AI-Based Embedded Framework for Cyber-Attack Detection Through Signal Processing and Anomaly Analysis

Sebastian-Alexandru Drăgușin *, Robert-Nicolae Boștinaru, Nicu Bizon *, Gabriel-Vasile Iana

Posted Date: 24 December 2025

doi: 10.20944/preprints202512.2225.v1

Keywords: embedded systems; cyber-security; artificial intelligence; cyber-threat detection; anomaly detection; signal processing; feature extraction

 doi:10.20944/preprints202512.2225.v1

Article

AI-Based Embedded Framework for Cyber-Attack Detection Through Signal Processing and Anomaly Analysis

Sebastian-Alexandru Drăguşin [1,*], Robert-Nicolae Boştinaru [1], Nicu Bizon [2,*] and Gabriel-Vasile Iana [2,3]

[1] Doctoral School, National University of Science and Technology POLITEHNICA Bucharest, 060042 Bucharest, Romania

[2] Faculty of Electronics, Communication and Computers, National University of Science and Technology POLITEHNICA Bucharest, Pitești University Centre, 110040 Piteşti, Romania

[3] Power Electronics R&D Department, Mira Technologies Group, Bucharest, Romania, Research & Development Centre, 164 Ciorogârlei Street, Joița, 087151 Giurgiu, Romania

* Correspondence: dragusin.sebi@yahoo.com (S.-A.D); nicu.bizon1402@upb.ro (N.B.)

Abstract

This paper proposes an applied framework for cyberattack and anomaly detection in resource-constrained embedded/IoT environments by combining signal-processing feature construction with supervised and unsupervised AI (Artificial Intelligence) models. The workflow covers dataset preparation and normalization, correlation-driven feature analysis, and compact representations via PCA (Principal Component Analysis), followed by classification and anomaly scoring. In addition to the original UNSW-NB15 (University of New South Wales—Network-Based Dataset 2015) traffic features, Fourier-domain descriptors, wavelet-domain descriptors, and Kalman-based smoothing/innovation features are considered to improve robustness under variability and measurement noise. Detection performance is assessed using classical and ensemble learning methods (SVM (Support Vector Machines), RF (Random Forest), XGBoost (Extreme Gradient Boosting), LightGBM (Light Gradient Boosting Machine)), unsupervised baselines (K-Means and DBSCAN (Density-Based Spatial Clustering of Applications with Noise)), and DL (Deep-Learning) anomaly detectors based on Autoencoder reconstruction and GAN (Generative Adversarial Network)-based scoring. Experimental results on UNSW-NB15 indicate that ensemble-based models provide the strongest overall detection performance, while the signal-processing augmentation and PCA-based compactness support efficient deployment in embedded contexts. The findings confirm that integrating lightweight signal processing with AI-driven models enables effective and adaptable identification of malicious network traffic supporting deployment-oriented embedded cybersecurity and motivating future real-time validation on edge hardware.

Keywords: embedded systems; cyber-security; artificial intelligence; cyber-threat detection; anomaly detection; signal processing; feature extraction

1. Introduction

In recent years, the integration of embedded systems into industrial, automotive, and IoT (Internet of Things) infrastructures has led to an exponential increase in both functionality and exposure to cyber threats. The growing interconnectivity of devices has transformed embedded platforms from isolated controllers into networked computing nodes that process sensitive information and operate critical functions. This evolution has made them a prime target for sophisticated cyberattacks capable of exploiting hardware, firmware, and software vulnerabilities [1,2].

 doi:10.20944/preprints202512.2225.v1

Traditional security mechanisms, often designed for desktop or cloud environments, are unsuitable for resource-constrained embedded systems due to their limited computational capacity, memory, and energy budgets. Consequently, new methodologies are required to ensure real-time and adaptive protection, leveraging intelligent algorithms that can detect and predict anomalous behavior. In this context, the combination of signal processing techniques with AI and ML (Machine Learning) models has proven to be a promising solution for the early detection of cyber-threats [3].

Building upon our previous theoretical study [4], in which the authors presented a comprehensive literature review of cyber-attack types and anomaly detection mechanisms, the present paper focuses on the practical implementation and evaluation of an AI-based framework for embedded cybersecurity. This new contribution aims to validate, through experimental analysis, the efficiency of combining signal processing and AI algorithms for the identification of anomalies in embedded environments.

Recent research on cyber-attack detection for embedded and edge environments has produced strong results, yet many studies evaluate supervised detection or unsupervised anomaly analysis as separate, non-comparable tracks, and often under protocols that do not explicitly quantify robustness, uncertainty, or deployment feasibility. The present work addresses this gap by proposing a compact end-to-end workflow that unifies signal-level feature construction with learning-based detection under a controlled and reproducible benchmark setting. The contribution is reinforced through consistent protocol definition, multi-metric reporting (including per-class analysis), statistical uncertainty quantification, perturbation-based robustness assessment, and edge-oriented profiling, enabling a coherent comparison and a deployment-relevant characterization of the resulting framework.

The remainder of this paper is structured as follows. Section 2 reviews recent literature on AI-enabled intrusion and anomaly detection, with emphasis on signal-level feature construction and resource-aware design. Section 3 presents the methodology, detailing the dataset protocol, window-based representation, feature engineering (including spectral descriptors), learning configurations, and the evaluation strategy (metrics, statistical uncertainty, robustness tests, and deployment profiling). Section 4 reports the implementation and experimental results, covering supervised detection performance (binary and multi-class), robustness under controlled perturbations, and embedded feasibility indicators. Section 5 concludes the paper by summarizing the main findings and outlining targeted directions for future research.

2. Literature Review

The growing complexity of cyberattacks targeting embedded devices has motivated researchers to explore AI-driven anomaly detection models capable of processing large volumes of heterogeneous data. In this context, several studies have proposed frameworks integrating signal analysis, feature extraction, and deep learning methods to improve the detection accuracy of intelligent embedded systems.

The following review examines representative studies that have applied artificial intelligence and signal processing techniques to anomaly detection, with emphasis on model architectures, experimental validation, and comparative performance across benchmark datasets.

Trilles et al. [5] map anomaly detection for AIoT (Artificial Intelligence of Things)/TinyML (Tiny Machine Learning) on MCUs (Microcontroller Units) across 18 studies (2021–2023), summarizing models, data, metrics, and edge platforms. CNN (Convolutional Neural Network) is most common (18.8%), followed by Autoencoders, LSTM (Long Short-Term Memory), DNN (Deep Neural Network), Isolation Forest, and GMM (Gaussian Mixture Model) (13.6% each), mainly on IMU (Inertial Measurement Unit)/image/temperature data and typically deployed on Raspberry Pi, then Arduino Nano 33 BLE and ESP32 using TensorFlow/TFLite with Accuracy/F1. The review highlights TinyML benefits (low latency, privacy, cost) and gaps in power reporting, HW (Hardware)/SW (Software) heterogeneity, limited multi-tier architectures, and scarce LoRaWAN (Long Range Wide Area Network)/5G support.

 doi:10.20944/preprints202512.2225.v1

Adhikari et al. [6] survey IoT anomaly detection across layered architectures and cloud/fog/edge paradigms, covering both traditional methods (entropy/KLD (Kullback–Leibler Divergence), graph/spectral, blockchain) and ML (Machine Learning)/DL (Deep Learning) approaches (AE (Autoencoders), such as RNN (Recurrent Neural Networks), RBM (Restricted Boltzmann Machines), CNN (Convolutional Neural Network), GAN (Generative Adversarial Network), and RL (Reinforcement Learning) ensembles. The review summarizes commonly used datasets (KDD'99 (Knowledge Discovery and Data Mining Cup 1999), NSL-KDD (Non-Redundant Standard Learning KDD), UNSW-NB15, Bot-IoT, CICIDS2017 (Canadian Institute for Cybersecurity Intrusion Detection System 2017), ADFA (Australian Defence Force Academy Intrusion Detection Dataset)) and metrics (Precision, DR (Detection Rate), FPR (False Positive Rate), ROC-AUC (Receiver Operating Characteristic–Area Under Curve), F1-score). Key gaps include computational complexity, privacy/robustness, interpretability, and limited benchmark standardization, motivating XAI (Explainable Artificial Intelligence), Edge AI, SDN (Software-Defined Networking), self-/weak-supervision, and Transformer-based directions.

Zhang et al. [7] review DL (Deep Learning) for IDS (Intrusion Detection Systems), attributing performance bottlenecks to limited temporal modeling and class imbalance. CNN (Convolutional Neural Network) is contrasted with RNN (Recurrent Neural Network), LSTM (Long Short-Term Memory), and GRU (Gated Recurrent Units), while hybrid CNN-LSTM/CNN-GRU architectures are highlighted on CIC-IDS2017 and UNSW-NB15. Imbalance mitigation via SMOTE (Synthetic Minority Over-sampling Technique) and GAN (Generative Adversarial Network)-based augmentation is discussed, with risks such as overfitting and mode collapse. Reported results show high Accuracy/DR (Detection Rate) for majority traffic but weak detection of rare attacks (U2R (User-to-Root), R2L (Remote-to-Local)), motivating cost-sensitive losses, attention/Transformers, and edge-oriented compression (pruning/quantization) with XAI (Explainable Artificial Intelligence).

DeMedeiros et al. [8] survey AI-based AD (Anomaly Detection) for IoT/IIoT (Industrial Internet of Things) and sensor networks (2019–2022), highlighting anomaly types (point, collective/windowed, continuous) and sources (attacks, faults, environmental shifts). The review spans ML (Machine Learning) and DL (Deep Learning) approaches and notes strong performance from attention-based neural models and GNN (Graph Neural Network) formulations that capture inter-device dependencies. An evaluation is summarized via precision/recall/F1-score and ROC (Receiver Operating Characteristic) with TPR (True Positive Rate) vs. FPR (False Positive Rate), while emphasizing limited comparability due to dataset/task heterogeneity. The paper stresses a robustness–specialization trade-off and motivates future work on explainability, edge energy efficiency, and graph-/attention-aware detectors for time-dependent sensor data.

Morshedi and Matinkhah [9] survey DL anomaly detection in IoT, comparing CNN (Convolutional Neural Network), LSTM (Long Short-Term Memory), AE (Autoencoders), GAN (Generative Adversarial Network), GNN (Graph Neural Network), and Transformer-based models on datasets such as CICIDS2017, Bot-IoT, NSL-KDD, and TON-IoT (Telemetry Operating Network IoT). The review highlights DL benefits for learning features from high-dimensional sequential traffic, but reports challenges in scarce labeled data, edge resource limits, adversarial/DDoS (Distributed Denial of Service)/MITM (Man-in-the-Middle) robustness, and interpretability. Proposed directions include FL (Federated Learning), pruning/quantization, GAN/VAE (Variational Autoencoders) augmentation, cross-dataset validation, and XAI (Explainable Artificial Intelligence).

Reis and Serôdio [10] propose an Edge AI framework for smart-home anomaly detection that combines IF (Isolation Forest) with an LSTM-AE (Long Short-Term Memory Autoencoder) to balance accuracy and resource use. On synthetic and real sensor streams (temperature, motion, energy), LSTM-AE reaches up to 93.6% accuracy with higher recall, while IF provides faster, lower-power inference; the hybrid achieves sub-50 ms on-device latency on Raspberry Pi and NVIDIA Jetson Nano. Quantization further accelerates LSTM-AE inference (≈76%) and reduces power (≈35%), and the authors discuss FL (Federated Learning) and lightweight tactics (dynamic thresholds, event-triggered execution) for edge deployment.

 doi:10.20944/preprints202512.2225.v1

Tahri et al. [11] compare supervised ML (Machine Learning) classifiers for IDS (Intrusion Detection Systems) on UNSW-NB15 within the NIDS (Network IDS) context, detailing preprocessing and confusion-matrix metrics. Evaluated models include DT (Decision Tree), NB (Naïve Bayes), KNN (K-Nearest Neighbors), RF, SVM, and LR (Logistic Regression), reported via Accuracy, Precision, Recall, TPR, TNR (True Negative Rate), FPR, FNR (False Negative Rate), and FAR (False Alarm Rate). The study concludes that ensemble methods (notably RF) provide the most stable performance, and that feature selection/dimensionality reduction improve both runtime and detection quality; hybrid pipelines and swarm-based selection (e.g., PSO (Particle Swarm Optimization)) are highlighted as promising extensions.

Mari et al. [12] built an ML (Machine Learning)-based IDS (Intrusion Detection System) on NSL-KDD and evaluated robustness using a GAN (Generative Adversarial Network) to generate evasive malicious traffic while preserving attack functionality. The GAN synthesizes adversarial flows targeting IDS evasion, and the trained classifiers are re-tested on these harder samples to quantify resilience. The study reports that GAN-generated traffic can initially bypass detection, but adversarial evaluation and subsequent tuning improve IDS performance, motivating adversarial-aware validation for realistic deployment.

Yang et al. [13] propose uncertainty-aware anomaly detection using BDL (Bayesian Deep Learning) via a BAE (Bayesian Autoencoder) to estimate aleatoric and epistemic uncertainty and reduce false alerts in SOC (Security Operations Center) settings. The method models heteroscedastic noise in the latent space under a VLB/ELBO (Variational Lower Bound/Evidence Lower Bound) formulation, separating anomaly probability (from reconstruction/ELBO scores mapped via a CDF (Cumulative Distribution Function)) from anomaly uncertainty. Experiments on UNSW-NB15 and CIC-IDS-2017 show improved calibration that lowers FPR at comparable TPR/DR, supporting more reliable accept/reject policies and operational trust alongside XAI (Explainable Artificial Intelligence).

Kale and Thing [14] improve FSL/WS (Few-Shot Learning/Weakly Supervised) anomaly detection with a three-stage deep pipeline, namely triplet-based augmentation, MLP (Multi-Layer Perceptron) representation learning, and ordinal regression, to align scores with anomaly intensity. Evaluated on NSL-KDD, CIC-IDS2018, and TON-IoT using AUROC (Area Under the ROC Curve), TPR, FPR, and confusion-matrix counts, the method matches or improves ROC performance versus DevNet (Deviation Network) and a weakly supervised pairwise baseline, with gains increasing as labeled anomalies grow (e.g., 30→60). Results are robust across anomaly percentages; TON-IoT is generally easier (high AUROC), while CIC-IDS2018 remains more challenging, motivating larger tuples and deeper backbones.

Goumidi and Pierre [15] propose a real-time IoMT (Internet of Medical Things) anomaly detection framework using a stacking ensemble with XGBoost as meta-learner over RF and ANN (Artificial Neural Network) base models and introduce a healthcare-oriented dataset derived from UNSW-NB15 with added medical attacks (e.g., falsification, DoS (Denial of Service)). Seven ML models (KNN, SVM, LR, RF, IF (Isolation Forest), XGBoost, ANN) are benchmarked using Accuracy, Precision, Recall, F1-score, and ROC-AUC after encoding, min–max scaling and SMOTE for imbalance. The stacking approach outperforms individual learners on both datasets (98.02% accuracy on the medical dataset) and reports low-latency inference in a live streaming setup emulating Arduino-to-edge transmission. The work motivates ensemble IDS for resource-constrained healthcare and emphasizes domain-specific datasets and real-time validation for diverse attack coverage.

Kopljar et al. [16] propose xAAD, a post-feedback XAI (Explainable Artificial Intelligence) framework that combines IF (Isolation Forest) with AAD (Active Anomaly Discovery) and an IF-specific attribution scheme, AWS (Assist-Based Weighting Scheme), to reduce false positives and improve interpretability after analyst input. The method re-weights IF tree nodes through an interactive AAD labeling loop and derives local feature-importance vectors by fusing isolation depth with feedback-adjusted node weights. Compared with SHAP (SHapley Additive exPlanations) and DIFFI (Depth-based Isolation Forest Feature Importance), xAAD retains AWS efficiency while better

 doi:10.20944/preprints202512.2225.v1

aligning explanations with user feedback. Experiments on synthetic and real datasets report improved anomaly ranking, high binary classification accuracy, and lower RMSE (Root Mean Square Error) of importance estimates versus baseline AWS.

Villarreal–Vasquez et al. [17] propose LADOHD (LSTM-based Anomaly Detector Over High-dimensional Data) to detect insider threats from enterprise EDR (Endpoint Detection and Response) logs via LSTM next-event prediction. Anomalies are flagged when observed events fall outside a dynamic top-k prediction set, addressing order-awareness limits of n-gram/HMM (Hidden Markov Model) baselines. On a real-world dataset (38.9 M benign events over 30 machines/28 days plus a 4-day red-team attack), the method reports TPR (True Positive Rate) 97.29% at FPR (False Positive Rate) 0.38%. LADOHD outperforms a production EDR stack and fixed-k LSTM baselines while maintaining low FPR, highlighting the importance of long-range temporal dependencies for runtime detection at scale.

Khan et al. [18] propose a graph anomaly detector that combines a GCAE (Graph Convolutional Autoencoder) with SSL (Self-Supervised Learning) objectives, proximity-preservation losses (first-/high-order), and GAN (Generative Adversarial Network)-based adversarial refinement. The pipeline applies Laplacian smoothing/sharpening, reconstructs attributes and adjacency via GCN (Graph Convolutional Network) layers, and regularizes embeddings with tasks such as attribute masking and edge prediction. Evaluated on Cora, Citeseer, BlogCatalog, and Flickr, the method reports higher AUC-ROC and AP (Average Precision) than OC-SVM (One-Class SVM), DOMINANT, Dual-SVDAE, DVAEGMM, and CaCo, with notable gains in noisy, high-dimensional settings.

