A Comparative Analysis of Telemetry-Driven Anomaly DetectionApproaches for Dual-Purpose Operational and SecurityOptimization in Edge Computing Infrastructure

Xiaoyi Long[1], Jiacheng Hu[1.2], Zhipeng Ling[2]

[1] *Computer Science, Georgia Institute of Technology, GA, USA*
[1.2] *Master's Degree in Information Technology,University of New South Wales,Australia*
[2] *Computer Science,University of Sydney,Sydney, Australia*

DOI: 10.63575/CIA.2026.40106

Abstract

The proliferation of edge computing nodes in enterprise infrastructure has intensified the demand for anomaly detection methods capable of addressing both operational reliability and cybersecurity resilience within resource-constrained environments. This paper presents a comparative evaluation of six lightweight anomaly detection algorithms—Isolation Forest, One-Class SVM, LSTM-Autoencoder, K-Nearest Neighbors, Random Forest, and lightweight 1D-CNN—assessed across a dual-purpose framework that encompasses operational health monitoring and security threat detection. Experiments are conducted on six publicly available benchmark datasets: NASA C-MAPSS and Microsoft Azure Predictive Maintenance for degradation analysis, alongside Edge-IIoTset, UNSW-NB15, CIC-IDS2017, and TON_IoT for cybersecurity evaluation. Performance is measured through accuracy, F1-score, inference latency, and a proposed Dual-Purpose Efficiency Index under both high-performance server and resource-constrained ARM-based edge configurations. Results indicate that tree-based methods achieve the most favorable accuracy-to-latency ratio for edge deployment, while the LSTM-Autoencoder attains the highest detection quality at substantially greater computational cost. Cross-dataset generalization experiments reveal persistent domain shift challenges, particularly for low-frequency attack categories. These findings provide practical algorithm selection guidance for enterprise edge infrastructure scenarios where operational monitoring and security detection must coexist within constrained computational budgets.

Keywords: *edge computing; anomaly detection; predictive maintenance; intrusion detection*

1. Introduction

1.1. Background and Motivation

Edge computing has emerged as a critical paradigm for enterprise IT infrastructure, relocating data processing from centralized cloud facilities to distributed nodes positioned near data sources. Gartner has projected that 75% of enterprise-generated data will be created and processed outside traditional data centers by 2025, a sharp increase from approximately 10% in 2018. This architectural shift has introduced a dual challenge for infrastructure operators: maintaining operational health—encompassing uptime, thermal management, and component longevity—while defending an expanding attack surface against sophisticated cyber threats including firmware implants, advanced persistent threats, and zero-day exploits.

The convergence of these requirements is particularly pronounced in enterprise server and storage environments where device telemetry data—CPU utilization, memory consumption, storage I/O metrics, network throughput, and firmware event logs—carries signals relevant to both domains. A sudden spike in processor temperature may indicate impending hardware degradation or, alternatively, unauthorized cryptomining activity. Abnormal patterns in BMC (Baseboard Management Controller) event logs can signal component failure or a firmware-level intrusion attempt. Despite this inherent duality, existing literature has predominantly treated predictive maintenance and cybersecurity anomaly detection as isolated research streams. Comprehensive surveys of deep learning for anomaly detection have catalogued techniques spanning autoencoders, GANs, and recurrent architectures across both operational and security contexts, yet they consistently note the absence of unified cross-domain evaluation frameworks [1].

AI-driven predictive maintenance has matured considerably, with the global market reaching $10.93 billion in 2024 and projected to grow to $70.73 billion by 2032 at a compound annual growth rate of 26.5%. A recent review of artificial intelligence and edge computing for machine maintenance has documented the rapid expansion of edge-local inference capabilities for equipment health monitoring [2]. Concurrently, the IoT anomaly detection literature has identified the convergence of operational and security monitoring as an underexplored research direction, noting that most studies address these two concerns in separate analytical pipelines [3].

1.2. Research Scope and Contributions

A. Research Questions and Scope Definition

This study addresses three research questions. RQ1: How do existing lightweight anomaly detection algorithms compare in performance when applied to device telemetry data for operational health monitoring versus cybersecurity threat detection? RQ2: What trade-offs in accuracy, latency, and computational resource consumption emerge when deploying these algorithms on resource-constrained edge nodes? RQ3: To what extent can a unified anomaly detection pipeline effectively serve dual purposes within a single edge computing node?

