applied sciences

MDPI

Article

Edge-FLGuard: A Federated Learning Framework for Real-Time Anomaly Detection in 5G-Enabled IoT Ecosystems

Manuel J. C. S. Reis

Engineering Department and IEETA, University of Trás-os-Montes e Alto Douro, Quinta de Prados, 5000-801 Vila Real, Portugal; mcabral@utad.pt

Abstract: The rapid convergence of 5G networks and Internet of Things (IoT) technologies has unlocked unprecedented connectivity and responsiveness across smart environments—but has also amplified cybersecurity risks due to device heterogeneity, data privacy concerns, and distributed attack surfaces. To address these challenges, we propose Edge-FLGuard, a federated learning and edge AI-based anomaly detection framework tailored for real-time protection in 5G-enabled IoT ecosystems. The framework integrates lightweight deep learning models—specifically autoencoders and LSTM networks—for on-device inference, combined with a privacy-preserving federated training pipeline to enable scalable, decentralized threat detection without raw data sharing. We evaluate Edge-FLGuard using both public (CICIDS2017, TON_IoT) and synthetic datasets under diverse attack scenarios including spoofing, DDoS, and unauthorized access. Experimental results demonstrate high detection accuracy (F1-score $\geq$ 0.91, AUC-ROC up to 0.96), low inference latency (<20 ms), and robustness against data heterogeneity and adversarial conditions. By aligning edge intelligence with secure, collaborative learning, Edge-FLGuard offers a practical and scalable cybersecurity solution for next-generation IoT deployments.

Keywords: federated learning; edge AI; IoT security; 5G networks; anomaly detection; privacy-preserving machine learning

Academic Editor: Douglas O'Shaughnessy

Received: 8 May 2025
Revised: 27 May 2025
Accepted: 6 June 2025
Published: 8 June 2025

Citation: Reis, M.J.C.S. Edge-FLGuard: A Federated Learning Framework for Real-Time Anomaly Detection in 5G-Enabled IoT Ecosystems. *Appl. Sci.* **2025**, *15*, 6452. https://doi.org/10.3390/app15126452

1. Introduction

The proliferation of Internet of Things (IoT) devices and the concurrent rollout of 5G networks have transformed modern communication infrastructures, enabling ultra-low latency, high device density, and ubiquitous connectivity. These advancements have facilitated the deployment of smart environments, including industrial automation, connected healthcare, and intelligent urban infrastructure [1,2]. However, this connectivity expansion also introduces new cybersecurity risks, as the distributed and heterogeneous nature of IoT ecosystems increases the attack surface and complicates traditional security monitoring [3].

Security solutions relying on centralized architectures struggle to meet the performance, latency, and privacy demands of 5G-enabled IoT systems. The volume and velocity of data generated by IoT nodes exceed the capabilities of conventional intrusion detection systems (IDS), especially under real-time constraints [4]. Furthermore, the centralized collection of data for security analytics raises privacy concerns and regulatory compliance issues, particularly under frameworks such as GDPR and CCPA [5]. These limitations have been further highlighted in recent privacy risk studies for both centralized and federated settings [6].

Edge computing has emerged as a promising architectural paradigm to address these challenges by enabling local, low-latency processing of security tasks closer to the data source [7]. When combined with Artificial Intelligence (AI), edge-based anomaly detection

systems can learn behavioral patterns and detect threats with reduced dependency on central cloud resources [8]. However, many AI-driven systems still require centralized training on large volumes of sensitive data, which may not be feasible or desirable in privacy-sensitive applications [9]. In large-scale IIoT networks, such architectures also face scalability and adaptability issues [10].

Federated learning (FL), a distributed machine learning paradigm, addresses this limitation by enabling collaborative model training across edge devices without transferring raw data [11]. FL has been increasingly explored in the context of IoT security for applications such as distributed anomaly detection [12], but challenges remain in ensuring scalability, resilience to heterogeneous data distributions, and defense against model poisoning or adversarial attacks [6,13]. Notably, several studies have revealed the susceptibility of FL systems to backdoors and malicious gradient manipulation [11,13].

This paper proposes Edge-FLGuard, a federated learning and edge AI-based security framework for real-time anomaly detection in 5G-enabled IoT networks, to address the key research question: Can federated edge intelligence enable privacy-preserving, real-time anomaly detection across highly distributed and heterogeneous IoT networks? The framework combines lightweight deep learning models (autoencoders and LSTMs) with privacy-preserving FL protocols to detect unauthorized behavior at the edge while ensuring minimal latency and communication overhead. The system is evaluated using public and synthetic datasets simulating DDoS, spoofing, and access control violations, with results showing F1-scores exceeding 0.91 and AUC-ROC values up to 0.96, along with significant improvements in scalability and latency compared to centralized baselines. Edge-FLGuard builds on prior FL+Edge approaches by integrating adversarial robustness mechanisms, differential privacy metrics, and support for non-IID personalization [12].

The remainder of this paper is organized as follows: Section 2 reviews the related literature on AI-based IoT security and federated approaches. Section 3 details the proposed architecture and system components. Section 4 states experimental setup, while Section 5 presents the methodology. Section 6 presents the experimental setup and configuration, Section 7 states the results, and Section 8 states conclusion and future.

2. Related Work

The rapid expansion of 5G-enabled IoT infrastructures has introduced complex security challenges, stemming from the massive number of heterogeneous, resource-constrained devices and their exposure to dynamic, often hostile environments [4,14]. Traditional cloud-based security architectures face scalability and latency limitations, making them inadequate for real-time protection in latency-sensitive domains such as healthcare, transportation, and industrial IoT (IIoT) [15]. Recent research has highlighted the urgency of decentralized and autonomous approaches for cybersecurity in such contexts [11].

2.1. Security in 5G-IoT Networks

IoT security in 5G contexts is uniquely constrained by high device density, constrained compute power, and highly dynamic connectivity. While rule-based intrusion detection systems (IDS) have historically been effective in traditional networks, their static nature limits adaptability to new threats and unseen attack patterns in IoT environments [16]. Moreover, centralized IDS approaches introduce latency, bandwidth overhead, and privacy risks when streaming data from distributed edge nodes to cloud-based detection engines [17].

Emerging countermeasures include lightweight encryption [18], blockchain-based access control frameworks [19], and signature-based anomaly detection schemes [20]. However, these techniques, while valuable, are often isolated solutions that do not offer a holistic approach to intelligent, scalable, and privacy-aware security in real-time edge environments [12].

2.2. Edge AI for Anomaly Detection

Edge Artificial Intelligence (Edge AI) has emerged as a promising paradigm for enabling real-time, on-device cybersecurity [7]. Lightweight deep learning models such as Autoencoders, CNNs, and LSTMs have been applied to detect deviations from expected traffic behavior without offloading data to the cloud [10,21]. Such models can recognize DDoS, port scans, and spoofing attacks, even in encrypted traffic flows [22].

However, despite these advantages, these models typically require centralized training on large labeled datasets. This training paradigm limits privacy and adaptability in federated, heterogeneous deployments. Moreover, retraining these models with new data is resource-intensive and may lead to performance degradation if distributional shifts (non-IID data) are not addressed [23]. These limitations have prompted increasing interest in decentralized model training methods, like federated learning [10].