Mansourian et al. [19] propose a prediction-based IDS for CAVs (Connected Autonomous Vehicles) on CAN (Controller Area Network) traffic, exploiting temporal and spatiotemporal ECU (Electronic Control Unit) correlations. The method uses per-ID LSTM (Long Short-Term Memory) predictors and a shared ConvLSTM (Convolutional LSTM) to forecast next payloads, scores anomalies via MAE (Mean Absolute Error), and classifies with GNB (Gaussian Naïve Bayes) rather than fixed thresholds. On the Car Hacking Dataset (DoS, fuzzing, spoofing), it outperforms OCSVM (One-Class Support Vector Machine), Isolation Forest, AE (Autoencoders), and prior CAN IDS designs, reporting near-perfect F-score/Accuracy with low latency suitable for embedded gateways (e.g., Jetson-class). The study notes a memory–robustness trade-off: per-ID LSTM models increase footprint, while shared ConvLSTM reduces parameters but is more sensitive to uncorrelated signals, motivating ID grouping by ECU dependence.

To provide a consolidated overview of the state of the art in AI-based anomaly detection within embedded and IoT environments, Table 1 summarizes the key characteristics of the studies reviewed in this section. The comparison emphasizes the methodological diversity (from classical machine learning to hybrid and graph-based deep learning), datasets used for benchmarking, evaluation metrics, and the primary contributions or limitations identified by each work. This synthesis allows for the positioning the proposed framework within the broader research landscape and highlights the evolution from model-centric approaches toward integrated, resource-aware, and explainable anomaly detection systems.

Table 1. Comparative summary of related works on AI-based anomaly detection.

Ref.	Focus/Domain	Techniques/ Models	Datasets	Evaluation Metrics	Key Findings/ Limitations
[5]	TinyML/AIoT anomaly detection on microcontrollers	CNN, AE, LSTM, DNN, IF, GMM	IMU signals, images, temperature datasets	Accuracy, F1	CNN dominant (18.8%). TinyML benefits (latency decreases and privacy increases) but HW/SW heterogeneity and energy reporting limited
[6]	IoT layered AD survey	AE, RNN, RBM, CNN, GAN, RL	KDD'99, NSL-KDD, UNSW-NB15, Bot-IoT,	Precision, DR, FPR,	Broad taxonomy (physical–network–application); identifies issues of complexity,

 doi:10.20944/preprints202512.2225.v1

			CICIDS2017, ADFA, Yahoo Webscope	Accuracy, ROC-AUC, F1	privacy, interpretability; future XAI and Transformer work
[7]	IDS with DL	CNN, RNN, LSTM, GRU, SMOTE, GAN augmentation	CIC-IDS2017, UNSW-NB15	Accuracy, DR, Recall for U2R/R2L	Hybrid CNN-RNN improves temporal modeling. imbalance remains issue; calls for cost-sensitive and attention models
[8]	AD in IoT and sensor networks	ML and DL hybrids with GNN and attention	Multiple public IoT datasets	Precision, Recall, F1, ROC (TPR/FPR)	Highlights trade-off between generality vs. specialization. future work on explainability and energy efficiency
[9]	DL for IoT network anomaly detection	CNN, LSTM, AE, GAN, GNN, Transformer, FL	CICIDS2017, Bot-IoT, NSL-KDD, TON-IoT	Accuracy, Precision, Recall, F1	DL excels in feature learning. issues with label scarcity and adversarial robustness. advocates FL and XAI
[10]	Edge AI for Smart Home IoT	IF + LSTM-AE hybrid, quantized TinyML	Real and synthetic sensor data	Accuracy, Recall, Latency, Energy use	Hybrid reduces latency (< 50 ms) on Raspberry Pi. privacy-preserving local inference; future Transformers
[11]	ML classifiers on UNSW-NB15 dataset	DT, NB, KNN, RF, SVM, LR, PSO for features	UNSW-NB15, NSL-KDD, CICIDS2017	Accuracy, Precision, Recall, TPR, TNR, FPR, FNR, FAR	RF most robust; feature selection improves speed and accuracy; ensemble methods recommended
[12]	GAN-based IDS robustness testing	ML classifiers + GAN for adversarial traffic generation	NSL-KDD	Accuracy, F1, Detection Rate	Adversarial GAN traffic initially bypasses IDS but enhances robustness after retraining; advocates adversarial validation
[13]	Uncertainty-aware cybersecurity AD	BDL, BAE, BVAE	UNSW-NB15, CIC-IDS-2017	TPR, FPR, ROC-AUC, F1	Models aleatoric + epistemic uncertainty; reduces false alerts; integrates uncertainty with XAI for trust
[14]	Few-shot/weakly supervised IDS	Triplet augmentation, MLP, ordinal regression	NSL-KDD, CIC-IDS2018, TON-IoT	AUROC, TPR, FPR	High ROC with few labels. scalable to limited annotation settings
[15]	IoMT real-time IDS	Stacking ensemble (XGBoost + RF + ANN base models)	UNSW-NB15 + Healthcare dataset	Accuracy, Precision, Recall, F1, ROC-AUC	98% accuracy with low latency; demonstrates value of ensemble learning and domain-specific datasets
[16]	Explainable semi-supervised AD	IF + AAD + AWS	Synthetic & real datasets	Accuracy, RMSE of importance scores	Improves interpretability via feature attributions, bridges active learning and explainable AD
[17]	Insider threat detection via LSTM	LSTM sequence prediction with dynamic top-k selection	Enterprise EDR logs	TPR, FPR	97% TPR @ 0.38% FPR. outperforms EDR baselines; suited for high-volume SOC
[18]	Attributed network AD with SSL	GCAE, GCN, GAN, Dual-SVDAE, DVAEGMM	Cora, Citeseer, BlogCatalog, Flickr	AUC, AP	Outperforms GCN-AE baselines; strong under noise; shows benefits of adversarial and proximity constraints

 doi:10.20944/preprints202512.2225.v1

[19]	CAV IDS	LSTM, ConvLSTM, GNB	Car Hacking Dataset	Accuracy, F1, Latency	Near-perfect F1 and real-time performance; analyzes memory/latency trade-offs for embedded CAV devices

Existing studies report strong results on multiple datasets and deployment targets; however, direct comparison is limited by differences in datasets, preprocessing, and evaluation protocols. For this reason, a structured comparison with representative literature is provided in the results section.

3. Methodologies

Embedded systems are specialized devices with limited resources, integrated into applications ranging from medical devices to autonomous vehicles and critical infrastructure. The security of these systems is crucial, and the process of collecting and analyzing security data is central to ensuring their integrity, privacy, and availability [20].

The proposed methodology describes the development and evaluation of an AI-based framework for cyber-attack detection, including dataset preparation and preprocessing, signal-processing–based feature construction, and the implementation of machine learning and deep-learning models for anomaly identification and prediction. For experimental validation, the pipeline is benchmarked using the publicly available UNSW-NB15 dataset, while the data-collection discussion reflects representative sources in real embedded deployments.

3.1. Collecting Data in Embedded Systems

In embedded cybersecurity, the data collection process represents a fundamental stage for building AI-based cyber-attack detection pipelines. To ensure accurate anomaly identification, multiple data sources must be integrated, combining hardware-level measurements, system logs, and network traffic analysis into a unified monitoring infrastructure [21].

Sensors and monitoring agents play a crucial role in this architecture, as they continuously observe the operational behavior of embedded devices and capture relevant security data. These components can be implemented both in hardware and software, depending on the required granularity and performance constraints. Hardware sensors are typically integrated directly into the embedded platform and allow the detection of physical irregularities, unauthorized modifications, or voltage and timing anomalies. In parallel, software monitoring agents operate at the operating system or application layer, collecting information such as process execution traces, memory access patterns, network usage, and system integrity checks. The synergy between hardware sensors and software agents enables a multi-layered observation approach that enhances visibility across the entire embedded stack and supports timely identification of suspicious activities [22].

Another essential aspect of the data acquisition process is logging, which ensures the continuous recording of system events, transactions, and configuration changes. Security logs contain information about authentication attempts, user access, system errors, and process execution sequences, thereby allowing both preventive analysis and forensic investigation. Maintaining the integrity of log data is critical; tamper-proof storage and access control mechanisms must be implemented to prevent unauthorized modification or deletion. As a result, logging systems not only support post-incident analysis but also provide early indicators of potential attacks through the identification of abnormal behavioral patterns [23].

In addition to local data acquisition, network monitoring plays a pivotal role in capturing communication flows between embedded devices and external entities. By employing technologies such as packet sniffers and NetFlow analyzers, it becomes possible to observe traffic characteristics, identify suspicious packets, and detect deviations from normal communication protocols. Depending on the operational requirements, this monitoring can occur in real time allowing immediate detection and mitigation of threats, or post-mortem, and enabling detailed analysis and reconstruction of attack vectors after incidents occur. Combining both modes provides comprehensive coverage, improving detection accuracy and supporting a more resilient embedded cybersecurity framework [24,25].

 doi:10.20944/preprints202512.2225.v1

Through the integration of sensor data, secure logging, and network traffic monitoring, the proposed system establishes a consistent foundation for anomaly detection. This multi-source approach not only increases detection granularity but also enables correlation across different data types, creating the contextual awareness necessary for intelligent AI-driven cyber-attack detection in embedded environments.

For the experimental validation reported in this paper, data acquisition is operationalized using the UNSW-NB15 dataset, which provides flow-level network records with labeled benign and malicious traffic. This choice enables reproducible benchmarking of the proposed pipeline on standardized traffic features and attack categories. Consequently, the present experimental analysis is based on network-flow records rather than on-device sensor streams or embedded system logs, which are discussed in this section as representative data sources in real deployments.

3.2. Analyzing Security Data in Embedded Systems

Security data analysis represents a core component of the proposed framework, enabling the transformation of raw information collected from embedded systems into actionable intelligence. This process integrates several complementary techniques that enhance the system's ability to identify and classify abnormal behavior. Each analytical approach: behavioral, signature-based, heuristic, and correlation-driven, contributes to the accuracy and completeness of the overall detection architecture.

Behavioral analysis focuses on examining the patterns and dynamics of user and system activities to detect deviations from normal operation. By establishing baseline behavioral profiles, the system can identify actions that diverge significantly from expected norms, potentially indicating malicious activity. Such analysis typically relies on machine learning models and statistical methods that can adapt to evolving data distributions and learn complex dependencies between variables. Algorithms based on clustering, anomaly detection, and neural networks are commonly employed to uncover unusual behavioral patterns that might escape traditional rule-based mechanisms [26].

Signature-based analysis complements behavioral methods by comparing captured data with a continuously updated database of known attack and malware signatures. This approach is highly effective in detecting previously documented threats, as it directly matches data characteristics with specific attack fingerprints. However, its limitation lies in the inability to recognize novel or zero-day attacks that lack predefined signatures. Therefore, maintaining an up-to-date repository and integrating it with intrusion detection and antivirus systems is crucial for achieving robust protection against recurring threats [27].

To address the limitations of purely signature-driven methods, heuristic analysis introduces rule-based and algorithmic reasoning designed to detect suspicious behaviors that fall outside known signature patterns. Heuristic mechanisms analyze code structure, execution flows, and behavioral traits to identify potential zero-day exploits or emerging attack strategies. By simulating execution scenarios or applying pattern recognition rules, heuristic analysis provides an adaptive layer of defense capable of discovering previously unseen malicious actions [28].

Finally, correlation-based analysis enhances the system's situational awareness by integrating and correlating multiple heterogeneous datasets to identify relationships and composite patterns indicative of coordinated attacks. This technique allows the system to generate a holistic view of security events by linking network logs, sensor data, and system alerts. In practice, correlation-based analysis is implemented using SIEM (Security Information and Event Management) platforms, which aggregate and process data streams from different sources to construct a unified security perspective. Such correlation mechanisms improve detection efficiency, reduce false positives, and provide a contextual understanding of threats in embedded networks [29].

By combining these analytical dimensions, behavioral profiling, signature recognition, heuristic reasoning, and event correlation, the proposed framework achieves a balanced approach to security analysis. This integration ensures that both known and unknown threats can be effectively identified, offering a resilient defense architecture suitable for modern embedded environments.

 doi:10.20944/preprints202512.2225.v1

In the present implementation, behavioral analysis is operationalized through supervised and unsupervised learning on flow-level traffic features (e.g., RF/XGBoost/LightGBM, SVM, K-Means, DBSCAN) and reconstruction-based anomaly scoring (PCA and Autoencoder). Signature-based and heuristic mechanisms are discussed as complementary layers commonly used in deployed intrusion detection systems; however, the experimental validation reported in this paper focuses on learning-based detection from labeled and unlabeled traffic patterns within the UNSW-NB15 benchmark.

3.3. Signal Preprocessing Pipeline and Time-Series Construction

The UNSW-NB15 dataset provides flow-level network records that are not natively formatted as time-series. To enable the application of temporal signal-processing methods (Wavelet/Kalman), flow records are first ordered using an ordering key and aggregated into fixed-length observation windows. Records are sorted by an ordering key and grouped into windows of length W flows (records) with step H flows, resulting in a sequence of windows $\{W_m\}_{m=1}^{M}$.

In this study, $W = 256$ flows and $H = 128$ flows are used. This configuration provides a practical compromise between temporal resolution and statistical stability: the window length is sufficient to capture burst-like traffic variations while reducing variance in window-level descriptors, and the 50% overlap preserves short-lived transitions without excessively increasing computational overhead.

For each window W_m, a set of time-dependent traffic attributes is summarized into a compact vector using robust statistics. Typical attributes include packet-related and byte-related indicators (e.g., packet counts, byte counts, rate-like quantities), TTL (Time-To-Live)-related behavior, and inter-arrival dynamics derived from consecutive timestamps. For each selected attribute a, window-level descriptors are computed (e.g., mean, standard deviation, minimum/maximum, and optionally the last value), yielding a window sequence $x_a[m]$ that forms a pseudo time-series for attribute a. Concatenating the series of multiple attributes produces a multivariate time-series representation $x[m] \in \mathrm{R}^{\mathrm{p}}$, where p is the number of window-level descriptors across all chosen attributes.

Table 2 summarizes the windowing configuration and the resulting pseudo time-series dataset characteristics, including the number of constructed windows, the dimensionality of the window-level descriptor vector, and the class distribution at window level.

Table 2. Parameters and output characteristics of the window-based time-series construction for the UNSW-NB15 dataset.

Item	Value
Window length W	256 flows
Step H	128 flows (50% overlap)
Number of window-level descriptors per attribute	mean, std, min, max
Number of attributes used	14
Total features per window p	56
Number of windows	2012
Window labels	1695 attack/ 317 normal

This construction transforms irregular, event-based flow records into uniformly indexed temporal observations, enabling temporal decomposition (Wavelet) and dynamic-state modeling (Kalman). The resulting signal-derived features are designed to capture burst-like variations and short-lived deviations that are typically attenuated when only static variance structure is considered (e.g., raw PCA projections).

The proposed methodology can be formalized as a composed mapping from windowed observations to a decision score, $f = C \circ \Phi \circ A$, where A denotes window-based aggregation, Φ denotes transform and filtering operators (Fourier/Wavelet/Kalman) that yield compact sufficient statistics under variability, and C denotes the final learning-based decision rule. The mathematical

 doi:10.20944/preprints202512.2225.v1

foundations presented in the following subsections are included not only to motivate the adopted signal-processing operators through their objective effect on the representation (spectral concentration, multi-scale localization, and minimum-variance state estimation), but also to formalize the associated learning objectives and decision mechanisms used by the AI models. This ensures that all variables, parameters, and optimization criteria employed in the software implementation are precisely defined, and that the end-to-end pipeline can be reproduced and audited as a theoretically grounded signal-to-decision model rather than a collection of loosely connected techniques.

3.4. Signal Processing Techniques Used to Detect Anomalies

Detecting anomalies in embedded systems is essential for identifying and preventing cyber-attacks. Signal processing techniques play a crucial role in this process, by extracting relevant characteristics and identifying deviant behaviors.

Within the proposed framework, signal-processing methods are applied to the window-based pseudo time-series representation constructed in Section 3.3. Specifically, flow-level records are aggregated into overlapping windows, and window-level descriptors are computed for packet/byte indicators, TTL-related attributes, and inter-arrival dynamics, yielding multivariate sequences indexed by the window order. This construction enables frequency-domain, time–frequency, and state-space analysis to be meaningfully applied to network traffic dynamics, rather than treating each flow as an isolated static point.

Signal processing techniques represent the analytical foundation of the proposed framework for detecting cyber-attacks and anomalies in embedded systems. These methods enable the transformation, decomposition, and interpretation of traffic-derived sequences to reveal latent patterns indicative of abnormal behavior. The core approaches considered in this work include Fourier analysis, wavelet-based time–frequency decomposition, PCA-based compact representations, and Kalman filtering for dynamic-state modeling.

The FT (Fourier Transform) enables a frequency-domain view of traffic dynamics by transforming window-indexed sequences (constructed in Section 3.3) into spectral representations. Using the DFT (Discrete Fourier Transform)/FFT (Fast Fourier Transform), periodic or quasi-periodic components can be identified in descriptors such as packet and byte activity across consecutive windows [30]. In the proposed pipeline, Fourier-derived features (e.g., magnitude spectrum summaries and band-energy indicators) support the detection of repetitive scanning behavior and burst-like attack patterns that may be less evident in purely static variance-based representations [31,32].

The mathematical formula of the FT is presented in the relation (1):

$$X(f) = \int_{-\infty}^{+\infty} x(t) \cdot e^{-j \cdot 2\pi \cdot f \cdot t} dt \tag{1}$$

where $x(t)$ is the signal in the time domain, $X(f)$ is the representation of the signal in the frequency domain, f is the frequency analyzed, j is the imaginary unity ($j^2 = -1$) and $e^{-j \cdot 2\pi \cdot f \cdot t}$ is the complex basic function that describes sinusoidal oscillations.

For discrete signals, DFT is used, with the definition relationship (2):

$$X[k] = \sum_{n=0}^{N-1} x[n] \cdot e^{-j \cdot 2\pi \cdot k \cdot n/N} \tag{2}$$

where N is the total number of samples, and k is the discrete frequency component. Efficient implementation, FFT, reduces computational complexity from $O(N^2)$ to $O(N \log N)$.

