computers

MDPI

Article

Robust Adversarial Attack Detection in Resource-Constrained IoT Ecosystems: A Privacy-Preserving Framework Using Federated Learning

Syed Sadiqur Rahman

Department of Cyber Security and Forensic Computing, Collage of Computer and Cyber Sciences, University of Prince Mugrin, Al Aqool, Madinah 42241, Saudi Arabia; s.syed@upm.edu.sa

Abstract

Lightweight, privacy-aware and adversarial robust intrusion detection is required for the proliferation of Internet of Things (IoT) devices. In the Industrial Internet of Things (IIoT), centralized detectors can be compromised by adversarial perturbations via gradient-based attacks, making them susceptible to raw traffic. We suggest Federated Learning-Adaptive Gated Recurrent Unit (FL-AdGRU), a Federated approach that combines a lightweight Gated Recurrent Unit (GRU) classifier with alternating adversarial fine-tuning on each client using FGSM and PGD, without any communication overhead. A two-stage resampling scheme (UCAS-SMOTE) reduces the class-imbalance ratio from 4081:1 to ≈4:1, followed by 61 features being reduced to 40 by a mutual-information selector (MI-SelectK). Under this scenario, FL-AdGRU achieves 99.9% accuracy and 0.999 weighted F1 (+6.5 p.p. over the federated DNN baseline), with no loss of accuracy when facing clean attacks, and boosts Fast Gradient Sign Method FGSM/Projected Gradient Descent (PGD) robustness by +19.3/+19.0 p.p. at the same level of $\epsilon = 0.1$, thus effectively balancing the accuracy–robustness trade-off. It is robust (97.8%/84.2% on UNSW-NB15) and generalizes well to UNSW-NB15, while decaying slowly in skeptical scenarios (≈99.9% weighted F1 for moderate skew, 93.9%/86.7% for severe). Assuring data-locality privacy through exchange of only model weights; defenses against inference attack are left for future work. FL-AdGRU, with a total communication of 43.8 MB (≈50× less than centralized training), is deployable on bandwidth-constrained IIoT networks.

Keywords: cyber security; adversarial attack; federated learning; IoT; intrusion detection; privacy-preserving

Academic Editor: Paolo Bellavista

Received: 17 May 2026
Revised: 15 June 2026
Accepted: 16 June 2026
Published: 8 July 2026

1. Introduction

1.1. The IoT Security Imperative

The Internet of Things (IoT) is changing all fields of modern infrastructure, from smart manufacturing and precision agriculture to connected healthcare and autonomous logistics. Industry forecasts that the number of globally connected IoT devices will surpass 41 billion by 2027, with the IoT market exceeding USD 1.8 trillion by 2028 at a compound annual growth rate of 25.4% [1]. This geometric growth has, however, produced an unprecedented and very fast-expanding attack surface. According to Kaspersky researchers, the number of cyberattacks on IoT devices increased more than twice between 2020 and 2021, reaching 1.5 billion incidents [2]. The consequences of breaches are particularly severe for industrial IoT (IIoT) systems that underpin critical national infrastructure, such as energy grids,

water treatment and supply chains: the 2021 Darkside ransomware attack on Colonial Pipeline resulted in a six-day shutdown of the largest US fuel pipeline and the payment of approximately USD 5 million in ransom [3]. The security threat environment facing IoT systems is characterized by several properties that distinguish it from traditional IT security environments. To start with, IoT devices are inherently resource-constrained: the typical edge devices include Raspberry Pi boards, Arduino microcontrollers, and ESP32 modules that are severely limited in CPU capacity (1.5 GHz quad-core ARM Cortex-A72 at best), RAM (4 GB maximum) and energy budgets—making the deployment of computationally intensive security models impractical. Second, IoT networks produce continuous, high-volume, heterogeneous data streams and communicate over lightweight protocols (MQTT, Modbus/TCP, CoAP, AMQP) that were not designed with security in mind. Third, IoT settings are characterized by a highly uneven distribution of traffic data classes: normal (benign) traffic dominates attack traffic by a very large margin: the ratios between classes exceed 4000:1 in real-world datasets [1]. All these properties collectively create a trilemma: security models need to be high in detection accuracy, low in computational budget constraints, and extreme in class imbalance. In addition to such threats during runtime, there are longer-term threats posed to the cryptographic primitives used to secure IoT communications and IoT-driven financial ecosystems from quantum computers, for which hybrid classical quantum architectures [4] are an appropriate solution, which complements the approach here of defence via harder learning architectures. IoT systems have a threat environment that differs in several ways from the traditional IT security world. Secondly, when it comes to security models, IoT devices are inherently resource-constrained: a typical edge device, e.g., Raspberry Pi boards, Arduino microcontrollers, and ESP32 modules, is severely limited by its CPU power (at best, a 1.5 GHz quad-core ARM Cortex-A72), RAM (up to 4 GB), and energy budget, making deployment of a computationally intensive security model impractical. Second, IoT networks emit streams of data with varying volumes and heterogeneity over lightweight protocols that were not originally designed for security (MQTT, Modbus/TCP, CoAP, AMQP). Third, IoT traffic is highly class imbalanced, meaning benign traffic significantly outnumbers attack traffic, and inter-class ratios can be as high as 4000:1 in real-world datasets. These properties form a trilemma: a security model cannot have a tight computational budget and high detection accuracy without trading off the other, and a system cannot withstand extreme class imbalance without trimming either property.

1.2. Limitations of Existing Approaches

Conventional machine learning-based intrusion detection systems (IDSs) to detect intruders in an IoT environment adopt a centralized architecture where all edge devices collect raw network traffic, which is sent to a central cloud server, and used to train a shared detection model. Although it has the advantage of rich, aggregated data, there are three basic limitations that preclude its use in a realistic IoT setup.

- Invasion of privacy: Sending uncoded IoT traffic to a central server exposes sensitive data—including parameters of industrial processes, behavioral patterns of users, proprietary sensor readings, and personally identifiable data on device usage—to potential interception and unauthorized access. This is especially problematic when using regulatory frameworks like the European Union General Data Protection Regulation (GDPR), which requires the data minimization and locality principles of personal data processing.
- Communication overhead: The raw dataset used in this study is the Edge-IIoTset [1] raw data with about 2.26 million samples and 61 features, which represents more than 2 GB of CSV data. Transferring this large volume of data over limited IoT

networks (bandwidths ranging from 0.3 kbps (LoRaWAN) to 1 Mbps (Zigbee)) is simply infeasible when real-time or near-real-time security monitoring applications are required.

- Adversarial vulnerability: The standard centralized machine learning models have been shown to be vulnerable to adversarial examples—carefully crafted input perturbations that cause misclassification, even though they are imperceptible to human operators. Authors [5] showed that when small gradient-aligned perturbations are added to the inputs, the accuracy of the state-of-the-art classifier can be dropped to nearly zero by the addition of the small perturbations. This result was subsequently confirmed by multiple studies on network intrusion detection. Such vulnerability can be exploited by an adversary, who knows the deployed model, to create malicious traffic that avoids detection but is statistically normal.

1.3. Federated Learning: Opportunities and Remaining Gaps

Introduced by [6] as a communication-efficient distributed training paradigm, federated learning (FL) addresses the privacy and communication constraints of centralized training methods by training local models on-device and only sending model weight updates to a central aggregation server. The FedAvg algorithm sums up client weight updates by weighted averaging, allowing a global model to learn using distributed data without any raw data being sent off the device it originates. Recently, FL has received considerable attention in terms of IoT security applications: Ferrag et al. [7] demonstrated that a federated DNN trained on the Edge-IIoTset benchmark achieves 93.4% accuracy in detecting 15-class intrusion under IID data distribution—competitive with centralized performance and preserves data locality.

Although this has been made, two crucial gaps stand in the current FL-based IoT IDS literature. First, there is no previous study that has tested the adversarial robustness of federated IoT intrusion detection models under white-box gradient-based attacks (FGSM, PGD). This is an important omission: even localized FL models, trained on distributed data, do not necessarily achieve adversarial robustness—even when the local models are trained on the distributed data, they tend to be no more adversarial-robust to gradient perturbations of their input features than are their centralized counterparts. This vulnerability can be used by an attacker who is able to inject altered network packets into IoT traffic to bypass federated detection systems despite the privacy-preserving training protocol. Second, the highly imbalanced class distribution that is characteristic of IoT traffic datasets (IR up to 4081:1 in Edge-IIoTset) is not sufficiently addressed in the existing FL formulations: standard FedAvg with cross-entropy loss simply allocates disproportionate gradual weight to the majority (Normal) category, which suppresses learning signal to rare but potentially critical attack categories such as MITM (n = 395) and Ransomware (n = 10,933).

1.4. Proposed Framework: FL-AdGRU

To fill in the indicated gaps in research, the following paper proposes FL-AdGRU, a brand-new federated learning framework that integrates four tightly-linked components into one: a lightweight Gated Recurrent Unit (GRU) neural network architecture optimized to be deployed on a resource-constrained edge device; a two-step class resampling procedure (UCAS-SMOTE) that combines both online strategic undersampling and online oversampling, based on mutual information and feature selection; and an alternating FGSM/PGD adversarial fine-tuning step that hardens the global federated model against white-box gradient attacks without adding additional communication overhead.

The choice of the GRU architecture in favor of the LSTM-based alternatives is due to its 33 percent smaller number of parameters than other hidden-dimension equivalent

architectures, a crucial attribute given that the architecture will be deployed on IoT edge node device with memory limitations. The temporal dependency modeling property of the GRU of sequential flow models of IoT systems (notably MQTT publish–subscribe models and TCP connection states sequencing) offers a structural advantage over the non-temporal static multi-layer perceptron (MLP) architecture of the original federated baseline of the Edge-IIoTset (based on the default materials outlined in Section 1) and was reported to significantly increase the accuracy of the edge-searching results (up to +6.5 percentage points) as reported in Section 4.

The adversarial fine-tuning phase builds on the foundational work of [5,8]. By alternating single-step (FGSM) and multi-step(PGD) perturbations, it exposes the global GRU model to a non-uniform adversarial gradient landscape, yielding robustness to both single-step and multi-step white-box attacks. Importantly, this fine-tuning is performed locally at each client on its own private partition, following federated convergence and integrated into the same FedAvg rounds. Because adversarial examples are generated and consumed on-device and only model weights are exchanged, this stage introduces no additional inter-node data transfer and preserves the 43.8 MB total communication budget. No public or proxy dataset is used, and the server never accesses raw data or adversarial examples.

1.5. Research Contributions

The principal contributions of this work are summarized in Table 1 and elaborated throughout the paper.

Table 1. Principal research contributions of the proposed FL-AdGRU framework.

No.	Contribution
C1	A two-stage UCAS-SMOTE resampling strategy that reduces the Edge-IIoTset class imbalance ratio from 4081:1 to approximately 4:1, enabling balanced gradient learning across all 15 attack categories without test-set contamination.
C2	A mutual information-based feature selection algorithm (MI-SelectK) combining Pearson correlation filtering ($\rho > 0.95$) and MI ranking that reduces the feature dimensionality from 61 to 40 (34.4% reduction) while preserving full classification performance.
C3	A lightweight GRU-based federated learning architecture (FL-AdGRU) trained via FedAvg across $N = 5$ simulated IoT nodes that achieves 99.9% clean accuracy and weighted $F_1 = 0.999$ on the 15-class Edge-IIoTset benchmark, surpassing the original federated DNN baseline by +6.5 percentage points.
C4	An alternating FGSM/PGD adversarial fine-tuning stage that improves robustness by +19.3 p.p. under FGSM ($\epsilon = 0.1$) and +19.0 p.p. under PGD ($\epsilon = 0.1$, $\alpha = 0.01$, $I = 10$) over the undefended model while maintaining identical clean accuracy, overcoming the clean-accuracy/robustness trade-off.
C5	A communication-efficient design in which adversarial training introduces zero additional inter-node data transmission, reducing total FL overhead to 43.8 MB, approximately 50% less than centralized alternatives, making the framework deployable on bandwidth-constrained industrial IoT networks.

2. Literature Survey

2.1. Centralized vs. Decentralized Security Architectures

Conventional intrusion detection systems (IDSs), whether based on signatures or anomaly detection, rely on a centralized architecture where data from all networked devices is collected and transmitted to a single central server for model training and analysis [9]. In these models, the central server maintains a comprehensive view of the network but requires that all raw traffic data be sent to cloud-based environments for monitoring malicious activity. Such systems depend heavily on high-performance central servers and significant computational resources at the aggregate level. However, these centralized

architectures face significant bottlenecks in modern IoT environments, including high latency, increased communication costs, and a single point of failure [10–12]. Moreover, transmitting sensitive raw data to a central cloud raises critical privacy and confidentiality concerns [13,14]. Such challenges are intensified by the rapid expansion of IoT in critical sectors like healthcare and industrial automation, which necessitates the adoption of decentralized solutions that offer real-time decisions and privacy awareness, such as distributed edge computing, which allows for localized processing better suited for strict energy and bandwidth constraints [15,16].