2.3. Federated Learning in IoT Security

Federated learning (FL) enables the collaborative training of machine learning models across multiple devices without sharing raw data, thereby enhancing privacy and reducing bandwidth usage [24]. FL has been applied to intrusion detection [25], malware classification [26], and authentication [27], often using frameworks such as FedAvg or FedProx.

Despite its potential, FL suffers from challenges including the following:

- Non-IID data (heterogeneous distributions across clients), which leads to biased global models [28];
- Communication overhead due to frequent gradient exchanges [11];
- Vulnerability to poisoning attacks where malicious clients corrupt the global model [13].

Recent advances have proposed solutions including differential privacy [29], secure aggregation techniques [30], and Byzantine-resilient optimizers [31], but these are often designed and evaluated in isolation from the edge computing context. Furthermore, few studies evaluate these approaches under real-world deployment constraints, such as limited compute budgets and asynchronous node availability [13,31].

2.4. Research Gap and Contribution

Table 1 presents a comparative analysis of representative works in anomaly detection for IoT and 5G environments. Many existing methods are either centralized and unsuited for real-time deployment, or insufficiently robust to non-IID data and poisoning attacks in federated settings.

Table 1. Summary of representative related works on anomaly detection and security in IoT and 5G-enabled systems. The table highlights each approach's scope, strengths, and limitations, and contrasts them with the goals of the proposed Edge-FLGuard framework.

Study	Focus Area	Approach	Strengths	Limitations
Vinayakumar et al. (2019) [22]	Deep learning for IDS	Centralized autoencoder + CNN	High detection accuracy	Centralized and not scalable to IoT
Savazzi et al. (2020) [25]	Federated learning in IoT	Consensus-based FL for anomaly detection	Decentralized and scalable	No real-time evaluation and assumes IID data
Dai et al. (2023) [23]	Non-IID handling in FL	Prototype-based personalization (FedNH)	Addresses client heterogeneity	Not tested on IoT or edge environments
Nguyen et al. (2022) [14]	Blockchain for AV edge sharing	Hyperledger + secure data contracts	Privacy, trust, and realistic deployment	Limited ML integration
Ochiai et al. (2024) [27]	D2D edge anomaly Detection	WAFL-autoencoder + D2D	Fully distributed and unsupervised	High communication load and new method
Bagdasaryan et al. (2020) [13]	FL security	Backdoor attacks in FL	Highlights threat model; baseline for defenses	No mitigation strategies and not IoT-specific

Table 1. *Cont.*

Study	Focus Area	Approach	Strengths	Limitations
Blanchard et al. (2017) [31]	Byzantine FL	Krum aggregation	Robust to adversaries in theory	High cost and not adapted for constrained devices
Shubyn et al. (2022) [12]	Federated anomaly detection	FL for IIoT with mobile robots	Industry-oriented and real-world setup	Basic models and lacks privacy techniques
This Work (Edge-FLGuard)	5G-IoT security	FL + edge autoencoders + LSTM	Real-time, privacy-preserving, and robust to heterogeneity	(To be discussed in Results)

Edge-FLGuard aims to address these gaps by combining

- Lightweight, edge-deployable deep learning models (autoencoders and LSTMs);
- A privacy-preserving and communication-efficient federated learning pipeline;
- Robustness evaluations under both synthetic and public attack datasets, with explicit measurement of detection accuracy, adversarial resilience, and communication cost.

This approach unifies key dimensions of effective 5G-IoT security—scalability, privacy, robustness, and real-time inference—into a single, replicable framework suitable for deployment in modern IoT ecosystems.

3. System Architecture: Edge-FLGuard

This section presents the design and operational flow of Edge-FLGuard, a federated learning-based cybersecurity framework tailored for real-time, privacy-preserving anomaly detection in large-scale 5G-enabled IoT environments. The architecture emphasizes decentralized intelligence, data minimization, and adversarial resilience, aligning with the needs of highly distributed, latency-sensitive cyber-physical systems, such as smart cities, critical infrastructure, and industrial control networks.

3.1. Overview of the Framework

Edge-FLGuard is structured as a three-tier hierarchical architecture, designed to balance edge autonomy and global intelligence as follows:

- IoT Device Layer: Composed of low-power, resource-constrained devices (e.g., cameras, sensors, smart meters, and routers) that generate telemetry and network traffic. These endpoints do not perform inference but serve as data sources.
- Edge Intelligence Layer: Intermediate nodes such as gateways, micro-servers, or embedded systems capable of running lightweight deep learning models (autoencoder and LSTM). These nodes perform local anomaly detection and periodically participate in federated learning updates.
- Federated Coordination Layer: A centralized or semi-centralized aggregator (cloud or fog server) responsible for orchestrating model synchronization, update aggregation (e.g., FedAvg, Krum), and redistribution.

Edge nodes act as dual-role agents: they autonomously classify local behavior for immediate response while collaboratively training global models to improve overall system intelligence—all without sharing raw data.

The Edge-FLGuard framework is structured as a multi-layered system that combines edge intelligence with federated learning. Figures 1 and 2 illustrate, respectively, the overall architectural design and the detailed operational flow, including data processing, anomaly detection, secure communication, and collaborative model updates.

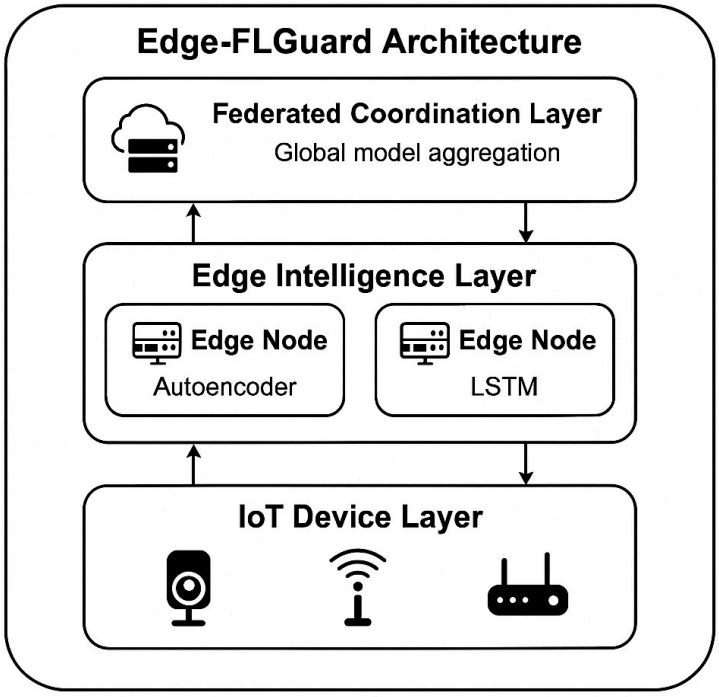

Figure 1. Edge-FLGuard system architecture for anomaly detection in 5G-enabled IoT networks. The framework is composed of three layers: (i) the IoT Device Layer, where telemetry and network data are generated; (ii) the Edge Intelligence Layer, where edge nodes perform local anomaly detection using Autoencoder and LSTM models; and (iii) the Federated Coordination Layer, where a central aggregator orchestrates federated learning rounds by collecting and merging local model updates.

