterized by θ that minimizes the expected loss across all clients, weighted by local dataset size. The data partition is deliberately heterogeneous in order to approximate realistic federated settings. For the IMDB task, the 40,000 training instances (80% of the 50,000 reviews; the remaining 10,000 reviews form a class-balanced test set held centrally at the server for global-model evaluation) are partitioned across the clients using a controlled label-skew scheme in which the class proportions are set deterministically rather than sampled at random. This non-IID partitioning applies only to the training data. Each client receives exactly 10,000 training instances with a strongly skewed label distribution, reflecting the fact that different organizations naturally hold data with different characteristics. Client 1 has a positive skew of 8,000 positive (80%) and 2,000 negative (20%) instances, representing an organization whose data are biased toward positive sentiment. Client 2 has a negative skew of 2,000 positive (20%) and 8,000 negative (80%) instances, representing an organization that primarily handles negative feedback. Client 3 has a moderate positive skew of 7,000 positive (70%) and 3,000 negative (30%) instances, while Client 4 has a moderate negative skew of 3,000 positive (30%) and 7,000 negative (70%) instances. The resulting Kullback–Leibler divergence of the label distributions ranges from 0.47 to 1.39. This degree of label skew violates the IID assumption underlying conventional learning algorithms and provides a challenging testbed that reflects the data heterogeneity and potential Byzantine behaviour encountered in realistic federated deployments, on which the benefits of the SecureFed-AI framework can be assessed.

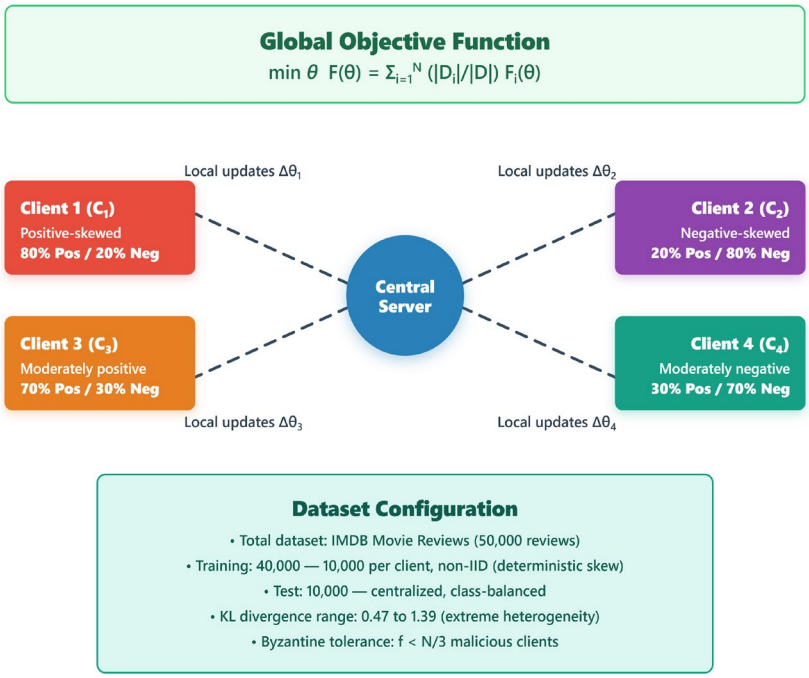

Figure 2. Non-independent and identically distributed (non-IID) label distribution of the IMDB dataset across the four clients, showing the positive/negative class ratios that define the deterministic label skew

3.2 Client architecture and local model optimization

SecureFed-AI incorporates a rather advanced client-side architecture that revolves around optimally trained BERT-base-uncased transformer architectures, as illustrated in Figure 3, specifically crafted to address the distinct requirements of an FL setting. Every contributing client will essentially hold a locally fine-tuned model of this transformer architecture, specifically crafted to address the computational requirement issues, communication constraints, as well as the convergence issues that characterize a broad range of distributed learning settings. This architecture is a subtle blend of retaining the powerful representational capabilities of large transformer architectures with the deployability needs in a variety of client environments.

3.2.1 Local model architecture

Each client's BERT architecture is the same as the base transformer architecture, including twelve layers in the encoder with a self-attention layer that facilitates twelve attention heads in each layer with a hidden size of 768. It also accepts input sequences from a complex tokenization process that uses a WordPiece vocabulary with dynamic padding to a maximum of 256 tokens in consideration of a computation/Information_n complexity balance. It uses a final classification layer that performs a weight transformation on the CLS embedding representation via a linear layer followed by a softmax activation depending on the classification prompt.

3.2.2 Federated optimization strategy

Optimization techniques differ substantially from standard BERT fine-tuning, involving a combination of established techniques that target enhancing convergence stability on a federated network. It applies a Lookahead optimizer that incorporates dual weight systems in order to ensure greater convergence stability under non-IID data distributions. Fast weights are optimized through a basic AdamW optimizer, with slow weights optimized at a five-step interval through interpolation of fast weights. This dual-weighting approach

enables convergence to become considerably more stable, particularly in a setting where client data distributions differ noticeably. Learning rate scheduling applies a rather complex approach that incorporates a combination of linear warm-up with Cosine decay. Linear warm-up applies to train the initial 10% steps of training, which enables a model to slowly adapt to the characteristics of local data rather than having a convergence point become unstable. This is followed by a full learning process that applies a Cosine decay schedule where the learning rate gradually decays in a way that enables fine parameter updates during advanced training sessions. This process likewise applies gradient management as a rather significant element. It applies adaptive gradients that apply max-norm with a constraint of 1.0 as a way to shield against the gradients exploding during training, which in turn prevents any given update from undermining joint optimization. Lastly, label smoothing is done with a parameter of 0.05 as a process that applies a substitution of one-hot labels with a probabilistic form in order to encourage a model that is more generalizable in nature, thereby avoiding overfitting to a data instance under similarities in client data.