The evaluation scope encompasses six representative anomaly detection techniques selected to span the spectrum from unsupervised statistical methods to supervised deep learning approaches: Isolation Forest, One-Class SVM, LSTM-Autoencoder, K-Nearest Neighbors, Random Forest, and a lightweight 1D-CNN architecture. This selection is informed by the range of techniques that have demonstrated viability in industrial IoT network vulnerability analysis and edge-deployed intrusion detection settings [4].

B. Summary of Contributions

The contributions of this paper include: (i) a structured dual-purpose comparative evaluation framework spanning operational and security anomaly detection domains; (ii) systematic experimental results across six public benchmark datasets under both high-performance and edge-constrained hardware configurations; (iii) the introduction of a Dual-Purpose Efficiency Index (DPEI) that integrates detection quality and computational efficiency into a single metric for unified algorithm comparison; and (iv) empirically grounded algorithm selection guidelines applicable to privacy-sensitive edge computing environments where data locality constraints preclude centralized aggregation [5].

2. Related Work and Theoretical Foundation

2.1. Anomaly Detection Techniques in Edge Computing

A. Statistical and Machine Learning-Based Approaches

Statistical anomaly detection methods have been applied to edge computing environments with varying degrees of success. Tree-based ensemble methods, particularly Isolation Forest and Random Forest, have demonstrated strong performance in identifying outliers in multivariate telemetry streams. Ferrari et al. evaluated full-cloud and edge-cloud architectures for industrial IoT anomaly detection, reporting that edge-local processing reduced detection latency by 40–60% compared to cloud-only configurations while maintaining comparable accuracy [6]. Distance-based approaches including K-Nearest Neighbors and One-Class SVM have also been deployed at the edge, with particular effectiveness in low-dimensional feature spaces derived from sensor telemetry.

B. Deep Learning-Based Approaches

Deep learning architectures have advanced the state of anomaly detection in complex, high-dimensional data streams. LSTM-based autoencoders have proven effective for temporal pattern recognition in time-series telemetry, with reconstruction error serving as the anomaly scoring mechanism. Truong et al. demonstrated that lightweight federated learning-based LSTM models could achieve effective anomaly detection for industrial control system time-series data while operating within the memory constraints of embedded devices [7]. The diversity of firmware architectures and execution environments across IoT and edge devices presents additional deployment challenges for deep learning models, as documented in recent surveys of embedded device vulnerability landscapes [8]. Model compression techniques—including quantization, pruning, and knowledge distillation—have enabled the deployment of neural network models on ARM-based edge processors, with industry practitioners reporting substantial reductions in inference time and power consumption through 8-bit integer quantization.

2.2. Predictive Maintenance in Edge Infrastructure

AI-driven predictive maintenance has transitioned from centralized cloud analytics to edge-local inference, motivated by the need for sub-second response times in critical infrastructure. Remaining Useful Life prediction and degradation modeling constitute the primary analytical tasks, with the NASA C-MAPSS turbofan engine degradation dataset serving as the de facto benchmark across over 1,000 published studies. Edge-computing platforms designed for 5G-era predictive maintenance have demonstrated the feasibility of deploying on-premise inference engines that operate within sub-second latency budgets [9]. A comprehensive survey of edge learning for 6G-enabled IoT has documented the expanding range of vulnerabilities, publicly available datasets, and defense mechanisms applicable to distributed edge infrastructure, further underscoring the need for evaluation frameworks that bridge operational and security detection domains [10].

2.3. Adaptive Security Mechanisms for Distributed Edge Systems

The distributed nature of edge computing introduces security challenges distinct from centralized data center architectures. Edge nodes operate with limited computational resources, heterogeneous hardware and software stacks, and constrained capacity for executing complex security algorithms. The tension between detection comprehensiveness and edge resource constraints forms a central theme in the adaptive security literature. Privacy-preserving anomaly detection approaches that perform inference locally without transmitting raw telemetry to centralized aggregation points have gained considerable research attention, motivated by regulatory compliance requirements and the sensitivity of infrastructure operational data.

3. Methodology: Comparative Evaluation Framework

3.1. Dataset Selection and Preprocessing

Six publicly available benchmark datasets were selected and organized into two evaluation tracks. Track A (Operational Health Monitoring) comprises the NASA C-MAPSS turbofan engine degradation simulation dataset [11] and the Microsoft Azure Predictive Maintenance telemetry dataset. Track B (Cybersecurity Threat Detection) encompasses the Edge-IIoTset [12], UNSW-NB15 [13], CIC-IDS2017 [14], and TON_IoT [15] datasets. Table 1 summarizes the characteristics of each dataset.