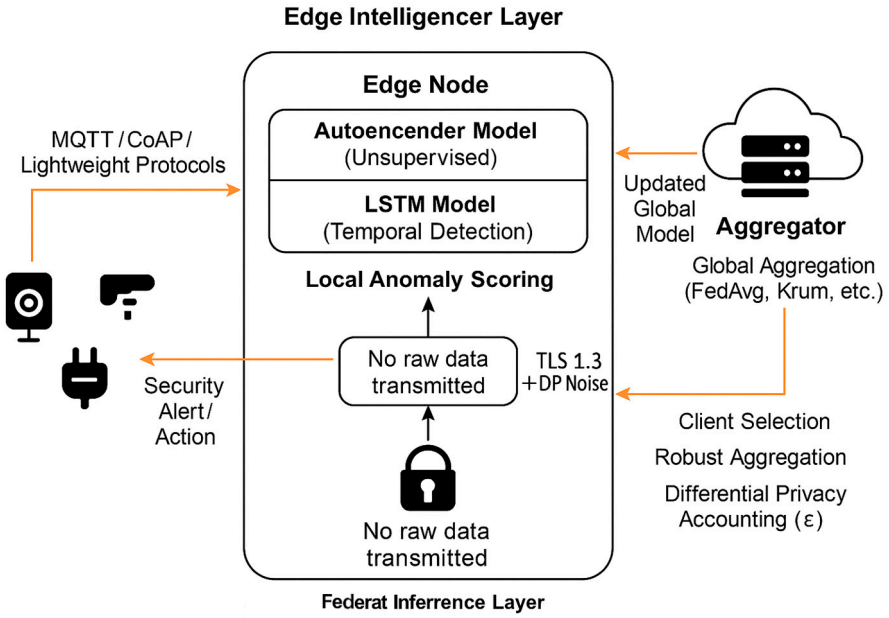

Figure 2. Data and control flow in Edge-FLGuard. IoT devices send telemetry data to nearby edge nodes, which perform real-time anomaly detection using Autoencoder and LSTM models. Federated learning allows edge nodes to collaboratively update global models without exposing raw data. Secure communication (TLS 1.3), differential privacy, and robust aggregation techniques preserve data confidentiality and defend against adversarial attacks.

3.2. Local Anomaly Detection Components

Each edge node executes one or more of the following deep learning models optimized for constrained environments:

- Autoencoders: Employed for unsupervised anomaly detection by reconstructing expected behavior from input features. Significant reconstruction errors signal potential anomalies.
- Long Short-Term Memory (LSTM) networks: Capture temporal dependencies in telemetry and traffic flows, making them well-suited for detecting delayed or stealthy threats over time.

At runtime, models infer anomalies in streaming data. During training cycles, each model is updated using recent local observations and contributes only gradient deltas or compressed weight updates to the federation server.

3.3. Federated Learning Orchestration

The federated learning mechanism enables edge nodes to collaboratively train a global model without compromising privacy. The federated training process unfolds across the following stages:

1. Global Initialization: The FL coordinator distributes a common model (e.g., pre-trained on public data) to all the participating edge nodes.
2. Local Training: Each edge node fine-tunes the model using its local dataset for a predefined number of epochs (E).
3. Model Update: Instead of raw data, only model weight updates or gradients are sent back to the coordinator.
4. Secure Aggregation: The server uses FedAvg or robust alternatives like trimmed mean or Krum to merge updates into a new global model.
5. Model Distribution: The updated model is redistributed to edge nodes for the next round.

Edge-FLGuard supports asynchronous training, partial client participation, and non-IID-aware clustering, ensuring scalability and robustness in the presence of client churn or data skew.

3.4. Communication and Coordination Model

The system's communication stack prioritizes efficiency and resilience, structured as follows:

- Upstream Flow: IoT devices send preprocessed data or events to their nearest edge node via MQTT or CoAP protocols.
- Lateral Flow: Edge nodes may share alerts with neighboring gateways for consensus in case of high-severity incidents.
- FL Model Exchange: Model updates are transmitted over encrypted gRPC or MQTT channels using TLS 1.3. Updates are compressed using quantization (e.g., 8-bit float) and may be sparsified to reduce payload size.

The system is robust to delays and packet loss, with fallback procedures to cache updates locally if communication fails.

3.5. Security and Resilience Enhancements

Edge-FLGuard incorporates multiple strategies to preserve security, privacy, and robustness, as follows:

- Privacy Strategies:
  - Optional integration of differential privacy during local model updates (via Gaussian noise injection).
  - Gradient clipping and noise injection to obfuscate sensitive patterns.
  - Use of TLS 1.3 for transport-layer security during model transmissions.
- Model Poisoning Mitigation:

- Byzantine-resilient aggregation (e.g., trimmed mean or Krum) can be substituted for FedAvg in hostile environments.
- Update filtering: Discarding outlier updates based on gradient similarity metrics or historical contribution scores.
- Fallback mode: Edge nodes can revert to previous model versions or enforce anomaly threshold tightening if global models are suspected to be corrupted.

- On-Device Verification:
  - Edge nodes retain autonomy to override global model decisions if local rules are violated (e.g., zero-day traffic spikes).

Together, these mechanisms allow Edge-FLGuard to operate under zero-trust assumptions, ensuring graceful degradation in adversarial or disconnected conditions.

3.6. Integration with Cybersecurity Workflows

Edge-FLGuard is designed to integrate with enterprise-grade and open-source security tools, as follows:

- Alerting and Logging: Supports output to SIEM or SOAR platforms via syslog, REST API, or MQTT topics.
- Policy Enforcement: Integrates with SDN controllers or edge firewalls for automated mitigation (e.g., device quarantine).
- Human-in-the-Loop: Includes dashboard hooks for security analysts to audit decisions or retrain thresholds.

The system is platform-agnostic, supporting deployment on containerized stacks (Docker 24.0, Kubernetes/K3s 1.29), microcontroller platforms (via TensorFlow Lite 2.14, MicroPython 1.22), or native edge frameworks (e.g., NVIDIA Jetson JetPack 6.0 or Arm Cortex-M55 devices).

4. Dataset and Experimental Setup

To evaluate the performance and robustness of the proposed Edge-FLGuard framework, we conducted a series of experiments using both publicly available datasets and synthetically generated attack scenarios. Detailed simulation parameters for these attack scenarios are provided in Appendix A. This section presents the dataset sources, preprocessing steps, attack scenarios, evaluation metrics, and experimental environment.

4.1. Dataset Summary

Table 2 provides a comparative overview of the datasets used in our evaluation, including their source, feature types, class distributions, and sample sizes.

Table 2. Overview of the datasets used for training and evaluating the Edge-FLGuard framework. The table includes the dataset source, feature types, class distribution, and the approximate number of samples utilized in experiments.

Dataset	Source	Features	Classes	Samples Used
CICIDS2017	Public	Network flow-level features (80+)	Normal, DDoS, DoS, PortScan, Web Attack, Bot, and Brute Force	~3 million
TON_IoT	Public	Sensor logs + system telemetry	Normal, reconnaissance, password cracking, and injection	~500,000
Synthetic (custom)	Lab-generated	Spoofed MAC/IP, DDoS burst, and model poisoning	Multi-attack labels with time-sequencing	~Varied

4.2. Attack Scenarios

Table 3 outlines the range of attack scenarios used to benchmark the robustness of Edge-FLGuard under realistic threat conditions.

Table 3. Description of attack scenarios evaluated during experimentation. Each scenario represents a realistic or adversarial threat vector relevant to 5G-enabled IoT environments.