Figure 1 illustrates the magnitude spectrum obtained by applying the FFT to a representative window-indexed traffic descriptor, highlighting dominant spectral components and broadband variations that can be used as compact Fourier-derived indicators for repetitive or burst-like traffic behavior.

 doi:10.20944/preprints202512.2225.v1

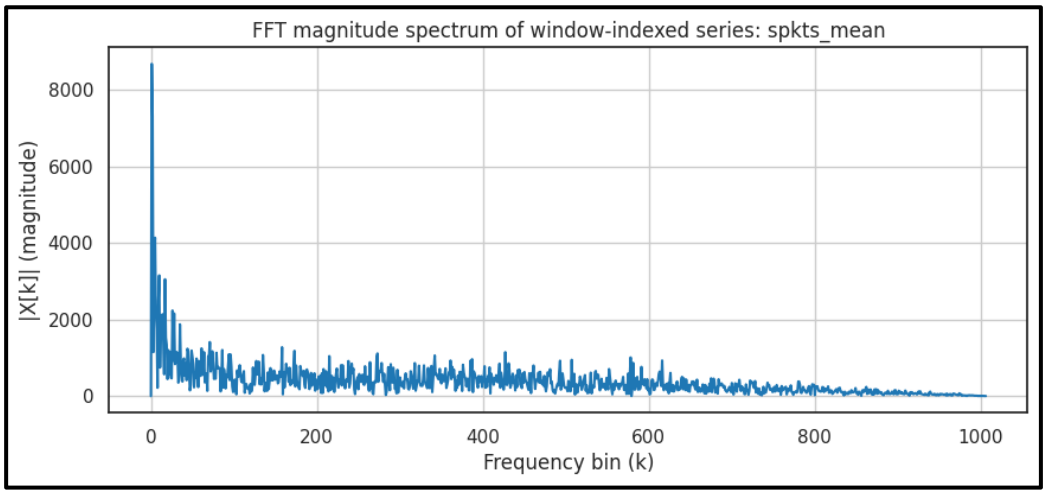

Figure 1. FFT magnitude spectrum of the window-indexed spkts_mean descriptor.

Complementing the frequency-domain perspective of Fourier analysis, the WT (Wavelet Transform) provides simultaneous time–frequency representation of signals, enabling the detection of short-lived or transient anomalies that may not be visible in pure frequency analysis [33]. The CWT (Continuous Wavelet Transform) facilitates detailed examination of complex signals, while the DWT (Discrete Wavelet Transform) offers efficient decomposition of discrete signals into wavelet coefficients that expose abrupt changes or localized irregularities [34,35]. In the proposed pipeline, wavelet-domain features are derived from window-indexed descriptor sequences using multiscale coefficient statistics and band-energy measures to capture transient deviations associated with bursty or evolving attack behavior. This makes WT suitable for the window-based traffic dynamics used in this study, as it can emphasize short-term fluctuations and localized burst-like variations that may be less visible in purely frequency-domain summaries.

WT is defined by the relation (3):

$$W(a,b) = \int_{-\infty}^{\infty} x(t) \cdot \psi^* \cdot \left(\frac{t-b}{a}\right) dt \tag{3}$$

where $x(t)$ is the analyzed signal, $\psi(t)$ is the mother wavelet function, $\psi^*(t)$ is the complex conjugate of the wavelet function, a is the scaling factor (frequency expansion/compression), b is the translational factor (time displacement) and $W(a,b)$ is the wavelet coefficient indicating the local contribution of the signal to the scale a and position b.

For discrete signals, the DWT is used, as defined by (4):

$$W_{j,k} = \sum_{n=0}^{N-1} x[n] \cdot \psi_{j,k}[n] \tag{4}$$

where j represents the level of resolution (scale) and k is the translation index.

A CWT scalogram is included in Figure 2 to illustrate time–frequency localization, where concentrated coefficient energy indicates transient bursts and short-lived deviations.

 doi:10.20944/preprints202512.2225.v1

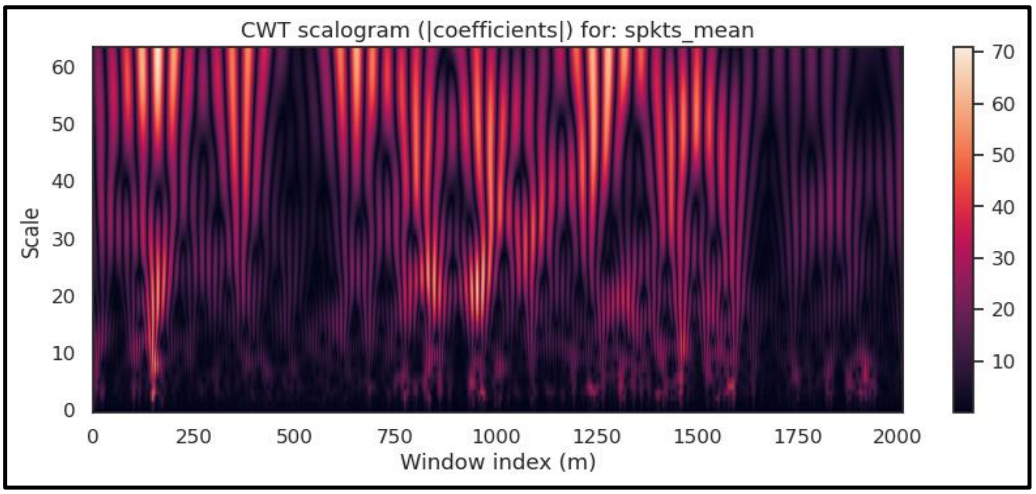

Figure 2. CWT scalogram of the window-indexed spkts_mean descriptor.

Another powerful feature extraction method is PCA, a dimensionality reduction technique that transforms data into a smaller set of orthogonal components while preserving maximum variance [36]. By projecting data onto principal components derived from the covariance matrix, PCA enables the identification of atypical variance structure or correlations that deviate from normal operating patterns [37]. In this study, PCA is used to obtain compact representations of the window-level traffic descriptors and to support reconstruction-based analysis, enabling efficient dimensionality reduction while preserving the dominant variance structure. Its efficiency in reducing computational complexity makes it ideal for embedded applications where memory and processing capabilities are limited. Within the proposed framework, PCA is also used as a baseline compactness mechanism for comparison against wavelet- and Kalman-derived temporal descriptors. PCA transforms the original data X into a new orthogonal space of the principal components Y, as shown in relation (5):

$$Y = X \cdot W \tag{5}$$

where: X is the initial data matrix ($m \times n$, with m samples and n features), W is the matrix of eigenvectors of the covariance matrix $C_X = \frac{1}{m-1} \cdot X^T \cdot X$ and Y is the data projection in small space (main components).

Each eigenvector corresponds to a direction of maximum variation, and eigenvalues indicate the importance (explained variance) of each component.

Figure 3 reports the cumulative explained variance of the PCA model fitted on the window-level descriptor space, illustrating that most of the variance is captured by a relatively small number of principal components, which supports compact representations for efficient downstream detection.

 doi:10.20944/preprints202512.2225.v1

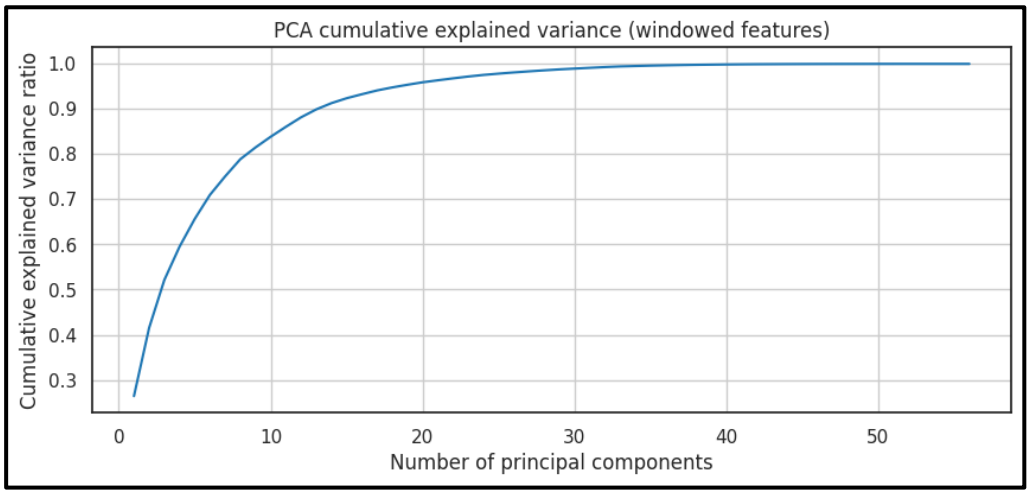

Figure 3. Cumulative explained variance of PCA for the window-level feature set.

The Kalman Filter represents a recursive state estimation technique designed to infer the true state of a system from noisy measurements [38]. By modeling the system through state and observation equations, the filter continuously updates its predictions as new data arrive, identifying discrepancies between expected and measured behavior. This capability allows for real-time detection of anomalies in systems such as autonomous vehicles, industrial monitoring setups, and biomedical signal acquisition devices. The Kalman approach is especially valuable for embedded security because it can efficiently distinguish between environmental noise and genuine signal deviations that may indicate cyber or operational anomalies. Within the proposed framework, Kalman filtering is applied to window-indexed traffic descriptor sequences, and innovation/residual statistics are extracted as features to highlight deviations from the expected window-to-window dynamics. The Kalman model consists of two main stages: prediction and updating.

State model (6):

$$x_k = A_{x_k-1} + B_{u_k} + w_k \tag{6}$$

Observation model (7):

$$z_k = H_{x_k} + v_k \tag{7}$$

where x_k is the vector of the estimated state at moment k, A is the transition matrix of states, B is the control matrix, u_k is the vector of control inputs, w_k is the process noise (with Gaussian distribution $N(0, Q)$), z_k is the vector of the measurements, H is the observation matrix and v_k is the measurement noise (with $N(0, R)$ distribution).

Update of the estimate (8)–(10):

$$K_k = P_k^- \cdot H^T \cdot (H \cdot P_k^- \cdot H^T + R)^{-1} \tag{8}$$

$$x_k = x_k^- + K_k(z_k - H \cdot x_k^-) \tag{9}$$

$$P_k = (I - K_k \cdot H) \cdot P_k^- \tag{10}$$

where K_k is the Kalman gain, P_k is the covariance matrix of the estimation error, and x_k^- is the prediction of the previous state.

Figure 4 illustrates Kalman filtering applied to a representative window-indexed traffic descriptor, where the state estimate smooths noisy fluctuations and the innovation magnitude highlights deviations from the expected window-to-window dynamics.

 doi:10.20944/preprints202512.2225.v1

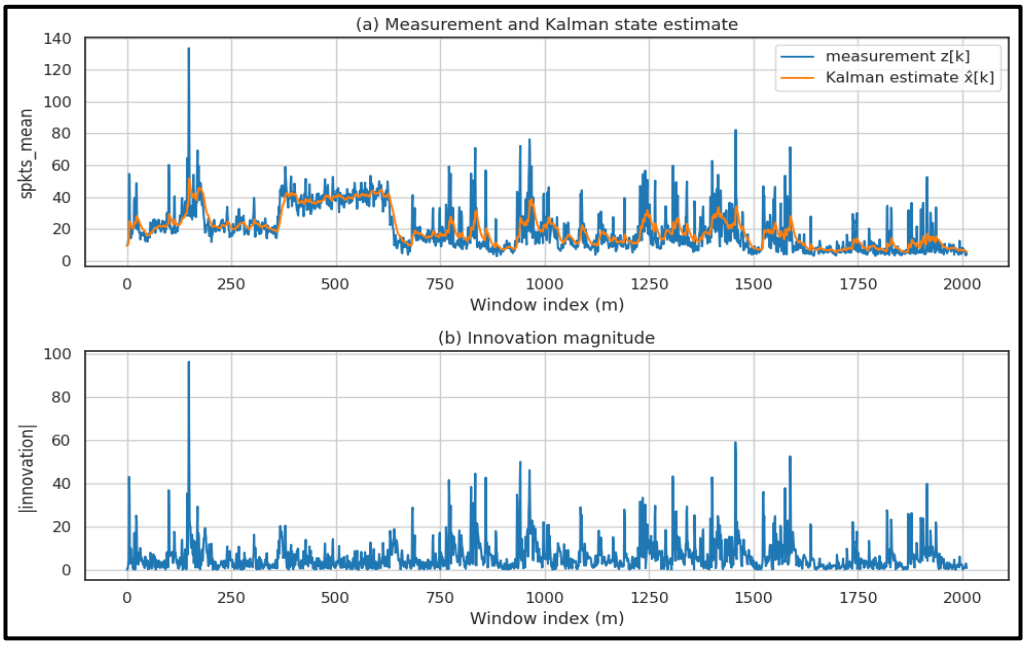

Figure 4. Kalman filtering on the window-indexed spkts_mean descriptor: (**a**) measurement and state estimate; (**b**) innovation magnitude.

Table 3 summarizes how each signal-processing module is instantiated in this study, including the concrete example used for visualization and the type of extracted descriptors used in the detection pipeline.

Table 3. Study-specific comparison of the signal-processing modules used in the proposed pipeline.

Technique	Key Settings	Output Feature Type	Measured Runtime	Main Limitation
Fourier (DFT/FFT)	Series: spkts_mean; FFT magnitude on window index	Spectral summaries/band-energy indicators	FFT rfft (2012 windows): 0.178 ms per run	Less informative for strongly non-stationary behavior if used alone
Wavelet (CWT/DWT)	Wavelet: Morlet; scales $1-64$ (CWT)	Multiscale coefficient statistics/energy features	CWT (scales 1–64, Morlet): 15.195 ms per run	Parameter sensitivity (wavelet/scales) and higher compute than FFT
PCA	Standardized window features; 14/19/31 components for 90%/95%/99% cumulative explained variance (train split).	Low-dimensional components; explained variance profile	PCA fit performed offline; projection is linear (matrix multiplication)	Linear model; may miss non-linear structure
Kalman Filter	1D model; $Q = 10^{-3}$, $R = 10^{-1}$	Innovation/residual statistics (anomaly-sensitive)	D Kalman (2012 windows): 0.709 ms per run	Model/parameter dependence (Q/R, assumptions)

These measurements indicate that FFT- and Kalman-based modules are lightweight for window-level sequences, while CWT is comparatively more computationally intensive and is therefore treated primarily as an offline feature construction step or a selectively enabled module depending on resource constraints.

3.5. AI Models for Cyberattack and Anomaly Detection

This section describes the supervised and unsupervised learning models used in the proposed detection pipeline and summarizes their operating principles and implementation considerations for resource-constrained settings.

 doi:10.20944/preprints202512.2225.v1

AI has become an indispensable tool in modern cybersecurity frameworks, providing advanced methods for identifying, predicting, and understanding anomalies across complex embedded environments. By leveraging algorithms capable of learning from large-scale data, AI enhances system resilience through pattern recognition, adaptive learning, and intelligent classification. The applied techniques are generally grouped into supervised, unsupervised, and semi-supervised or transfer learning approaches, each contributing distinct capabilities for anomaly detection and attack identification [39].

In supervised learning, models are trained on labeled samples to separate benign from malicious behavior in the engineered feature space. In this study, supervised detection is evaluated using SVM [40] as a margin-based classifier and tree/ensemble learners, RF [41], XGBoost, and LightGBM, which are well-suited to heterogeneous tabular traffic descriptors and often provide strong accuracy–robustness trade-offs in intrusion detection [42].

When labels are unavailable or incomplete, unsupervised learning provides complementary anomaly discovery by modeling structure in the feature space rather than relying on class targets. In this study, K-Means [43] is used as a clustering baseline, while DBSCAN [44] is used as a density-based alternative that can isolate sparse outliers without predefining the number of clusters.

In the present framework, all models operate on the same normalized feature space obtained from UNSW-NB15, augmented with window-based signal descriptors (Fourier/Wavelet/Kalman) and compact PCA representations. This ensures that performance differences reflect the learning mechanism (margin-based, ensemble, clustering, reconstruction/distribution scoring) rather than inconsistencies in preprocessing.

Reconstruction-based DL detectors, particularly Autoencoders, are considered to complement classical models by assigning anomaly scores from reconstruction error. Autoencoders are neural network architectures designed to learn compact, latent representations of input data in an unsupervised or semi-supervised manner. Their main objective is to capture the intrinsic structure of data by compressing it through an encoder and reconstructing it through a decoder [45,46]. Autoencoders are defined by the composite mapping shown in relation (11):

$$\hat{x} = f_{dec}\left(f_{enc}(x)\right) \tag{11}$$

where x is the input data vector, $f_{enc}(\cdot)$ is the encoding function that transforms the input into a lower-dimensional latent representation h, $f_{dec}(\cdot)$ is the decoding function that reconstructs the input from the latent space and $\hat{x}$—reconstructed output.

The encoding process can be expressed as (12):

$$h = f_{enc}(x) = \sigma(W_e x + b_e) \tag{12}$$

The decoding process can be expressed as (13):

$$\hat{x} = f_{dec}(h) = \sigma(W_d h + b_d) \tag{13}$$

where W_e, W_d are the encoder and decoder weight matrices, b_e, b_d are the bias vectors and $\sigma(\cdot)$ is the nonlinear activation function (e.g., sigmoid, ReLU (Rectified Linear Unit)).

The reconstruction loss is used as an anomaly score and is defined by relation (14):

$$L(x, \hat{x}) = \|x - \hat{x}\|^2 \tag{14}$$

Large reconstruction errors indicate that the model was unable to reproduce the input accurately, suggesting that the input pattern deviates from the learned normal distribution, an essential signal for anomaly detection.