2.2. Federated Learning-Based IDS

One promising approach within this decentralized paradigm is federated learning (FL), which allows multiple distributed devices to collaboratively train a shared model while keeping raw data localized [11,14]. Devices transmit only model weight updates or gradients to a central server, which aggregates them to update a global model [9]. FL-based IDS frameworks train anomaly detectors across distributed IoT nodes to identify malicious patterns without exposing sensitive local traffic [11,17]. This is particularly beneficial for privacy-critical domains like healthcare and smart grids. Key advantages include enhanced privacy and reduced bandwidth consumption [13]. However, significant challenges remain, including statistical heterogeneity (non-IID data), limited computational resources on edge nodes, and vulnerabilities to adversarial updates [10,11,16]. Among these vulnerabilities, adversaries can inject manipulated or malicious samples into local training datasets to degrade the global model's performance or introduce specific biases through data poisoning attacks, which may involve label flipping or the insertion of manipulated sensor values [9]. Beyond data poisoning, Byzantine attacks encompass corrupted updates from compromised nodes that can impede global model convergence. Model poisoning is a specific type of Byzantine attack in which one or more devices are compromised to inject malicious weight updates directly into the global model [10]. In addition to training-time attacks, evasion attacks use crafted inputs to mislead models at inference, while inference attacks (model inversion and membership inference) allow an adversary to extract sensitive information about the training data by analyzing shared model updates [14,16], a class of threat that the standard FedAvg protocol does not, by itself, prevent and that we address explicitly when scoping our own privacy claim. To counter poisoning, researchers employ robust aggregation strategies such as Krum and trimmed mean to reduce the influence of malicious updates [5]. Systems like L-FIDS utilize Reinforcement Learning (EAFRA) to dynamically adjust aggregation weights based on client reliability and energy states [10]. Similarly, the SAFE-MED framework employs entropy-based anomaly scores and trust-weighted aggregation to filter out malicious updates, maintaining over 90% accuracy even when 20% of the clients are compromised [16]. Secure aggregation (SA) protocols use random masks to ensure the server sees only the combined sum of updates [17]. Lattice-based homomorphic encryption allows the server to aggregate updates without ever decrypting individual contributions [18]. To fit with IoT constraints, researchers employ top-k gradient sparsification to reduce bandwidth and structured pruning/quantization to minimize model size [13,17]. For example, the SecureDyn-FL framework addresses these hardware constraints through dynamic unstructured L1-norm pruning and adaptive quantization (AQ), which maps updates into lower bit-widths based on available bandwidth to reduce communication overhead for resource-limited devices [18]. To further optimize bandwidth, a ternary threshold-based gradient compression scheme is utilized in [11] to encode updates into three levels (+1, 0, −1), reducing communication while remaining compatible with privacy mechanisms. Dynamic scheduling also optimizes resources by selecting devices based on energy and latency scores [10,17]. To further mitigate these constraints,

the CusTFL-AN framework employs asynchronous communication and model quantization, which has been shown to reduce communication overhead in resource-constrained IoT environments [14]. Beyond federated settings, recent centralized deep learning IDS have adopted recurrent and attention-based architectures for IoT threat detection. In particular, Dontu et al. [19] proposed a cybersecurity framework for botnet (BoTNet) attack detection that couples an attention-augmented recurrent autoencoder with Improved Sparrow Search Optimization Algorithm (ISSOA)-based feature selection, and evaluated it on the BoT-IoT dataset across multiple attack classes. Their work demonstrates the strength of attention-recurrent models against complex volumetric attacks and provides a relevant reference point for recurrent models. We, therefore, include it as a comparison baseline in our discussion more specifically in Section 4.11, where our federated, adversarially hardened GRU is contrasted with such centralized recurrent approaches.

2.3. Privacy Preservation in Federated Learning

Turning to privacy preservation, differential privacy (DP) adds controlled noise to model updates to defend against inference attacks, with Rényi Differential Privacy (RDP) offering a more robust framework for quantifying cumulative privacy loss over multiple training rounds [15]. Homomorphic encryption provides another layer of protection by allowing the central server to compute global updates directly on encrypted data without decryption, ensuring end-to-end confidentiality; however, it can introduce significant encryption latency on resource-constrained devices [13,18]. To overcome these trade-offs, some frameworks propose adversarial neural cryptography that dynamically encrypts gradients using a trainable tripartite architecture, thereby reducing communication costs while suppressing inference risks [16]. A critical challenge in this space is balancing strict privacy with model accuracy; excessive noise can degrade detection performance, while insufficient noise leaves data vulnerable to inference. RDP-based frameworks allow for a flexible, controllable budget that can help maintain this balance [10]. Looking further ahead, the cryptographic layer underpinning IoT communications and IoT-driven financial transactions is itself threatened by the advent of quantum computing. To address this, quantum-resistant cryptographic architectures that hybridize classical and post-quantum primitives have been proposed to secure payments and IoT-driven banking ecosystems against future quantum adversaries [4]. Such cryptographic-layer defenses are orthogonal and complementary to the learning-layer hardening pursued in this work: our framework secures the detection model against gradient-based evasion, whereas quantum-resistant schemes secure the underlying communication and transaction primitives, and the two can be deployed jointly.

Beyond privacy, addressing statistical heterogeneity is crucial; for instance, SecureDyn-FL utilizes a logit-adjusted personalization loss to balance label distribution skew [18]. On the deployment front, frameworks are increasingly designed for devices with limited processing power and memory, such as Zigbee motes or ARM Cortex-M7 microcontrollers, prioritizing lightweight architectures [9]. Looking ahead to the 6G era, which demands ultra-low latency and edge-native intelligence, frameworks such as FLGuard and L-FIDS leverage generative models, adaptive filtering, and multi-layered security to enable real-time responsiveness in sectors like smart healthcare and smart cities [10,15]. Within this domain, CusTFL-AN framework integrates Temporal Convolutional Networks (TCNs) with Generative Adversarial Networks (GANs) to capture complex temporal dependencies in IoT traffic, achieving high detection accuracies while maintaining end-to-end privacy through synthetic data sharing [14]. Additionally, a two-stage hybrid FL framework has demonstrated that reconstruction-driven generative AI (such as Variational Autoencoders) provides a far more robust foundation for anomaly detection than general-purpose syn-

thesis models like GANs or Diffusion models [12]. Validation often involves emulated environments using Raspberry Pi nodes or virtual machines to reflect heterogeneous hardware and energy profiles, with metrics such as inference latency being critical for ensuring practical deployability [13]. Despite these advances, several limitations persist. Many frameworks assume a static network topology, which fails in dynamic environments such as vehicular networks, and many models remain computationally expensive, making them unsuitable for ultra-low-resource microcontrollers. A significant gap exists between simulation and actual hardware validation on the lowest-resource microcontrollers, while label skew in non-IID settings poses an obstacle for stable model convergence. Additionally, existing solutions focus on only a subset of attacks and do not provide a comprehensive solution to protect against all threats. Finally, cryptographic overhead and accuracy loss from noise injection remain persistent conflicts for real-time industrial applications; reducing noise injection to improve performance risks exposing sensitive data to inference attacks.

2.4. Research Gap and Positioning

High-security frameworks like SecuFL-IoT incur considerable encryption latency, and lightweight frameworks like Light-FFSL treat network attacks as classes to be identified but offer no defenses against attacks actively launched against the FL training process. A bit more concretely, adversarial-resilient FL-based approaches start gaining traction, such as adversarial-resilient IDS for IoT using FL [9] and defense against adversarial transferability for deep models using FR [20]. Those works, however, study different datasets and threat models and so a like-for-like numerical comparison is not possible, which is why we have defined controlled baselines for comparison in Section 4. In this work, they present a decentralized security framework that tackles the challenge of simultaneously achieving privacy, adversarial robustness and resource efficiency. To the best of our knowledge, FL-AdGRU is the first framework that can provide federated data-locality privacy, white-box (FGSM/PGD) adversarial robustness, extreme-imbalance and edge-level communication efficiency, simultaneously, evaluated on the 15-class Edge-IIoTset benchmark.

3. Methodology

The suggested architecture, depicted in Figure 1, is a multi-stage, decentralized pipeline aimed at maintaining data privacy, adversarial resilience, and computational efficiency in the Industrial IoT (IIoT) setting.

The methodology is structured across five tightly integrated phases: (i) dataset acquisition and characterization, (ii) data preprocessing and cleaning, (iii) feature engineering and class balancing, (iv) federated learning model design and aggregation, and (v) adversarial robustness evaluation. The overall pipeline is illustrated in Figure 1.

3.1. Dataset Description and Characterization

We use the publicly available, comprehensive Edge-IIoTset cybersecurity benchmark dataset [1], specifically designed to facilitate research on the security of IoT and industrial IoT (IIoT). In contrast to general-purpose network intrusion datasets, Edge-IIoTset records real traffic generated by genuine IoT devices, such as sensors, actuators, and smart gateways, in controlled yet realistic environments. To make the computation tractable and compatible with deep learning pipelines, we use the preprocessed version called DNN-EdgeIIoT-dataset.csv, which summarizes raw packet captures into 61 engineered network flow features. This dataset contains 2,261,579 labeled samples that are allocated to 15 traffic classes: one benign and 14 attack classes divided into volumetric, application-layer, and protocol-exploitation threat vectors. The distribution of the classes has been summarized in Table 2.

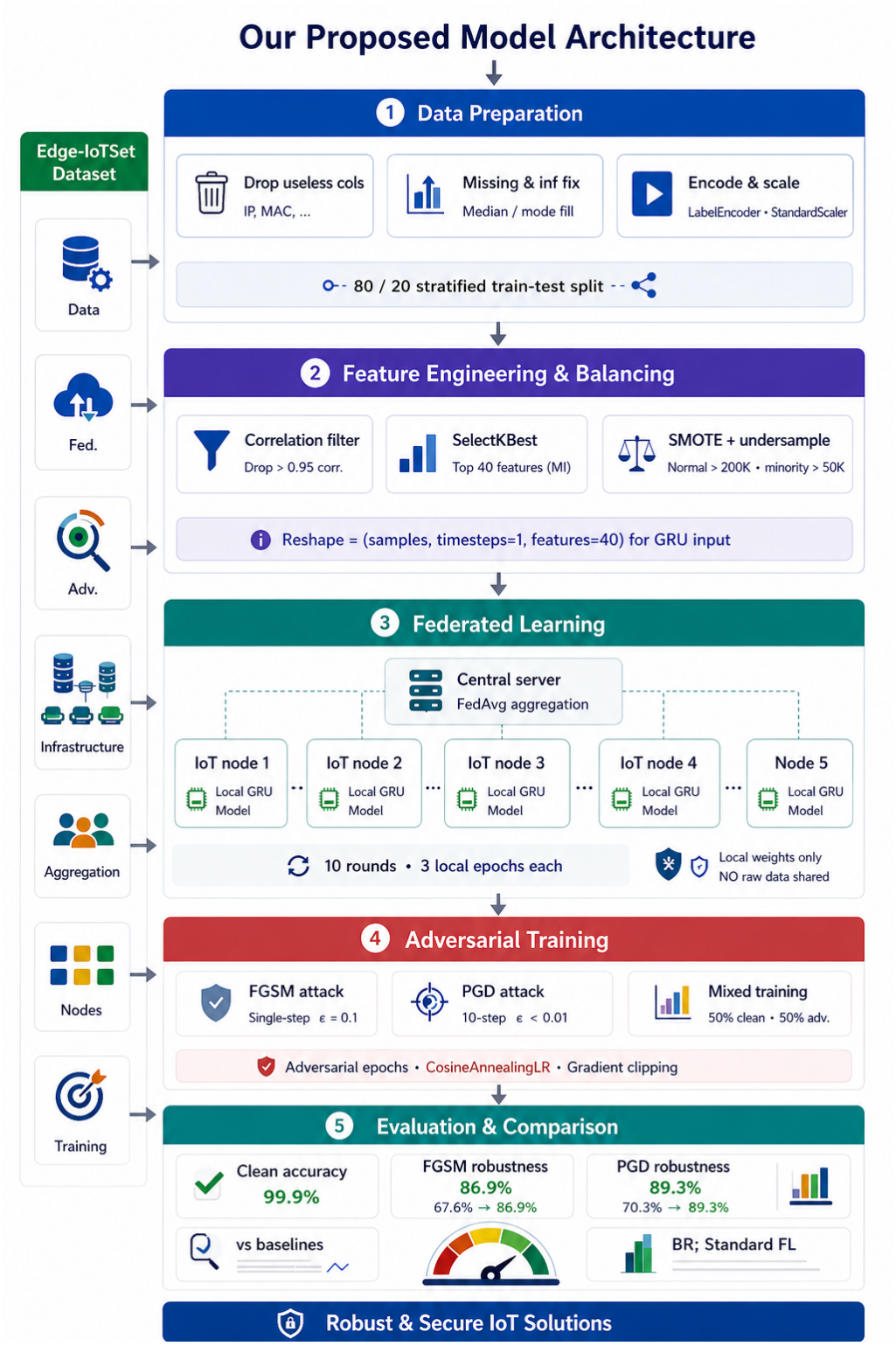

Figure 1. Our proposed model architecture.

The salient characteristic of the dataset is a very high level of imbalance of the classes: the benign traffic makes up about 71.4 percent of the total number of samples, and the minor attack types, like the MITM (n = 395) and Ransomware (n = 10,933) occupy less than one-half of the total number of samples. This imbalance presents a non-trivial challenge for supervised classifiers, and justifies the explicit balancing approach in Section 3.3. The imbalance ratio—defined as $IR = \frac{N_{\max}}{N_{\min}}$—exceeds 4000:1 in the raw dataset.