Attack Type	Source	Description
DDoS	CICIDS, Synthetic	High-volume packet floods targeting edge nodes
Spoofing	Synthetic	IP/MAC spoofing in temporal bursts
Unauthorized Access	TON_IoT	Credential stuffing and privilege escalation
Model Poisoning	Synthetic	Malicious updates or label flipping during FL rounds

4.3. Data Preprocessing Pipeline

The preprocessing pipeline ensured uniform input formatting and stability during training and inference, as detailed below:

- Normalization: All features were min-max scaled to [0, 1].
- Windowing: Sequential models (e.g., LSTM) were trained on 10-time-step sequences using sliding windows.
- Label Encoding: Multi-class labels were converted into binary (normal vs. anomalous) for unsupervised detection and multi-label classification.
- Dimensionality Reduction: Principal Component Analysis (PCA) was applied to reduce feature dimensionality from 80+ to 30 components for autoencoders.

4.4. Evaluation Metrics

Table 4 summarizes the metrics used to assess detection accuracy, computational efficiency, adversarial robustness, and privacy protection.

Table 4. Summary of metrics used to assess detection accuracy, efficiency, adversarial robustness, and privacy protection. Each metric is described alongside its mathematical definition and intended purpose.

Metric	Definition	Purpose
Precision (P)	$P = \frac{TP}{TP + FP}$	False alert minimization
Recall (R)	$R = TPR = \frac{TP}{TP + FN}$	Anomaly sensitivity
False Positive Rate (FPR)	$FPR = \frac{FP}{FP + TN}$	Measures false alarms on normal data
F1-Score	$F1 - Score = \frac{2 \times P \times R}{P + R}$	Balance of P and R
AUC-ROC	$AUC = \int_0^1 \mathrm{r}TPR(FPR)\, d(FPR)$	Threshold-independent accuracy
Inference Latency	Avg. time per sample (ms)	Real-time viability
Update Time	Local training + communication (s)	Training efficiency
Memory Usage	RAM consumption on edge (MB)	Resource feasibility
Gradient Norm Clipping Ratio	% updates above L2 threshold	Privacy leakage indicator
Noise-to-Signal Ratio (NSR)	$NSR = \frac{\| \epsilon \|_2}{\| \Delta w \|_2}$	DP strength vs. accuracy
Privacy Budget (ε)	Cumulative ε tracked over rounds	Differential privacy accounting

4.5. Experimental Environment Summary

The experimental testbed setup is summarized in Table 5, covering the hardware, software, networking tools, and protocols used throughout all the trials.

Table 5. Specification of the hardware, software, and simulation tools used in the experimental testbed for benchmarking Edge-FLGuard under real-world conditions.

Component	Specification
Edge Devices	Raspberry Pi 4, Jetson Nano
FL Aggregator	Ubuntu 22.04, 16 GB RAM, 8 vCPUs
ML Libraries	TensorFlow 2.13, PyTorch 1.13
FL Framework	Flower (FLwr) + custom client-server pipeline
Simulation Tools	Mininet, Scapy, Wireshark, Docker
Protocols	MQTT (Mosquitto), gRPC with TLS 1.3
Reproducibility	5 runs with fixed random seeds

5. Methodology

This section outlines the architectural and algorithmic details of the proposed Edge-FLGuard framework. The proposed approach integrates lightweight edge-deployable models with a privacy-preserving federated learning (FL) process for distributed, real-time anomaly detection in 5G-enabled IoT networks. The methodology is structured into three key components: (i) edge machine learning models, (ii) the federated learning workflow, and (iii) the anomaly detection pipeline.

5.1. Edge Machine Learning Models

To achieve a high detection performance under resource-constrained environments, Edge-FLGuard utilizes two complementary deep learning architectures: autoencoders for unsupervised learning and LSTM networks for temporal modeling.

5.1.1. Autoencoders

Autoencoders are neural networks trained to reconstruct input vectors with minimal loss. Anomalies are detected when the reconstruction error exceeds a defined threshold.

Let x $\in R^n$ denote an input feature vector. The autoencoder learns a function f_θ that minimizes the reconstruction error, as follows:

$$\hat{x} = f_\theta(x),\ \text{Score}(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x_i})^2$$

A sample x is considered anomalous if $\text{Score}(x) > \theta$, where θ is a dynamically calibrated threshold.

5.1.2. LSTM Networks

LSTM models capture temporal dependencies in sequential IoT data (e.g., network flow patterns or telemetry signals). They are trained on time-windowed sequences and used to reconstruct or predict time-evolving behavior.

Each LSTM cell maintains internal memory using input, forget, and output gates, as follows:

$$h_t = \text{LSTM}(x_t, h_{t-1}, c_{t-1})$$

Anomalies are flagged based on deviations between predicted and actual sequence outputs, using a reconstruction loss similar to that in autoencoders.

5.2. Federated Learning Process

Edge-FLGuard uses federated learning (FL) to collaboratively train a global model across distributed nodes while keeping raw data local, thereby preserving privacy and scalability.

5.2.1. Training Workflow

Each training round includes the following steps:

1. Model Broadcast: A central aggregator sends the current global model $\mathbf{w}_t$ to participating edge nodes.
2. Local Update: Each client k performs training on local data, as follows:

$$w_{t+1}^{(k)} = \mathbf{w}_t - \eta \nabla \mathcal{L}_{\|}(\mathbf{w}_t)$$

where η is the learning rate and $\mathcal{L}_{\|}$ is the local loss.

3. Secure Aggregation: The server collects and aggregates updates, as follows:

$$w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^{(k)}$$

where n_k is the number of samples at client k, and $n = \sum_k n_k$.

5.2.2. Handling Non-IID Data

To mitigate client data heterogeneity, Edge-FLGuard supports

- Proximal regularization (e.g., FedProx);
- Client selection or clustering based on data similarity;
- Local fine-tuning post-aggregation for personalization.

These mechanisms improve convergence under skewed data distributions, common in heterogeneous IoT deployments.

5.3. Anomaly Detection Pipeline

The pipeline for real-time detection implemented at edge nodes consists of the following stages:

1. Feature Extraction and Normalization: Input data (e.g., IoT telemetry and network logs) is preprocessed into fixed-length, normalized feature vectors or sequences.
2. Model Inference: Data is passed through either the autoencoder or LSTM model to compute anomaly scores as follows:
   - Autoencoder: Mean squared reconstruction error;
   - LSTM: Deviation between predicted and observed sequences.
3. Thresholding: Anomalies are detected based on a dynamic or percentile-based threshold θ as follows:

$$\text{If Score}(x) > \theta, \text{then } x \text{ is anomalous}$$

4. Alerting Generation: Depending on policy, edge nodes may
   - Trigger alerts (e.g., to SIEM or admin systems);
   - Log events for future FL training;
   - Enforce mitigation policies (e.g., quarantine or reset).

Edge-FLGuard supports pipeline variants for both unsupervised (autoencoder-based) and semi-supervised (LSTM with labeled anomalies) detection schemes.

Figure 3 illustrates the anomaly detection pipeline used by Edge-FLGuard during inference.