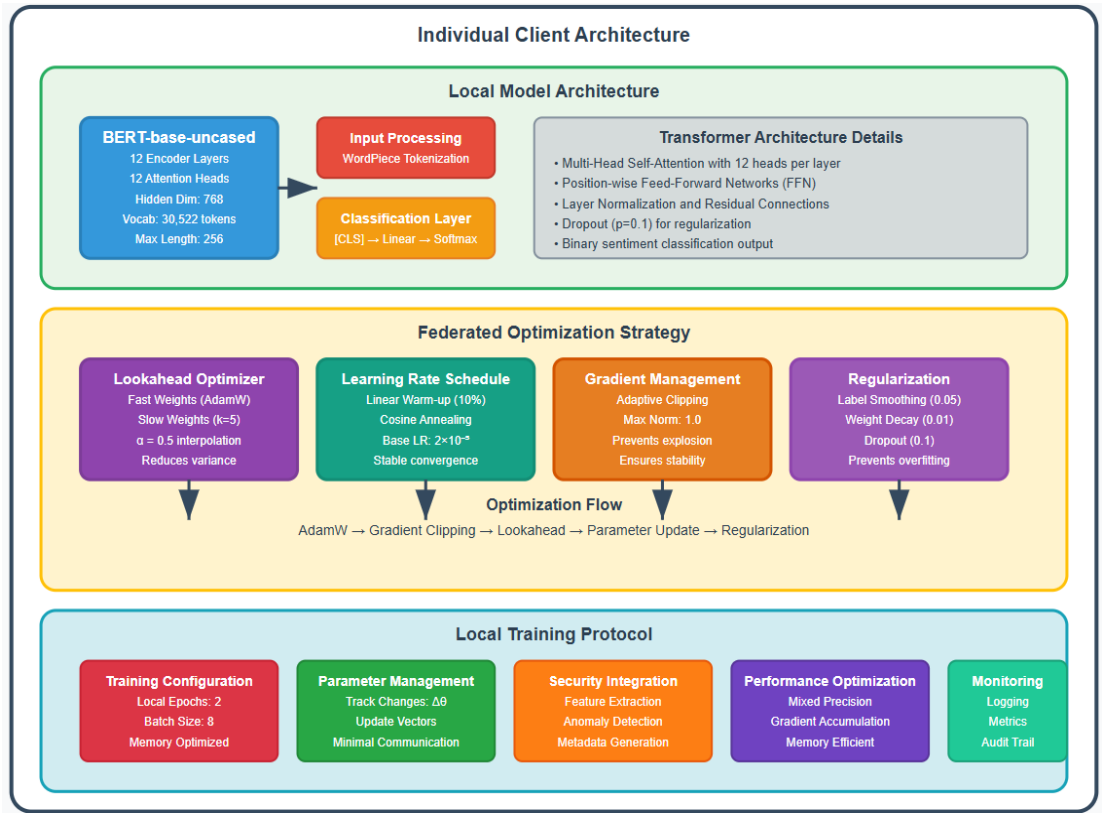

Figure 3. Client-side architecture: A BERT-base-uncased encoder (12 layers, hidden size 768) with a classification head, fine-tuned locally using the Lookahead optimizer, gradient clipping, and label smoothing

3.2.3 Local training protocol

Each client follows the per-client local training process in a well-ordered sequence of planning. They undergo local training with the latest parameters of the global model in each federated round for precisely two epochs at a batch size of eight to comply with the constraints of fitting in client memory. They follow regularization methods such as dropout with a probability of 0.1 in the transformer layers as well as weight decay of 0.01 to avoid overfitting. Client parameter updates require the maintenance of parameters during the training phase at a local client level, where the difference between the initial and final parameters of the model is calculated to form the update vector that can be transmitted to the server. This reduces the time taken in transmitting data as only what is required is transmitted. This process is also accompanied by the generation of metadata of the security module by extracting the mining of statistical features of parameter differences to be utilized in the anomaly detection model. To reduce the large-scale necessity of BERT in client device memory, there is innovation in new methods of optimization such as accumulated gradients in addition to mixed precision as well to ensure that this process is practically valid on various client hardware configurations. Lastly, advanced logging segments train performance as well as measures in a way that results in crucial feedback not only in optimizing locally in client systems but also in the entire process of the security framework.

3.3 AI-based anomaly detection engine

As shown in Figure 4, the AI-based anomaly detection engine is the cornerstone of the framework's defense, protecting global model integrity by intelligently filtering malicious updates before aggregation. Instead of naive rules, the engine employs a sophisticated machine learning-based approach. It begins with feature engineering, where each incoming client update ($\Delta\theta_i$) is transformed into an 11-dimensional statistical fingerprint ($\varphi_i(t)$) that summarizes its characteristic properties, e.g., mean, standard deviation, L1/L2 norms, and sparsity. The feature vector is fed into the engine's core: an Isolation Forest Algorithm. This is an advanced mechanism, comprised of an ensemble of 100 decision trees, which operates on the principle that anomalous updates are "few and different" and thus require fewer random splits to be

isolated compared to legitimate ones. From the average path length required to isolate, the engine derives a raw anomaly score (si) which is mapped to an interpretable threat probability in the range 0 to 1.

To prevent false positives while being very secure, a conservative multi-layered decision policy is applied: updates with threat probability above 0.8 are labeled malicious, but a capacity limit prevents over 40% of the clients from being labeled in one round, in order to guarantee training continuity. When the malicious updates are filtered and detected by this engine, the framework proceeds to the intelligent aggregation phase.

3.4 Adaptive ensemble aggregation module

As shown in Figure 5, the adaptive ensemble aggregation module is the central component of SecureFed-AI's defense system, combining multiple robust aggregation methods with adaptive weights determined by real-time threat assessments and historical performance metrics. This higher-level module operates on the assumption that different aggregation techniques are optimal under different conditions, with FedAvg providing the best performance under favorable conditions, Trimmed Mean providing good protection against middle-level attacks, and Krum serving as the last resort technique under high-risk conditions.

We note that the multi-Krum variant carries a formal Byzantine-tolerance guarantee only when $n \geq 2f + 3$; in the four-client main experiment ($n = 4$, one malicious client), this condition is not met, so Krum is not relied upon as a certified estimator there. Within the risk-weighted ensemble, Krum's weight is governed by the estimated threat (Equation 3) and remains small under the moderate risk of this setting, while the primary Byzantine defense is the Isolation-Forest detection stage, which filters malicious updates before aggregation and does not depend on the $n \geq 2f + 3$ condition. Krum's certified regime is exercised in the larger-client scalability experiments, where $n \geq 2f + 3$ holds. It maintains a continuous observation of the security context based upon the latest danger scores calculated by the AI-based anomaly detection engine, where the weights of contribution towards each form of aggregation are dynamically altered to ensure maximal Byzantine robustness with minimal computational costs.