GAN-based scoring is considered as a distribution-learning alternative, where deviations from the learned normal data distribution can be treated as anomalous. GANs consist of two competing neural networks, namely a generator G and a discriminator D, trained simultaneously in an adversarial process [47]. The optimization is expressed in relation to (15):

$$\min_G \min_D V(D,G) = \mathrm{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathrm{E}_{z\sim p_z(z)}\left[\log\left(1 - D\left(G(z)\right)\right)\right] \tag{15}$$

 doi:10.20944/preprints202512.2225.v1

where $D(x)$ is the discriminator output (probability that sample x is real), $G(z)$ is the synthetic data generated from latent variable z, $p_{data}(x)$ is the true data distribution and $p_z(z)$ —prior distribution of the latent variable (e.g., Gaussian).

Through this minimax optimization, the generator learns to produce data indistinguishable from real samples, while the discriminator learns to differentiate between real and generated inputs. In anomaly detection, deviations from the learned data distribution are flagged as potential attacks or abnormal events.

Table 4 summarizes the evaluated model families and highlights their role within the proposed pipeline, emphasizing the learning paradigm and the type of anomaly evidence (classification, clustering-based outliers, reconstruction error, or distribution deviation).

Table 4. Comparative overview of the evaluated AI model families within the proposed anomaly-detection pipeline.

Algorithm	Learning Paradigm	Computational Cost	Typical Training/ Inference time	Noise/ Variability Robustness	Resource Footprint	Main Limitations
SVM	Supervised	Training can be high with large N; inference moderate	Moderate training; fast inference	Moderate (kernel + scaling sensitive)	CPU/RAM moderate	Kernel/parameter sensitivity; scalability issues on large datasets; limited interpretability
RF	Supervised (ensemble)	baseline; robust decision rules for heterogeneous tabular descriptors; inference $\sim O(T \cdot D)$ (T-trees, D-depth)	Fast training; fast inference	High (stable under feature noise/variance)	CPU/RAM moderate; model size grows with T	Model size may grow; less efficient than boosting when heavily tuned; limited calibrated probabilities
XGBoost	Supervised (boosting)	Training higher than RF; inference $\sim O(T \cdot D)$	Moderate training; fast inference	High (with regularization)	CPU/RAM moderate	Sensitive to hyperparameters; risk of overfitting without careful regularization
LightGBM	Supervised (boosting)	Efficient training; inference $\sim O(T \cdot D)$	Moderate training; fast inference	High (often robust on engineered descriptors)	CPU/RAM moderate	Parameter-sensitive; performance depends on tuning and handling class imbalance
K-Means	Unsupervised (clustering)	$\sim O(N \cdot k \cdot d)$ per iteration	Fast training/inferenc e	Low–Moderate	Low	Requires k; sensitive to initialization and feature scaling; weak on irregular cluster shapes
DBSCAN	Unsupervised (density-based)	without k; isolates sparse regions as anomalies; Often $\sim O(N\ log\ N)$ with indexing (data-dependent)	Moderate	Moderate–High (good at isolating sparse outliers)	CPU moderate	Sensitive to $\varepsilon/MinPts$; degrades in high-dimensional spaces; unstable under varying density
AE	DL, reconstruction-based)	Training depends on architecture/epoch	Moderate training; fast inference	High (if trained on “normal” distribution;	CPU/GPU depending on architecture	Requires careful training protocol; thresholding/calibrati

 doi:10.20944/preprints202512.2225.v1

		s; inference low–moderate		benefits from denoising)		on needed; can reconstruct some attacks (missed anomalies)
GAN	DL, distribution-based	Training expensive/unstable; inference moderate	High training cost; inference moderate	High in principle, but depends strongly on stable training	Typically, GPU for training	Training instability (mode collapse); sensitive to hyperparameters; harder to reproduce consistently

All evaluated models are trained and tested on the same normalized feature space derived from UNSW-NB15, including the window-level descriptors and the optional signal-processing augmentation (Fourier/Wavelet/Kalman) together with PCA-based compact representations. This design ensures that observed performance differences primarily reflect the learning mechanism (margin-based classification, tree ensembles, clustering-based outliers, reconstruction error, or distribution deviation) rather than inconsistencies in preprocessing. Robustness to noise and variability, as well as efficiency indicators, are quantified in the Results section through controlled perturbation tests and runtime/complexity reporting.

4. Implementation and Results

The implementation of a cyber-attack detection capability for resource-constrained embedded devices requires an integrated detection architecture that remains reliable under strict computational, memory, and energy budgets, while maintaining sensitivity to heterogeneous abnormal behaviors within network traffic. To satisfy these constraints, the detection pipeline is instantiated through complementary learning paradigms that jointly cover discriminative classification and deviation-driven anomaly screening. The resulting framework combines classical classifiers such as SVM with ensemble learners including Random Forest, XGBoost, and LightGBM, while also incorporating clustering baselines such as K-Means and DBSCAN and deep anomaly detectors based on reconstruction through autoencoders and adversarial scoring through GAN-derived anomaly measures. In addition, compact representations and reconstruction-oriented analysis using PCA support dimensionality reduction, facilitate interpretability through low-dimensional projections, and enable thresholded scoring via reconstruction error.

Within this methodological setting, robust feature representations and calibrated decision thresholds improve detection coverage across diverse attack behaviors while limiting spurious alarms. Real-time operation is treated as a deployment-oriented objective and is discussed in terms of feasibility under embedded constraints, with attention to model complexity and inference cost. Platform-level validation targeting latency, memory footprint, CPU utilization, and energy consumption is positioned as a subsequent step toward operational readiness on representative edge hardware.

4.1. Embedded Deployment

Embedded and real-time applicability is constrained by inference latency, memory footprint, CPU (Central Processing Unit) utilization, and energy consumption. The experimental validation is carried out on UNSW-NB15; therefore, the reported results quantify detection performance through Accuracy, Precision, Recall, F1 score, ROC-AUC, and confusion matrices, while platform-level runtime characterization is addressed through feasibility indicators and envelope-based bounds. Deployment feasibility is consequently framed using a hardware-capability envelope derived from platform documentation and datasheets. For an edge-class ARM system-on-module such as Jetson Nano, the computed envelope is defined by a quad-core Arm Cortex-A57 CPU up to 1.43 GHz, 4 GB LPDDR4 memory, and software-defined 10 W and 5 W power modes that cap operating frequencies and online CPU cores to remain within the respective budgets.

 doi:10.20944/preprints202512.2225.v1

Real-time capability is evaluated by comparing the measured per-window inference time t_{inf} against the analysis window duration T_w, with streaming feasibility requiring $t_{inf} \leq T_w$. To preserve operational margin for I/O (Input/Output), feature normalization, and logging, a stricter headroom target can be stated as $t_{inf} \leq 0.7T_w$, which corresponds to a conservative CPU-only utilization bound $U \leq t_{inf}/T_w$.

Memory feasibility is characterized using peak RSS (Resident Set Size) observed during inference, a directly measurable quantity that captures the maximum runtime footprint; sustained deployment is considered feasible when peak RSS remains below the effectively available memory without paging-induced latency inflation. For power and energy, average power draw P_{avg} is monitored during steady-state streaming, and energy per inference is derived from measured quantities as $E = P_{avg} \cdot t_{inf}$; equivalently, for a streaming rate $f = 1/T_w$, the average energy per window can be reported as $E \approx P_{avg}/f$.

Overall, the embedded deployment envelope is grounded in measured behavior of the implemented pipeline, enabling feasibility statements that are tied to observed latency, CPU headroom, memory footprint, and energy cost under continuous edge execution.

Table 5 summarizes the streaming feasibility envelope by mapping periodic windowed inference to real-time latency constraints, with a headroom target and to deterministic energy upper bounds computed from the standard relation, using the target's software-defined 5 W and 10 W power regimes as experimental budgets.

Table 5. Streaming inference feasibility bounds under windowed operation, reporting latency budgets and deterministic energy upper bounds for software-configured power modes.

Inference Rate f	Window Duration T_w	Latency Budget $t_{inf,max} = T_w$	Headroom Latency $t_{inf} = 0.7T_w$	E_{max} @ 5 W	E_{max} @ 10 W	$E_{headroom}$ @ 5 W	$E_{headroom}$ @ 10 W
10 Hz	100 ms	100 ms	70 ms	0.5 J (500 mJ)	1.0 J (1000 mJ)	0.35 J (350 mJ)	0.70 J (700 mJ)
50 Hz	20 ms	20 ms	14 ms	0.1 J (100 mJ)	0.2 J (200 mJ)	0.07 J (70 mJ)	0.14 J (140 mJ)
100 Hz	10 ms	10 ms	7 ms	0.05 J (50 mJ)	0.1 J (100 mJ)	0.035 J (35 mJ)	0.07 J (70 mJ)

These values represent envelope-based bounds derived from documented power budgets and the selected inference rate, while platform-specific latency, peak memory, and CPU utilization remain measurable quantities for subsequent on-device profiling.

4.2. Dataset Description and Benchmark Protocol

The experimental evaluation employs the UNSW-NB15 benchmark [48], a labeled network security dataset generated in a controlled cyber-range environment that combines modern benign traffic with contemporary synthetic attack behaviors. The dataset is distributed both as a full collection (2,540,044 records) and as an official predefined benchmark split comprising a training subset (175,341 records) and a testing subset (82,332 records). Each record includes engineered flow-level attributes and a corresponding class label, and the benchmark covers nine attack categories reflecting heterogeneous intrusion behaviors [48].

A representative sample of the UNSW-NB15 benchmark records used in the present experiments is illustrated in Figure 5 [48].

 doi:10.20944/preprints202512.2225.v1

	dur	proto	service	state	spkts	dpkts	sbytes	dbytes	rate	sttl	dttl	label	attack_cat
0	0.121478	tcp	-	FIN	6	4	258	172	74.087490	252	254	0	Normal
1	0.649902	tcp	-	FIN	14	38	734	42014	78.473372	62	252	0	Normal
2	1.623129	tcp	-	FIN	8	16	364	13186	14.170161	62	252	0	Normal
3	1.681642	tcp	ftp	FIN	12	12	628	770	13.677108	62	252	0	Normal
4	0.449454	tcp	-	FIN	10	6	534	268	33.373826	254	252	0	Normal
5	0.380537	tcp	-	FIN	10	6	534	268	39.417980	254	252	0	Normal
6	0.637109	tcp	-	FIN	10	8	534	354	26.683033	254	252	0	Normal
7	0.521584	tcp	-	FIN	10	8	534	354	32.593026	254	252	0	Normal

Figure 5. Sample of the UNSW-NB15 benchmark dataset used for training and testing.

Subsequently, the distribution of attack categories was analyzed to characterize class imbalance across threat types and to contextualize model performance under skewed category prevalence. Figure 6 reports the relative composition of attack classes in the considered benchmark subset, highlighting that the dataset is dominated by high-frequency categories. Generic and Exploits account for the largest shares, representing approximately 35.8% and 27.0%, respectively, followed by Fuzzers at approximately 14.7% and DoS at approximately 9.9%. The remaining categories contribute comparatively smaller fractions, which motivates the use of complementary detectors and threshold calibration strategies to limit sensitivity degradation on minority attack types.

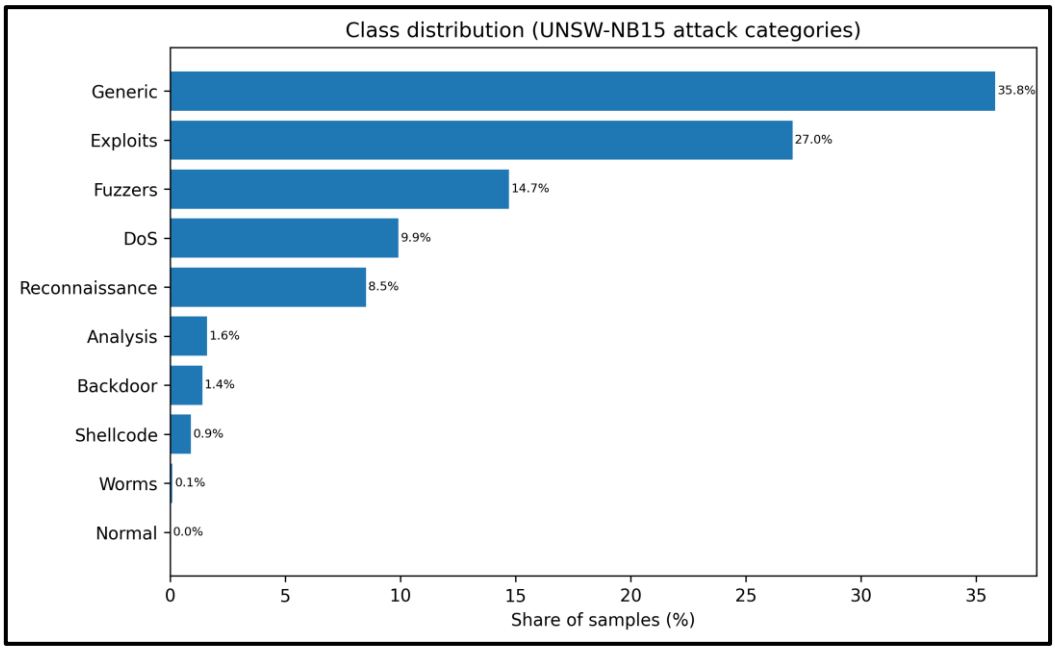

Figure 6. Distribution of attack categories in the UNSW-NB15 benchmark subset.

The evaluation protocol preserves the official UNSW-NB15 predefined split [48] by keeping the training subset and testing subset strictly disjoint. The training subset contains 175,341 records, while the testing subset contains 82,332 records. The testing subset is kept fully held out and is used exclusively for final performance reporting. To support model selection and calibration without test leakage, an internal 80/20 split is derived only from the training subset, yielding a fit subset of 140,272 records and a validation subset of 35,069 records. The validation subset is used solely for procedures such as threshold selection for one-class detectors and hyperparameter tuning, while ensuring that no information from the testing subset influences model fitting, calibration, or model selection.

Beyond protocol compliance, interpretability is supported through a compact DT (Decision-Tree) visualization that provides an explicit view of how representative flow-level attributes separate benign and attack samples. Although the full detection pipeline relies on multiple model families, the

 doi:10.20944/preprints202512.2225.v1

DT view is used as an explanatory diagnostic to reveal class-conditional structure, identify dominant split attributes, and substantiate the motivation for combining complementary detectors under heterogeneous attack behaviors. This perspective is particularly useful for understanding why certain attack categories concentrate around specific feature regimes and for validating that the selected feature representations preserve separability under constrained model capacity [49].

Figure 7 illustrates this interpretable structure using a shallow tree fitted on the training-fit subset and a small set of representative features, where the node sample counts reflect the subset reaching each split rather than the global subset cardinalities reported above.

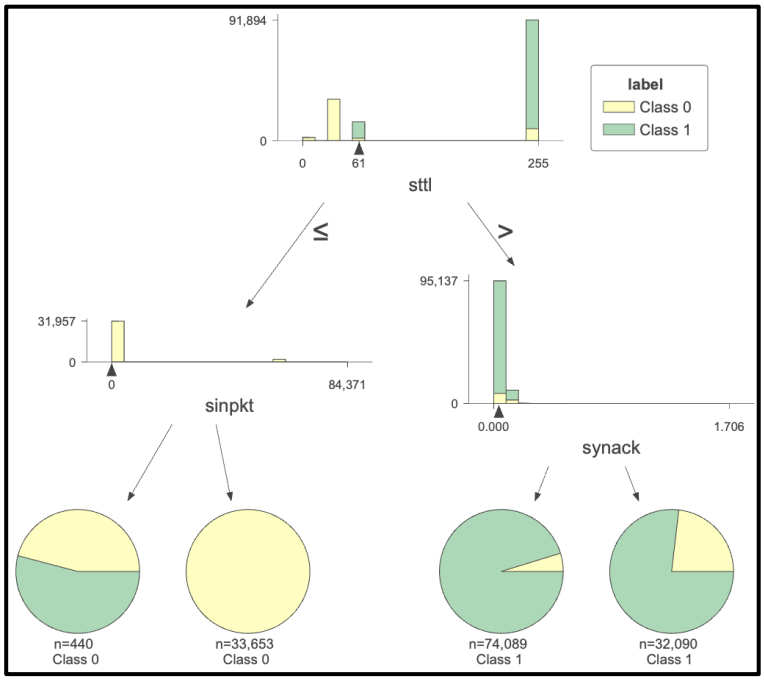

Figure 7. Decision-tree view of UNSW-NB15 features and labels, fitted on the training-fit subset.

To characterize statistical dependencies among network attributes and to assess potential feature redundancy, correlation heatmaps were generated for the UNSW-NB15 dataset. Figure 8 reports the pairwise Pearson correlation coefficients across numeric features, enabling the identification of strongly correlated feature groups that may induce redundancy in downstream learning stages. Such analysis is particularly relevant for embedded detection pipelines, where computational and memory constraints motivate compact representations, feature pruning, or dimensionality reduction to reduce inference cost while preserving discriminative structure [50].

 doi:10.20944/preprints202512.2225.v1

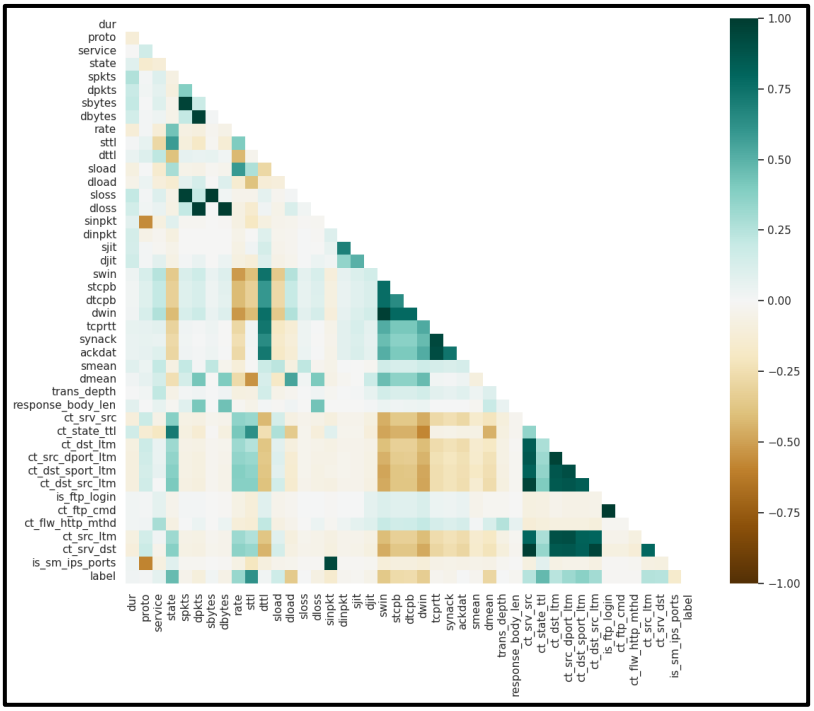

Figure 8. Pearson correlation heatmap of UNSW-NB15 numeric features.