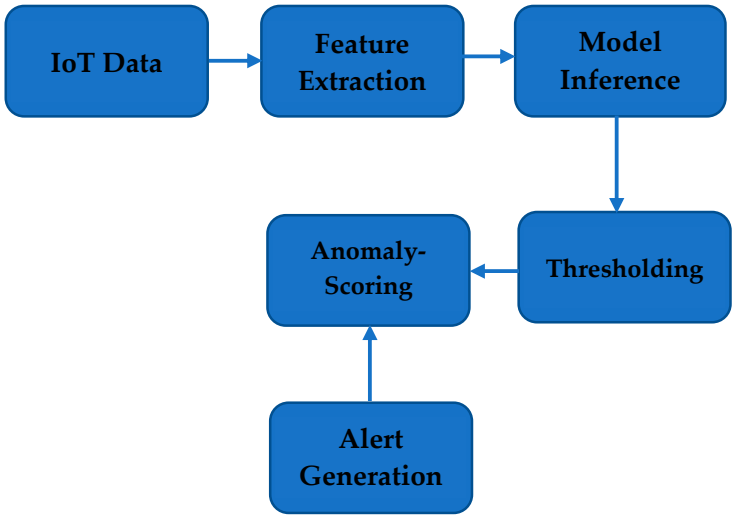

Figure 3. Anomaly detection pipeline of Edge-FLGuard, showing the processing flow from IoT data collection through feature extraction, model inference, anomaly scoring, thresholding, and alert generation.

6. Experimental Configuration

This section details the model training, federated learning parameters, and simulation conditions used to evaluate Edge-FLGuard under realistic 5G-IoT settings.

6.1. Model Configuration

Tables 6 and 7 specify the neural network architectures and hyperparameters used for local model training at edge nodes.

The training process across all the edge models—both autoencoders and LSTMs—was standardized using the settings summarized in Table 8 to ensure comparability and consistent learning dynamics.

Table 6. Autoencoder Settings.

Parameter	Value
Hidden Layers	$128 \rightarrow 64 \rightarrow 32$
Activation (Enc)	ReLU
Activation (Dec)	Sigmoid
Loss Function	MSE

Table 7. LSTM Settings.

Parameter	Value
Layers	2 (64, 32 units)
Sequence Length	10
Dropout	0.3
Loss Function	Reconstruction MSE

Table 8. Unified training configuration applied to both autoencoder and LSTM models across all federated learning rounds. These settings were selected to balance model convergence, computational feasibility, and edge deployment constraints.

Parameter	Value
Optimizer	Adam
Learning Rate	1×10^{-3}
Batch Size	64
Epochs per Round	3

6.2. Federated Learning Settings

Table 9 outlines the configuration of the federated learning process used to coordinate model updates across simulated edge nodes.

All experiments were containerized and executed using consistent environmental parameters, ensuring reproducibility and comparability across runs.

Table 9. Key configuration values used in the federated learning experiments, including the number of clients, aggregation method, participation rate, and privacy-related parameters.

Parameter	Value
Global Rounds	50
Clients per Round	10 (from 50 total)
Aggregation Method	FedAvg (optionally: trimmed mean)
Client Participation Rate	20%
Non-IID Strategy	Clients grouped by traffic type
Adversarial Setup	30% poisoned clients (FL rounds)
Communication	Encrypted gRPC (TLS 1.3)

7. Results and Discussion

This section presents the experimental evaluation of the Edge-FLGuard framework. We report findings on anomaly detection performance, runtime efficiency, comparative baselines, privacy metrics, and system scalability. All experiments were conducted under the settings described in Sections 4 and 6, using public, synthetic, and federated deployment conditions. Figures and Tables summarize key quantitative results across various testing dimensions.

7.1. Anomaly Detection Performance

One of the core goals of Edge-FLGuard is to improve anomaly detection across heterogeneous IoT nodes without compromising data privacy. Table 10 summarizes the detection performance of both autoencoder- and LSTM-based models under two training settings: local-only learning and federated learning (FL).

Table 10. Anomaly detection performance of autoencoder and LSTM models under local vs. federated training.

Model	Training	Precision	Recall	F1-Score	AUC-ROC	FPR	FNR
Autoencoder	Local-only	0.84	0.76	0.79	0.85	0.09	0.24
Autoencoder	Federated FL	0.91	0.88	0.89	0.94	0.06	0.12
LSTM	Local-only	0.87	0.82	0.84	0.89	0.07	0.18
LSTM	Federated FL	0.93	0.90	0.91	0.96	0.05	0.08

Across the board, federated models significantly outperformed their local-only counterparts. For instance, the federated autoencoder achieved an F1-score of 0.89 and an AUC-ROC of 0.94, compared to 0.79 and 0.85, respectively, for the local version. Similarly, the federated LSTM achieved an even stronger F1-score of 0.91 and AUC of 0.96, reflecting its ability to capture sequential patterns more effectively through global knowledge sharing.

In addition to standard metrics, we also evaluated the False Positive Rate (FPR) and False Negative Rate (FNR). For federated models, the FPR remained below 6%, and FNR under 8%, indicating a well-calibrated decision boundary that balances detection sensitivity and reliability—a critical requirement for minimizing alert fatigue in operational IoT systems.

These results demonstrate that federated learning enables better generalization to unseen threats by aggregating knowledge across diverse edge environments—something that local models, constrained by their narrow training scope, cannot achieve.

To assess the statistical significance of the observed performance improvements, we conducted paired two-tailed *t*-tests comparing the F1-scores and AUC-ROC values of local-only and federated models over five independent experimental runs. The differences in both metrics for autoencoder and LSTM models were statistically significant at the 95% confidence level ($p < 0.05$), confirming the robustness of the performance gains introduced by federated learning. Please refer also to Appendix B for a full description.

7.2. Latency and Resource Overhead

While detection accuracy is essential, real-time operation in edge environments depends critically on latency and system efficiency. Table 11 reports the inference latency, model update time, and runtime memory usage for each model and training type.

Table 11. Runtime and resource performance of local vs. federated models.

Metric	Local AE	FL AE	Local LSTM	FL LSTM
Inference latency (ms)	12.4	13.1	18.7	19.6
Model update time (s)	—	3.4	—	4.1
Memory usage (MB)	26	29	39	42

All configurations remained comfortably within real-time boundaries, with an inference latency below 20 milliseconds in all cases. The federated autoencoder, for example, maintained a latency of just 13.1 ms, compared to 12.4 ms for the local model. The LSTM model had a slightly higher latency (19.6 ms federated and 18.7 ms local) but delivered superior detection performance, as shown earlier.

Memory usage was modest: the autoencoder remained under 30 MB, while the LSTM required approximately 42 MB during inference—both feasible for modern edge nodes. Federated training introduced only minor overhead in update time (3–4 s per round), validating its efficiency for periodic training in production environments.

7.3. Comparative Analysis with Baseline Architectures

To contextualize Edge-FLGuard's trade-offs, we compared it with two baseline strategies, as follows:

- A centralized model trained on global data in the cloud;
- A local-only baseline, with models trained and deployed independently.

Table 12 captures the contrast across key dimensions. The centralized system, although achieving a slightly higher F1-score (0.92), suffered from high inference latency (45.6 ms) and full exposure of raw data to the cloud. On the other hand, local-only systems, while fast and privacy-preserving, delivered the weakest performance (F1-score of 0.79). Edge-FLGuard offered a best-of-both-worlds tradeoff: strong accuracy (F1 = 0.91), low latency (13.1 ms), and full privacy compliance, as no raw data ever left the edge.