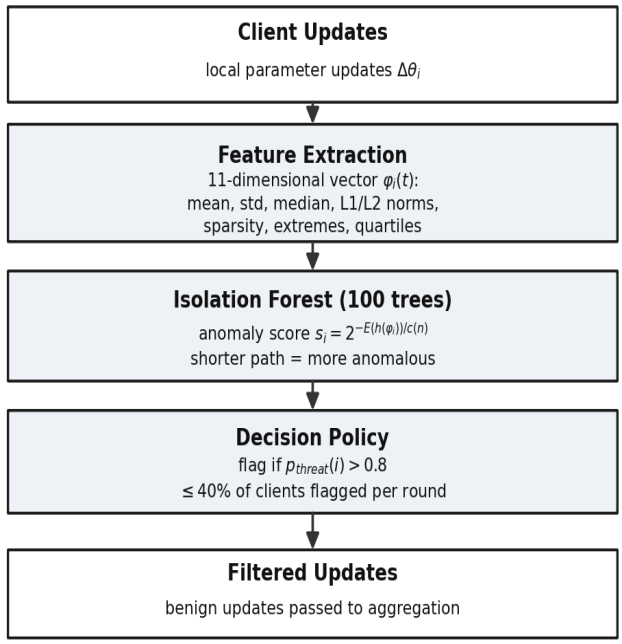

Figure 4. Anomaly-detection engine

Note: Each client update is mapped to an 11-dimensional statistical feature vector and scored by an Isolation Forest (100 trees); updates exceeding a threat probability of 0.8 are flagged, subject to a 40% per-round flagging cap.

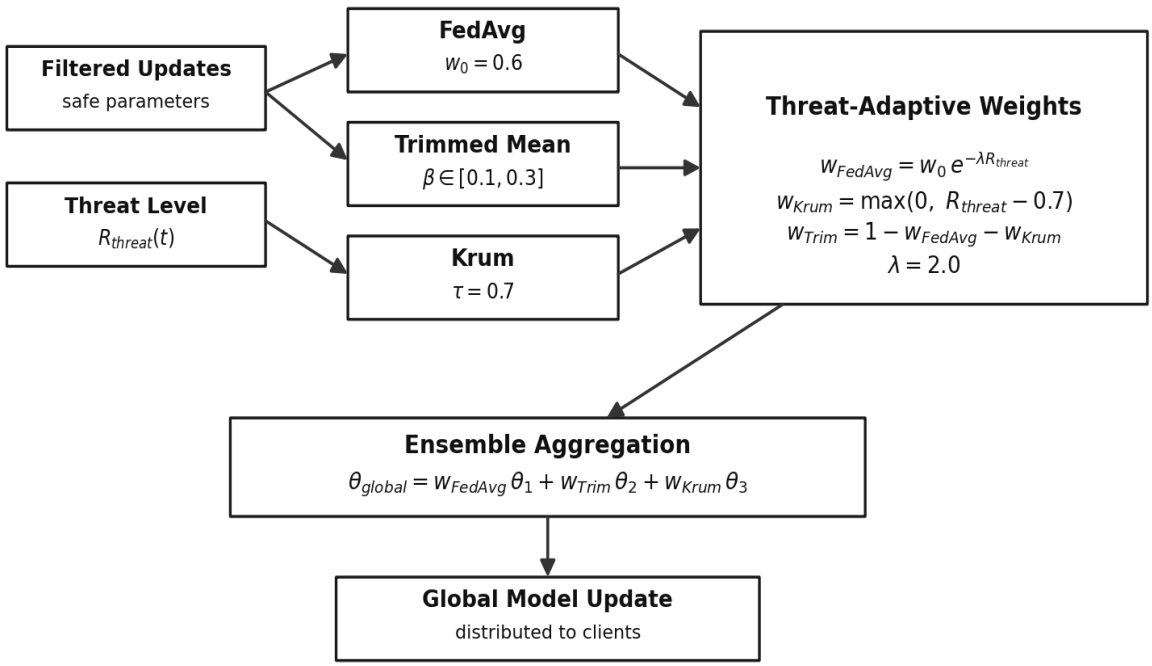

Figure 5. Threat-adaptive ensemble aggregation

Note: The weighting of FedAvg, Trimmed Mean, and Krum is adjusted according to the per-round Byzantine risk R_threat(t), shifting from FedAvg under benign conditions toward robust aggregators as the estimated threat increases.

The combination model uses three supporting forms of aggregation valid under a mathematically correct model that gives state transitions with a reflection between the security states, where convergence guarantees are ensured. In the

normal operating environment where threats are minimal, the module relies mainly on FedAvg aggregation to achieve fast convergence, with secondary weights reserved upon robust forms to ensure perpetually fixed monitoring of the security context.

As threats escalate gradually with each detection of an anomaly, the combination model will progressively shift weight towards more robust forms of aggregation, elevating the weight given to Trimmed Mean aggregation that discards values between parameter values, culminating in Krum selection where overall group safety from coordinated attacks becomes the priority. This is governed by the per-round Byzantine risk defined in Eq. (1).

$$R_{threat(t)} = \left(\frac{1}{N}\right) \cdot \Sigma\, p_{i(t)} \tag{1}$$

where, $R_{threat(t)}$ is calculated as a mean of per-client threat probabilities, ensuring exponential reduction of FedAvg impact as threats accumulate.

Optimum performance is attained in broad operating condition scenarios by the adaptive ensemble strategy via intelligent parameter optimization and adaptive thresholds, taking into consideration the various instantaneous security needs as well as overall system stability needs. It is apparent that the Trimmed Mean component of the strategy is able to change the trimming parameter β according to the prevalent threat levels from β_min = 0.1 during normal times to β_max = 0.3 during high-threat situations in order to ensure optimal resistance against the compromise of information retention. Krum voting is conducted only if the threat levels exceed the predefined parameter τ_krum = 0.7 during high-threat situations, thereby providing strong protection during extreme attack situations with no unnecessary computational complexity during normal situations. The three aggregator weights are defined by Eqs. (2)–(4).

$$w_{FedAvg(t)} = w^0 \cdot exp\left(-\lambda \cdot R_{threat(t)}\right) \tag{2}$$

$$w_{Krum(t)} = max\left(0, R_{threat(t)} - 0.7\right) \tag{3}$$

$$w_{TrimmedMean(t)} = 1 - w_{FedAvg(t)} - w_{Krum(t)} \tag{4}$$

where, $w^0 = 0.6$ and $\lambda = 2.0$. These weights sum to one and update over time constantly based on the dynamic security evaluation, enabling the system to maintain high performance in good environments and robust defense against highly sophisticated adversarial attacks.