In addition, a feature-to-label correlation ranking was extracted to provide a coarse, model-agnostic indication of which individual attributes exhibit the strongest linear association with the attack label, as summarized in Figure 9. High positive correlation values suggest features that tend to increase with malicious traffic in the considered benchmark distribution, while negative correlation values indicate features that decrease on average for labeled attacks. This ranking is not interpreted as causal evidence, but it serves as a useful diagnostic for guiding lightweight feature selection and for motivating the subsequent use of compact feature sets and reconstruction-based analysis within the implemented pipeline.

 doi:10.20944/preprints202512.2225.v1

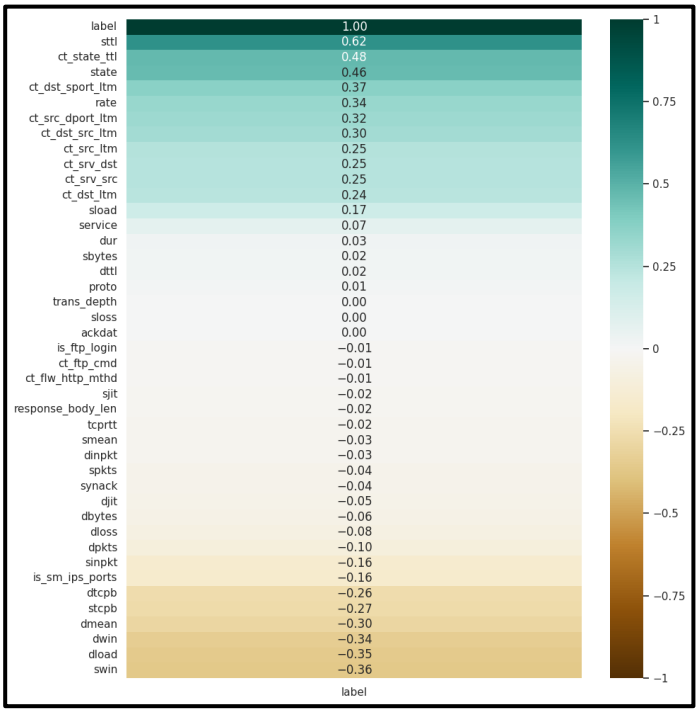

Figure 9. Feature-to-label Pearson correlation ranking for UNSW-NB15.

Taken together, the correlation heatmap and the feature-to-label ranking provide a complementary, model-agnostic diagnostic of which temporal, cumulative, and transport-layer attributes exhibit consistent linear association with the attack label in UNSW-NB15. This exploratory evidence aligns with the feature relevance patterns typically observed in tree-based learners, supporting the use of these attributes as informative predictors under embedded constraints, where interpretability and computational efficiency are critical design objectives.

4.3. FFT-Based Feature Representation and Detector Suite

This subsection reports the implementation and performance of frequency-domain features obtained via FFT for window-level intrusion detection under the strict benchmark protocol defined in the dataset description section. All model fitting, calibration, and hyperparameter selection were conducted using only the official training split of UNSW-NB15, while the official testing split was kept fully held out for final reporting. In addition, any internal calibration procedures, such as threshold selection for one-class detectors, were performed exclusively on a validation subset derived from the training split, thereby preventing any form of test leakage and directly addressing the reproducibility constraints raised during review.

The FFT representation was constructed by transforming each preprocessed window into a fixed-length spectral feature vector of 42 components. This compact representation was selected to support computationally efficient learning while preserving discriminative structure in the frequency domain. For the supervised FFT experiments, the resulting windowed datasets comprised 4381 fit windows, 1093 validation windows, and 2570 test windows, each described by 42 FFT features. For one-class evaluation, the benign-only training protocol was applied by restricting model fitting to benign windows extracted from the training split, with 1399 benign windows used for fitting and 349 benign windows used for validation-based threshold calibration. The held-out FFT test set contained 2570 windows with an observed attack prevalence of 55.06%, which provides a realistic and challenging evaluation regime for both discriminative and novelty-detection models.

To assess the behavior of a reconstruction-oriented one-class approach on FFT features, GANomaly (GAN Anomaly) was trained exclusively on benign FFT windows from the training subset and monitored on benign validation windows to define an operating threshold. Thresholds

 doi:10.20944/preprints202512.2225.v1

were selected from the benign validation score distribution at the 95th, 97th, and 99th percentiles, and the resulting decision rules were applied once to the held-out test set. In contrast to earlier experiments, the updated configuration yielded near-perfect separability between benign and attack windows. Specifically, at the p95 threshold, the method achieved Acc = 0.9949, Prec = 0.9969, Rec = 0.9969, and F1 = 0.9969, with AUC = 0.9999. Using stricter thresholds reduced false positives at the cost of slightly higher false negatives: both p97 and p99 achieved Acc = 0.9923, Prec = 0.9999, Rec = 0.9908, F1 = 0.9954, and AUC = 0.9999. Overall, these results indicate that, under the current FFT parametrization and benign-only regime, GANomaly provides a stable and highly discriminative anomaly score.

In the supervised setting, classical discriminative baselines achieved similarly strong performance on the same FFT representation, with extremely low error rates. The SVM model attained Acc = 0.9949, Prec = 0.9999, Rec = 0.9939, and F1 = 0.9969, with AUC = 0.9999, indicating virtually complete attack sensitivity while preserving perfect benign precision on the test split. The RF model provided a marginally higher accuracy, achieving Acc = 0.9974, Prec = 0.9970, Rec = 0.9999, and F1 = 0.9985, with AUC = 0.999, corresponding to zero missed attacks and a single false alarm on benign traffic.

In contrast, alternative unsupervised and one-class baselines on the same FFT features exhibit more pronounced trade-offs, which clarifies the practical value of reconstruction-based scoring. The AE one-class detector matches the supervised operating range when a strict threshold is used: at p99, it reached Acc = 0.9949, Prec = 0.9999, Rec = 0.9939, F1 = 0.9969, and AUC = 0.9999, while lower percentiles increase false positives. K-Means one-class (k = 5) remains competitive but slightly below AE/GANomaly (e.g., p95: Acc = 0.9769, F1 = 0.9863, AUC = 0.9965). By comparison, DBSCAN one-class enforces an effectively zero false-positive regime (FP = 0 across tested eps quantiles) but does so by rejecting many attack samples as normal (e.g., p95: Acc = 0.8359, F1 = 0.8915). Collectively, the updated FFT results position supervised RF/SVM as the highest-performing detectors when labels are available, while AE (p99) and GANomaly (p95–p99) provide comparably strong performance under a benign-only training assumption, with controllable sensitivity–specificity trade-offs via percentile-based thresholding.

The FFT-based detection results obtained under the strict UNSW-NB15 benchmark protocol are summarized in Table 6.

Table 6. FFT-based window-level detection performance on the held-out UNSW-NB15 test subset under the strict benchmark protocol.

Model	Acc	Prec	Rec	F1	AUC
SVM	0.9949	0.9999	0.9939	0.9969	0.9999
RF	0.9974	0.9970	0.9999	0.9985	0.9999
XGBoost	0.9949	0.9969	0.9969	0.9969	0.9999
LightGBM	0.9974	0.9970	0.9999	0.9985	0.9998
K-Means proxy (k = 2)	0.8385	0.8385	0.999	0.9121	0.2312
K-Means one-class (k = 5, p95)	0.9769	0.9818	0.9908	0.9863	0.9965
K-Means one-class (k = 5, p97)	0.9718	0.9877	0.9786	0.9831	0.9965
K-Means one-class (k = 5, p99)	0.9615	0.9906	0.9633	0.9767	0.9965
DBSCAN one-class (p95)	0.8359	0.9999	0.8043	0.8915	0.9224
DBSCAN one-class (p97)	0.8282	0.9999	0.7951	0.8859	0.9224
DBSCAN one-class (p99)	0.7897	0.9999	0.7492	0.8566	0.9224
AE one-class (p95)	0.9769	0.9732	0.9999	0.9864	0.9999
AE one-class (p97)	0.9821	0.9790	0.9999	0.9894	0.9999
AE one-class(p99)	0.9949	0.9999	0.9939	0.9969	0.9999
GAN one-class (p95)	0.9949	0.9969	0.9969	0.9969	0.9999
GAN one-class (p97)	0.9923	0.9999	0.9908	0.9954	0.9999
GAN one-class (p99)	0.9923	0.9999	0.9908	0.9954	0.9999

 doi:10.20944/preprints202512.2225.v1

As reported in Table 6, the FFT representation yields near-ceiling discrimination across the strongest supervised and reconstruction-based detectors: RF and SVM achieve AUC ≈ 1.00 with ≥99.5% accuracy, while one-class GANomaly and AE (with percentile-based thresholds) match this operating range and allow explicit control of the false-positive/false-negative trade-off; in contrast, density-based DBSCAN prioritizes zero false positives but at the cost of noticeably higher false negatives, indicating a more conservative but less sensitive detection regime.

4.4. Wavelet-Based Feature Representation and Detector Suite

Wavelet-domain features were constructed to capture localized, transient deviations in the windowed traffic stream through a compact time–frequency descriptor that remains suitable for embedded-oriented inference. The implementation preserves the strict benchmark separation enforced by the pipeline, where all training, calibration, and any model-selection steps are performed using only training-derived windows, and the held-out test windows are used exclusively for final reporting. This design prevents test leakage during feature construction and operating-point selection, ensuring that reported performance reflects genuine generalization under the predefined evaluation regime.

In the implemented configuration, each preprocessed window is transformed into a fixed-length 42-dimensional wavelet feature vector, obtained from a multi-level discrete wavelet decomposition and summarized into a compact coefficient-based descriptor. After windowing and wavelet feature extraction, the resulting subsets comprise 4382 fit windows, 1094 validation windows, and 2571 held-out test windows, each represented by 42 wavelet features. This setup defines the wavelet-based input space used by the detector suite, allowing a consistent comparison against other feature representations under the same fit/validation/test partitioning.

To evaluate the robustness of wavelet-domain representations under the same strict UNSW-NB15 protocol, the detection suite was re-applied to the Wavelet feature matrix derived from multi-level discrete wavelet decomposition over sliding histories, yielding a compact 42-dimensional descriptor (7 statistics per monitored signal). In the supervised setting, the gradient-boosted tree family achieved the strongest overall performance, indicating that the wavelet representation preserves highly separable structure for discriminative learning. LightGBM provided the best aggregate results, reaching Acc = 0.9949, Prec = 0.9923, Rec = 0.9999, F1 = 0.9962, and AUC = 0.9999, with near-zero missed attacks and only two benign false alarms. XGBoost followed closely with Acc = 0.9897, Prec = 0.9885, Rec = 0.9961, F1 = 0.9923, and AUC = 0.9996, confirming consistent attack sensitivity at a slightly higher benign error rate. Classical baselines remained highly competitive: SVM achieved Acc = 0.9872, Prec = 0.9847, Rec = 0.9961, F1 = 0.9904, AUC = 0.9994, while RF reached Acc = 0.9821, Prec = 0.9773, Rec = 0.9961, F1 = 0.9866, AUC = 0.9988, both demonstrating that wavelet features are sufficient for near-perfect supervised discrimination when labels are available.

In contrast, purely unsupervised clustering baselines on the same wavelet representation show substantially larger trade-offs, clarifying the practical benefit of anomaly scoring under benign-only or label-scarce regimes. The K-Means proxy baseline (k = 2) exhibits limited separability (Acc = 0.6282, F1 = 0.7320, AUC = 0.6026), suggesting that coarse partitioning of the wavelet space does not align well with the benign/attack boundary. Under a benign-only one-class formulation, K-Means (k = 5) becomes markedly more conservative: at p95 it yields high precision but reduced sensitivity (Acc = 0.7231, Prec = 0.9521, Rec = 0.6139, F1 = 0.7465, AUC = 0.8979), while stricter thresholds further suppress false alarms at the cost of sharp recall degradation (e.g., p99: Acc = 0.5410, Prec = 0.9762, Rec = 0.3166, F1 = 0.4781, with unchanged AUC because the underlying score ranking remains similar). DBSCAN one-class provides a stronger unsupervised alternative by enforcing a tighter notion of density-consistent "normality": at p95 it attains Acc = 0.8564, Prec = 0.9721, Rec = 0.8069, F1 = 0.8819, AUC = 0.9753, and as eps becomes more permissive (p97→p99) precision approaches perfection (p99: Prec≈0.9999) while recall decreases (Rec = 0.6062), illustrating a clear sensitivity–specificity control via percentile-based eps selection.

 doi:10.20944/preprints202512.2225.v1

Reconstruction-based one-class detectors remain the most reliable benign-only mechanisms on wavelet features, offering high AUCs and tunable operating points. The AE one-class detector achieves its best-balanced performance at p97, reaching Acc = 0.9538, Prec = 0.9547, Rec = 0.9768, F1 = 0.9656, AUC = 0.9882, whereas p95 prioritizes recall (Rec = 0.9961) with lower precision (Prec = 0.9085), and p99 reduces false positives while incurring additional missed attacks (Rec = 0.9112). The GAN-based one-class scorer is consistently more conservative on this representation: at p95 it achieves Acc = 0.8103, Prec = 0.9602, Rec = 0.7452, F1 = 0.8391, AUC = 0.9612, with stricter percentiles reducing recall further (p99: Rec = 0.5560) despite maintaining very high precision (Prec = 0.9863). Overall, the wavelet results position LightGBM/XGBoost as the top-performing detectors in the supervised regime, while the AE one-class (p97) emerges as the strongest benign-only option among the evaluated unsupervised/one-class baselines, and DBSCAN provides an interpretable density-based alternative with a controllable precision–recall trade-off.

The wavelet-based detection results obtained under the strict UNSW-NB15 benchmark protocol are summarized in Table 7.

As reported in Table 7, the wavelet-based representation also delivers near-ceiling discrimination for the best supervised learners: LightGBM and XGBoost achieve AUC ≈ 1.00 with ~99–99.5% accuracy, confirming that the proposed multi-level wavelet descriptor retains highly separable structure at the window level. Among benign-only methods, the AE one-class model provides the most stable high-AUC operating range (AUC ≈ 0.99) and supports explicit sensitivity–specificity tuning via percentile thresholds, whereas the GAN-based one-class scorer remains more conservative, preserving high precision but incurring larger recall drops as the threshold is tightened.

Table 7. Wavelet-based window-level detection performance on the held-out UNSW-NB15 test subset under the strict benchmark protocol.

Model	Acc	Prec	Rec	F1	AUC
SVM	0.9872	0.9847	0.9961	0.9904	0.9994
RF	0.9821	0.9773	0.9961	0.9866	0.9988
XGBoost	0.9897	0.9885	0.9961	0.9923	0.9996
LightGBM	0.9949	0.9923	0.9999	0.9962	0.9999
K-Means proxy (k = 2)	0.6282	0.7021	0.7645	0.7320	0.6026
K-Means one-class (k = 5, p95)	0.7231	0.9521	0.6139	0.7465	0.8979
K-Means one-class (k = 5, p97)	0.5974	0.9811	0.4015	0.5699	0.8979
K-Means one-class (k = 5, p99)	0.5410	0.9762	0.3166	0.4781	0.8979
DBSCAN one-class (p95)	0.8564	0.9721	0.8069	0.8819	0.9753
DBSCAN one-class (p97)	0.8205	0.9846	0.7413	0.8458	0.9833
DBSCAN one-class (p99)	0.7385	0.9999	0.6062	0.7548	0.9894
AE one-class (p95)	0.9308	0.9085	0.9961	0.9503	0.9882
AE one-class (p97)	0.9538	0.9547	0.9768	0.9656	0.9882
AE one-class(p99)	0.9282	0.9793	0.9112	0.9440	0.9882
GAN one-class (p95)	0.8103	0.9602	0.7452	0.8391	0.9612
GAN one-class (p97)	0.7231	0.9809	0.5946	0.7404	0.9612
GAN one-class (p99)	0.7000	0.9863	0.5560	0.7111	0.9612

In contrast, clustering baselines exhibit clearer limitations: K-Means proxy shows weak separability, while K-Means one-class becomes high-precision but increasingly insensitive under stricter percentiles; DBSCAN offers a stronger density-based alternative with very high precision and competitive AUC, but its recall decreases as eps becomes more permissive, reflecting a conservative regime that trades missed attacks for fewer false alarms.

4.5. Kalman-Based Feature Representation and Detector Suite

The Kalman-based processing stage was introduced to obtain a compact, noise-robust representation of window-level traffic descriptors by explicitly smoothing short-horizon fluctuations

 doi:10.20944/preprints202512.2225.v1

and emphasizing deviations from expected dynamics. The implementation follows the same strict benchmark discipline used throughout the pipeline: all parameter choices, calibration actions, and any optional model-selection procedures are derived only from training-originated windows, while the held-out test subset remains untouched until final evaluation. As a result, the Kalman-based feature stream is generated and assessed without any information flow from the test subset into model fitting or threshold selection.