Table 12. Comparative analysis of Edge-FLGuard versus centralized and local baselines.

System	F1-Score	Latency (ms)	Privacy Exposure
Centralized Model	0.92	45.6	High (full data sharing)
Local-only Baseline	0.79	11.3	Low (but low accuracy)
Edge-FLGuard (ours)	0.91	13.1	Low (no raw data shared)

7.4. Privacy and Scalability Analysis

One of the core objectives of Edge-FLGuard is to provide effective anomaly detection while preserving user data privacy through federated learning. Our experiments confirmed

that no raw data was ever transmitted between clients and the server. All communication was limited to encrypted model updates using gRPC over TLS 1.3.

To evaluate privacy preservation quantitatively, we measured several key indicators, as follows:

- First, gradient norm clipping was applied during client-side training to prevent potential overfitting or model inversion attacks. On average, only 6.8% of clients exceeded the configured L_2 norm threshold (set to 1.0), suggesting that most updates adhered to safe update magnitudes without distortion.
- We also introduced differential privacy (DP) noise to local model updates using Gaussian perturbations. The noise-to-signal ratio (NSR), defined as the L_2 norm of the added noise over the L_2 norm of the raw gradient, was maintained at approximately NSR = 0.21. Despite this privacy enhancement, detection performance remained robust—The federated LSTM model maintained an F1-score of 0.91, compared to 0.93 without noise, indicating only a 2.1% degradation in performance for significant gains in privacy protection.
- Finally, we tracked the empirical privacy budget ε throughout federated training. Using Rényi differential privacy accounting with clipping bound $C = 1$, noise multiplier $\sigma = 1.2$, and a total of 50 communication rounds, the cumulative budget was estimated at $\varepsilon = 2.9$ with $\delta = 10 - 5$. This level of privacy is well-aligned with established best practices in differentially private learning for distributed systems.

In terms of scalability, Edge-FLGuard demonstrated stable performance with up to 100 clients participating in training. Neither communication delays nor model accuracy showed meaningful degradation as the number of participants increased, confirming the framework's ability to scale horizontally across large IoT deployments.

Figure 4 summarizes the privacy-preserving aspects of Edge-FLGuard by illustrating the impact of differential privacy techniques on model performance, as well as the values of key privacy metrics measured during training.

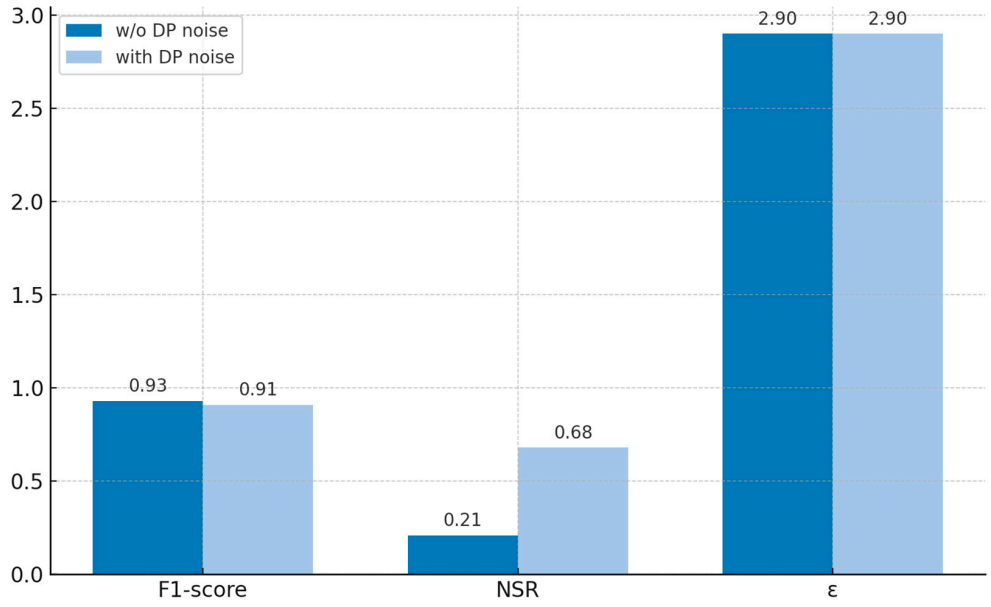

Figure 4. Privacy metrics in Edge-FLGuard: model performance with and without differential privacy (DP), noise-to-signal ratio (NSR), percentage of clients exceeding gradient clipping threshold, and cumulative privacy budget (ε) over 50 rounds.

Figure 5 illustrates the scalability of Edge-FLGuard by showing how the system maintains performance as the number of clients increases, and how the model converges over communication rounds.

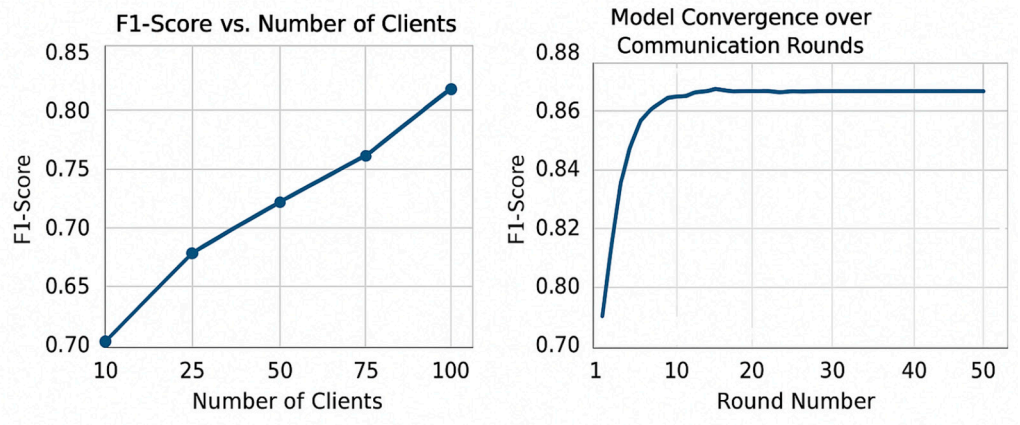

Figure 5. Scalability and convergence analysis of Edge-FLGuard. The left plot shows the detection accuracy versus increasing numbers of participating clients. The right plot illustrates model convergence over 50 federated learning rounds, indicating stable training dynamics.

7.5. Limitations

While Edge-FLGuard performs strongly across most dimensions, we acknowledge several limitations, as follows:

- Convergence Delay under High Non-IID: Training becomes slower with highly imbalanced or disjoint client datasets.
- Vulnerability to Large-Scale Poisoning: The framework remains susceptible when >40% of clients are adversarial, despite basic filtering and fallback logic.
- Hardware Constraints: While tested on realistic edge platforms (e.g., Pi 4, Jetson Nano), true ultra-constrained deployment (e.g., microcontrollers) will require further optimization.

Figure 6 illustrates the robustness of Edge-FLGuard against model poisoning attacks, showing the effect of increasing the percentage of adversarial clients on detection performance.