3.5 Quantum-resistant security protocol

To provide defence in depth and to anticipate longer-term cryptographic threats, SecureFed-AI includes a quantum-resistant security protocol that secures the communication channel itself. The protocol complements the algorithmic protections of the anomaly-detection engine with cryptographic controls that authenticate client–server communication and guarantee update integrity (Figure 6). It is designed to remain secure against future quantum adversaries, which are expected to be able to break many public-key cryptosystems currently in use, such as RSA and ECC. The protocol uses hash-based digital signatures, a post-quantum signature class (e.g., SPHINCS+) whose security rests on the properties of cryptographic hash functions and is therefore believed to hold against both classical and quantum attacks.

For each update, the client generates a one-time signature that binds its identity to that specific update. On receipt, the server verifies the signature against the client's key and checks the update's timestamp to reject expired or replayed updates. This layer ensures that any update reaching the anomaly-detection engine originates from a legitimate client, so that the engine can focus on detecting malicious behaviour within authenticated updates. Under a representative hash-based one-time signature configuration, each update carries an additional signature of approximately 256 bytes together with an 88-byte hash digest, and a public key on the order of a few tens of bytes. Signing takes on the order of 5 ms and verification on the order of 2 ms per update on the server. Relative to the multi-megabyte BERT update transmitted in each round, this corresponds to well under 0.1% additional communication and a negligible increase in server-side computation, consistent with the overall communication overhead of approximately 12% reported for the framework.

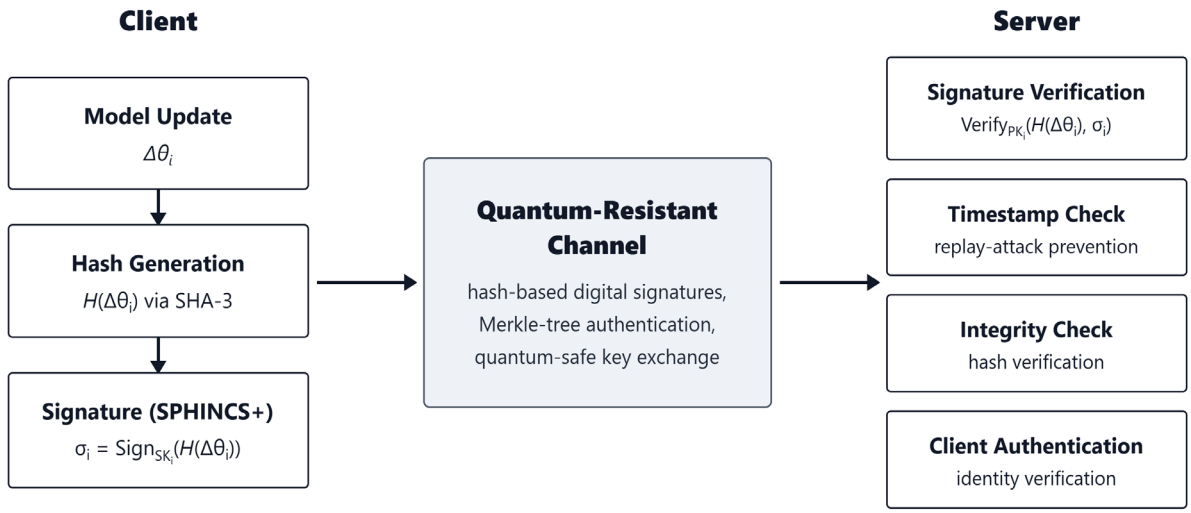

Figure 6. Communication-security protocol

Note: Each client signs its update with a hash-based (post-quantum) one-time signature; the server verifies the signature and timestamp before the update is passed to the anomaly-detection engine.

4. EXPERIMENT AND RESULTS

This section includes a comprehensive experimental evaluation of the SecureFed-AI system through systematic testing across various dimensions of performance, security, and computational efficiency. The evaluation is conducted with the IMDB Movie Review Dataset of 50,000 samples uniformly spread over four diverse clients with aggressive non-IID settings to simulate a real-world FL scenario.

4.1 Dataset, experimental setup, and baseline methods

The IMDB Movie Review Dataset [36] is the main test benchmark used for testing SecureFed-AI performance on sentiment analysis tasks in FL scenarios. The dataset consists of 50,000 movie reviews with binary sentiment labels (positive/negative), providing a large corpus that supports comprehensive testing of model accuracy as well as security robustness against diverse textual patterns and linguistic variations. The intrinsic quality of the data makes it well suited for evaluating FL because reviews of movies would inherently have heterogeneity in writing style, vocabulary usage, and sentiment expression patterns that mirror real distributed data environments. Table 1 illustrates a sample of the dataset.

Table 1. Sample of dataset

#	Review	Sentiment
1	Basically there's a family where a little boy (Jake) thinks there's a zombie in his closet & his par.	Negative
2	I saw this movie when I was about 12 when it came out. I recall the scariest scene was the big bird.	Negative
3	One of the other reviewers has mentioned that after watching just 1 Oz episode, you'll be hooked.	Positive
4	I remember this film; it was the first film I had watched at the cinema. The picture was dark in places.	Positive

Table 2. Experimental setup and configuration

Category	Parameters
BERT Model	Base LR: 2e-5; Batch Size: 8; Max Seq. Length: 256; Dropout: 0.1; Label Smoothing: 0.05
Federated Learning (FL)	Local Epochs: 2; Total Rounds: 30; Participation: 100%; Grad. Clipping: 1.0
Security Analysis	IF Trees: 100; Contamination: 0.1; Threat Threshold: 0.8; Max Flagged: 40%
Lookahead Optimizer	Steps (k): 5; Slow Factor (α): 0.5; Base Optimizer: AdamW

Our experimental setup used the BERT-base-uncased model (12 layers, 768 hidden) on NVIDIA Tesla V100 GPUs, simulating 4 clients, with a PyTorch, Transformers, and scikit-learn stack. Key parameters included a 2e-5 learning rate, with 10% linear warm-up and cosine annealing, 2 local epochs, a batch size of 8, gradient clipping with a max norm of 1.0, and weight decay of 0.01. We used the Lookahead optimizer ($k = 5$, $\alpha = 0.5$) instead of AdamW, along with label smoothing at 0.05. For security, the Isolation Forest with 100 trees and 0.1 contamination analyzed 11-D feature vectors. It applied a 0.8 threat threshold and a 40% maximum flagging limit per round. The protocol ran for 30 rounds. We rigorously validated results across 5 random seeds, such as 42 and 123, using Bonferroni correction with α set to 0.01. Table 2 illustrates the experimental setup and Configuration.