In the current implementation, Kalman filtering is applied at the window level by interpreting each window descriptor as an ordered observation sequence and performing recursive prediction–update smoothing with fixed filter parameters. The filtered trajectory and innovation behavior are then summarized into a compact 6-dimensional Kalman feature vector per window, yielding feature matrices of (4381, 6) for the fit subset, (1093, 6) for the validation subset, and (2570, 6) for the held-out test subset. This transformation reduces dimensionality substantially relative to the original 42-feature window representation, while preserving the exact same partition structure, enabling the detector suite to operate on a lightweight Kalman-derived feature stream aligned with embedded deployment constraints.

To evaluate the effectiveness of Kalman-smoothed statistical window descriptors under the same strict UNSW-NB15 protocol, the detection suite was re-applied to the Kalman feature stream, obtained by filtering each numeric window-level statistic with a 1D Kalman estimator to suppress short-term fluctuations while preserving longer-term dynamics. In the supervised setting, tree-based ensembles and gradient boosting achieved the strongest overall performance, indicating that Kalman smoothing yields a highly discriminative feature space for window-level attack detection. LightGBM and XGBoost provided the best aggregate results, both reaching Acc = 0.9926, Prec = 0.9963, Rec = 0.9926, and F1 = 0.9945, with AUC = 0.9998 and AUC = 0.9993, respectively, confirming near-ceiling separability with minimal error rates on the held-out test split. RF remained comparably strong (Acc = 0.9901, Prec = 0.9926, Rec = 0.9926, F1 = 0.9926, AUC = 0.9995), reinforcing that the Kalman-smoothed descriptors preserve stable decision structure for classical ensemble learning. By comparison, SVM was notably weaker on this representation (Acc = 0.9231, Prec = 0.9167, Rec = 0.9742, F1 = 0.9445, AUC = 0.9444), suggesting that a margin-based RBF classifier does not exploit the Kalman feature geometry as effectively as boosted and bagged tree learners.

In contrast, purely unsupervised clustering baselines on the same Kalman feature stream exhibit more pronounced trade-offs, clarifying the value of calibrated anomaly scoring under benign-only or label-scarce regimes. The K-Means proxy baseline (k = 2) achieves moderate hard-label performance (Acc = 0.9181, Prec = 0.9103, Rec = 0.9742, F1 = 0.9412) but yields an AUC below chance (AUC = 0.4192), indicating that the cluster-to-class mapping can mimic a reasonable decision rule while the underlying distance score provides limited ranking fidelity for attack likelihood. Under a benign-only one-class formulation, K-Means (k = 5) becomes more conservative: at p95 it yields Acc = 0.9107, Prec = 0.9681, Rec = 0.8967, F1 = 0.9310, AUC = 0.9705, while stricter thresholds reduce false positives at the cost of substantially higher false negatives (e.g., p99: Acc = 0.7494, Prec = 0.9670, Rec = 0.6494, F1 = 0.7770, with unchanged AUC because the score ordering remains similar). DBSCAN enforces an even stricter density-consistency notion of "normality" and is therefore highly conservative in this feature space: at p95 it attains Acc = 0.5434, Prec = 0.9579, Rec = 0.3358, F1 = 0.4973, AUC = 0.9888, and as the operating point is tightened (p97→p99) precision approaches perfection (p99: Prec = 1.0000) while recall collapses (Rec = 0.0886), reflecting a regime that strongly minimizes false alarms but misses a large fraction of attacks.

Reconstruction-based one-class detectors provide the most reliable benign-only operating range on the Kalman feature stream, offering high AUCs and tunable thresholds. The AE detector achieves consistently strong performance across percentiles and approaches supervised accuracy, with its best-balanced configuration at p97 reaching Acc = 0.9801, Prec = 0.9852, Rec = 0.9852, F1 = 0.9852, AUC = 0.9956; by comparison, p95 prioritizes recall (Rec = 0.9926) with a modest precision reduction (Prec = 0.9711), while p99 shifts toward fewer false positives (Prec = 0.9888) at a small recall cost (Rec = 0.9779). The GAN-based scorer is similarly competitive, achieving Acc = 0.9677, Prec = 0.9674, Rec =

 doi:10.20944/preprints202512.2225.v1

0.9852, F1 = 0.9762, AUC = 0.9953 at p95 and improving to Acc = 0.9801, Prec = 0.9852, Rec = 0.9852, F1 = 0.9852, AUC = 0.9953 at p97–p99, indicating stable anomaly ranking and a robust benign-only decision rule under percentile thresholding. Overall, the Kalman results position LightGBM/XGBoost/RF as the highest-performing detectors when labels are available, while AE and GAN (p97–p99) provide strong benign-only alternatives with controllable sensitivity–specificity trade-offs, and density-based DBSCAN remains an interpretable but overly conservative baseline in this smoothed feature space.

The Kalman-based detection results obtained under the strict UNSW-NB15 benchmark protocol are summarized in Table 8.

Table 8. Kalman-based window-level detection performance on the UNSW-NB15 test subset under the strict benchmark protocol.

Model	Acc	Prec	Rec	F1	AUC
SVM	0.9231	0.9167	0.9742	0.9445	0.9444
RF	0.9901	0.9926	0.9926	0.9926	0.9995
XGBoost	0.9926	0.9963	0.9926	0.9945	0.9993
LightGBM	0.9926	0.9963	0.9926	0.9945	0.9998
K-Means proxy (k = 2)	0.9181	0.9103	0.9742	0.9412	0.4192
K-Means one-class (k = 5, p95)	0.9107	0.9681	0.8967	0.9310	0.9705
K-Means one-class (k = 5, p97)	0.8189	0.9670	0.7565	0.8489	0.9705
K-Means one-class (k = 5, p99)	0.7494	0.9670	0.6494	0.7770	0.9705
DBSCAN one-class (p95)	0.5434	0.9579	0.3358	0.4973	0.9888
DBSCAN one-class (p97)	0.4591	0.9492	0.2066	0.3394	0.9888
DBSCAN one-class (p99)	0.3871	1.0000	0.0886	0.1627	0.9888
AE one-class (p95)	0.9752	0.9711	0.9926	0.9818	0.9956
AE one-class (p97)	0.9801	0.9852	0.9852	0.9852	0.9956
AE one-class(p99)	0.9777	0.9888	0.9779	0.9833	0.9956
GAN one-class (p95)	0.9677	0.9674	0.9852	0.9762	0.9953
GAN one-class (p97)	0.9801	0.9852	0.9852	0.9852	0.9953
GAN one-class (p99)	0.9801	0.9852	0.9852	0.9852	0.9953

As reported in Table 8, the Kalman-feature stream yields near-ceiling discrimination for the strongest supervised learners. Both XGBoost and LightGBM achieve Acc = 0.9926, Prec = 0.9963, Rec = 0.9926, and F1 = 0.9945, with AUC = 0.9993 and AUC = 0.9998, respectively, corresponding to one false alarm and two missed attacks on the held-out test split. Overall, these results indicate that temporal smoothing via the proposed Kalman filtering stage preserves highly separable structure in the compact statistical window descriptors, enabling robust, low-error attack detection under the strict UNSW-NB15 protocol.

4.6. PCA-Based Feature Representation and Anomaly Visualization

The PCA model was fitted on the training split and configured to retain at least 95% of the explained variance, yielding n_components = 21 (cumulative explained variance ratio 0.9540). After transformation, the PCA feature matrices had shapes of (4381, 21) for training (fit), (1093, 21) for validation, and (2570, 21) for testing. Beyond dimensionality reduction, PCA enabled an interpretable unsupervised diagnostic via reconstruction error: samples were projected into the reduced subspace and reconstructed back to the standardized feature space, with the per-window MSE used as an anomaly score. On the validation split, the reconstruction error had mean = 0.0457 and std = 0.0351, and the 95th-percentile threshold was set to thr(p95) = 0.1078. Applying this validation-derived threshold to the test split resulted in 147 out of 2570 windows being flagged as anomalous, corresponding to 5.72% of the test data, while the test reconstruction error exhibited mean = 0.0452 and std = 0.0444.

 doi:10.20944/preprints202512.2225.v1

For interpretability, PCA was also employed as an unsupervised anomaly-visualization mechanism by reconstructing samples from the low-dimensional space and computing a reconstruction-error score per window. As illustrated in Figure 10, the workflow provides three complementary views: (a) the 2D PCA projection highlighting the global structure of the traffic manifold, (b) the distribution of reconstruction errors together with a data-driven p95 threshold, and (c) an anomaly overlay in PCA space in which flagged windows are emphasized relative to the dominant normal region. This combined visualization supports transparent validation of deviation intensity and spatial separability, which is especially useful in embedded cyber-security monitoring when explainability is required.

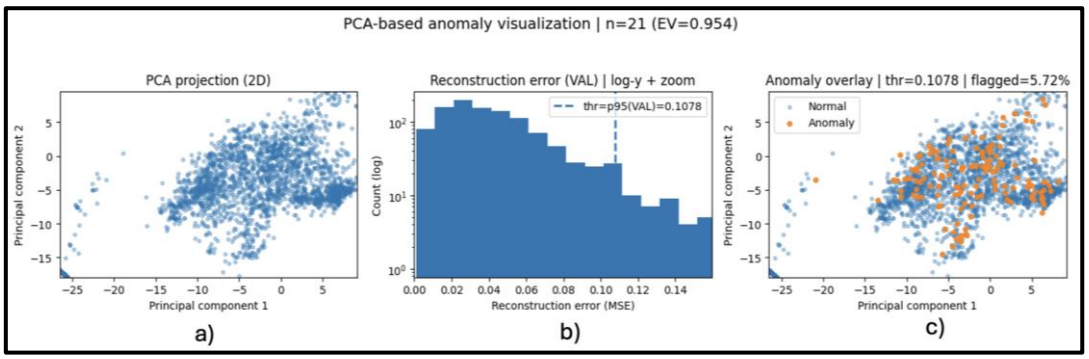

Figure 10. PCA-based anomaly visualization on the evaluated network-traffic subset: (**a**) 2D PCA projection, (**b**) reconstruction-error histogram with the p95 threshold, and (**c**) anomaly overlay in PCA space (p95).

Under the PCA-based setup (n_components = 21; cumulative explained variance 0.9540), the supervised detectors delivered clearly stronger and more stable performance than the unsupervised/density-based alternatives. In particular, Random Forest achieved near-ceiling validation performance (Acc = 0.9890, Prec = 0.9919, Rec = 0.9919, F1 = 0.9919 with a validation-tuned threshold), indicating that the PCA representation preserves highly discriminative structure for learning-based classification. Comparable high validation scores were also obtained by gradient-boosting models, including XGBoost (Acc = 0.9963, Prec = 0.9960, Rec = 0.9987, F1 = 0.9973) and LightGBM (Acc = 0.9945, Prec = 0.9933, Rec = 0.9987, F1 = 0.9960), confirming that tree-ensemble learners can reliably separate benign versus attack patterns in the reduced feature space. By contrast, density- and distance-driven unsupervised approaches were markedly less reliable under cross-split distribution shifts: the DBSCAN-style kNN-distance method exhibited severe instability when applying a validation-derived threshold to the test split, flagging 67.74% of test windows as anomalous and yielding poor detection quality (F1 = 0.4265, AUC = 0.2812). Similar fragility was observed for generative and centroid-based methods (AE/GAN/K-Means), where validation-calibrated thresholds produced excessively high alarm rates and weaker class-separation signals, demonstrating that these unsupervised schemes are substantially more sensitive to distribution drift and often require recalibration or additional adaptation mechanisms to remain operational.

The comparative performance of all evaluated PCA-based detectors reported using the five-standard metrics (Accuracy, Precision, Recall, F1-score, and AUC), is summarized in Table 9, highlighting the consistently superior validation performance of the supervised ensembles (RF/XGBoost/LightGBM) relative to the unsupervised density- and reconstruction-based baselines.

Table 9. Comparative evaluation of PCA-based detectors.

Model	Acc	Prec	Rec	F1	AUC
SVM	0.9954	0.9933	1.0000	0.9967	0.7494
RF	0.9890	0.9919	0.9919	0.9919	0.8028
XGBoost	0.9963	0.9960	0.9987	0.9973	0.7785

 doi:10.20944/preprints202512.2225.v1

LightGBM	0.9945	0.9933	0.9987	0.9960	0.7898
K-Means	0.9314	0.9539	0.9449	0.9494	0.5002
DBSCAN	0.2957	0.3866	0.4756	0.4265	0.2812
AE	0.8719	0.8784	0.9422	0.9092	0.5724
GAN	0.8454	0.8468	0.9435	0.8926	0.4344

These results show that PCA-based reconstruction error can serve as a practical and transparent mechanism for highlighting structural deviations in embedded cyber-security traffic without requiring labels. The approach combines compact dimensionality reduction with a validation-calibrated percentile threshold, enabling interpretable anomaly scoring through both error-distribution inspection and latent-space visualization.

However, the comparative table also indicates that supervised learners operating on the same PCA representation consistently deliver substantially stronger detection performance. In our experiments, ensemble-based models (e.g., RF, XGBoost, and LightGBM) achieved near-ceiling validation scores under a validation-tuned operating point, confirming that the reduced feature space preserves highly discriminative structure for benign-versus-attack separation.

By contrast, unsupervised density- and reconstruction-driven methods were notably more sensitive to cross-split distribution shifts: validation-derived thresholds often led to excessive alarm rates on the test split and weaker ranking quality (as reflected by degraded AUC/F1). Therefore, the PCA module is best interpreted as a lightweight, explainable surveillance layer that complements the primary supervised detectors, particularly in scenarios where labels are limited, while acknowledging that robust deployment may require periodic threshold recalibration or drift-aware adaptation.

4.7. Embedded Deployment Framework

Based on the final experimental evidence and an implementation-oriented cost perspective, the final system configuration was selected to balance detection effectiveness with computational practicality in embedded/edge settings. On the signal-processing side, the FFT-based transformation was retained as the core representation step because it provides a compact and stable descriptor while maintaining predictable runtime under sliding-window operation, which is essential for resource-constrained deployment. On the learning side, RF was chosen as the primary detector due to its consistently strong performance together with low inference overhead and straightforward integration into a real-time pipeline. Consequently, the final framework adopts an FFT-driven feature representation followed by RF inference as the default processing chain for the proposed embedded deployment scenario.

To avoid interpreting the framework as a mere juxtaposition of modules, each component is included with a distinct functional role and a non-overlapping objective. The FFT stage is retained in the deployment chain because it provides an explicit transform-domain representation that is computationally predictable under sliding windows and exposes stable band-energy patterns that are not guaranteed to emerge from purely data-driven compression. In contrast, PCA is treated primarily as a projection mechanism that reduces dimensionality and supports compact anomaly scoring/visualization; it does not replace spectral analysis but can be applied either to time-domain descriptors or to spectral features to obtain low-dimensional structure and to facilitate interpretation of separability and failure modes. Wavelet features are evaluated as a complementary multi-resolution alternative to FFT, motivated by the fact that embedded traffic and event-driven behaviors can exhibit non-stationary or transient patterns where time–frequency localization is more informative than global spectral concentration; consequently, wavelets serve as an ablation/comparison path rather than a mandatory deployment step. Kalman filtering contributes at a different level by providing minimum-variance state estimation for noisy or jittered measurements, improving temporal consistency of the descriptor stream before transform-domain mapping, which is relevant under quantization, sampling variability, or sensor-like acquisition effects at the edge. Finally, deep unsupervised scores (AE/GAN-based) are positioned as auxiliary detectors that add

 doi:10.20944/preprints202512.2225.v1

value in scenarios where labeled data are limited, novel attacks appear, or distribution shift occurs; in such cases, reconstruction- or likelihood-based scores provide an orthogonal signal to supervised ensembles (RF/XGBoost), which primarily optimize discrimination under the training distribution. This separation of roles clarifies how spectral processing, filtering, projection-based analysis, and supervised/unsupervised decision rules interact within a coherent signal-to-decision pipeline and why the final embedded configuration focuses on the FFT→RF chain while retaining other modules for comparative evaluation and extended robustness coverage.

To improve reproducibility and to provide a compact, structured view of the proposed workflow, Algorithm 1 summarizes the end-to-end methodology from raw UNSW-NB15 flows to the final embedded framework selection, evaluation, robustness testing, and deployment profiling.

Algorithm 1. End-to-End Methodology for Embedded Cyber-Attack Detection

Input:
- UNSW-NB15 train set Dtrain and test set Dtest (official split)
- Window length W, hop H
- Descriptor set C (numeric flow attributes)
- FFT descriptor subset Cfft ⊂ C
- FFT length L, frequency bands B
- Final classifier MF (Random Forest, θRF)
- Noise levels Σ and random seeds S
- Bootstrap resamples R

Output:
- Trained model MF and final framework metrics
- Test metrics (Acc, Prec, Rec, F1, AUC)
- Bootstrap 95% CI and robustness (ΔAUC, ΔRec)
- Deployment profiling metrics (latency, memory, footprint)

Load Dtrain and Dtest using the official UNSW-NB15 predefined split.
Select numeric base attributes C from the intersection of train/test columns.
For each dataset D ∈ {Dtrain, Dtest}:
    Segment D into overlapping windows using (W, H).
    For each window, compute mean/std/min/max for each attribute in C→T(D).
    Assign window label ywin = 1 if any sample in window is malicious; else 0.
For each dataset D ∈ {Dtrain, Dtest}:
    For each descriptor d ∈ Cfft:
        Apply sliding rFFT over length L and extract spectral features over bands B.
    Concatenate features across descriptors → X(D); drop first (L − 1) steps.
Split X(Dtrain) into fit/validation (80/20); keep X(Dtest) held out.
Train MF on the fit subset using cost-sensitive learning (balanced class weighting).
Select the operating threshold and any hyperparameters on validation only.
Evaluate MF on held-out test: report Acc/Prec/Rec/F1/AUC and macro/per-class metrics.
Bootstrap test instances (R resamples) to estimate 95% CI for AUC (and Rec).
For each noise level α ∈ Σ and seed k ∈ S:
    Create Xtest' = Xtest + N(0, α·std(Xfit)).
    Compute AUCα, k and Recα, k; aggregate mean ± std; compute Δ vs. clean.
On target edge device, benchmark footprint, latency (mean/p50/p95), and peak RSS.