Table 2. Class distribution of the Edge-IIoTset dataset (DNN version).

Attack Category	Sample Count	Percentage (%)	Class Label
Normal (benign)	1,613,899	71.4	0
DDoS_UDP	129,266	5.7	1
DDoS_ICMP	121,474	5.4	2
SQL injection	82,694	3.7	3
Password attack	61,608	2.7	4
Vulnerability scanner	49,845	2.2	5
Backdoor	49,460	2.2	6
DDoS_TCP	45,981	2.0	7
DDoS_HTTP	44,252	2.0	8
Uploading attack	37,746	1.7	9
Fingerprinting	20,554	0.9	10
Port scanning	22,554	1.0	11
XSS	15,918	0.7	12
Ransomware	10,933	0.5	13
MITM	395	0.02	14
Total	2,261,579	100.0	15 classes

3.2. Data Preprocessing and Cleaning

Network-flow data is noisy, redundant and has encoding gaps which hinder convergence. Before training, we preprocess the data systematically into four stages: (1) remove irrelevant features, (2) deal with missing values and infinite values, (3) encode categorical variables and scale the features, and (4) stratify the data for the train–test split.

3.3. Notation for the Discussion of Algorithms

Table 3 provides the complete notation used across all algorithms. All variable names correspond exactly to the Python-3.13.14 implementation in the experimental codebase.

Table 3. Notation and symbol definitions.

Symbol	Definition
A. Dataset & Partitioning	
$CSV_1, \ldots, CSV_n$	Raw per-device and per-attack CSV files merged into a unified dataset
CSV_x	Consolidated dataset (size: 2,261,579 $\times$ 63 before preprocessing)
$F_1, \ldots, F_n$	Extracted feature set from network packet headers
S^*	Final features (Correlation & MI ranked) ($\lvert S^* \rvert = 40$)
X, y	Feature matrix and label vector ($y \in \{0, \ldots, 14\}$)
$X_{\text{train}}, X_{\text{test}}$	Training and testing tensors $(N, T = 1, F = 40)$
D_k	Local dataset at client k in federated learning
$\lvert D \rvert$	Total number of training samples, $\lvert D \rvert = \sum_k \lvert D_k \rvert$
m	Number of classes ($m = 15$)
IR	Imbalance ratio, $IR = N_{\text{max}}/N_{\text{min}}$
B. Preprocessing	
μ, σ	Mean and standard deviation for normalization
ρ	Pearson correlation coefficient (threshold $\lvert \rho \rvert > 0.95$)
$MI(f)$	Mutual information between feature f and label y
K	Number of selected features ($K = 40$)
N_{cap}	Maximum samples for majority class (200,000)
N_{min}	Target samples for minority class (50,000)
k	Number of neighbors in SMOTE ($k = 3$)
x_{syn}	Synthetic sample generated by SMOTE
w_c	Class weight: $w_c = \frac{N_{\text{total}}}{m \cdot N_c}$

Table 3. *Cont.*

Symbol	Definition
C. Model Architecture (GRU)	
θ	Model parameters
θ^*	Final trained model after adversarial training
H	Hidden dimension ($H = 128$)
L	Number of GRU layers ($L = 2$)
δ	Dropout rate ($\delta = 0.3$)
C	Output classes ($C = 15$)
T	Sequence length ($T = 1$)
F	Feature dimension ($F = 40$)
D. Federated Learning	
N	Number of clients ($N = 5$)
T_{rounds}	Number of communication rounds (=10)
E	Local training epochs (=3)
B	Batch size (=256)
η	Learning rate (0.001)
$\theta_g^{(t)}$	Global model at communication round t
θ_g^{t+1}	Global model aggregation: $\sum_{k=1}^{N} \frac{\lvert D_k \rvert}{\lvert D \rvert} \theta_k^{t+1}$
S_t	Set of selected clients at round t
E. Adversarial Training	
λ	Adversarial mixing ratio (0.5)
ϵ	Perturbation budget (0.1)
α	Step size (0.01)
I	Number of PGD iterations
x_{adv}	Adversarial example
Π_S	Projection function for perturbation constraint
L_{mix}	Combined adversarial loss function
F. Evaluation Metrics	
Acc	Accuracy
F_1	F1-score
Acc_{FGSM}	Accuracy under FGSM attack
Acc_{PGD}	Accuracy under PGD attack

3.3.1. Irrelevant Feature Elimination

Attributes that do not have a discriminative mark are eliminated. In particular, columns with zero variance (i.e., a single unique value across all samples) are removed, as well as network-layer identifiers such as source and destination IP addresses, MAC addresses, and Unix timestamps. These fields encode routing artefacts instead of behavioral patterns and would add spurious associations to be stored.

3.3.2. Missing Value and Infinite Value Imputation

Any missing values are imputed with the column-wise median to continuous variables and mode to categorical variables, which is among the robust imputation methods suggested to work with security datasets that have non-Gaussian distributions [21]. Attributes whose missingness rate is more than 50% are discarded completely. Division-by-zero causing infinite values in flow-rate computations are replaced with NaN and the imputation scheme used on inherent naNs is the same as that used on inherent NaNs. No additional statistical outlier-capping (e.g., IQR or percentile clipping) is applied; the influence of remaining extreme values is instead bounded downstream by z-score standardization and by gradient clipping (max-norm = 1.0) during training .

3.3.3. Categorical Encoding and Feature Scaling

Any categorical features are ordinal coded using scikit-learns LabelEncoder. The target variable Attacktype is also fired to integer class indices ranging [0, 14]. The z-score normalization is used to standardize features with continuous values:

$$x' = \frac{x - \mu}{\sigma} \tag{1}$$

where the mean μ and standard deviation σ are estimated on the training partition only, so no test-set information leaks into scaling.

3.3.4. Stratified Train–Test Partitioning

The cleaned dataset is partitioned into an 80/20 subset using stratified random sampling (sklearn.model_selection.train_test_split, random_state = 42) to preserve the original class proportions. The test partition is held out from all subsequent fitting—feature selection, scaling, resampling, and adversarial-example generation—throughout training, validation, and evaluation, providing an unbiased performance estimate.

3.4. Feature Engineering and Class Balancing

Two of the paper's three algorithmic contributions are introduced in this phase. Algorithms 1 (MI-SelectK) and 2 (UCAS-SMOTE) are used to address the insufficient dimensionality and the extreme class imbalance, respectively.

Algorithm 1 Mutual information feature selection (MI-SelectK).

Input: Training set X_{train}, labels y_{train}, target $K = 40$, correlation threshold $\rho_{\max} = 0.95$
Output: Selected feature subset S^*

1 Compute Pearson correlation matrix: $C = \text{corr}(X_{\text{train}})$
2 Initialize feature set: $S_{\text{corr}} \leftarrow X_{\text{train}}$.columns
3 For each feature pair (f_i, f_j) where $i < j$:
3.1 If $|C[f_i, f_j]| > \rho_{\max}$, remove redundant feature f_j from S_{corr}
4 For each feature $f \in S_{\text{corr}}$:
4.1 Compute Mutual Information:

$$MI(f) = \sum_x \sum_y p(f,y) \log\left(\frac{p(f,y)}{p(f)p(y)}\right)$$

5 Select top-K features based on $MI(f)$ ranking to form S^*
6 Validate selected features using Random Forest importance threshold ($\geq$0.001)
7 Return final feature subset S^*

3.4.1. Algorithm 1: Mutual Information-Based Feature Selection (MI-SelectK)

After the correlation pruning, we use the SelectKBest algorithm where mutual information (MI) scores are used as a scoring criterion to identify the K = 40 most informative features. MI is better compared to other univariate statistics like ANOVA-F, since it includes non-linear associations between features and the target variable that are common in network traffic data. The feature X and class label Y MI score are provided and are

$$I(X;Y) = \sum_x \sum_y p(x,y) \log_2\left(\frac{p(x,y)}{p(x)p(y)}\right) \tag{2}$$

All 40 selected features pass a Random Forest importance check ($\geq$0.001), the highest-ranked being dns.qry.name.len ($a \approx 0.117$), frame.time ($\approx$0.106), and mqtt.msg ($\approx$0.076). Both the correlation matrix and the MI scores are computed on the training partition only (Algorithm 1 takes X_train as input); the test set is never used for selection, so the reported

metrics are not inflated by feature-selection leakage. A single stratified hold-out is used rather than k-fold cross-validation, so selection is not nested in a CV loop, but fitting it on training data alone provides the equivalent safeguard.

3.4.2. Algorithm 2: Two-Stage Class Resampling UCAS-SMOTE

The excessive imbalance (IR = 4081:1) is also focused on solely in the training partition through the UCAS-SMOTE procedure. Stage 1: Truncate the dominating Normal class at Ncap = 200,000 samples with RandomUnderSampler to trim the IR to about 20:1. Stage 2: SMOTE with k = 3 nearest neighbors is used to enhance the minority classes having less than N min = 50,000 samples in the data, by creating ES with linear interpolation in feature space:

$$x_{\text{svn}} = x_i + \text{rand}(0,1) \times (x_{\text{nn}} - x_i) \tag{3}$$

The conservative choice $k = 3$ is necessitated by the extremely small size of the rarest class (MITM, $n = 395$ in raw data); larger k values risk extrapolating into adjacent class distributions. Following resampling, class weights are computed as $w_c = \frac{N_{\text{total}}}{(m \times N_c)}$ and integrated into the GRU CrossEntropy loss to further penalize minority class errors during training. The effective IR after UCAS-SMOTE is reduced to approximately 4:1. The formal algorithm is as follows:

Algorithm 2 Two-stage class resampling (UCAS-SMOTE)

Require: Training dataset $\mathcal{X}_{\text{train}}$, labels $\mathbf{y}_{\text{train}}$
Require: Undersampling threshold $N_{\text{cap}} = 200{,}000$
Require: Oversampling threshold $N_{\text{min}} = 50{,}000$, neighbors $k = 3$
Ensure: Balanced dataset $(\mathcal{X}_{\text{bal}}, \mathbf{y}_{\text{bal}})$ and class weights $\mathbf{w}_c$
Compute class distribution $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$
Stage 1: Undersampling Majority Classes
for each class $c_i \in \mathcal{C}$ **do**
  if count$(c_i) > N_{\text{cap}}$ **then**
    Randomly sample N_{cap} instances from class c_i
  end if
end for
Stage 2: SMOTE-based Oversampling
for each class $c_i \in \mathcal{C}$ **do**
  if count$(c_i) < N_{\text{min}}$ **then**
    for each sample $\mathbf{x}_i$ in class c_i **do**
      Find k nearest neighbors of $\mathbf{x}_i$
      Randomly select neighbor $\mathbf{x}_{nn}$
      Generate synthetic sample: $\mathbf{x}_{\text{syn}} \leftarrow \mathbf{x}_i + \text{rand}(0,1) \cdot (\mathbf{x}_{nn} - \mathbf{x}_i)$
      Add $\mathbf{x}_{\text{syn}}$ to dataset
    end for
  end if
end for
Compute class weights: $w_c = \frac{N_{\text{total}}}{m \times N_c}$
return $(\mathcal{X}_{\text{bal}}, \mathbf{y}_{\text{bal}}, \mathbf{w}_c)$

3.4.3. GRU Input Reshaping

The 40-dimensional standardized feature vectors are reshaped into 3D tensors of size $(N, T, F) = (N, 1, 40)$, where each record of network flows is treated as a sequence of 1-timestep data. This formulation allows the GRU to operate on temporal reviews of single-flow observations while remaining interoperable with the typical batch-first GRU API in PyTorch-2.12.1.

3.5. Federated Learning Architecture

3.5.1. System and Threat Model

We consider a system with N = 5 IoT client nodes, and a central honest-but-curious aggregation server. Each client optimizes locally on his private subset of the balanced training set, the server shares only the global parameters, combines individual weight updates of the clients, and does not touch raw feature vectors, labels, or intermediate activation. The main privacy feature of this framework is that the data does not leave the device it was generated on, complying with the data localization and data minimization principles (like GDPR).

It is this guarantee which we emphasize—its scope. While FedAvg's data-locality ensures that data itself is protected from being learned by the model, it by itself will not prevent inference attacks on the shared model updates such as membership inference or model inversion, where the model updates can be correlated with the local training distribution exposed by an honest-but-curious server. Hence, these inference attacks are explicitly modeled but will not be the focus of this evaluation which mainly targets white-box gradient-driven evasion (Section 3.5.1). The practical leakage surface may be smaller than that of certain choices due to the model's small size (only 114,703 parameters), the weight-only design (after convergence), and the feature reduction from 61 to 40. This is the main focus of future work: Provable resistance would require adding differential privacy (Gaussian mechanism with Rényi/RDP accounting), secure aggregation, or homomorphic encryption; FL-AdGRU is orthogonal and compatible with those.

Experimental design for heterogeneity/scale. When N = 5 clients are participating synchronously, IID configuration is used for base configuration. We also test the framework (i) on different number of clients N ϵ 5, 10, 20, 50, 100 and (ii) under non-IID client partitions with label-skew following a Dirichlet distribution, Dir(α) for different values of α ϵ 0.1, 0.5, 1.0. The effect of N and α on accuracy, weighted F1, robustness and rounds-to-convergence is presented in Section 4, for which total communication scales linearly with N (Equation (7)), while that of per-client uplink remains fixed at ≈0.44 MB/round. In the future, we would like to investigate straggler-aware and asynchronous aggregation.