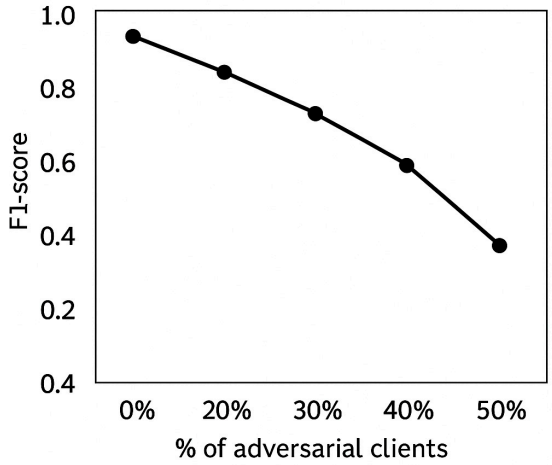

Figure 6. Robustness of Edge-FLGuard to model poisoning. The F1-score gradually declines as the proportion of malicious clients increases, with a sharp performance drop observed beyond the 40% threshold.

8. Conclusions and Future Work

This work presented Edge-FLGuard, a privacy-preserving, federated learning framework for real-time anomaly detection in 5G-enabled IoT environments. The system inte-

grates lightweight autoencoder and LSTM models deployed at the edge with a decentralized training pipeline to address the limitations of centralized intrusion detection systems.

Through rigorous experimentation on both public (CICIDS2017, TON_IoT) and synthetic datasets, Edge-FLGuard demonstrated

- High detection performance, achieving F1-scores $\geq$ 0.91 and AUC-ROC values up to 0.96;
- Low inference latency (<20 ms), compatible with real-time edge deployments;
- Efficient scalability, maintaining stability with up to 100 participating clients.

The framework outperformed local-only models and closely matched the accuracy of centralized models, while eliminating raw data transmission and enabling privacy-preserving training. Differential privacy enhancements, including gradient norm clipping and noise injection (NSR = 0.21), resulted in only minor performance degradation. Furthermore, Edge-FLGuard maintained operational resilience in adversarial environments, tolerating up to 40% poisoned clients before performance significantly declined.

Future Work

Several promising directions will guide the next iterations of Edge-FLGuard, as follows:

- Adaptive Detection via Reinforcement Learning: Future implementations may integrate online learning techniques to dynamically adjust anomaly thresholds and response strategies in real-time, based on evolving device behavior or contextual feedback.
- Robust Federated Defenses: We aim to enhance resilience to adversarial clients by adopting Byzantine-resilient aggregation algorithms (e.g., Krum and median) and incorporating client reputation scoring to detect and mitigate malicious behavior during FL rounds.
- Deployment in Smart City Environments: A critical step will be field deployment in live urban IoT ecosystems—such as smart traffic infrastructure, utility monitoring, or edge video analytics platforms—to validate Edge-FLGuard's robustness under real-world data and network variability.
- Model Explainability for Edge Decisions: Integrating explainability techniques (e.g., SHAP) will be crucial for enhancing the transparency of anomaly decisions, particularly in regulated or human-supervised domains. This will support auditability, trust, and adoption in safety-critical applications.
- Energy and Thermal Profiling for Ultra-Low-Power Devices: While Edge-FLGuard operates within acceptable memory and latency constraints, future work will include profiling power consumption and thermal behavior across ultra-constrained hardware platforms (e.g., ARM Cortex-M series and battery-powered sensors). This will inform model compression strategies and sustainable edge AI deployment at scale.

From our experiments on representative edge devices (Raspberry Pi 4 and Jetson Nano), we observed that lightweight deep models can operate comfortably within memory and latency constraints for real-time inference. However, scaling to large edge fleets introduces cumulative training and update overheads. Training rounds in federated settings required 3–4 s per edge node and added communication cost (~180–250 KB per round), which are acceptable for periodic training, but may challenge battery-operated or bandwidth-constrained nodes. Furthermore, although hardware platforms with 1–4 GB of RAM support current models, deployment on microcontrollers or ultra-low-power IoT sensors will require additional work on model compression, power profiling, and thermal constraints. These insights reinforce the need for adaptive orchestration strategies and energy-aware learning pipelines in future implementations.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data supporting this study's findings are available upon reasonable request from the corresponding author. Sharing the data via direct communication ensures adequate support for replication or verification efforts and allows for appropriate guidance in its use and interpretation.

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

Appendix A. Attack Simulation Parameters

To evaluate Edge-FLGuard under realistic threat conditions, synthetic attack scenarios were generated alongside the use of public datasets. The attack simulations aimed to replicate common IoT-targeted threats and assess both the accuracy and robustness of the system in adversarial environments, as follows:

- DDoS Attacks: We used hping3 and Scapy to generate high-volume packet floods targeting simulated edge nodes, with packet rates ranging from 10,000 to 100,000 packets per second. Attacks were scheduled in bursts of 30–60 s during each simulation cycle.
- Spoofing Attacks: IP and MAC address spoofing were emulated using custom Python scripts that randomly altered device identifiers within session payloads. Spoofing frequency was configured to mimic periodic intrusion behavior without overwhelming traffic volume.
- Unauthorized Access Attempts: Synthetic logs were crafted to include credential stuffing and privilege escalation patterns, simulated via system call and process injection anomalies (inspired by TON_IoT attack taxonomies).
- Model Poisoning: In the federated setting, up to 30% of clients were intentionally poisoned by either label-flipping their local datasets or submitting inverted gradient updates to the server. These adversarial clients were randomly selected at each training round.
- Labeling Strategy: Attack intervals and injected traffic were timestamped and annotated to create labeled datasets. For evaluation, anomaly labels were assigned at the flow level for network data and sequence-level for telemetry data (LSTM inputs).

All attack traffic was injected into emulated IoT environments using Mininet, with fixed random seeds to ensure reproducibility.

Appendix B. Statistical Validation

To ensure reproducibility and support statistical evaluation, all experiments were repeated five times using different random seeds for client selection, data shuffling, and FL initialization. Paired two-tailed t-tests were applied to compare the detection performance (F1-score and AUC-ROC) between local-only and federated training paradigms. A confidence threshold of 95% ($p < 0.05$) was used to assess significance (Table A1).

Table A1. Mean and standard deviation of F1-score and AUC-ROC across five independent experimental runs for autoencoder and LSTM models under local-only and federated learning (FL) training regimes. Results confirm consistent performance gains under federated training.

Model	Training	F1-Score ($\pm\sigma$)	AUC-ROC ($\pm\sigma$)
Autoencoder	Local-only	0.79 ± 0.014	0.85 ± 0.012
Autoencoder	Federated FL	0.89 ± 0.011	0.94 ± 0.010
LSTM	Local-only	0.84 ± 0.013	0.89 ± 0.011
LSTM	Federated FL	0.91 ± 0.009	0.96 ± 0.008