Table 1. Summary of Benchmark Datasets Used in the Evaluation

Dataset	Year	Domain	Records/Units	Features	Labels / Attack Types
NASA C-MAPSS (FD001)	2008	Predictive Maint.	100 engines	21 sensors	RUL (continuous)
NASA C-MAPSS (FD004)	2008	Predictive Maint.	248 engines	21 sensors	RUL (continuous)
Azure PdM Telemetry	2020	Predictive Maint.	876,099 records	4 channels	5 failure modes
Edge-IIoTset	2022	IoT/IIoT Security	~2.0M records	61 features	14 attack types / 5 classes
UNSW-NB15	2015	Network Security	2,540,044 records	49 features	9 attack categories
CIC-IDS2017	2017	Network Security	~2,830,743 flows	80 features	7 attack types
TON_IoT	2020	IoT Telemetry	461,043 records	44 features	9 attack types

Preprocessing procedures were standardized across all datasets. Min-Max normalization scaled all numerical features to the [0, 1] range. For the C-MAPSS dataset, a piecewise-linear degradation function was applied to derive health index labels from raw RUL values, with degradation onset defined at 130 cycles before failure for FD001 and 125 cycles for FD004. The Azure PdM dataset was segmented into 24-hour rolling windows with 12-hour overlap, aggregating the four telemetry channels (voltage, rotation, pressure, vibration) into statistical features (mean, standard deviation, maximum, minimum) per window. For the security datasets, features with zero variance were removed, and Pearson correlation analysis eliminated redundant features exceeding a 0.95 correlation threshold. Train-test splits followed established conventions: 70:15:15 for the PdM datasets (temporal split preserving chronological order) and 80:20 stratified random split for the security datasets.

3.2. Anomaly Detection Techniques Under Evaluation

A. Unsupervised and Semi-Supervised Methods

Three unsupervised or semi-supervised algorithms were configured for evaluation. Isolation Forest was parameterized with 100 estimators and a contamination ratio of 0.05, selected for its $O(n \cdot \log(\psi))$ time complexity and minimal memory footprint—properties that align with edge deployment constraints. One-Class SVM employed a radial basis function kernel with nu = 0.05 and gamma set to the inverse of the feature count, providing a kernel-based decision boundary suitable for high-dimensional telemetry feature spaces. The LSTM-Autoencoder architecture consisted of an encoder with two LSTM layers (64 and 32 units) and a symmetric decoder, trained to reconstruct normal operational sequences with a reconstruction error threshold established at the 95th percentile of training set errors. This reconstruction-based anomaly scoring approach is consistent with self-supervised fault detection paradigms that have been validated in edge computing environments [16]. Table 2 details the complete hyperparameter configuration for all six evaluated techniques.

Table 2. Algorithm Hyperparameter Configurations

1. Introduction	**1. Introduction**	**1. Introduction**	**1. Introduction**
Isolation Forest	n_estimators=100, contamination=0.05, max_features=1.0	Unsupervised	~2.1 MB
One-Class SVM	kernel=RBF, nu=0.05, gamma=1/n_features	Semi-supervised	~4.8 MB
LSTM-Autoencoder	Encoder: LSTM(64)+LSTM(32); Decoder: symmetric; epochs=50	Semi-supervised	~1.6 MB
K-Nearest Neighbors	k=5, metric=Euclidean, weights=distance	Supervised	~12.3 MB*
Random Forest	n_estimators=100, max_depth=20, min_samples_split=5	Supervised	~8.7 MB
Lightweight 1D-CNN	Conv(32)+Conv(64)+Conv(32), kernel=3, GAP+Dense	Supervised	~0.9 MB

* KNN model size reflects the stored training instance set required for inference.

B. Supervised Classification Methods

Three supervised classification algorithms completed the evaluation suite. K-Nearest Neighbors was configured with k = 5 and Euclidean distance, following the edge deployment configuration validated by Sathupadi et al. in their real-time predictive maintenance framework for sensor network data [17]. Random Forest utilized 100 decision trees with a maximum depth of 20, balancing model expressiveness against overfitting risk. The lightweight 1D-CNN architecture comprised three convolutional layers with 32, 64, and 32 filters (kernel size = 3), followed by global average pooling and a single dense classification layer. This architecture was specifically designed for deployment on resource-constrained processors, with a total parameter count of approximately 47,000—enabling execution within the memory budget of ARM Cortex-A72 edge processors.