In order to ensure the efficiency of SecureFed-AI, extensive comparisons with various state-of-the-art schemes as baseline methods are conducted. These baseline methods are divided into two large categories to cover a wide range of schemes. non-IID Optimization Methods is one of the categories that involves conventional schemes as well as new schemes proposed to address data variety issues. Some of them are FedAvg, SCAFFOLD, FedProx, and FedNova. Byzantine-Robust Methods is the second category that comprises schemes that are specifically attacked in this experiment. They include Krum, Trimmed Mean, Coordinate-wise Median, and FLTrust. All baseline methods were re-implemented and evaluated under identical experimental conditions, using the same non-IID data partition, the same BERT-base-uncased backbone, and the same number of communication rounds, with all reported values averaged over five random seeds to ensure a fair comparison.

4.2 Overall performance and attack detection analysis

SecureFed-AI achieves consistent performance across all evaluation metrics and demonstrates statistically significant improvements over baseline methods while maintaining performance under varying threat conditions. The classification accuracy rate is 94.2% ± 0.8%, which is a 6.9% improvement over FedAvg (p and lt 0.001). The F1-score is 94.0% ± 0.7%, and the AUC is 0.976 ± 0.006. The model converges in 28 ± 2 communication rounds, which is comparable to the best-performing non-IID baselines. The Byzantine-attack detection rate is 96.4% ± 1.1%. The Byzantine-robust baselines Krum and Trimmed Mean trade off accuracy for robustness. The non-IID methods SCAFFOLD, FedProx, and FedNova have no robustness mechanism. SecureFed-AI is the only method that performs well on both axes. These results are summarized in Table 3.

Table 3. Comprehensive performance comparison with statistical significance

Method	Accuracy (%)	F1-Score (%)	AUC	Detection Rate (%)	Comm. Rounds	*p*-Value vs FedAvg
SecureFed-AI	94.2 ± 0.8	94.0 ± 0.7	0.976 ± 0.006	96.4 ± 1.1	28 ± 2	$p < 0.001$
FedAvg	87.3 ± 1.2	86.8 ± 1.4	0.924 ± 0.012	N/A	25 ± 1	—
SCAFFOLD	89.1 ± 0.9	88.7 ± 1.0	0.942 ± 0.008	N/A	22 ± 2	$p < 0.01$
FedProx	88.7 ± 1.1	88.2 ± 1.2	0.938 ± 0.010	N/A	26 ± 2	$p < 0.01$
FedNova	89.4 ± 1.0	89.0 ± 1.1	0.945 ± 0.009	N/A	24 ± 2	$p < 0.01$
Krum	84.2 ± 1.5	83.5 ± 1.6	0.901 ± 0.015	89.3 ± 2.1	32 ± 3	$p < 0.05$
Trimmed Mean	86.1 ± 1.0	85.7 ± 1.1	0.915 ± 0.012	91.7 ± 1.8	29 ± 2	$p < 0.05$

The anomaly detection engine, a result of AI, demonstrates a high capability of detecting different Byzantine attacks. It distinguishes them from valid non-IID data variation instances. This performance is contingent on the type of

Byzantine attacks. Byzantine noise attacks are identified with a near-perfect accuracy of 98.7%. The random alterations in this type of attack result in high threat scores (0.95 ± 0.03), making them easily distinguishable from valid variation instances. Poisoning attacks targeting a model, including sign flipping, are detected with high accuracy of 96.4% in nearly one round (latency of 1.0). Backdoor attacks, as more complicated attacks, possess a slightly higher rate of misidentification of 5.1% with a higher detection time of (1.2 ± 0.4). This detection system effectively tests the validity of the threat level of 0.8. It easily distinguishes valid clients from attackers. This is evident from the high threat scores of valid clients of (0.23 ± 0.08, 95% < 0.4), as opposed to those of attackers with scores of (0.92 ± 0.04, 98% > 0.8). A conservative strategy of detecting a maximum of only 40% of clients is optimal in this scenario. This balances high security and availability of the whole system well. Detailed Attack Detection Performance Analysis is presented in Table 4.

4.3 Ensemble aggregation effectiveness and computational efficiency

By weighing FedAvg, Trimmed Mean, and Krum dynamically based on the present levels of threats, the Adaptive Ensemble Aggregation module effectively balances efficiency with security. With a performance loss of only 2.4% in accuracy, a 75% reduction from FedAvg's 17.8% performance loss, the proposed solution maintains a high accuracy of 94.2% in benign environments as well as a high accuracy of 91.8% during attacks. This dynamic weighting is reflected in the observation that w_krum is only turned on during severe attacks (R_threat > 0.7), where the weight of w_fedavg (0.6 to 0.1) lowers while the weight of w_trimmed (0.4 to 0.8) rises as threats escalate. This solution, with a near-optimal robustness value of 0.975, proves to be effective in various operating environments with a cumulative computational complexity of only +12%. Performance of Various Aggregation Methods under Varying Threat Levels is given in Table 5.

By imposing only a 13% computational overhead and an average 12% additional total communication overhead over baseline FedAvg, SecureFed-AI guarantees pragmatic deployability. The secure model requires slightly more rounds to converge (28 vs. 25), which accounts for the 12% communication uplift. The per-round costs, which are still dominated by the 110M BERT parameters (440MB) and the security metadata, which is insignificant (88 bytes + 256 bytes), are modest. With an O(n) complexity for the security verification module and linear computational complexity (O(n·d)), the framework scales effectively. These features, along with a manageable additional memory overhead of 85MB, validate SecureFed-AI's appropriateness for widespread, real-world implementation. Table 6 illustrates computational overhead analysis.