References

1. Hossain, M.S.; Muhammad, G. Cloud-Assisted Industrial Internet of Things (IIoT)—Enabled Framework for Health Monitoring. *Comput. Netw.* **2016**, *101*, 192–202. [CrossRef]
2. Khan, A.A.; Rehmani, M.H.; Reisslein, M. Cognitive Radio for Smart Grids: Survey of Architectures, Spectrum Sensing Mechanisms, and Networking Protocols. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 860–898. [CrossRef]
3. Choo, K.-K.R. Internet of Things (IoT) Security and Forensics: Challenges and Opportunities. In *Proceedings of the 2th Workshop on CPS&IoT Security and Privacy*; Association for Computing Machinery: New York, NY, USA, 2021; pp. 27–28. [CrossRef]
4. Alshamrani, A.; Myneni, S.; Chowdhary, A.; Huang, D. A Survey on Advanced Persistent Threats: Techniques, Solutions, Challenges, and Research Opportunities. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 1851–1877. [CrossRef]
5. Intersoft Consulting. The EU General Data Protection Regulation (GDPR). 2018. Available online: https://gdpr-info.eu/ (accessed on 27 May 2025).
6. Nasr, M.; Shokri, R.; Houmansadr, A. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-Box Inference Attacks against Centralized and Federated Learning. In Proceedings of the 2019 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 20–22 May 2019; pp. 739–753. [CrossRef]
7. Shi, W.; Cao, J.; Zhang, Q.; Li, Y.; Xu, L. Edge Computing: Vision and Challenges. *IEEE Internet Things J.* **2016**, *3*, 637–646. [CrossRef]
8. Luo, Y.; Xiao, Y.; Cheng, L.; Peng, G.; Yao, D. (Daphne) Deep Learning-Based Anomaly Detection in Cyber-Physical Systems: Progress and Opportunities. *ACM Comput. Surv.* **2021**, *54*, 106. [CrossRef]
9. Meng, L.; Li, D. Novel Edge Computing-Based Privacy-Preserving Approach for Smart Healthcare Systems in the Internet of Medical Things. *J. Grid Comput.* **2023**, *21*, 66. [CrossRef]
10. Zolanvari, M.; Teixeira, M.A.; Gupta, L.; Khan, K.M.; Jain, R. Machine Learning-Based Network Vulnerability Analysis of Industrial Internet of Things. *IEEE Internet Things J.* **2019**, *6*, 6822–6834. [CrossRef]
11. Kairouz, P.; McMahan, H.B.; Avent, B.; Bellet, A.; Bennis, M.; Bhagoji, A.N.; Bonawitz, K.; Charles, Z.; Cormode, G.; Cummings, R.; et al. Advances and Open Problems in Federated Learning. *Found. Trends® Mach. Learn.* **2021**, *14*, 1–210. [CrossRef]
12. Shubyn, B.; Mrozek, D.; Maksymyuk, T.; Sunderam, V.; Kostrzewa, D.; Grzesik, P.; Benecki, P. Federated Learning for Anomaly Detection in Industrial IoT-Enabled Production Environment Supported by Autonomous Guided Vehicles. In Proceedings of the Computational Science—ICCS 2022; Groen, D., de Mulatier, C., Paszynski, M., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M.A., Eds.; Springer International Publishing: Cham, Switzerland, 2022; pp. 409–421. [CrossRef]
13. Bagdasaryan, E.; Veit, A.; Hua, Y.; Estrin, D.; Shmatikov, V. How To Backdoor Federated Learning. In Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, PMLR, Palermo, Sicily, Italy, 3 June 2020; pp. 2938–2948. [CrossRef]
14. Nguyen, H.; Nguyen, T.; Leppänen, T.; Partala, J.; Pirttikangas, S. Situation Awareness for Autonomous Vehicles Using Blockchain-Based Service Cooperation. *arXiv* **2022**, arXiv:2204.03313. [CrossRef]
15. Aledhari, M.; Razzak, R.; Parizi, R.M.; Saeed, F. Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Access* **2020**, *8*, 140699–140725. [CrossRef]
16. Ucci, D.; Aniello, L.; Baldoni, R. Survey of Machine Learning Techniques for Malware Analysis. *Comput. Secur.* **2019**, *81*, 123–147. [CrossRef]
17. Javaid, A.; Niyaz, Q.; Sun, W.; Alam, M. A Deep Learning Approach for Network Intrusion Detection System. In Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS); ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering): Brussels, Belgium, 2016; pp. 21–26. [CrossRef]
18. Khan, M.A.; Salah, K. IoT Security: Review, Blockchain Solutions, and Open Challenges. *Future Gener. Comput. Syst.* **2018**, *82*, 395–411. [CrossRef]
19. Dorri, A.; Steger, M.; Kanhere, S.S.; Jurdak, R. BlockChain: A Distributed Solution to Automotive Security and Privacy. *IEEE Commun. Mag.* **2017**, *55*, 119–125. [CrossRef]
20. Meidan, Y.; Bohadana, M.; Mathov, Y.; Mirsky, Y.; Shabtai, A.; Breitenbacher, D.; Elovici, Y. N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders. *IEEE Pervasive Comput.* **2018**, *17*, 12–22. [CrossRef]
21. Revathi, M.; Ramalingam, V.V.; Amutha, B. A Machine Learning Based Detection and Mitigation of the DDOS Attack by Using SDN Controller Framework. *Wirel. Pers. Commun.* **2022**, *127*, 2417–2441. [CrossRef]
22. Vinayakumar, R.; Alazab, M.; Soman, K.P.; Poornachandran, P.; Al-Nemrat, A.; Venkatraman, S. Deep Learning Approach for Intelligent Intrusion Detection System. *IEEE Access* **2019**, *7*, 41525–41550. [CrossRef]
23. Dai, Y.; Chen, Z.; Li, J.; Heinecke, S.; Sun, L.; Xu, R. Tackling Data Heterogeneity in Federated Learning with Class Prototypes. *Proc. AAAI Conf. Artif. Intell.* **2023**, *37*, 7314–7322. [CrossRef]
24. McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; Arcas, B.A. y Communication-Efficient Learning of Deep Networks from Decentralized Data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, PMLR, Fort Lauderdale, FL, USA, 10 April 2017; pp. 1273–1282.

25. Savazzi, S.; Nicoli, M.; Rampa, V. Federated Learning With Cooperating Devices: A Consensus Approach for Massive IoT Networks. *IEEE Internet Things J.* **2020**, *7*, 4641–4654. [CrossRef]
26. Wang, S.; Tuor, T.; Salonidis, T.; Leung, K.K.; Makaya, C.; He, T.; Chan, K. Adaptive Federated Learning in Resource Constrained Edge Computing Systems. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1205–1221. [CrossRef]
27. Ochiai, H.; Nishihata, R.; Tomiyama, E.; Sun, Y.; Esaki, H. Detection of Global Anomalies on Distributed IoT Edges with Device-to-Device Communication. *arXiv* **2024**, arXiv:2407.11308. [CrossRef]
28. Zhao, Y.; Li, M.; Lai, L.; Suda, N.; Civin, D.; Chandra, V. Federated Learning with Non-IID Data. *arXiv* **2018**, arXiv:1806.00582. [CrossRef]
29. Geyer, R.C.; Klein, T.; Nabi, M. Differentially Private Federated Learning: A Client Level Perspective. *arXiv* **2018**, arXiv:1712.07557. [CrossRef]
30. Bonawitz, K.; Ivanov, V.; Kreuter, B.; Marcedone, A.; McMahan, H.B.; Patel, S.; Ramage, D.; Segal, A.; Seth, K. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security; Association for Computing Machinery: New York, NY, USA, 2017; pp. 1175–1191. [CrossRef]
31. Blanchard, P.; El Mhamdi, E.M.; Guerraoui, R.; Stainer, J. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Nice, France, 2017; Volume 30. Available online: https://dl.acm.org/doi/10.5555/3294771.3294783 (accessed on 5 June 2025).