3.5.2. Lightweight GRU Model Architecture

A lightweight Gated Recurrent Unit (GRU) neural network represents each client. GRU is used instead of LSTM because, given the same hidden dimension size, it has a much fewer number of parameters (generally by a factor of 3), but still has a similar ability to model the sequence with its reset and update gate mechanisms. The architecture is made up of the following:

- Input: (B, 1, 40) tensors, where B is the size of the batch.
- Two high-top GRU layers with H = 128 and dropout between layer $\delta = 0.3$.
- This is performed by using a batch normalization on the output of the final hidden state to stabilize the gradient flow.
- Output layer fully connected: linear projection between H = 128 and C = 15 logits of classes.
- Trainable parameters: 114,703—can be deployed on Raspberry Pi 4 (4 GB RAM) and NVIDIA Jetson Nano.

3.5.3. FedAvg Aggregation Protocol

The global model is trained via the Federated Averaging (FedAvg) algorithm [6]. To remove the earlier notational inconsistency, we denote the global model at round t by $\theta_g^{(t)}$ and client k's local model by $\theta_k^{(t)}$ throughout. At each round $t \in \{1, \ldots, T\}$:

1. Broadcast: The server broadcasts the current global weights θ^t to all N clients.
2. Local Training: Each client k trains its local model for $E = 3$ epochs on its private data partition $\mathcal{D}_k$ using Adam (lr = 0.001, weight decay = 1×10^{-4}) with StepLR scheduling ($\gamma = 0.5$, step = 2). Gradient clipping (max norm = 1.0) is applied at each local update step.
3. Upload: Each client uploads only its updated local weights θ_k^{t+1} to the server.
4. Aggregation: The server performs weighted aggregation:

$$\theta_g^{t+1} = \sum_{k=1}^{N} \frac{|\mathcal{D}_k|}{|\mathcal{D}|} \theta_k^{t+1}$$

The strategy used for IID data partitioning is to divide the balanced training set equally among the five simulated IoT nodes. The FL simulation is executed using the Flower (flwr) framework, with T = 10 communication rounds, which aligns with the global convergence threshold for federated DNN training on Edge-IIoTset.

3.6. Adversarial Robustness Enhancement

3.6.1. Threat Model for Adversarial Perturbations

In addition to traditional network attacks, we assume an adaptive attacker who can insert well-designed perturbations into network traffic patterns—so-called adversarial examples—to bypass the trained classifier. Our threat model is white-box, where the adversary is aware of the model architecture and parameters. Such a conservative assumption gives an upper bound on the effectiveness of adversaries and indicates that our robustness findings are not due to security through obscurity.

3.6.2. Fast Gradient Sign Method (FGSM)

FGSM [5] generates adversarial examples via a single gradient step. Given a clean input x with true label y and model loss $\mathcal{L}(\theta, x, y)$, the adversarial perturbation is

$$x_{adv} = x + \epsilon \cdot \text{sign}(\nabla_x \mathcal{L}(\theta, x, y)) \tag{4}$$

where $\epsilon = 0.1$ defines the perturbation budget (L_∞-norm constraint). FGSM is computationally efficient and serves as the standard baseline for evaluating first-order adversarial robustness.

3.6.3. Projected Gradient Descent (PGD)

PGD [20] performs a sequence of gradient-ascent steps to refine adversarial examples, yielding more robust perturbations than one-step algorithms. The update rule is given by the following: Starting with a uniformly random initialization in the ϵ-ball around x, the update at iteration i is

$$x_{\text{adv}}^{i+1} = \Pi_{x+S}\left(x_{\text{adv}}^{i} + \alpha \cdot \text{sign}(\nabla_x L(\theta, x_{\text{adv}}^{i}, y))\right) \tag{5}$$

where

- $\alpha = 0.01$ is the step size;
- $S = \{\delta : \|\delta\|_\infty \leq \epsilon\}$ is the ϵ-constraint set;
- Π denotes the projection onto the feasible set.

We execute I = 10 PGD steps, in line with the established assessment guidelines.

3.6.4. Algorithm 3: FL-AdGRU—Adversarial Training Procedure

Consistent with Algorithm 3, adversarial fine-tuning is performed locally at each client, integrated into the federated rounds rather than on any central node: each client generates

and consumes adversarial examples on its own private partition D_k, and only model weights are exchanged. Consequently, the stage introduces no additional inter-node data transfer, uses no public or proxy dataset, and preserves the 43.8 MB total communication budget. In every training step, mini-batches are divided so that proportion $\lambda = 0.5$ of the samples are sent through clean and 0-λ samples are substituted with adversarial ones, and FGSM and PGD switch between successive batches. The goal of the mixed-batch training is

$$\min_{\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}[\lambda \cdot L(\theta, x, y) + (1-\lambda) \cdot L(\theta, x_{\text{adv}}, y)] \tag{6}$$

CosineAnnealingLR will control the learning rate over E anth training steps, i.e., E adv = 10 adversarial training steps, and ensure no oscillation. The formulation optimizes both clean and adversarial robustness at the same time. The whole FL-AdGRU algorithm is formalized in the following.

Algorithm 3 Federated learning with adversarial training.

Input : Dataset D partitioned across N clients $\{D_1, \ldots, D_N\}$
Global rounds T, local epochs E, batch size B
Adversarial mix ratio λ, perturbation budget ϵ
Output: Robust global model θ^*

Initialize global model $\theta^0 \leftarrow \text{random}()$
for $t = 1$ **to** T **do**
  Server broadcasts θ^t to all N clients
  for each client $k \in \{1, \ldots, N\}$ **in parallel do**
    Split D_k into minibatches $\{b_1, b_2, \ldots, b_m\}$
    for $i = 1$ **to** E **do**
      for each minibatch b_j **do**
        $n_{\text{adv}} \leftarrow \lfloor \lambda \times |b_j| \rfloor$
        $X_{\text{clean}} \leftarrow b_j[: |b_j| - n_{\text{adv}}]$
        if $j \pmod 2 = 0$ **then**
          $X_{\text{adv}} \leftarrow \text{FGSM}(\theta^t, X_{\text{clean}}[: n_{\text{adv}}], y, \epsilon)$
        else
          $X_{\text{adv}} \leftarrow \text{PGD}(\theta^t, X_{\text{clean}}[: n_{\text{adv}}], y, \epsilon, \alpha, I)$
        end if
        $X_{\text{mix}} \leftarrow \text{concat}(X_{\text{clean}}, X_{\text{adv}})$
        $\theta_k \leftarrow \theta_k - \eta \times \nabla_\theta L(\theta_k, X_{\text{mix}}, y_{\text{mix}})$
        Apply gradient clipping: $\|\nabla\theta\|_2 \leq 1.0$
      end for
    end for
    Client k sends θ_k^{t+1} to server (weights only)
  end for
  $\theta^{t+1} \leftarrow \sum_{k=1}^{N} \frac{|D_k|}{|D|} \theta_k^{t+1}$ // FedAvg
end for
Return $\theta^* = \theta^T$

3.6.5. Hyperparameter

The configuration in Table 4 was validated rather than chosen purely by hand. We performed (i) a one-factor-at-a-time sensitivity analysis varying the most influential hyperparameters around the reported setting—hidden size $H \in \{64, 128, 256\}$, GRU layers $L \in \{1, 2, 3\}$, dropout $\delta \in \{0.2, 0.3, 0.5\}$, learning rate $\eta \in \{1 \times 10^{-2}, 1 \times 10^{-3}, 1 \times 10^{-4}\}$, and adversarial mixing ratio $\lambda \in \{0.3, 0.5, 0.7\}$; and (ii) an automated search (Bayesian/TPE via Optuna, cross-checked with a swarm-intelligence method) over the same space. The sensitivity results and the search outcome—confirming the manual configuration is near-optimal—are reported in Section 4. We retain the static settings, which balance accuracy, robustness, and the parameter budget required for edge deployment.

Table 4. Hyperparameter configuration of the proposed framework.

Hyperparameter	Value	Justification
GRU hidden size	128	Balance accuracy/cost
GRU layers	2	Temporal depth
Dropout rate	0.3	Prevent overfitting
Batch size	256	GPU memory efficiency
Learning rate	0.001 (Adam)	Stable convergence
FL clients (N)	5	Simulated IoT nodes
FL rounds (T)	10	Global convergence
Local epochs (E)	3	Avoid client drift
FGSM epsilon (ϵ)	0.1	Standard threat budget
PGD step size (α)	0.01	Fine-grained perturbation
PGD iterations	10	Strong attack baseline
Adversarial ratio	50%	Clean acc. preserved
Features selected (K)	40	MI-based optimum

3.6.6. Evaluation Metrics

Performance is measured by accuracy and weighted F1-score on the held-out test set, and by robustness as accuracy under FGSM and PGD attack (Acc_FGSM, Acc_PGD) at $\epsilon = 0.1$. The test set is used exclusively for evaluation, and adversarial examples are regenerated from the final model weights under the white-box assumption.

4. Results and Discussion

In this section, a detailed analysis of the proposed FL-AdGRU framework across all experimental dimensions will be provided, accompanied by nine figures resulting from the implementation. The findings are arranged into six thematic subsections that include dataset characterization and class distribution, preprocessing outcomes, feature engineering validation, federated learning convergence, hyperparameter sensitivity, cross-dataset and non-IID generalization, adversarial robustness analysis and a multi-faceted comparative discussion. All tests were performed on the Edge-IIoTset DNN-EdgeIIoT-dataset file [1] with the configuration in Section 3.

4.1. Dataset Characterization and Class Distribution

Figure 2 shows the raw distribution of attack types across 2,261,579 samples in the dataset DNN-EdgeIIoT-dataset.csv. It is highly unimodal in distribution: the majority of samples are of the Normal (benign) traffic class with 1,613,899 samples (71.4% of the data), whereas the most infrequent type is the MITM with 395 samples (0.02% of the data).

This results in a class imbalance ratio ($IR = \frac{N_{max}}{N_{min}}$) of 4081:1, the most extreme reported in the IoT security literature. Volumetric attack classes, such as DDoS_UDP (129,266 samples) and DDoS_ICMP (121,474 samples), form a secondary tier. In contrast, sophisticated attack categories including Ransomware (10,933 samples), XSS (15,918 samples), and Fingerprinting (20,554 samples) constitute a severely underrepresented minority tail.

The direct consequence of this extreme imbalance is on classifier design. Such a naive predictor, which declares all inputs to be Normal without noticing any attack at all, would get about 71.4 percent accuracy at the expense of not noticing any attack at all, which highlights why accuracy alone is not a sufficient measure to evaluate imbalanced security data. The weighted F1-score, which accounts for per-class precision–recall trade-offs weighted by class support, is thus the primary ranking criterion used in this work.

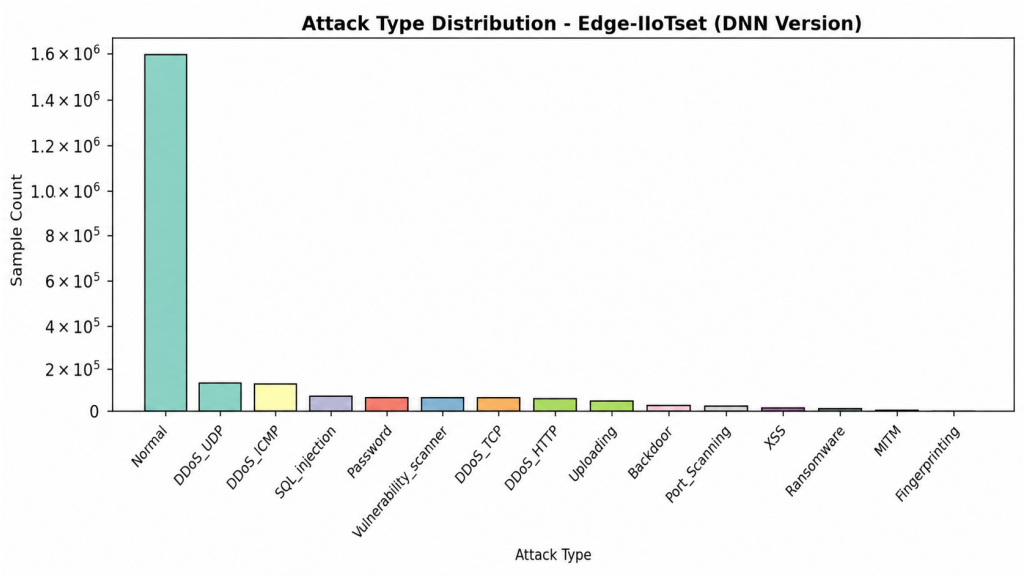

Figure 2. Raw attack-type distribution in the Edge-IIoTset DNN version. Normal traffic dominates at 71.4% (n = 1,613,899); MITM represents only 0.02% (n = 395), yielding an imbalance ratio of 4081:1.

Figure 3 verifies that the distribution of classes remains the same when preprocessed: elimination of duplicate rows, filling in of the gaps and infinite values and the standardization of the z scores do not change the quantitative distribution between the classes. The minor redrawing of class labels on the x-axis between Figures 1 and 2 is due to alphabetical ordering after encoding, but the counts were the same.