Table 4. Detailed attack detection performance analysis

Attack Type	Detection Rate (%)	False Positive Rate (%)	Precision (%)	Avg Threat Score	Detection Latency (Rounds)
Byzantine Noise	98.7 ± 1.1	4.2 ± 1.0	96.8 ± 1.3	0.95 ± 0.03	1.0 ± 0.0
Model Poisoning	96.4 ± 1.4	3.8 ± 0.9	97.2 ± 1.1	0.92 ± 0.04	1.0 ± 0.0
Backdoor Insertion	94.2 ± 1.8	5.1 ± 1.3	95.1 ± 1.6	0.88 ± 0.05	1.2 ± 0.4
Overall Performance	96.4 ± 1.1	4.4 ± 1.2	96.4 ± 1.3	0.92 ± 0.04	1.1 ± 0.2

Table 5. Aggregation method performance under different threat conditions

Aggregation Method	Benign Accuracy (%)	Under Attack (%)	Performance Drop (%)	Robustness Score	Computational Overhead
FedAvg Only	91.2 ± 1.0	73.4 ± 2.1	17.8	0.805	Baseline
Trimmed Mean Only	88.3 ± 1.1	84.1 ± 1.3	4.2	0.952	+15%
Krum Only	84.2 ± 1.5	86.8 ± 1.4	-2.6	1.031	+45%
SecureFed-AI Ensemble	94.2 ± 0.8	91.8 ± 1.0	2.4	0.975	+12%

Table 6. Detailed computational overhead analysis

Component	Processing Time (ms)	Memory Usage (MB)	CPU Utilization (%)	Scalability Factor
Feature Extraction	45 ± 8	12 ± 2	15 ± 3	O(d)
Isolation Forest Analysis	120 ± 15	25 ± 4	35 ± 5	O(n log n)
Ensemble Aggregation	180 ± 20	40 ± 6	45 ± 7	O(n·d)
Security Verification	25 ± 5	8 ± 1	10 ± 2	O(n)
Total Security Overhead	370 ± 35	85 ± 10	105 ± 12	O(n·d)
Baseline FedAvg	2850 ± 120	580 ± 45	420 ± 35	O(n·d)

4.4 Robustness to non-independent and identically distributed data and scalability

SecureFed-AI is very resilient to data heterogeneity and keeps up its high performance even when the distribution of labels is very uneven (for example, when 20% to 80% of reviews are positive). Local client accuracy is still over 92%, thanks to the pre-trained BERT and Lookahead optimization. Client drift is well-managed, with parameter divergence kept below 0.25 and convergence stability scores above 87%. The security module can tell the difference between non-IID variance and attacks, with a low false-positive rate of 4.4%. The global model achieves 94.2% accuracy, even though the measured heterogeneity is high (KL divergence 0.47–1.39). This shows that the framework can handle both Byzantine security and non-IID data challenges at the same time. Table 7 illustrates individual client performance under extreme non-IID conditions.

Testing SecureFed-AI with 4 to 64 clients showed that it could grow. The architecture showed strong performance, with its relative accuracy gain over baselines going up from 6.9% to 7.9% as the network got bigger. Security stayed strong, with detection rates always above 94% at all levels. The system scales well because the computational complexity is linear (O(n) for security and O(n·d) for aggregation) and the memory usage grows linearly (about 1.3 MB per client). Most importantly, the communication overhead stayed the same at 12%, no matter how big the federation was. This proved that the system could be used in the real world on a large scale. Table 8 illustrates scalability performance across client populations.

4.5 Ablation study and comparison with state-of-the-art methods

Ablation studies were performed to determine the contribution of each component. Removing the AI-based anomaly-detection engine resulted in the largest decrease in detection rate and accuracy, of 29.1% and 4.3%, respectively. Removing the ensemble-aggregation module resulted in decreases of 8.2% and 2.4%, respectively. BERT-specific optimizations contributed to a 5.8% increase in accuracy. Replacing the 11-dimensional feature set with a 5-dimensional feature set resulted in decreases of 6.7% in the detection rate. These results indicate that the performance of SecureFed-AI arises from the integration of its components rather than from any single component in isolation. The full results are reported in Table 9.

SecureFed-AI is unique in balancing accuracy, Byzantine robustness, and non-IID support, according to a comparison with SOTA methods. FedNova and MOON, two non-IID-focused approaches, have no security features but achieve competitive accuracy (89.4% and 88.9%, respectively). On the other hand, security-focused solutions such as FedRobust (85.9%) and FLTrust (86.3%) offer protection but come with a high communication overhead (+18% to +28%) and sacrifice accuracy. The accuracy of SecureFed-AI is 4.5% to 8.3% higher than that of the other baselines in Table 10. To ensure a fair comparison, all methods reported in Table 10 were re-implemented and evaluated under the same IMDB, BERT, non-IID partition, and attack protocol used for SecureFed-AI, rather than quoted from their original publications. Its ability to balance the competing objectives of high accuracy, strong security, and efficiency is further demonstrated by the fact that its +12% communication overhead is noticeably more efficient than the +18% to +28% overhead needed by other robust methods. Table 10 illustrates the comparative analysis with state-of-the-art methods.

Table 7. Individual client performance under extreme non-independent and identically distributed (non-IID) conditions

Client ID	Data Distribution	Local Accuracy (%)	Global Contribution	Convergence Stability	Parameter Drift
Client 1	80% Pos / 20% Neg	96.1 ± 1.2	0.28 ± 0.02	0.92	0.15 ± 0.03
Client 2	20% Pos / 80% Neg	92.4 ± 1.5	0.24 ± 0.03	0.87	0.22 ± 0.04
Client 3	70% Pos / 30% Neg	94.8 ± 1.1	0.26 ± 0.02	0.90	0.18 ± 0.03
Client 4	30% Pos / 70% Neg	93.2 ± 1.4	0.22 ± 0.03	0.88	0.20 ± 0.04