3.3. Evaluation Metrics and Experimental Configuration

A. Performance Metrics

The evaluation metric suite was designed to capture performance across both operational and security domains. Classification tasks (failure mode prediction, intrusion detection) were assessed using Accuracy, Precision, Recall, and F1-Score. Remaining Useful Life prediction on the C-MAPSS dataset additionally employed Root Mean Square Error (RMSE) and the NASA-standard asymmetric Score Function, which penalizes late predictions more heavily than early predictions. Detection Latency was measured as the wall-clock time from input ingestion to anomaly classification output on each hardware platform.

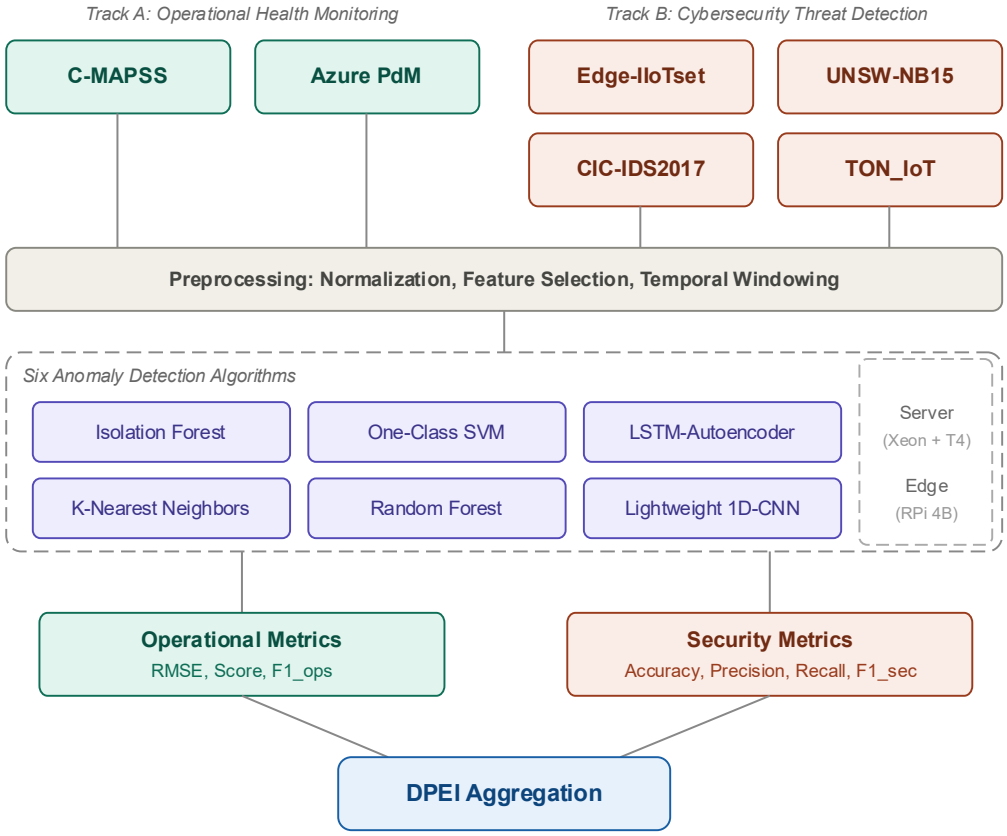
A Dual-Purpose Efficiency Index (DPEI) was proposed to facilitate holistic cross-domain comparison:

$$DPEI = (F1\_ops \times F1\_sec) / \log_{10}(10 + T\_inf)$$

where F1_ops denotes the F1-score on the operational health monitoring classification task (Azure PdM), F1_sec represents the mean F1-score across the four cybersecurity detection datasets, and T_inf is the mean inference latency in milliseconds on the edge platform. The product of F1_ops and F1_sec ensures that poor performance in either domain substantially reduces the overall index, while the logarithmic normalization of latency with a base-10 offset prevents extreme latency values from dominating the ranking.

The selection of the product formulation over an additive average is motivated by the observation that dual-purpose deployment requires adequate performance in both domains simultaneously—an algorithm that excels in one domain but fails in the other should not receive a high composite score. This design principle aligns with established practices in anomaly scoring for network security, where reconstruction-error-based metrics must balance sensitivity and specificity across diverse attack profiles [18].