To strengthen interpretability within the final embedded framework, feature importance was extracted from the RF classifier to identify the network attributes that most strongly contribute to the decision function. The resulting ranking (Figure 11) highlights sttl, ct_state_ttl, rate, dload, and sload among the top contributors, together capturing a substantial fraction of the overall importance mass. These variables describe flow-level temporal behavior (TTL dynamics and state transitions) and traffic intensity (rate and load-related indicators), which are known to be discriminative for separating malicious activity from benign communications in network intrusion settings.

 doi:10.20944/preprints202512.2225.v1

Accordingly, the feature ranking provides a transparent view of the dominant cues leveraged by the final framework, complementing the deployment-oriented robustness and uncertainty analysis.

	Name	Importance
0	sttl	0.141648
1	flow_id	0.127475
2	ct_state_ttl	0.100930
3	rate	0.046808
4	sload	0.046792
5	dload	0.039321
6	sbytes	0.039275
7	smean	0.035066
8	ct_dst_src_ltm	0.034970
9	dbytes	0.032074

Figure 11. Feature importance ranking computed using the RF classifier.

In Figure 11, the background color intensity encodes the relative magnitude of feature importance, where darker shading indicates higher contribution to the RF decision function.

To complement point-estimate reporting and to support deployment-oriented claims for the final FFT→RF pipeline, both statistical uncertainty and robustness under controlled variability are quantified. First, uncertainty of the main operating metrics is estimated on the held-out test set via non-parametric bootstrap resampling of window instances (B resamples), yielding 95% confidence intervals for AUC and recall without imposing distributional assumptions. Second, robustness against feature-level noise/variability is assessed through Gaussian noise injection at inference time, where perturbations are scaled relative to the feature-wise standard deviation computed on the training-fit subset to avoid test leakage. Performance stability is summarized as the relative degradation with respect to clean inputs (ΔAUC and ΔRecall), reported across multiple noise levels and random seeds. This validation layer provides a practical stress test of the selected FFT-driven representation and the RF classifier under controlled perturbations consistent with embedded/edge operation, where moderate variability, quantization effects, and measurement noise can affect the observed feature stream.

Under clean conditions, the final FFT→RF pipeline achieved an AUC of 0.9679, with a bootstrap 95% confidence interval of [0.9582, 0.9760], and a recall estimate of 0.9103 with a 95% confidence interval of [0.8916, 0.9288]. The robustness assessment indicates a gradual and controlled degradation as the injected noise magnitude increases. In particular, the average AUC decreased by 0.0061 at $\sigma = 0.01$, by 0.0181 at $\sigma = 0.05$, and by 0.0450 at $\sigma = 0.10$ relative to the clean baseline, while recall remained above 0.89 even at the highest perturbation level. Table 10 summarizes the robustness statistics, while Figures 12 and 13 illustrate the AUC trend and the corresponding ΔAUC profile as a function of the injected noise level.

Table 10. Robustness of the final embedded framework under Gaussian noise injection.

Gaussian Noise Level (σ)	AUC (Mean ± Std)	ΔAUC vs. Clean	Recall (Mean ± Std)	ΔRecall vs. Clean
0.00	0.9679 ± 0.0000	0.0000	0.9186 ± 0.0000	0.0000
0.01	0.9618 ± 0.0021	−0.0061	0.9093 ± 0.0028	−0.0092
0.05	0.9498 ± 0.0043	−0.0181	0.9015 ± 0.0050	−0.0171

 doi:10.20944/preprints202512.2225.v1

0.10	0.9229 ± 0.0051	−0.0450	0.8903 ± 0.0033	−0.0283

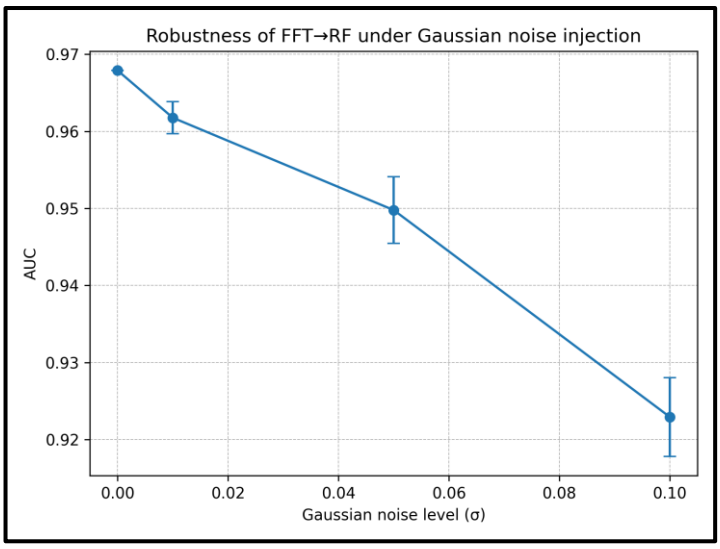

Figure 12. Robustness of the final embedded framework under Gaussian noise injection: AUC versus noise level σ (mean ± std across seeds).

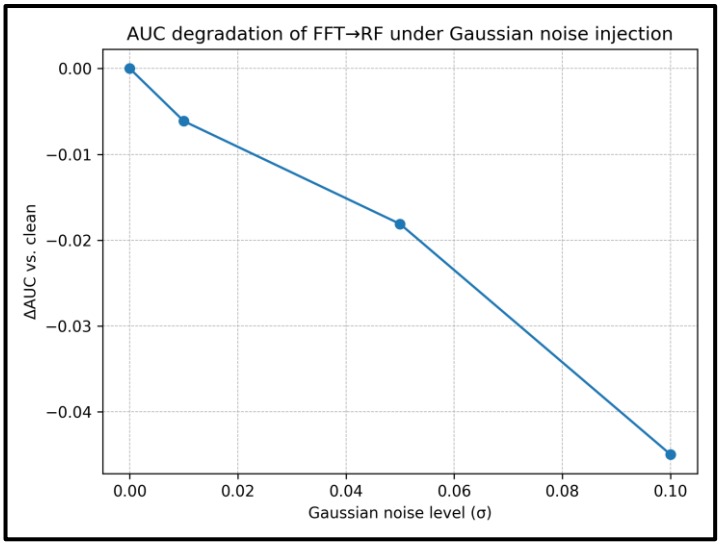

Figure 13. AUC degradation profile (ΔAUC relative to the clean condition) of the final embedded framework under Gaussian noise injection.

Class imbalance is treated explicitly through cost-sensitive learning and operating-point calibration. The RF backend is trained using class-weighted sampling (balanced class weighting) to increase the penalty associated with minority-class errors, while the decision threshold is selected on a validation split (rather than fixed at 0.5) to improve minority-class sensitivity under controlled false-positive behavior. In addition to overall accuracy, macro-averaged metrics and per-class Precision/Recall/F1 are reported to ensure that low-prevalence categories are not masked by majority-class dominance. Oversampling approaches (e.g., SMOTE) are not used as a primary mechanism of the final embedded configuration due to deployment overhead and leakage risk; when referenced, they are restricted to training-only comparative baselines.

To explicitly quantify the contribution of each representation module, an ablation-style comparison is summarized (Table 11) by fixing the classifier backbone (RF) and varying only the upstream feature-construction stage, thereby isolating the impact of each transform/filter choice on detection performance and embedded-oriented cost under an identical evaluation protocol.

 doi:10.20944/preprints202512.2225.v1

Table 11. Ablation-style comparison of representation modules with a fixed RF backbone.

Representation Module	Feature Dim.	Acc	Prec	Rec	F1	AUC	Key Interpretation
FFT + RF	42	0.9974	0.9970	0.9999	0.9985	0.9999	Best overall trade-off for deployment: near-ceiling detection with predictable sliding-window runtime and explicit feasibility envelope.
DWT + RF	42	0.9821	0.9773	0.9961	0.9866	0.9988	Strong AUC/recall but lower accuracy than FFT; best treated as offline/selectively enabled when transient localization is needed.
Kalman + RF	6	0.9901	0.9926	0.9926	0.9926	0.9995	Excellent performance with very low feature dimension; attractive for tight memory/compute budgets where "spectral richness" is not required.
PCA + RF	21	0.9890	0.9919	0.9919	0.9919	0.8028	PCA is best positioned as compact/interpretability support and anomaly visualization; not the primary discriminative driver compared to FFT/Kalman in the final embedded framework.

The ablation results indicate that, under a fixed RF decision rule and an identical evaluation discipline, detection performance is strongly shaped by the upstream representation. FFT-derived spectral summaries provide the highest overall separability on the held-out test subset, while the Kalman-derived stream achieves a competitive accuracy-compactness trade-off with a substantially reduced feature dimensionality. Wavelet features remain highly effective but exhibit slightly lower aggregate performance under the adopted configuration. The PCA module is primarily retained as a compact representation and interpretability layer; its supervised scores are reported under a validation-tuned operating point, and it is therefore interpreted as complementary rather than a strictly like-for-like replacement of the test-reported spectral/filter representations.

4.8. Final System Validation: Multi-Class Attack Category Detection and Edge Feasibility

Following the consolidation of the proposed pipeline into a deployment-ready configuration, this subsection presents the final validation stage, integrating attack-category performance analysis with edge feasibility profiling to substantiate practical embedded applicability.

The RF classifier was further evaluated in a multi-class setting to assess its ability to discriminate between all attack categories defined in the UNSW-NB15 taxonomy. Trained on the official training subset and assessed on the held-out test subset, the model attained a global accuracy of 0.944, indicating robust performance under heterogeneous traffic conditions. Figure 14 reports the resulting confusion matrix over the ten classes. Clear separability is observed for Generic, Normal, and Reconnaissance, which form compact and well-defined clusters, whereas DoS, Shellcode, and Backdoor remain comparatively more challenging. These confusions are consistent with temporal and behavioral overlap between certain attack patterns, class-imbalance effects, and the limited diversity of samples available for some categories, which collectively constrain separability at the category level.

 doi:10.20944/preprints202512.2225.v1

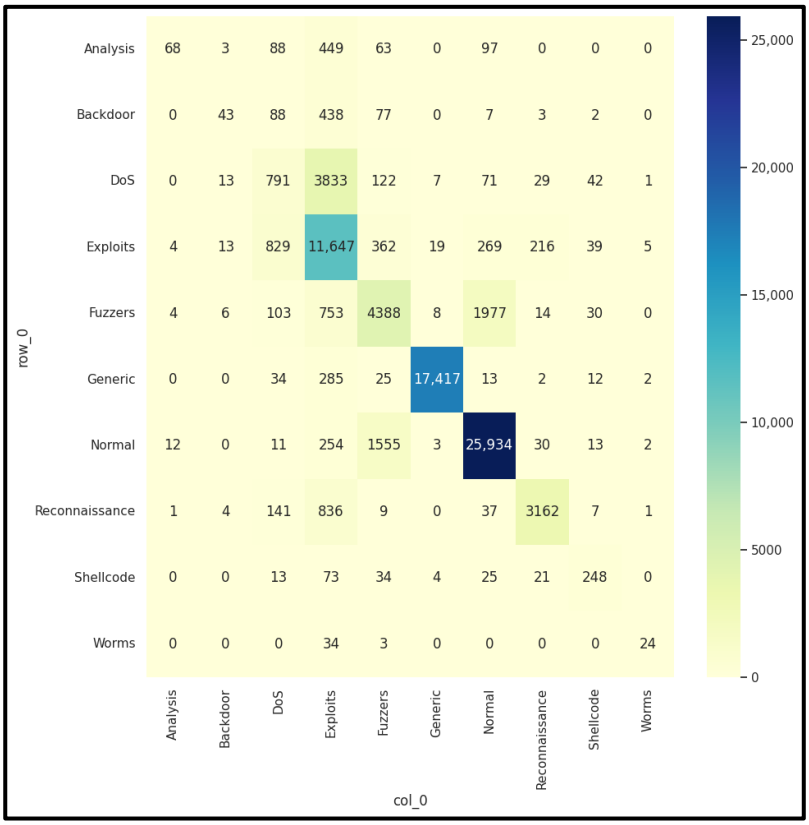

Figure 14. Confusion matrix of the RFClassifier across all attack categories.

To complement the confusion-matrix interpretation, Table 12 reports per-class performance (Precision, Recall, F1-score, and Specificity), revealing substantial variability in detection difficulty across attack categories. High scores are observed for majority classes with more consistent traffic signatures (e.g., Reconnaissance and Analysis), whereas minority or stealthier categories (e.g., Backdoor, Shellcode, Worms, and certain Exploit patterns) exhibit reduced recall and F1-score. This behavior is consistent with class imbalance and with partial feature-level overlap between specific attack behaviors and benign background traffic, which increases the likelihood of false negatives and cross-class confusions in multi-class intrusion detection.

Table 12. Per-class detection accuracy and confusion structure for each cyberattack category using the RFClassifier.

Attack Type	Accuracy %	Precision %	Recall %	F1 %	Specificity %
Reconnaissance	99.48	99.51	98.23	98.86	99.86
Analysis	98.53	97.44	98.50	97.97	98.55
DoS	96.74	84.45	79.90	82.11	98.48
Backdoor	97.86	84.05	74.98	79.26	99.18
Fuzzers	90.06	69.14	77.04	72.87	92.79
Shellcode	99.52	61.03	55.91	58.36	99.78
Worms	99.94	68.97	33.90	45.45	99.99
Exploits	91.96	36.65	36.23	36.44	95.75
Normal	98.80	32.82	16.29	21.77	99.65
Generic	98.80	23.97	15.90	19.11	99.54

The results indicate that the RF classifier provides strong overall performance within the evaluated protocol, while the per-class breakdown highlights that residual errors are concentrated in a subset of minority or overlapping attack categories. This observation motivates future extensions

 doi:10.20944/preprints202512.2225.v1

focusing on drift-aware evaluation, cross-domain validation, and enhanced temporal representations for improved sensitivity on challenging classes.

To substantiate the embedded/real-time applicability of the final embedded framework, deployment-oriented micro-benchmarks were conducted using the exact inference implementation intended for edge execution. The profiling reports model footprint, runtime latency, and memory footprint as primary feasibility indicators for resource-constrained deployment. The serialized RF model occupies approximately 336 KB, supporting lightweight storage on edge systems. During steady-state execution, the process-level memory footprint reached 476 MB RSS with a peak of 522 MB RSS, reflecting the Python 3.12.12 runtime environment and associated libraries (numpy: 2.0.2, pandas: 2.2.2, scikit-learn: 1.6.1, matplotlib: 3.10.0, joblib: 1.5.3, psutil: 5.9.5) rather than the tree model itself. In terms of runtime, RF inference exhibited a mean latency of 74.6 ms per call (p50: 75.7 ms, p95: 78.5 ms) for batch size 1, with similar timing for batch size 16 due to implementation overheads. End-to-end streaming-step profiling, including FFT-based feature computation for L = 64 and inference on the resulting 42-dimensional feature vector, yielded a mean latency of 72.9 ms (p50: 76.7 ms, p95: 80.1 ms). These measurements provide an explicit deployment envelope for the final framework under the adopted window-based processing regime, enabling reproducible assessment of real-time feasibility on edge-class hardware.

5. Conclusions

This study demonstrates that an embedded-oriented cyber-attack detection pipeline can be built by combining signal-level feature construction with resource-efficient learning mechanisms, while maintaining strong detection performance under a controlled and reproducible evaluation protocol. Using the official UNSW-NB15 split and a window-based representation of network flows, the experimental analysis covered both binary and multi-class intrusion detection and integrated complementary elements for accuracy, interpretability, and deployability, including compact feature design, cost-sensitive learning, and deployment-oriented validation.

The primary contribution of this work is to address a key research gap, namely the absence of a unified, deployment-oriented comparison that jointly evaluates signal-level representations and detector families under a consistent protocol, by consolidating these components into a reproducible benchmark and translating the comparative evidence into a concrete embedded-ready processing chain supported by uncertainty quantification, robustness assessment, and edge feasibility profiling.

Across the supervised baselines, ensemble learners provided consistently strong discrimination, and the final embedded configuration was selected by jointly considering effectiveness and implementation cost. An FFT-driven representation was retained as the core signal-processing stage because it yields a compact descriptor under predictable sliding-window runtime, while RF was selected as the primary detector due to its stable performance and low integration overhead for edge inference. The resulting embedded framework attains high ranking quality under clean conditions (AUC $\approx$ 0.968) and preserves robust separability across heterogeneous traffic patterns, while the multi-class evaluation further confirms that residual errors concentrate primarily in minority or behaviorally overlapping attack categories rather than in the dominant classes.