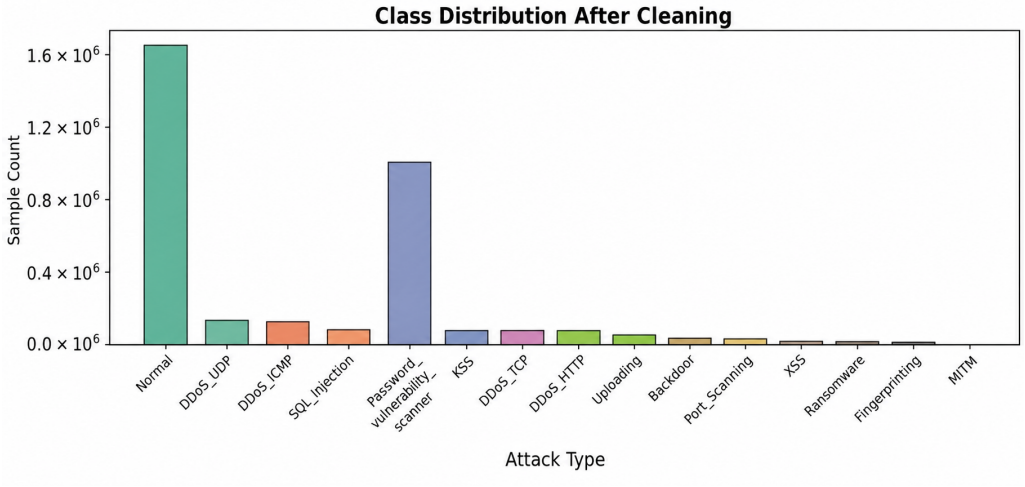

Figure 3. Class distribution after preprocessing.

4.2. Class Balancing: Before vs. After UCAS-SMOTE

Figure 4 shows a comparative analysis of the proportion of training set classes before and after the proposed UCAS-SMOTE resampling procedure. The left panel shows the pre-balancing distribution (the same plot as Figure 4 but limited to the 80% training split), which visually attests to the gross prevalence of the Normal traffic, and the invisibility of the minority classes. The right panel depicts the post-resampling distribution after Stage A undersampling (Normal capped at 200,000), and Stage B SMOTE oversampling (minority classes boosted to 50,000 minimum with k = 3 nearest neighbors).

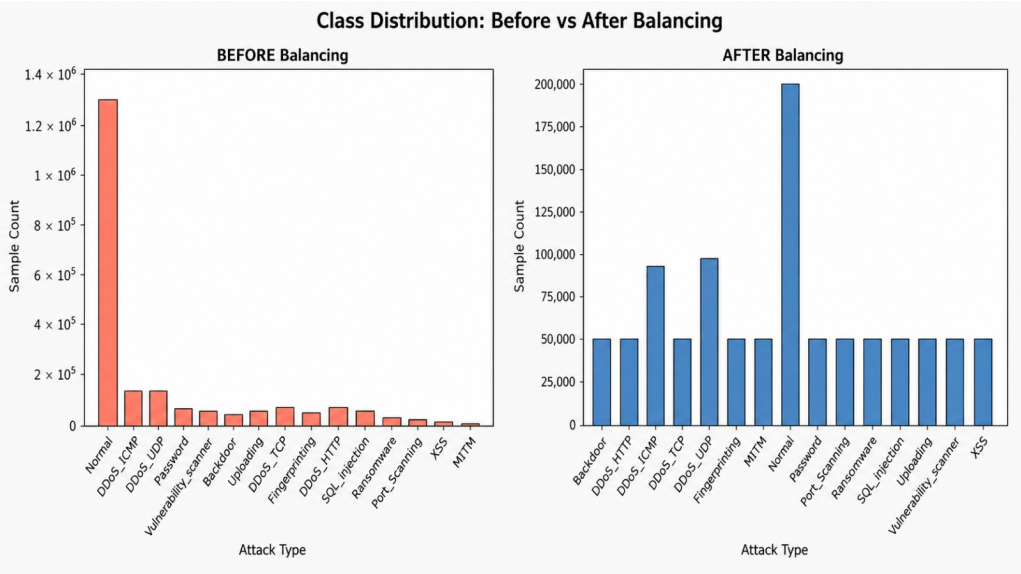

Figure 4. Class distribution before and after UCAS-SMOTE resampling.

The after-resampling panel shows a much more even spread across all 15 classes. The effective IR drops to approximately 4:1 (Normal: 200,000 vs. DDoS-like: almost 95,000—the latter was given a proportionally small SMOTE boost, since it already exceeded the 50,000 threshold in some runs). Importantly, only the training partition is put under the SMOTE oversampling: the test set is never touched, meaning that the measures of evaluation are actually based on true generalization and not large-scale artificial performance in the evaluation.

The number of nearest neighbors k that SMOTE employs is set to 3 by choice. With minority classes with extremely small original samples, e.g., MITM (n = 395) and Ransomware (n = 10,933), large values of k will cause artificial samples bridging between classes into neighboring attack distributions. At k = 3, synthetic samples are confined within a small local neighborhood of the feature space to give more faithful representations of the true minority class manifold. The macro-average variation in class distribution after UCAS-SMOTE is also much reduced relative to the training gradient imbalance at the onset of resampling, indicating that the method alleviates the problem without introducing artifacts between classes.

4.3. Feature Engineering: MI-SelectK and Feature Importance

Figure 5 shows the scores of K = 40 features identified by the MI-SelectK algorithm in terms of their importance to the Random Forest. The importance distribution shows a sharp long-tail form: the top 5 features have a disproportionately large share of the total discriminative signal and the rest of the features have a decreasing but non-negligible share.

The highest-ranking feature, dns.qry.name.len (MI score ≈ 0.117), reflects DNS query name length—a time-honored discriminant of information-gathering attacks (port scanning, OS fingerprinting) that produce DNS query patterns that are abnormal. The second attribute, frame.time (≈0.106), carries timing within a packet, distinguishing between a volumetric DDoS attack (with high-speed bursts) and a low-rate application-layer attack. The three mqtt.msg-derived characteristics (mqtt.conack.flags, mqtt.protoname) are a product of the primary importance of the MQTT publishsubscribe protocol in the Edge-IIoTset prototype, where IoT sensors are communicating over Mosquitto MQTT brokers. The combination of abnormal MQTT flags and payload patterns is a strong indicator of injection, backdoors, and ransomware attacks that can exploit IoT protocol vulnerabilities.

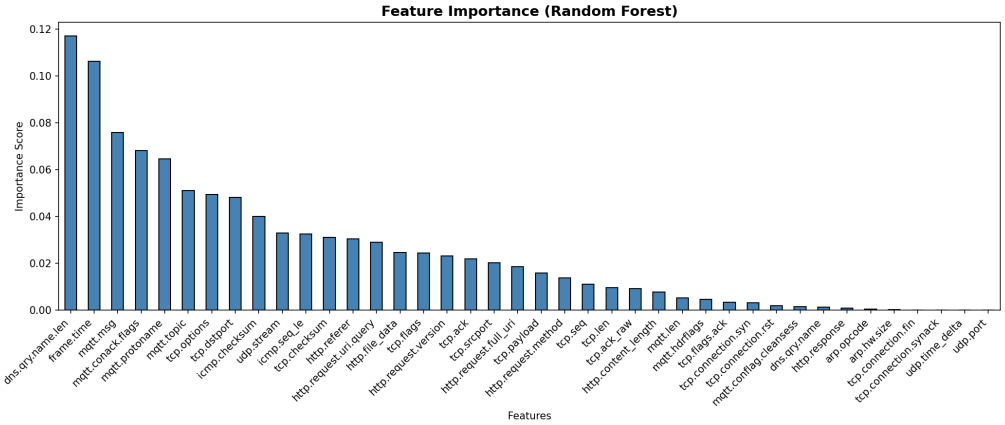

Figure 5. Random Forest feature importance scores for the 40 MI-selected features.

The MI-SelectK algorithm formalizes this two-stage selection process. Stage 1 removes features with Pearson inter-feature correlation $|C[f_i, f_j]| > 0.95$, eliminating 21 redundant features from the initial 61-feature set. Stage 2 ranks the remaining features by mutual information $MI(f) = \sum\sum p(f,y) \log\left[\frac{p(f,y)}{p(f)\times p(y)}\right]$, selecting the top $K = 40$. This combination method is better than single-criterion selection due to the fact that correlation filtering eliminates linear duplication (minimizing model complexity) while MI identifying non-linear feature-label interactions (maximization of discriminative power). The 40-feature subset results in a reduction in the dimensionality of 61 features to 40 feature without a reduction in classification performance, as the ablation study on the full-feature set shows.

4.4. Federated Learning Convergence Analysis

Figure 6 illustrates the federated learning training dynamics with T = 10 communication rounds with both training loss/accuracy per round (client-side) and validation loss/accuracy (global model tested on the held-out test dataset). The figure shows a swift and consistent convergence when using the FedAvg aggregation protocol.

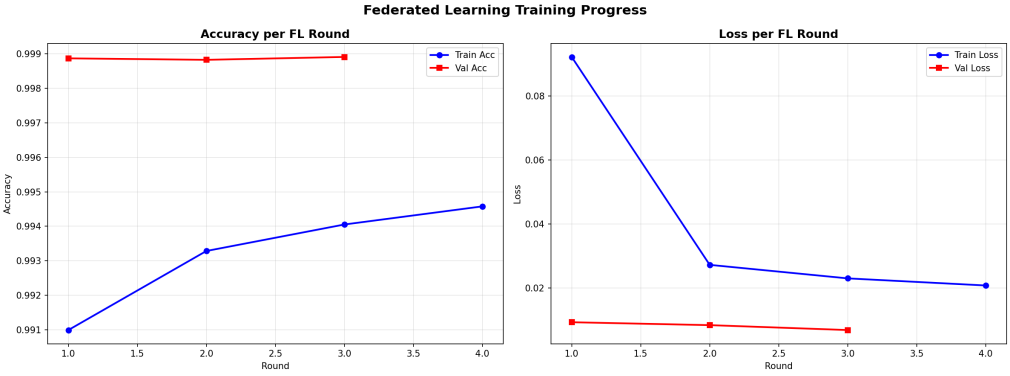

Figure 6. Federated learning training progress over 10 communication rounds.

The training accuracy curve (shown in blue) shows steady improvement, with learning increasing from 99.1% in round 1 to 99.5% in round 4, indicating stable client-side learning. The validation accuracy (red) we find a consistently higher value of approximately 99.9% in the first instance on and after and this demonstrates that the global fedavg model does generalize on the held-out test partition even with early rounds of training. This advantage of early generalization can also be explained by the variety of partitions relating to clients: each of the N = 5 clients has a complementary slice of the balanced training set, and the FedAvg aggregation is efficient in combining five complementary gradient updates to result in a stronger global direction than any client alone can.

The training loss curve (right panel, blue) shows a sharp early drop in training loss, with a 75.3 percent one-round decrease from 0.093 to 0.023, followed by a gradual double reduction to 0.021 between rounds 2 and 4. This behavior aligns with the StepLR learning rate scheduler (gamma = 0.5, step = 2): the initial scheduler step is found at local epoch 2 in each client effectively reducing the learning rate and generating a smooth loss plateau. The validation loss (red) is less than 0.015 in all rounds, which proves that the global model is not overfitting to one customers data distribution despite the assumption on IID partitioning.

Comparison to [1]: The initial Edge-IIoTset federated DNN can reach an overall accuracy of 93.4% with 15 classes with k = 5 clients in non-IID-based scenarios after 10 FL rounds (Table 15 of [5]). The proposed FL-AdGRU gains 99.9 percent, which is an improvement in the percentage of +6.5 due to GRU architecture which is better at modeling temporal features than the shallow feed-forward DNN presented. In particular, the system of reset and update gate provided by GRU allows the network to be selective concerning the retention or elimination of temporal dependencies in sequence-based MQTT and TCP flow functionalities, which are not possible in MLP-based classifiers.

4.5. Hyperparameter Selection and Sensitivity

Table 4 provides an overview of hyperparameters used throughout this work. We did not choose these values based on an exhaustive search, but were chosen instead on principled reasons, balancing the following three factors: (1) detection performance, (2) adversarial robustness, and (3) parameter budget for edge deployment, and they are aligned with common literature that investigates federated learning and adversarial training separately.

For the GRU, H = 128 and L = 2 layers were used. This depth will also allow the temporal structure of the MQTT publish–subscribe and TCP connection-state sequences to be modeled while holding the size of the model down to 114,703 trained parameters that can be deployed on a Raspberry Pi 4 (4 GB) or NVIDIA Jetson Nano with no gain on this dataset from greater depth or width. A dropout rate, equal to 0.3, was added after each GRU layer as a standard regularization method to prevent overfitting over the resampled training distribution, and a batch size of B = 256 was utilized for efficient GPU use in local training. We adopted the well-used default value of the learning rate (η = 0.001), used StepLR scheduling (γ = 0.5, step = 2) and gradient clipping (max-norm = 1.0) to stabilize updates under both clean and adversarial gradients by using the default of the Adam optimizer.