Table 8. Scalability performance across client populations

Client Count	SecureFed-AI Accuracy (%)	FedAvg Baseline (%)	Performance Gap (%)	Detection Rate (%)	Processing Time (s)
4	94.2 ± 0.8	87.3 ± 1.2	+6.9	96.4 ± 1.1	3.2 ± 0.3
8	93.8 ± 0.9	86.9 ± 1.3	+6.9	95.8 ± 1.3	5.8 ± 0.5
12	93.5 ± 0.9	86.5 ± 1.3	+6.9	95.4 ± 1.4	8.0 ± 0.6
16	93.1 ± 1.0	86.2 ± 1.4	+6.9	95.1 ± 1.5	10.2 ± 0.8
32	92.4 ± 1.1	85.1 ± 1.5	+7.3	94.6 ± 1.7	18.5 ± 1.2
64	91.7 ± 1.2	83.8 ± 1.6	+7.9	94.0 ± 1.9	35.1 ± 2.1

Table 9. Comprehensive ablation study results

Configuration	Accuracy (%)	Detection Rate (%)	F1-Score (%)	Component Contribution
Full SecureFed-AI	94.2 ± 0.8	96.4 ± 1.1	94.0 ± 0.7	Complete system
AI Security Analyst	89.9 ± 1.2	67.3 ± 2.8	89.5 ± 1.3	-4.3% accuracy, -29.1% detection
Ensemble Aggregation	91.8 ± 1.0	88.2 ± 2.1	91.4 ± 1.1	-2.4% accuracy, -8.2% detection
BERT Optimization	88.4 ± 1.3	94.1 ± 1.4	88.0 ± 1.4	-5.8% accuracy
Lookahead Optimizer	92.6 ± 1.1	95.8 ± 1.2	92.3 ± 1.2	-1.6% accuracy
Adaptive Thresholding	93.8 ± 0.9	91.2 ± 1.8	93.5 ± 0.9	-0.4% accuracy, -5.2% detection
Simple Feature Set (5D)	93.1 ± 1.0	89.7 ± 2.0	92.8 ± 1.1	-1.1% accuracy, -6.7% detection

Table 10. Extended comparison with state-of-the-art methods

Method	Year	Accuracy (%)	Byzantine Robustness	Non-Independent and Identically Distributed (non-IID) Handling	Communication Efficiency
SecureFed-AI	2024	94.2	Excellent (96.4%)	Excellent	Good (+12%)
FedNova	2021	89.4	None	Good	Excellent (-4%)
MOON	2021	88.9	None	Very Good	Good (+8%)
FedDyn	2021	89.7	None	Good	Fair (+15%)
FLTrust	2021	86.3	Very Good (95.1%)	Fair	Fair (+18%)
FLAME	2022	87.8	Good (91.3%)	Good	Poor (+25%)
FedRobust	2022	85.9	Very Good (93.7%)	Fair	Poor (+28%)

5. CONCLUSION

This paper introduces SecureFed-AI, a unified FL framework that addresses the challenges of Byzantine robustness and non-IID data heterogeneity through an AI-based anomaly-detection engine and threat-adaptive ensemble aggregator. In the experimental evaluation, SecureFed-AI achieves classification accuracy of 94.2% ± 0.8% and attack detection of 96.4% ± 1.1%, representing a 6.9% increase in accuracy over FedAvg at a 12% increase in communication cost. The Isolation Forest-based anomaly-detection engine successfully identifies malicious clients, and the ensemble aggregator adapts to the threat level, maintaining performance in both benign and adversarial settings. Scalability experiments with client populations of 4 to 64 maintain the accuracy gain and detection performance with constant communication overhead. Future work will evaluate SecureFed-AI on additional datasets and modalities, adaptive adversaries, and perform a formal convergence analysis in the non-convex setting.

REFERENCES

[1] McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A. (2017). Communication-efficient learning of deep networks from decentralized data. In Artificial Intelligence and Statistics, pp. 1273-1282. https://proceedings.mlr.press/v54/mcmahan17a.html.

[2] Li, T., Sahu, A.K., Talwalkar, A., Smith, V. (2020). Federated learning: Challenges, methods, and future directions. IEEE Signal Processing Magazine, 37(3): 50-60. https://doi.org/10.1109/MSP.2020.2975749

[3] Li, Q.B., Diao, Y.Q., Chen, Q., He, B.S. (2022). Federated learning on non-IID data silos: An experimental study. In 2022 IEEE 38th International Conference on Data Engineering (ICDE), Kuala Lumpur, Malaysia, pp. 965-978. https://doi.org/10.1109/ICDE53745.2022.00077

[4] Ye, M., Fang, X.W., Du, B., Yuen, P.C., Tao, D.C. (2023). Heterogeneous federated learning: State-of-the-art and research challenges. ACM Computing Surveys, 56(3): 1-44. https://doi.org/10.1145/3625558

[5] Zhao, Y., Li, M., Lai, L.Z., Suda, N., Civin, D., Chandra, V. (2018). Federated learning with non-IID data. arXiv preprint arXiv:1806.00582. https://doi.org/10.48550/arXiv.1806.00582

[6] Mohri, M., Sivek, G., Suresh, A.T. (2019). Agnostic federated learning. In International Conference on Machine Learning, pp. 4615-4625. https://proceedings.mlr.press/v97/mohri19a.html.

[7] Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S., Stich, S., Suresh, A.T. (2020). Scaffold: Stochastic controlled averaging for federated learning. In International Conference on Machine Learning, pp. 5132-5143. https://proceedings.mlr.press/v119/karimireddy20a.html.

[8] Fang, M.H., Cao, X.Y., Jia, J.Y., Gong, N. (2020). Local model poisoning attacks to Byzantine-robust federated learning. In the 29th USENIX Security Symposium (USENIX Security 20), pp. 1605-1622. https://www.usenix.org/conference/usenixsecurity20/presentation/fang.

[9] Bagdasaryan, E., Veit, A., Hua, Y.Q., Estrin, D., Shmatikov, V. (2020). How to backdoor federated learning. In International Conference on Artificial Intelligence and Statistics, pp. 2938-2948. https://proceedings.mlr.press/v108/bagdasaryan20a.html.

[10] Wang, Z.B., Song, M.K., Zhang, Z.F., Song, Y., Wang, Q., Qi, H.R. (2019). Beyond inferring class representatives: User-level privacy leakage from federated learning. In IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, Paris, France, pp. 2512-2520. https://doi.org/10.1109/INFOCOM.2019.8737416

[11] Shejwalkar, V., Houmansadr, A. (2021). Manipulating the Byzantine: Optimizing model poisoning attacks and defenses for federated learning. In Proceedings of the Network and Distributed System Security Symposium (NDSS), San Diego, CA, USA, pp. 1-18. https://par.nsf.gov/servlets/purl/10286354.