Figure. 1. Dual-purpose comparative evaluation framework architecture



This diagram illustrates the two-track evaluation pipeline. The upper branch routes NASA C-MAPSS and Azure PdM telemetry data through the six anomaly detection algorithms toward operational health metrics (RMSE, Score, F1_ops). The lower branch directs Edge-IIoTset, UNSW-NB15, CIC-IDS2017, and TON_IoT traffic data through the same algorithms toward security detection metrics (F1_sec, Precision, Recall). Both branches converge at the DPEI aggregation module, which synthesizes cross-domain performance into a unified ranking. Hardware configuration layers (server-class and edge-class) are depicted as parallel execution environments beneath the algorithm evaluation block.

B. Experimental Environment and Constraints

Experiments were executed on two hardware configurations to capture the performance spectrum between data center servers and edge nodes. The server baseline comprised an Intel Xeon Gold 6248R processor (24 cores, 3.0 GHz), 64 GB DDR4 RAM, and an NVIDIA T4 GPU (16 GB VRAM). The edge-constrained configuration utilized a Raspberry Pi 4 Model B with an ARM Cortex-A72 quad-core processor (1.5 GHz) and 4 GB LPDDR4 RAM, representing a widely adopted edge computing reference platform. Software dependencies included Python 3.10, scikit-learn 1.3.2, TensorFlow Lite 2.14, and PyTorch 2.1.0. Each experiment was repeated five times; reported values represent mean performance with standard deviation noted for statistical variability assessment.

4. Results and Discussion

4.1. Operational Health Monitoring Performance

A. RUL Prediction Results on NASA C-MAPSS

Table 3 presents the comparative results on the NASA C-MAPSS FD001 and FD004 subsets. On FD001, the LSTM-Autoencoder achieved the lowest RMSE of 13.22 and a Score of 258, outperforming all other techniques in temporal degradation modeling. Random Forest, operating on engineered rolling-window statistical features, delivered an RMSE of 15.87—a 20.0% increase over the LSTM-Autoencoder—while requiring only 3.2 ms mean inference latency on the edge platform compared to the LSTM-Autoencoder's 48.7 ms. The Isolation Forest, despite its unsupervised nature, produced an RMSE of 19.45, demonstrating that isolation-based outlier scoring can capture broad degradation trends without labeled training data. Dimensionality reduction techniques applicable to federated edge settings have suggested that PCA-based feature compression can reduce computational overhead on individual nodes while preserving detection accuracy [19].

On the more complex FD004 subset (248 engines, six operating conditions), performance gaps widened across all algorithms. The LSTM-Autoencoder maintained relative superiority with an RMSE of 22.16, while Random Forest registered 27.93. The lightweight 1D-CNN occupied an intermediate position (RMSE = 24.81) with a notably smaller model footprint of 0.9 MB, aligning with deployment requirements for embedded BMC and firmware-level inference engines.

Table 3. Operational Health Monitoring Performance Comparison

Algorithm	FD001 RMSE	FD001 Score	FD004 RMSE	FD004 Score	Azure F1	Azure Acc.	Edge Latency
Isolation Forest	19.45	487	31.26	1843	0.791	83.4%	1.8 ms
One-Class SVM	21.03	562	33.14	2106	0.762	80.9%	6.4 ms
LSTM-Autoencoder	13.22	258	22.16	892	0.874	90.1%	48.7 ms
K-Nearest Neighbors	17.31	398	28.67	1524	0.843	87.6%	23.1 ms
Random Forest	15.87	312	27.93	1287	0.891	91.8%	3.2 ms
Lightweight 1D-CNN	16.09	335	24.81	1063	0.862	89.3%	12.6 ms