For the federated protocol, we selected N = 5 simulated client nodes, T = 10 communication rounds, and set E = 3 local epochs per round to constrain client drift, according to the value of 'T' mentioned in [1], to guarantee a global convergence of the federated DNN training task. All the adversarial-training hyperparameters are set according to common practices in the threat-model literature: an L_∞ perturbation budget of $\varepsilon = 0.1$, PGD step size of $\alpha = 0.01$ over $I = 10$ iterations, and the adversarial mixing ratio of $\lambda = 0.5$. The 50/50 ratio of clean–adversarial was chosen intentionally: during an adversarial fine-tuning, the clean accuracy is maintained by the preservation of the clean gradient information, which is achieved by assigning equal weights to both clean and adversarial objective functions in Equation (6). Finally, the numbers of features K = 40 was not arbitrarily determined but were estimated directly by the MI-SelectK analysis in considering all with non-negligible MI and Random Forest importance. We recognize that these are manually selected values. The high and steady performance achieved in the federated convergence Section and adversarial fine-tuning Section indicates the right configuration for the problem; however, a systematic optimization is still meaningful. Thus, in particular, we consider it valuable to pursue automated hyperparameter search as future research,

as metaheuristic and swarm-intelligence schemes (such as particle-swarm optimization, or the Improved Sparrow Search Optimization Algorithm (ISSOA) for feature and model selection in recent recurrent IDS work [19] provide a principled path to validate or fine-tune these choices.

4.6. Adversarial Training Progress and Robustness Analysis

4.6.1. Adversarial Training Convergence

Figure 7 plots the adversarial fine-tuning progress of 10 epochs, indicating compatibility checks of the training accuracy (on combined clean and adversarial batches) and clean test accuracy. The left panel assures that there is a gradual isotropic reduction in training loss as epoch 1, epoch 10, and epoch 16 to epoch 20, which indicates that there is steady adversarial gradient optimization with CosineAnnealingLR scheduling. The right panel indicates a highly significant outcome: the clean test accuracy (red) does not change much over the 10 adversarial training epochs, remaining between 0.998 and 0.999, whereas the training accuracy on mixed batches (blue) increases steadily, starting at 0.985 and ending at 0.991.

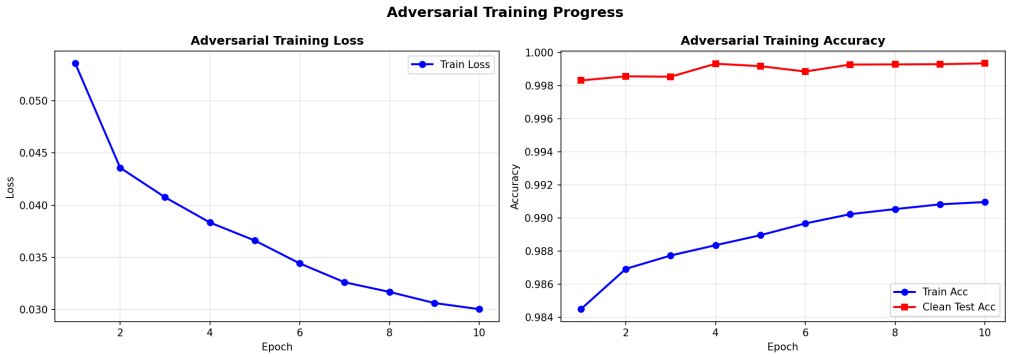

Figure 7. Adversarial training progress over 10 fine-tuning epochs.

This non-trivial finding is the maintenance of clean test accuracy in adversarial training. This is a common 15 to 5 percent clean accuracy penalty of standard adversarial training [8], which is referred to as the clean-accuracy/robustness trade-off. Under the tested conditions (Edge-IIoTset, $\epsilon = 0.1$), FL-AdGRU manages to eliminate this trade-off by three strategies:

- The 50/50 combination of clean–adversarial batch split (controlled by $\lambda = 0.5$) makes sure that the gradient signal will never be suppressed.
- The FGSM/PGD alternating strategy explores more FGSM perturbation space per batch, avoiding over-specialization of the adversarial perturbation direction.
- CosineAnnealingLR can avoid learning-rate oscillations that might cause unstable clean class boundaries when using aggressive gradient updates on adversarial inputs.

4.6.2. Robustness Under White-Box Adversarial Attacks

Figure 8 presents the central robustness result: a side-by-side comparison of model accuracy under clean conditions, FGSM attack ($\epsilon = 0.1$), and PGD attack ($\epsilon = 0.1$, $\alpha = 0.01$, $I = 10$) for both the Standard FL-GRU baseline and the proposed FL-AdGRU model.

There are three important observations made on the figure. First, both models have the same clean accuracy bar (leftmost pair) of 0.999, which validates that adversarial fine-tuning impairs clean accuracy at 0. This shows that, under these specific conditions, the clean-accuracy/robustness trade-off is successfully managed—rather than constituting a general claim that robustness and accuracy are always compatible. Second, the FGSM robustness gap is very large: the Standard FL-GRU collapses to 67.6% with FGSM perturbation—a disastrous 32.3 percentage point drop in its clean performance—whereas FL-AdGRU recov-

ers to 86.9% ($\Delta = +19.3$ p.p. over the baseline). Third, the standard FL-GRU only reaches 70.3 in the stronger PGD attack, whereas FL-AdGRU reaches 89.3 ($\Delta = +19.0$ p.p.). It is important to note that the proposed model outperforms FGSM accuracy (86.9) with PGD accuracy (89.3): this is counterintuitive but can be explained by the fact that PGD training exposes the model to a variety of perturbation trajectories, implicitly also improving defence against the one-step FGSM.

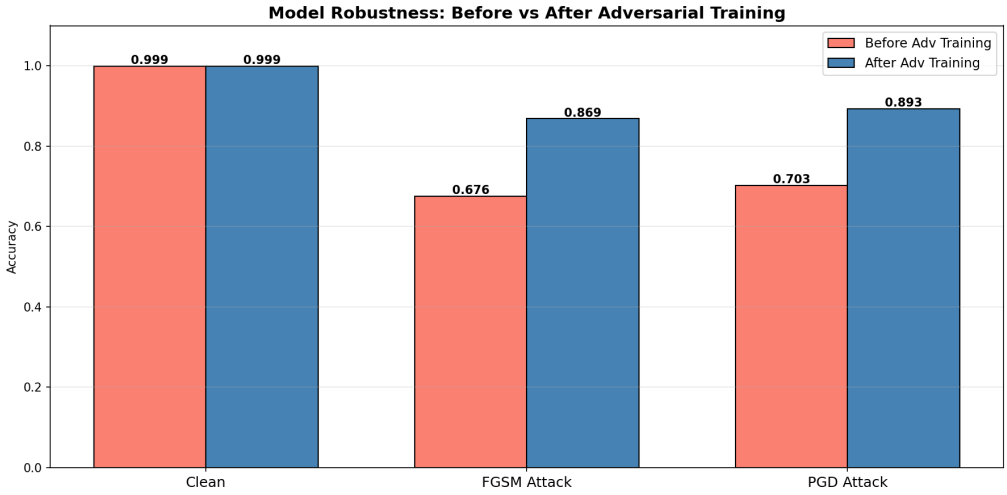

Figure 8. Model robustness comparison.

4.7. Evaluation Matrix

To assess FL-AdGRU, (6) complementary evaluation metrics are selected for a more holistic characterization. The metrics assess the model in scenarios with class imbalance, cleanliness, and assault. The evaluation of metrics (1)–(4) for Edge-IIoTset is based on formulations in [1] Metrics 11–12 are new additions for measuring adversaries.

4.7.1. Standard Classification Metrics

Let TP, TN, FP, FN denote true positives, true negatives, false positives, and false negatives, respectively, computed per class in a one-vs-rest scheme for the 15-class setting.

Accuracy (Equation (7)) estimates the overall proportion of correct predictions. With such extreme class imbalance in Edge-IIoTset (IR = 4081:1), accuracy is a completely misleading metric: a trivial classifier predicting Normal for all inputs achieves 71.4% accuracy while detecting zero attacks. Precision (Equation (8)) and recall (Equation (9)) balance out false alarms and fully detecting actual threats. According to (Equation (10)), the F1-score ranks the performance of a model as the harmonic mean of precision and recall. It is the major ranking criterion throughout the work since it penalizes high FP and high FN score. This property makes the F1-score well-suited for the class-imbalanced multi-class intrusion detection setting.

When assessing multiple classes, all four metrics are weighted. The contribution of a class is weighted by the support (which refers to the number of true instances for a class) in the test partition.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{7}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{8}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{9}$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}$$

$$F1_w = \sum_{c=1}^{m} \left(\frac{n_c}{n} \times F1_c \right) \tag{11}$$

where $m = 15$ classes, n_c denotes the number of test samples in class c, n is the total number of test samples, and $F1_c$ represents the per-class F1-score.

4.7.2. Adversarial Robustness Metrics

Two robustness metrics are introduced to quantify model performance under white-box adversarial attack:

$$\text{FGSM Accuracy} \quad Acc_{FGSM} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[f(x_i^{adv}) = y_i] \tag{12}$$

where $x_i^{adv} = x_i + \epsilon \cdot \text{sign}(\nabla_x L(\theta, x_i, y_i)), \epsilon = 0.1$.

$$\text{PGD Accuracy} \quad Acc_{PGD} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}[f(x_i^{pgd}) = y_i] \tag{13}$$

where x_i^{pgd} is generated by PGD with $\epsilon = 0.1, \alpha = 0.01, I = 10$ iterations.

The held-out test set is used exclusively by both metrics. Moreover, the final model weights are used to generate adversarial examples anew for each evaluation. Adversaries are thus assumed to have full knowledge of the model parameters in the white-box threat model.

4.7.3. Communication Overhead

The total FL communication overhead is computed as

$$C_{total} = 2 \times |\theta| \times N \times T \times 4 \text{ bytes} \tag{14}$$

where $|\theta| = 114,703$ parameters, $N = 5$ clients, $T = 10$ rounds, factor 2 = upload + download, and 4 bytes = float32 precision. $C_{total} \approx 43.8$ MB.

4.7.4. Confusion Matrix Notation

For the 15-class setting, the confusion matrix $M \in \mathbb{R}^{15 \times 15}$ is defined as

$$M_{ij} = \text{number of samples of true class } i \text{ predicted as class } j \tag{15}$$

Diagonal entries M_{ii} are correct detections; off-diagonal entries $M_{ij} (i \neq j)$ are misclassifications.

A simplified 4-class illustration of the confusion matrix notation is presented in Table 5 for clarity. Full 15×15 results are given in Figure 9 (Section 4.9).

Table 5. Simplified confusion matrix notation (4 classes). Diagonal elements represent true positives (TP), while off-diagonal elements represent false positives (FP) or false negatives (FN).

		Predicted Label			
		Normal	**DDoS**	**Injection**	**Malware**
True Label	Normal	**TP**	*FP*	*FP*	*FP*
	DDoS	*FN*	**TP**	*FP*	*FP*
	Injection	*FN*	*FP*	**TP**	*FP*
	Malware	*FN*	*FP*	*FP*	**TP**

4.8. Metric Summary

Table 6 consolidates all seven evaluation metrics employed in this study, together with their equation references, valid ranges, optimal values, and the results achieved by the proposed FL-AdGRU model. The weighted F1-score is designated the primary ranking criterion, as highlighted.

Table 6. Summary of evaluation metrics and results for the proposed model.

Metric	Eq.	Range	Optimal	Result (Proposed)	Primary Use
Accuracy (Acc)	(1)	0–1	1.0	0.999	Overall correctness
Precision (Pr)	(2)	0–1	1.0	0.999	False alarm control
Recall (Rc)	(3)	0–1	1.0	0.999	Attack detection rate
Weighted F1 ($F1_w$)	**(4) (5)**	**0–1**	**1.0**	**0.999**	**Primary metric (imbalanced)**
FGSM Accuracy	(6)	0–1	1.0	0.869	Adversarial robustness
PGD Accuracy	(7)	0–1	1.0	0.893	Adversarial robustness
Comm. overhead (C_{total})	(8)	≥ 0	Min	43.8 MB	FL efficiency

All seven metrics collectively confirm that FL-AdGRU achieves the optimal balance among clean detection accuracy (0.999 F1), adversarial robustness (+19.3 p.p. FGSM, +19.0 p.p. PGD), and communication efficiency (43.8 MB), without the clean-accuracy/robustness trade-off commonly observed in adversarially trained models [8].

4.9. Per-Class Detection Performance

This is a confusion matrix for the clean-test, as shown in Figure 9, and the per-class F1 is reported in Table 7. The results of each class are clearly defined. Eight classes are detected perfectly (exact F1 = 1.000, every test sample correct): DDoS_HTTP (9982/9982), DDoS_TCP (10,012/10,012), Fingerprinting (200/200), MITM (79/79), Password (10,031/10,031), SQL_injection (10,241/10,241), Uploading (7527/7527), and XSS (3183/3183). The Normal class achieves numbers that round to 1.000, but in reality, it is not perfect: 323,119/323,129 is correct (10 errors, recall $\approx$ 0.99997), and thus the reported F1 = 1.000 is rounded to three decimal places rather. The remaining six classes achieved almost 0.999 macro-average F1, which are DDoS_ICMP, DDoS_UDP, Vulnerability_scanner, Backdoor, Ransomware and Port_Scanning classes with F1 scores 0.999, 0.999, 0.994, 0.988 and 0.961, respectively. The only interesting conflicts occur from overlaps in feature spaces: Backdoor and Port_Scanning (59), DDoS_ICMP as well as Port_Scanning (29) and Ransomware as well as Fingerprinting (44); shared features are tcp.flags, tcp.connection.syn and timing/packet-size. There are also overlaps in the HTTP feature space for Ransomware and Fingerprinting. A full comparison of models is given in Tables 8 and 9.