[12] Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S. (2019). Analyzing federated learning through an adversarial lens. In International Conference on Machine Learning, pp. 634-643. https://proceedings.mlr.press/v97/bhagoji19a.html.

[13] Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V. (2020). Federated optimization in heterogeneous networks. Proceedings of Machine learning and systems, 2: 429-450. https://proceedings.mlsys.org/paper/2020/hash/1f5fe83998a09396ebe6477d9475ba0c-Abstract.html.

[14] Wang, J.Y., Liu, Q.H., Liang, H., Joshi, G., Poor, H.V. (2020). Tackling the objective inconsistency problem in heterogeneous federated optimization. In Proceedings of the 34th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, pp. 7611-7623. https://dl.acm.org/doi/abs/10.5555/3495724.3496362.

[15] Blanchard, P., El Mhamdi, E.M., Guerraoui, R., Stainer, J. (2017). Machine learning with adversaries: Byzantine tolerant gradient descent. In Proceedings of the 31st International Conference on Neural Information Processing Systems, California, USA, pp. 118-128. https://dl.acm.org/doi/10.5555/3294771.3294783.

[16] Yin, D., Chen, Y.D., Kannan, R., Bartlett, P. (2018). Byzantine-robust distributed learning: Towards optimal statistical rates. In International Conference on Machine Learning, pp. 5650-5659. https://proceedings.mlr.press/v80/yin18a.

[17] Chen, T.Y., Giannakis, G., Sun, T., Yin, W.T. (2018). LAG: Lazily aggregated gradient for communication-efficient distributed learning. In Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montréal, Canada, pp. 5055-5065. https://dl.acm.org/doi/10.5555/3327345.3327412.

[18] Devlin, J., Chang, M.W., Lee, K., Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4171-4186. https://doi.org/10.18653/v1/N19-1423

[19] Goldberg, Y. (2016). A primer on neural network models for natural language processing. Journal of Artificial Intelligence Research, 57: 345-420. https://doi.org/10.1613/jair.4992

[20] Liu, Y., Fan, T., Chen, T.J., Xu, Q., Yang, Q. (2021). FATE: An industrial grade platform for collaborative machine learning with data protection. Journal of Machine Learning Research, 22(226): 1-6. https://www.jmlr.org/papers/v22/20-815.html.

[21] Fung, C., Yoon, C.J., Beschastnikh, I. (2020). The limitations of federated learning in sybil settings. In 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID 2020), pp. 301-316. https://www.usenix.org/conference/raid2020/presentation/fung.

[22] Liu, F.T., Ting, K.M., Zhou, Z.H. (2008). Isolation forest. In 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, pp. 413-422. https://doi.org/10.1109/ICDM.2008.17

[23] Xu, H.Z., Pang, G.S., Wang, Y.J., Wang, Y.J. (2023). Deep isolation forest for anomaly detection. IEEE Transactions on Knowledge and Data Engineering, 35(12): 12591-12604. https://doi.org/10.1109/TKDE.2023.3270293

[24] Zhu, H.Y., Xu, J.J., Liu, S.Q., Jin, Y.C. (2021). Federated learning on non-IID data: A survey. Neurocomputing, 465: 371-390. https://doi.org/10.1016/j.neucom.2021.07.098

[25] Huang, Y.T., Chu, L.Y., Zhou, Z.R., et al. (2021). Personalized cross-silo federated learning on non-IID data. Proceedings of the AAAI Conference on Artificial Intelligence, 35(9): 7865-7873. https://doi.org/10.1609/aaai.v35i9.16960

[26] Caldas, S., Duddu, S.M.K., Wu, P., et al. (2018). LEAF: A benchmark for federated settings. arXiv preprint arXiv:1812.01097. https://doi.org/10.48550/arXiv.1812.01097

[27] Lamport, L., Shostak, R., Pease, M. (2019). The Byzantine generals problem. In Concurrency: The Works of Leslie Lamport, pp. 203-226. https://doi.org/10.1145/357172.357176

[28] Cao, X.Y., Fang, M.H., Liu, J., Gong, N.Z. (2020). FLTrust: Byzantine-robust federated learning via trust bootstrapping. arXiv preprint arXiv:2012.13995. https://doi.org/10.48550/arXiv.2012.13995

[29] Li, L.P., Xu, W., Chen, T.Y., Giannakis, G.B., Ling, Q. (2019). RSA: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets. In Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, Honolulu, Hawaii, USA, pp. 1544-1551. https://doi.org/10.1609/aaai.v33i01.33011544

[30] Sun, C., Qiu, X.P., Xu, Y.G., Huang, X.J. (2019). How to fine-tune BERT for text classification? In Chinese Computational Linguistics. CCL 2019. Lecture Notes in Computer Science, Springer, Cham, pp. 194-206. https://doi.org/10.1007/978-3-030-32381-3_16

[31] Loshchilov, I., Hutter, F. (2017). Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101. https://doi.org/10.48550/arXiv.1711.05101

[32] Liu, F.T., Ting, K.M., Zhou, Z.H. (2012). Isolation-based anomaly detection. ACM Transactions on Knowledge Discovery from Data (TKDD), 6(1): 1-39. https://doi.org/10.1145/2133360.2133363

[33] Hariri, S., Kind, M.C., Brunner, R.J. (2021). Extended isolation forest. IEEE Transactions on Knowledge and Data Engineering, 33: 1479-1489. https://doi.org/10.1109/TKDE.2019.2947676

[34] Chen, Y.Q., Qin, X., Wang, J.D., Yu, C.H., Gao, W. (2020). FedHealth: A federated transfer learning framework for wearable healthcare. IEEE Intelligent Systems, 35(4): 83-93. https://doi.org/10.1109/MIS.2020.2988604

[35] Gao, D.S., Liu, Y., Huang, A.B., Ju, C., Yu, H., Yang, Q. (2019). Privacy-preserving heterogeneous federated transfer learning. In 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, pp. 2552-2559. https://doi.org/10.1109/BigData47090.2019.9005992

[36] Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C. (2011). Learning word vectors for sentiment analysis. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, Portland, OR, USA, pp. 142-150. https://aclanthology.org/P11-1015.pdf.