To strengthen methodological rigor beyond point estimates, statistical uncertainty and robustness under controlled variability are explicitly quantified for the final framework. Non-parametric bootstrap resampling on the held-out test set provides 95% confidence intervals for key operating metrics (e.g., AUC and recall), supporting reproducible uncertainty reporting without distributional assumptions. Robustness is further assessed via inference-time Gaussian perturbations scaled by training-fit feature variability to avoid test leakage, with stability summarized through ΔAUC and ΔRecall across multiple noise levels and random seeds. The observed degradation remains gradual, indicating that the selected FFT-based representation and the Random Forest backend maintain stable behavior under moderate feature-level variability consistent with quantization and measurement noise in edge scenarios.

 doi:10.20944/preprints202512.2225.v1

Embedded applicability is additionally substantiated through deployment-oriented micro-benchmarks using the exact inference implementation intended for edge execution. The serialized model footprint is approximately 336 KB, and runtime profiling reports mean/p50/p95 latency for RF-only inference and for the end-to-end streaming step that includes FFT computation. These results define an explicit feasibility envelope for real-time operation on edge-class ARM platforms (e.g., Jetson Nano), complementing predictive performance with measurable indicators of latency and memory usage under continuous execution.

Several limitations remain important for operational generalization. First, while robustness to controlled noise is quantified, broader distribution shifts require drift-aware evaluation and calibration policies, including time-sliced testing, rolling-window recalibration, and explicit reporting of edge false-positive behavior under drift. Second, cross-domain validation on additional datasets (e.g., CIC-IDS2017 and TON-IoT) is a high-priority extension, requiring careful harmonization of feature schemas and label taxonomies to avoid confounded conclusions. Third, deployment realism can be further strengthened by incorporating embedded-specific traces (e.g., CAN-bus or microcontroller/IoT telemetry) and by reporting energy-per-inference and CPU utilization under native execution. Finally, future work should expand stress testing to adversarial perturbations and should provide systematic component-wise ablations and failure-mode analysis to quantify how performance and error propagation evolve across stages of the pipeline in realistic embedded conditions.

Author Contributions: Conceptualization, G.-V.I. and S.-A.D.; methodology, S.-A.D. and G.-V.I.; software, S.-A.D. and R.-N.B.; validation, G.-V.I. and N.B.; formal analysis, G.-V.I. and N.B.; investigation, S.-A.D. and R.-N.B.; resources, G.-V.I. and N.B.; data curation, G.-V.I. and S.-A.D.; writing—original draft preparation, S.-A.D.; writing—review and editing, G.-V.I., R.-N.B. and N.B.; visualization, G.-V.I. and N.B.; supervision, G.-V.I. and N.B.; project administration N.B.; funding acquisition, N.B. All authors have read and agreed to the published version of the manuscript.

Funding: The research was fully supported by the PubArt program of the National University of Science and Technology POLITEHNICA Bucharest.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data supporting this research have been made available in this paper.

Conflicts of Interest: The authors declare that they have no conflict of interest.

References

1. Dragusin, S.A.; Bizon, N.; Bostinaru, R.N. Comprehensive Analysis of Cyber-Attack Techniques and Vulnerabilities in Communication Channels of Embedded Systems. In Proceedings of the 16th International Conference on Electronics, Computers and Artificial Intelligence, Iasi, Romania, 27–28 June 2024. https://doi.org/10.1109/ECAI61503.2024.10607432.
2. Dragusin, S.A.; Bizon, N.; Bostinaru, R.N.; Enescu, F.M.; Teodorescu, R.M.; Savulescu, C. Analysis of Vulnerabilities in Communication Channels Using an Integrated Approach Based on Machine Learning and Statistical Methods. In Proceedings of the 16th International Conference on Electronics, Computers and Artificial Intelligence, Iasi, Romania, 27–28 June 2024. https://doi.org/10.1109/ECAI61503.2024.10607483.
3. Borangiu, T.; Morariu, O.; Răileanu, S.; Trentesaux, D.; Leitão, P.; Barata, J. Digital transformation of manufacturing. Industry of the Future with Cyber-Physical Production Systems. *Rom. J. Inf. Sci. Technol.* **2020**, *23*, 3–37.
4. Dragusin, S.; Bizon, N. Emerging Cybersecurity Threats in Embedded Systems: A Review of Attack Techniques, Anomaly Detection, and AI-Based Prediction Approaches. *J. Electr. Eng. Electron. Control. Comput. Sci.* **2025**, *11*, 11–22.

 doi:10.20944/preprints202512.2225.v1

5. Trilles, S.; Hammad, S.S.; Iskandaryan, D. Anomaly detection based on Artificial Intelligence of Things: A Systematic Literature Mapping. *Internet Things* **2024**, *25*, 101063. https://doi.org/10.1016/J.IOT.2024.101063.
6. Adhikari, D.; Jiang, W.; Zhan, J.; Rawat, D.B.; Bhattarai, A. Recent advances in anomaly detection in Internet of Things: Status, challenges, and perspectives. *Comput. Sci. Rev.* **2024**, *54*, 100665. https://doi.org/10.1016/J.COSREV.2024.100665.
7. Zhang, Y.; Muniyandi, R.C.; Qamar, F. A Review of Deep Learning Applications in Intrusion Detection Systems: Overcoming Challenges in Spatiotemporal Feature Extraction and Data Imbalance. *Appl. Sci.* **2025**, *15*, 1552. https://doi.org/10.3390/APP15031552.
8. DeMedeiros, K.; Hendawi, A.; Alvarez, M. A Survey of AI-Based Anomaly Detection in IoT and Sensor Networks. *Sensors* **2023**, *23*, 1352. https://doi.org/10.3390/S23031352.
9. Morshedi, R.; Matinkhah, S.M. A Comprehensive Review of Deep Learning Techniques for Anomaly Detection in IoT Networks: Methods, Challenges, and Datasets. *Eng. Rep.* **2025**, *7*, e70415. https://doi.org/10.1002/ENG2.70415.
10. Reis, M.J.C.S.; Serôdio, C. Edge AI for Real-Time Anomaly Detection in Smart Homes. *Future Internet* **2025**, *17*, 179. https://doi.org/10.3390/FI17040179.
11. Tahri, R.; Jarrar, A.; Lasbahani, A.; Balouki, Y. A comparative study of Machine learning Algorithms on the UNSW-NB 15 Dataset. In Proceedings of the 4th International Conference on Computing and Wireless Communication Systems, Tangier, Morocco, 21–23 June 2022. https://doi.org/10.1051/ITMCONF/20224803002.
12. Rahman, M.; Saad, S.; Satam, P.; Mari, A.-G.; Zinca, D.; Dobrota, V. Development of a Machine-Learning Intrusion Detection System and Testing of Its Performance Using a Generative Adversarial Network. *Sensors* **2023**, *23*, 1315. https://doi.org/10.3390/S23031315.
13. Yang, T.; Qiao, Y.; Lee, B. Towards trustworthy cybersecurity operations using Bayesian Deep Learning to improve uncertainty quantification of anomaly detection. *Comput. Secur.* **2024**, *144*, 103909. https://doi.org/10.1016/J.COSE.2024.103909.
14. Kale, R.; Thing, V.L.L. Few-shot weakly-supervised cybersecurity anomaly detection. *Comput. Secur.* **2023**, *130*, 103194. https://doi.org/10.1016/J.COSE.2023.103194.
15. Goumidi, H.; Pierre, S. Real-Time Anomaly Detection in IoMT Networks Using Stacking Model and a Healthcare-Specific Dataset. *IEEE Access* **2025**, *13*, 70352–70365. https://doi.org/10.1109/ACCESS.2025.3563158.
16. Kopljar, D.; Drvar, V.; Babic, J.; Podobnik, V. XAAD–Post-Feedback Explainability for Active Anomaly Discovery. *IEEE Access* **2024**, *12*, 181914–181924. https://doi.org/10.1109/ACCESS.2024.3510233.
17. Villarreal-Vasquez, M.; Modelo-Howard, G.; Dube, S.; Bhargava, B. Hunting for Insider Threats Using LSTM-Based Anomaly Detection. *IEEE Trans. Dependable Secur. Comput.* **2023**, *20*, 451–462. https://doi.org/10.1109/TDSC.2021.3135639.
18. Khan, W.; Ishrat, M.; Ahmed, M.N.; Abidin, S.; Husain, M.; Izhar, M.; Zamani, A.T.; Hussain, M.R.; Ali, A. Enhancing Anomaly Detection in Attributed Networks Using Proximity Preservation and Advanced Embedding Techniques. *IEEE Access* **2025**, *13*, 42777–42796. https://doi.org/10.1109/ACCESS.2025.3544260.
19. Mansourian, P.; Zhang, N.; Jaekel, A.; Kneppers, M. Deep Learning-Based Anomaly Detection for Connected Autonomous Vehicles Using Spatiotemporal Information. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 16006–16017. https://doi.org/10.1109/TITS.2023.3286611.
20. Dragusin, S.A.; Bizon, N.; Bostinaru, R.N. A Brief Overview of Current Encryption Techniques Used in Embedded Systems: Present and Future Technologies. In Proceedings of the 15th International Conference on Electronics, Computers and Artificial Intelligence, Bucharest, Romania, 29–30 June 2023. https://doi.org/10.1109/ECAI58194.2023.10194034.
21. Zhang, D.; Feng, G.; Shi, Y.; Srinivasan, D. Physical Safety and Cyber Security Analysis of Multi-Agent Systems: A Survey of Recent Advances. *IEEE/CAA J. Autom. Sin.* **2021**, *8*, 319–333. https://doi.org/10.1109/JAS.2021.1003820.
22. Chirilă, A.; Sărăcin, C.; Deaconu, D.; Nicolescu, D.; Radulian, A. Remote monitoring and control system with increased operational technology cybersecurity resilience. *UPB Sci. Bull. Ser. C* **2024**, *86*, 223–232.

 doi:10.20944/preprints202512.2225.v1

23. Nguyen, T.H. Cybersecurity Logging & Monitoring Security Program, DigitalCommons@SHU, Sacred Heart University. School of Computer Science & Engineering Undergraduate Publications, 2022, Available online: https://digitalcommons.sacredheart.edu/computersci_stu/3/ (accessed on 29 January 2025).
24. Jing, X.; Yan, Z.; Pedrycz, W. Security data collection and data analytics in the internet: A survey. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 586–618. https://doi.org/10.1109/COMST.2018.2863942.
25. Acosta, J.C.; Medina, S.; Ellis, J.; Clarke, L.; Rivas, V.; Newcomb, A. Network Data Curation Toolkit: Cybersecurity Data Collection, Aided-Labeling, and Rule Generation. In Proceedings of the Military Communications Conference, San Diego, CA, USA, 29 November–2 December 2021. https://doi.org/10.1109/MILCOM52596.2021.9653049.
26. Shashanka, M.; Shen, M.Y.; Wang, J. User and entity behavior analytics for enterprise security. In Proceedings of the International Conference on Big Data, Washington, DC, USA, 5–8 December 2016. https://doi.org/10.1109/BIGDATA.2016.7840805.
27. Agoramoorthy, M.; Ali, A.; Sujatha, D.; Michael Raj, T.F.; Ramesh, G. An Analysis of Signature-Based Components in Hybrid Intrusion Detection Systems. In Proceedings of the Intelligent Computing and Control for Engineering and Business Systems, Chennai, India, 14–15 December 2023. https://doi.org/10.1109/ICCEBS58601.2023.10449209.
28. Oppliger, R.; Grunert, A.; Michael, J.B. How to Measure Cybersecurity and Why Heuristics Matter. *Computer* **2024**, *57*, 111–115. https://doi.org/10.1109/MC.2023.3334054.
29. Noble, J.; Adams, N.M. Correlation-Based Streaming Anomaly Detection in Cyber-Security. In Proceedings of the 16th International Conference on Data Mining Workshops, Barcelona, Spain, 12–15 December 2016. https://doi.org/10.1109/ICDMW.2016.0051.
30. Dragusin, S.A.; Bizon, N.; Bostinaru, R.N.; Toma, D. Command Recognition System Using Convolutional Neural Networks. In Proceedings of the 16th International Conference on Electronics, Computers and Artificial Intelligence, Iasi, Romania, 27–28 June 2024. https://doi.org/10.1109/ECAI61503.2024.10607517.
31. Lappas, D.; Argyriou, V.; Makris, D. Fourier transformation autoencoders for anomaly detection. In Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Toronto, ON, Canada, 6–12 June 2021. https://doi.org/10.1109/ICASSP39728.2021.9415010.
32. Collins Jackson, A.; Lacey, S. Seasonality and Anomaly Detection in Rare Data Using the Discrete Fourier Transformation. In Proceedings of the 1st International Conference on Digital Data Processing, London, UK, 15–17 November 2019. https://doi.org/10.1109/DDP.2019.00013.
33. Jiang, D.; Zhang, P.; Xu, Z.; Yao, C.; Qin, W. A wavelet-based detection approach to traffic anomalies. In Proceedings of the 7th International Conference on Computational Intelligence and Security, Sanya, China, 3–4 December 2011. https://doi.org/10.1109/CIS.2011.222.
34. Golgowski, M.; Osowski, S. Detection of bearing failures using wavelet transformation and machine learning approach. In Proceedings of the International Joint Conference on Neural Networks, Padua, Italy, 18–23 July 2022. https://doi.org/10.1109/IJCNN55064.2022.9892755.
35. Thill, M.; Konen, W.; Bäck, T. Online Adaptable Time Series Anomaly Detection with Discrete Wavelet Transforms and Multivariate Gaussian Distributions. *Arch. Data Sci. Ser. A* **2018**, *5*, 17. https://doi.org/10.5445/KSP/1000087327/04.
36. Bostinaru, R.N.; Bizon, N.; Dragusin, S.A.; Iana, G.V.; Toma, D. Dimensionality Reduction with Principal Component Analysis for Fire and Non-Fire Audio Classification: A New Approach. In Proceedings of the 17th International Conference on Electronics, Computers and Artificial Intelligence, Targoviste, Romania, 26–27 June 2025. https://doi.org/10.1109/ECAI65401.2025.11095534.
37. Dani, S.K.; Thakur, C.; Nagvanshi, N.; Singh, G. Anomaly Detection using PCA in Time Series Data. In Proceedings of the International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation, Gwalior, India, 14–16 March 2024. https://doi.org/10.1109/IATMSI60426.2024.10502929.
38. Knorn, F.; Leith, D.J. Adaptive Kalman filtering for anomaly detection in software appliances. In Proceedings of the IEEE INFOCOM Workshops, Phoenix, AZ, USA, 13–18 April 2008. https://doi.org/10.1109/INFOCOM.2008.4544581.

 doi:10.20944/preprints202512.2225.v1

39. Ji, I.H.; Lee, J.H.; Kang, M.J.; Park, W.J.; Jeon, S.H.; Seo, J.T. Artificial Intelligence-Based Anomaly Detection Technology over Encrypted Traffic: A Systematic Literature Review. *Sensors* **2024**, *24*, 898. https://doi.org/10.3390/S24030898.
40. Zhang, X.; Gu, C.; Lin, J. Support vector machines for anomaly detection. In Proceedings of the 6th World Congress on Intelligent Control and Automation, Dalian, China, 21–23 June 2006. https://doi.org/10.1109/WCICA.2006.1712831.
41. Primartha, R.; Tama, B.A. Anomaly detection using random forest: A performance revisited In Proceedings of the International Conference on Data and Software Engineering, Palembang, Indonesia, 1–2 November 2017. https://doi.org/10.1109/ICODSE.2017.8285847.
42. Elmrabit, N.; Zhou, F.; Li, F.; Zhou, H. Evaluation of Machine Learning Algorithms for Anomaly Detection. In Proceedings of the International Conference on Cyber Security and Protection of Digital Services, Dublin, Ireland, 15–19 June 2020. https://doi.org/10.1109/CYBERSECURITY49315.2020.9138871.
43. Kumari Sheetanshu, R.; Singh, M.K.; Jha, R.; Singh, N.K. Anomaly detection in network traffic using K-mean clustering. In Proceedings of the 3rd International Conference on Recent Advances in Information Technology, Dhanbad, India, 3–5 March 2016. https://doi.org/10.1109/RAIT.2016.7507933.
44. Ranjith, R.; Athanesious, J.J.; Vaidehi, V. Anomaly detection using DBSCAN clustering technique for traffic video surveillance. In Proceedings of the 7th International Conference on Advanced Computing, Chennai, India, 15–17 December 2015. https://doi.org/10.1109/ICOAC.2015.7562795.
45. Kiziltas, B.; Gul, E. Network Anomaly Detection with Convolutional Neural Network Based Auto Encoders. In Proceedings of the 28th Signal Processing and Communications Applications Conference, Gaziantep, Turkey, 5–7 October 2020. https://doi.org/10.1109/SIU49456.2020.9302202.
46. Chen, Z.; Yeo, C.K.; Lee, B.S.; Lau, C.T. Autoencoder-based network anomaly detection. In Proceedings of the Wireless Telecommunications Symposium, Phoenix, AZ, USA, 17–20 April 2018. https://doi.org/10.1109/WTS.2018.8363930.
47. Kumarage, T.; Ranathunga, S.; Kuruppu, C.; De Silva, N.; Ranawaka, M. Generative Adversarial Networks (GAN) based Anomaly Detection in Industrial Software Systems. In Proceedings of the Moratuwa Engineering Research Conference, Moratuwa, Sri Lanka, 3–5 July 2019. https://doi.org/10.1109/MERCON.2019.8818750.
48. The UNSW-NB15 Dataset | UNSW Research. Available online: https://research.unsw.edu.au/projects/unsw-nb15-dataset (accessed on 28 October 2025).
49. Zhao, Y.; Ma, D.; Liu, W. Efficient Detection of Malicious Traffic Using a Decision Tree-Based Proximal Policy Optimisation Algorithm: A Deep Reinforcement Learning Malicious Traffic Detection Model Incorporating Entropy. *Entropy* **2024**, *26*, 648. https://doi.org/10.3390/E26080648.
50. Putrada, A.G.; Alamsyah, N.; Fauzan, M.N.; Oktaviani, I.D. Pearson Correlation for Efficient Network Anomaly Detection with Quantization on the UNSW-NB15 Dataset. In Proceedings of the International Conference on ICT for Smart Society, Bandung, Indonesia, 4–5 September 2024. https://doi.org/10.1109/ICISS62896.2024.10751550.