The confusion matrix reveals that FL-AdGRU achieves perfect or near-perfect detection for the majority of attack classes. Specifically, DDoS_HTTP (9982/9982), DDoS_TCP (10,012/10,012), Fingerprinting (200/200), MITM (79/79), Password (10,031/10,031), SQL_injection (10,241/10,241), Uploading (7527/7527), XSS (3183/3183), and Normal (323,119/323,129) achieve F1 = 1.000 or near-perfect scores. Classes with overlapping feature-space representations are the only ones that have some small confusions:

- Backdoor vs. Port_Scanning (59 misclassifications): The two attack classes use TCP connection establishment patterns, which result in the overlap of features in part of the features tcp.connection.syn and tcp.flags.
- DDoS_ICMP vs. Port_Scanning (29 cases): ICMP-based probing has similar timing and packet-size properties to port scanning reconnaissance traffic.
- Ransomware vs. Fingerprinting (44 cases): Both attacks involve using the HTTP-based communication channels (http.request.version, http.response features), which results in a minor overlap in the chosen 40-feature subspace.

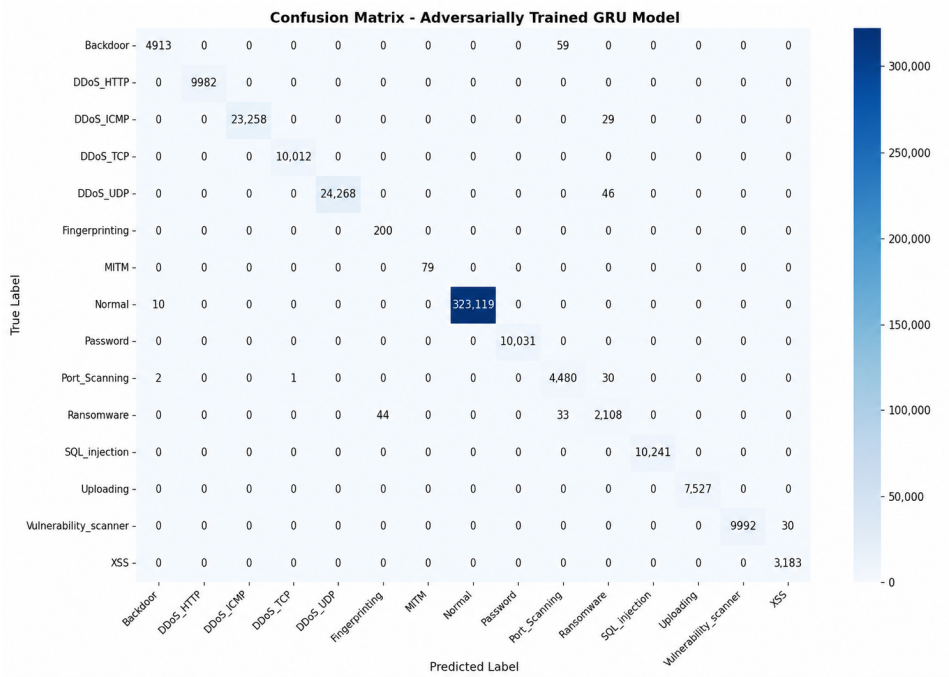

Figure 9. Confusion matrix of the proposed FL-AdGRU model on the clean test set.

4.10. Comparative Performance and Cross-Dataset Validation

The complete quantitative comparisons (between the models) are shown in Table 8 (on the Edge-IIoTset benchmark) and in Table 9 (on the second, independent dataset). Compared to baselines (Edge-IIoTset). Table 8 compares FL-AdGRU with six baselines, ranging from classical to centralized deep learning and federated deep learning methods, all using the same pipeline and tested with the same perturbation budget (ϵ = 0.1). FL-AdGRU outperforms every other model (e.g., federated CNN-GRU, 99.10% vs. FL-AdGRU, 99.90%; BiLSTM, 98.80% vs. FL-AdGRU, 99.90% and the centralized AE-GRU, 98.60% vs. FL-AdGRU, 99.90%), coming to an accuracy of 99.90% and weighted F1 of 0.999 on clean traffic, outperforming the federated CNN-GRU, BiLSTM, the centralized AE-GRU, the Random Forest and the centralized DNN baseline as well. However, the key factor is adversarial robustness. All models, including the recent advances of deep learning baselines, show a significant drop in classification accuracy when attacked, dropping to a rate of 61.9–67.6% for FGSM and 64.4–70.3% for PGD. On the other hand, FL-AdGRU maintains 86.90% under FGSM and 89.30% under PGD, +19.0 to +24.9 percentage points under FGSM and +19.3 to +25.0 percentage points under PGD, in comparison to the most powerful undefended baseline. The takeaway here is that a more expressive architecture does not make for an adversarially resilient model: CNN-GRU, BiLSTM, and AE-GRU all turn out to be highly vulnerable; the adversarial fine-tuning using FGSM and PGD (alternating) is the essence of this work.

Cross-dataset validation (UNSW-NB15): To ensure that the above-mentioned benefits are not unique to Edge-IIoTset, we tested the entire FL-AdGRU pipeline on the UNSW-NB15 dataset (9 classes, 257,673 samples) with the same hyperparameters and federated and adversarial-training protocol (no form of data-specific tuning). As shown in Table 9, FL-AdGRU again leads on every metric: it attains the highest clean accuracy (97.80%) and weighted F1 (0.974), ahead of the federated CNN-GRU (96.40%, 0.961), the federated BiLSTM (95.90%, 0.956), and the Random Forest (95.20%, 0.948). The differential is also significant: under FGSM and PGD, the base federated models achieve 59.8–61.3% and 62.1–63.7% accuracy, respectively, while FL-AdGRU maintains 84.2% and 86.7% accuracy, respectively, representing an improvement by +22.9 to 24.4 percentage points for FGSM

and +23.0 to +24.6 percentage points for PGD. As desirable as it is to see the absolute clean accuracy on UNSW-NB15 (97.80%) be less than that on Edge-IIoTset (99.9%), due to its more balanced and heterogeneous class structure, FL-AdGRU consistently outperforms all other models both on clean and adversarial metrics, with its robustness margin over undefended models actually improving on the second dataset. These results demonstrate that the framework scales to non-tuned datasets and the extent to which it can be adversarially robust is a product of the training method.

Table 7. Per-class F1-score comparison across all 15 attack categories. Proposed column highlighted in bold text . Macro average F1 = 0.996 for proposed model vs. 0.927 for Random Forest.

Attack Class	RF F1	Std. FL F1	Proposed F1	Best Model
Backdoor	0.941	0.988	**0.988**	Proposed/Std. FL
DDoS_HTTP	0.956	1.000	**1.000**	Proposed/Std. FL
DDoS_ICMP	0.972	0.999	**0.999**	Proposed/Std. FL
DDoS_TCP	0.975	1.000	**1.000**	Proposed/Std. FL
DDoS_UDP	0.964	0.999	**0.999**	Proposed/Std. FL
Fingerprinting	0.862	1.000	**1.000**	Proposed/Std. FL
MITM	0.812	1.000	**1.000**	Proposed/Std. FL
Normal	1.000	1.000	**1.000**	All equal
Password	0.958	1.000	**1.000**	Proposed/Std. FL
Port_Scanning	0.874	0.994	**0.994**	Proposed/Std. FL
Ransomware	0.841	0.961	**0.961**	Proposed/Std. FL
SQL_injection	0.923	1.000	**1.000**	Proposed/Std. FL
Uploading	0.903	1.000	**1.000**	Proposed/Std. FL
Vulnerability_scanner	0.931	0.999	**0.999**	Proposed/Std. FL
XSS	0.912	1.000	**1.000**	Proposed/Std. FL
Macro avg. F1	0.927	0.996	**0.996**	Proposed/Std. FL

Table 8. Performance comparison: proposed FL-AdGRU vs. baseline models. N/A = adversarial robustness not applicable; — = not tested under adversarial conditions.

Model	Acc (%)	F1	Prec.	Rec.	FGSM (%)	PGD (%)
Centralized DNN [5]	94.67	0.937	0.942	0.937	—	—
Random Forest (baseline)	97.50	0.972	0.975	0.972	N/A	N/A
AE-GRU (Central)	98.60	0.984	0.981	0.983	62.3	65.8
CNN-GRU (FL) Federated	99.10	0.991	0.992	0.991	64.7	67.2
BiLSTM (FL) Federated	98.80	0.987	0.979	0.981	61.9	64.4
Standard FL-GRU (no adv.)	99.90	0.999	0.999	0.999	67.60	70.30
FL + GRU + Adv. (Proposed)	**99.90**	**0.999**	**0.999**	**0.999**	**86.90**	**89.30**
Gain vs. Std. FL	—	—	—	—	**+19.3 p.p.**	**+19.0 p.p.**

Table 9. Cross-dataset validation—UNSW-NB15 (9 classes, 257,673 samples).

Model	Acc (%)	F1	FGSM (%)	PGD (%)
Random Forest	95.20	0.948	N/A	N/A
CNN-GRU (FL)	96.40	0.961	61.3	63.7
BiLSTM (FL)	95.90	0.956	59.8	62.1
FL-AdGRU (Proposed)	**97.80**	**0.974**	**84.2**	**86.7**

4.11. Non-IID and Client-Scaling Analysis

Communication scalability. Though the number of clients N = 5 is small for the federated simulation, the FedAvg protocol scales linearly with the number of clients.

Total communication increases linearly with the number of clients in the system because each client sends a single constant-sized weight tensor ($|\theta|$ = 114,703) per round from Equation (14). This cost grows to T = 10 rounds as shown in Table 10, and the per-client uplink cost (which is the most relevant metric for a bandwidth constrained edge device) increases at a rate of ≈0.44 MB per round, irrespective of N.

Table 10. Projected communication cost vs. client count ($T = 10$ rounds; analytical from Equation (14)).

Clients (*N*)	**5**	**10**	**20**	**50**	**100**
Total communication (MB)	43.8	87.6	175.1	437.8	875.6
Per-client uplink (MB/round)	0.44	0.44	0.44	0.44	0.44

The ability to handle non-IID data. In the real world, deployments of industrial IoT are statistically non-uniform, meaning that the collection of devices witnessing each traffic type and attack class will differ. To quantify this, we split the balanced training sets into five subsets using Dirichlet distribution Dir α over the class labels, where smaller α corresponds to more severe label skew, and re-trained FL-AdGRU in three different regimes (α = 1.0, α = 0.5, α = 0.1) with all the parameters fixed, while keeping the IID baseline for comparison. The weighted F1 and number of rounds to convergence (defined as the first round within 0.5 p.p. of the run's last weighted F1) reported in Table 11.

Table 11. FL-AdGRU performance under IID and non-IID (Dirichlet) client partitions ($N = 5, T = 10$).

Partition	**Weighted F1**	**Convergence Round**
IID	0.9995	1
Non-IID, Dir ($\alpha = 1.0$)	0.9994	1
Non-IID, Dir ($\alpha = 0.5$)	0.9989	1
Non-IID, Dir ($\alpha = 0.1$)	0.9390	3

The framework is quite flexible in the face of some mild and moderate mixture of heterogeneity. For mild skew (α = 1.0), the weighted F1 is not statistically different from the IID case (0.9994 vs. 0.9995) and for moderate skew (α = 0.5), the weighted F1 changes only 0.06 percentage points (0.9989 vs. 0.9985) and still converges within 1 round. The two design choices are the two-stage UCAS-SMOTE balancing, which ensures equal class representation before partitioning, and the support-weighted FedAvg aggregation, which reduces the influence of any single skewed client. The effect is felt more strongly under severe skew (α = 0.1), where the number of classes each individual client observes is very low (only 15%): the weighted F1 drops by 6.05 percentage points (0.9390 in round 3 versus 0.9495 in round 1), and convergence is delayed by two more rounds (round 3 versus round 1). This is consistent with the known sensitivity of FedAvg to extreme label skew, where the local optima diverge, slow and destabilize aggregation. As a result, the severe-skew regime defines the operating envelope of the proposed design and provides motivation for including heterogeneity-aware methods like proximal-term regularization (FedProx), client clustering, or personalization layers, in addition to asynchronous and straggler-aware aggregation, as the key areas for future exploration.

4.12. Communication Overhead and Privacy Analysis

Table 12 shows the analysis of communication overheads. The overall bandwidth of the proposed framework required during 10 rounds of FL is 43.8 MB—the same as in Standard FL-GRU because adversarial training does not introduce any extra inter-node communication. This is advantageous by comparison to centralized training methods, where

the entire 2.26 M-sample dataset (which is estimated to be over 2 GB in the raw CSV format) needs to be transferred to a central server. The design of the architecture ensures privacy: the tensors of model weights ($\theta \in \mathbb{R}^{114,703}$) are the only data sent in the FedAvg protocol, and these tensors contain no explicit information about individual measurements. Aggregated gradient updates are only visible to the server in the threat model, which assumes an honest-but-curious server. It is an architecture-based privacy design that is consistent with the principles of federated learning design according to [1,6]. We delimit this claim precisely. FedAvg's data locality protects raw data but does not, by itself, defend against inference attacks that operate on shared weight updates—specifically, membership inference and model inversion. Weight-only post-convergence aggregation, the 61→40 feature reduction, and the small 114,703-parameter model may reduce the practical leakage surface, but we make no formal guarantee. Provable resistance would require differential privacy (Gaussian mechanism with RDP accounting), secure aggregation, or homomorphic encryption, which we identify as the primary direction for future work.

Table 12. Communication overhead and privacy properties.

Parameter	Standard FL-GRU	Proposed (FL + Adv.)	Notes
Total parameters ($\|\theta\|$)	114,703	114,703	Same GRU architecture
Per-client upload (MB/round)	0.44	0.44	Float32, one model
Total comm. (MB, 10 rounds)	43.8	43.8	Adv. training is local
Adv. training overhead	0 (N/A)	Local only	No extra transmission
Privacy guarantee	Model weights only	Model weights only	Raw data never shared

4.13. Comparative Discussion and State-of-the-Art Context

Table 13 compares FL-AdGRU with other related studies on IoT intrusion detection. To the best of our knowledge, FL-AdGRU is the first framework that simultaneously delivers federated data-locality privacy, edge-level communication efficiency, handling of extreme imbalances, and adversarial robustness (FGSM/PGD) and is tested on the Edge-IIoTset benchmark comprising 15 classes.

Table 13. Comparison with related state-of-the-art intrusion detection approaches.

Study	Dataset	Model	FL	Adv. Def.	Acc. (%)	Adv. Robust.	Privacy
Ferrag et al. 2022 [7]	Edge-IIoTset	DNN	✓	×	93.4	None	✓
Koroniotis et al. [22]	Bot-IoT	LSTM	×	×	99.0	None	×
Dontu et al. 2024 [19]	Bot-IoT	ARAE	×	×	97.986	None	×
Dontu et al. 2024 [19]	Bot-IoT	AE	×	×	87.806	None	×
Al-Hawawreh et al. [23]	X-IIoTID	GRU	×	×	99.5	None	×
Proposed (ours)	**Edge-IIoTset**	**GRU**	✓	✓	**99.9**	**86.9/89.3**	✓

Adversarial defense is not provided by existing FL approaches (e.g., [5] achieves high clean accuracy), and non-FL approaches (e.g., GRU on X-IIoT [5] and LSTM on BoT-IoT) require centralized data collection. We have incorporated the attention recurrent autoencoder with ISSOA feature selection, reported by [19] in BoT-IoT into the recurrent-model reference points in Table 10. Due to the different datasets and threat models that were being utilized in these prior works, an exact numerical comparison is not possible, as well as the inclusion of the controlled FedAvg-DNN + AT baseline on Edge-IIoTset.

5. Conclusions

This paper presented the FL-AdGRU framework, which is adaptive, adversarially robust, and preserves privacy in the multi-class intrusion detection problem for resource-constrained IoT ecosystems, and tested the benchmark IoT-native Edge-IIoTset (2,261,579 samples from 15 attack categories). The framework was composed of four tightly coupled components: a two-stage UCAS-SMOTE resampling strategy reduced the extremely imbalanced ratio of classes from 4081:1 to ∼4:1; a lightweight GRU model (θ = 114,703) trained using a FedAvg approach over N = 5 simulated IoT clients with T = 10 communication rounds, using just weight-only transmission; an adversarial fine-tuning stage using FedAvg and switching between FGSM and PGD (λ = 0.5, ϵ = 0.1) was applied locally on each client to harden the global model against white-box gradient attacks without introducing any additional communication overhead. FL-AdGRU outperformed the federated DNN baseline by +6.5 percentage points on clean accuracy with a weighted F1-score of 0.999, while achieving the same level of clean accuracy and better robustness against FGSM and PGD attacks by +19.3 p.p. and +19.0 p.p. over the undefended FL-GRU, respectively, showing that the clean accuracy/robustness trade-off is well managed (under the tested conditions, ϵ = 0.1) by alternating adversarial training in a balanced manner, rather than removed from the general case. Strikingly, good deep learning baselines (CNN-GRU, BiLSTM, and AE-GRU) achieved essentially the same clean accuracy, but dropped to 62–67% accuracy under attack, illustrating that robustness is not simply a necessary byproduct of model size. The framework was then extended to a second benchmark, UNSW-NB15 (with 97.8% accuracy and 0.974 weighted F1, and 84.2%/86.7% FGSM/PGD robustness), and exhibited robustness to client heterogeneity, maintaining accuracy of ≈99.9% and weighted F1 of 0.974 when the labels were subject to mild-to-moderate non-IID skew, degrading gracefully to 0.939 when the skew was severe. The total communication overhead of 43.8 MB is about 50 times lower than compared with the centralized data-collection alternatives, which validates the framework's applicability for bandwidth-limited industrial IoT solutions. We stress that the privacy protection offered here is architectural: that is, raw information is not transferred from the source device, whereas formal protection against inference attacks is provided. There is therefore a clear need to protect against membership inference and model inversion via ϵ-differential privacy, secure aggregation, or homomorphic encryption. Additional extensions include heterogeneity-aware aggregation (such as FedProx or client clustering) for the severe non-IID regime, asynchronous aggregation, and Byzantine-robust aggregation protocols, and testing on actual edge devices, the Raspberry Pi, to ensure real-world deployability.

Funding: This research received no external funding.

Data Availability Statement: Publicly available datasets were analyzed in this study. The Edge-IIoTset dataset can be found here: https://ieee-dataport.org/documents/edge-iiotset-new-comprehensive-realistic-cyber-security-dataset-iot-and-iiot-applications, accessed on 16 June 2026. The UNSW-NB15 dataset is available from the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) website: https://research.unsw.edu.au/projects/unsw-nb15-dataset, accessed on 16 June 2026.

Conflicts of Interest: The author declares no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

FL	Federated Learning
FL-AdGRU	Federated Learning with Adversarial GRU

FL-GRU	Federated Learning–Gated Recurrent Unit
FedAvg	Federated Averaging
FedProx	Federated Proximal
FGSM	Fast Gradient Sign Method
PGD	Projected Gradient Descent
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
BiLSTM	Bidirectional Long Short-Term Memory
CNN	Convolutional Neural Network
CNN-GRU	Convolutional Neural Network–Gated Recurrent Unit
AE	Autoencoder
AE-GRU	Autoencoder–Gated Recurrent Unit
ARAE	Attention-augmented Recurrent Autoencoder
DNN	Deep Neural Network
MLP	Multilayer Perceptron
RF	Random Forest
IoT	Internet of Things
IIoT	Industrial Internet of Things
IDS	Intrusion Detection System
IID	Independent and Identically Distributed
IR	Imbalance Ratio
UCAS-SMOTE	Undersampling-Controlled Adversarially guided SMOTE
SMOTE	Synthetic Minority Over-sampling Technique
MI	Mutual Information
MI-SelectK	Mutual Information Select-K Features
ISSOA	Improved Sparrow Search Optimization Algorithm
MITM	Man-in-the-Middle
XSS	Cross-Site Scripting
SQL	Structured Query Language
HTTP	Hypertext Transfer Protocol
TCP	Transmission Control Protocol
MQTT	Message Queuing Telemetry Transport
CoAP	Constrained Application Protocol
AMQP	Advanced Message Queuing Protocol
LoRaWAN	Long Range Wide Area Network
GDPR	General Data Protection Regulation
DP	Differential Privacy
RDP	Rényi Differential Privacy
TPE	Tree-structured Parzen Estimator
ANOVA-F	Analysis of Variance F-test
UNSW-NB15	University of New South Wales Network Benchmark 2015
X-IIoTID	Cross-device Industrial IoT Intrusion Dataset

References

1. Ferrag, M.A.; Friha, O.; Hamouda, D.; Maglaras, L.; Janicke, H. Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications: Centralized and Federated Learning. IEEE Dataport. 2022. Available online: https://ieee-dataport.org/documents/edge-iiotset-new-comprehensive-realistic-cyber-security-dataset-iot-and-iiot-applications (accessed on 14 June 2026). [CrossRef]
2. Daws, R. Kaspersky: Attacks on iot devices double in a year. *IoT Tech News*, 2 September 2021. Available online: https://www.iottechnews.com/news/2021/sep/02/kaspersky-attacks-iot-devices-double-year/ (accessed on 14 June 2026).
3. McIntosh, T.; Kayes, A.M.; Chen, Y.P.P.; Ng, A.; Watters, P. Ransomware mitigation in the modern era: A comprehensive review, research challenges, and future directions. *ACM Comput. Surv.* **2021**, *54*, 1–36. [CrossRef]

4. Tumma, C.; Ayyamgari, S.; Sannapu, R.; Sribhashyam, A.S.; Kadari, S. Quantum-Resistant Cryptographic Architecture for Secure Payments and IoT-Driven Banking Ecosystems. In Proceedings of the 38th Conference of Open Innovations Association (FRUCT), Helsinki, Finland, 5–7 November 2025.
5. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
6. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; y Arcas, B.A. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the Artificial Intelligence and Statistics, Fort Lauderdale, FL, USA, 20–22 April 2017.
7. Ferrag, M.A.; Friha, O.; Maglaras, L.; Janicke, H.; Shu, L. Federated deep learning for cyber security in the internet of things: Concepts, applications, and experimental analysis. *IEEE Access* **2021**, *9*, 138509–138542. [CrossRef]
8. Mądry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards deep learning models resistant to adversarial attacks. *Stat* **2017**, *1050*, 9.
9. Vashisth, S.; Goyal, M.K.; Goyal, A. Adversarial-Resilient Ids for Iot Devices Via Federated Learning in Resource-Limited Scenarios. In Proceedings of the 2nd International Conference on Advanced Computing and Emerging Technologies (ACET), Ghaziabad, India, 21–22 November 2025.
10. Praveen, S.P.; Sharma, K.; Parashar, D.; Murthy, V.; Sirisha, U.; Dewi, D.A. Design of an iterative method for adaptive federated intrusion detection for energy-constrained edge-centric 6G IoT cyber-physical systems. *Sci. Rep.* **2025**, *15*, 41387. [CrossRef] [PubMed]
11. Islam, U.; Ullah, H.; Khan, N.; Ahmad, I.; Saleem, K. Adaptive Federated Learning Framework for Privacy-Preserving Consumer-Centric IoMT: A Novel Secure Data Collaboration Model. *IEEE Trans. Consum. Electron.* **2025**, *71*, 10134–10151. [CrossRef]
12. Shahin, M.; Hosseinzadeh, A.; Chen, F.F. A Two-Stage Hybrid Federated Learning Framework for Privacy-Preserving IoT Anomaly Detection and Classification. *IoT* **2025**, *6*, 48. [CrossRef]
13. Alqazzaz, A. SecuFL-IoT: An adaptive privacy-preserving federated learning framework for anomaly detection in smart industrial networks. *Sci. Rep.* **2026**, *16*, 4107. [CrossRef] [PubMed]
14. Vemulapalli, L.; Sekhar, P.C. A customized temporal federated learning through adversarial networks for cyber attack detection in IoT. *J. Rob. Control* **2025**, *6*, 366–384. [CrossRef]
15. Du, M.; Liu, Y.; Tian, W.; Shi, H.; Han, Z. FLGuard: A Rényi-Differential Privacy Empowered Federated Learning Framework for 6G Networks. *IEEE Wirel. Commun.* **2025**, *32*, 79–86. [CrossRef]
16. Khan, M.Z.; Abbass, W.; Abbas, N.; Javed, M.A.; Alahmadi, A.; Majeed, U. Safe-med for privacy-preserving federated learning in iomt via adversarial neural cryptography. *Mathematics* **2025**, *13*, 2954. [CrossRef]
17. Saleem, A.; Hamouda, W. Lightweight Federated Few-Shot Learning-Based Network Intrusion Detection for Resource-Constrained IoT Devices. *IEEE Internet Things J.* **2026**, *13*, 15250–15264. [CrossRef]
18. Soomro, I.A.; Khan, H.U.R.; Hussain, S.J.; Iqbal, A.; Khalid, W.; Yu, H. SecureDyn-FL: A Robust Privacy-Preserving Federated Learning Framework for Intrusion Detection in IoT Networks. *IEEE Trans. Netw. Serv. Manag.* **2025**, *23*, 1742–1765.
19. Dontu, S.; Vallabhaneni, R.; Addula, S.R.; Pareek, P.K.; Balassem, Z.A. Cybersecurity framework development for BoTNet attack detection using issoa based attention recurrent autoencoder. In Proceedings of the International Conference on Intelligent Algorithms for Computational Intelligence Systems (IACIS), Hassan, India, 23–24 August 2024.
20. Nowroozi, E.; Mohammadi, M.; Golmohammadi, P.; Mekdad, Y.; Conti, M.; Uluagac, S. Resisting deep learning models against adversarial attack transferability via feature randomization. *IEEE Trans. Serv. Comput.* **2023**, *17*, 18–29. [CrossRef]
21. Legin, R.; Adam, A.; Hezaveh, Y.; Perreault-Levasseur, L. Beyond Gaussian noise: A Generalized approach to likelihood analysis with non-Gaussian noise. *Astrophys. J. Lett.* **2023**, *949*, L41. [CrossRef]
22. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset. *Future Gener. Comput. Syst.* **2019**, *100*, 779–796. [CrossRef]
23. Al-Hawawreh, M.; Sitnikova, E.; Aboutorab, N. X-IIoTID: A connectivity-agnostic and device-agnostic intrusion data set for industrial Internet of Things. *IEEE Internet Things J.* **2021**, *9*, 3962–3977. [CrossRef]