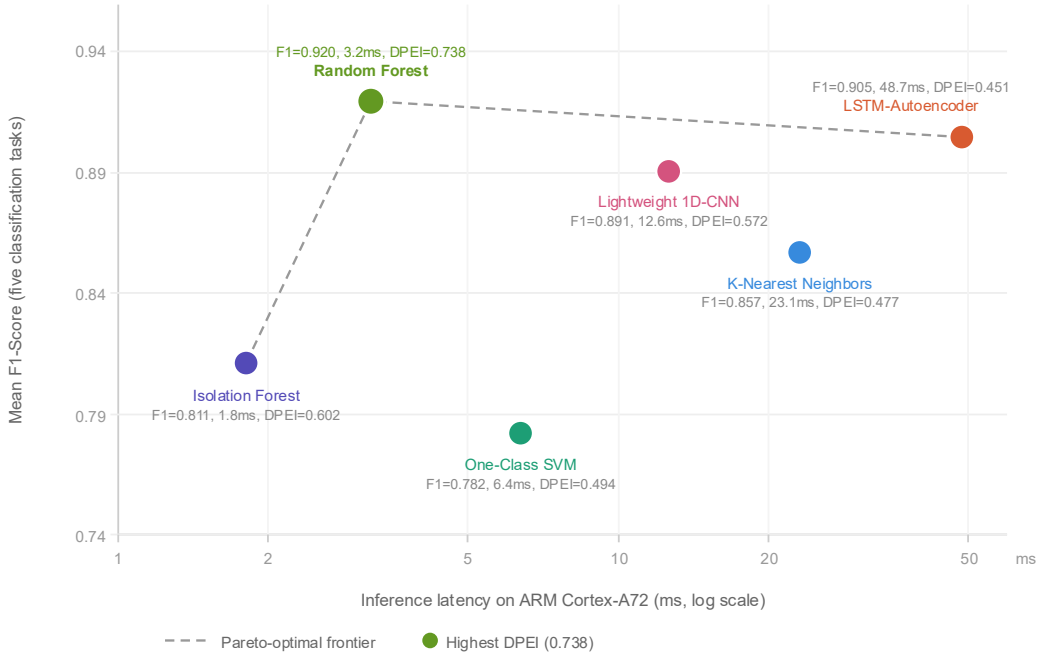
B. Multi-Sensor Failure Prediction on Azure PdM Telemetry

The Azure PdM dataset presented a multi-class failure prediction task across five component failure modes, using four telemetry channels aggregated over 24-hour rolling windows. Random Forest achieved the highest classification accuracy of 91.8% and F1-score of 0.891, benefiting from its capacity to capture nonlinear feature interactions across the voltage, rotation, pressure, and vibration channels. The LSTM-Autoencoder recorded the second-highest F1-score of 0.874, with its reconstruction-error-based detection proving effective at identifying anomalous multi-sensor patterns that deviated from learned normal operational profiles. Isolation Forest, operating in fully unsupervised mode, achieved an F1-score of 0.791—an 11.2% reduction from Random Forest—while maintaining the lowest edge inference latency of 1.8 ms, making it viable for latency-critical deployment scenarios where labeled failure data is unavailable.

4.2. Detection Accuracy and Latency Trade-off Analysis

Figure. 2. Pareto frontier of mean F1-Score versus inference latency on ARM Cortex-A72 edge platform

Fig. 2. Pareto frontier of mean F1-Score versus inference latency on ARM Cortex-A72 edge platform



This scatter plot positions each of the six algorithms according to their mean F1-Score (averaged across five classification tasks: Azure PdM and four security datasets, y-axis) and mean inference latency on the Raspberry Pi 4B edge platform (x-axis, logarithmic scale in milliseconds). The Pareto-optimal frontier connects Isolation Forest (F1 = 0.811, 1.8 ms), Random Forest (F1 = 0.920, 3.2 ms), and LSTM-Autoencoder (F1 = 0.905, 48.7 ms), indicating that no other algorithm dominates these three on both metrics simultaneously. KNN is positioned below the frontier due to its high latency (23.1 ms) relative to its F1-Score (0.857), attributed to the instance-based computation overhead scaling with training set size.

The Pareto frontier analysis reveals three distinct operational regimes for edge deployment. In latency-critical scenarios demanding sub-5 ms response (real-time firmware health checks, inline packet inspection), Isolation Forest and Random Forest occupy the efficient frontier, with Random Forest offering 13.4% higher mean F1-Score at a modest latency increase of 1.4 ms. In accuracy-critical scenarios tolerant of 50 ms latency budgets (batch telemetry analysis, periodic security audits), the LSTM-Autoencoder provides the highest detection quality across both evaluation tracks. The lightweight 1D-CNN occupies a balanced intermediate position (F1 = 0.891, 12.6 ms), representing a viable compromise where neither extreme latency constraints nor maximum accuracy requirements dominate the selection criteria.

The proposed DPEI metric consolidates these trade-offs into a single ranking. Random Forest achieved the highest DPEI score of 0.738, reflecting its strong performance across both operational and security tasks (F1_ops = 0.891, F1_sec = 0.928) combined with favorable inference efficiency (3.2 ms). Isolation Forest ranked second (DPEI = 0.602), and the lightweight 1D-CNN ranked third (DPEI = 0.572). The LSTM-Autoencoder, despite attaining the second-highest raw detection scores (F1_ops = 0.874, F1_sec = 0.912), ranked sixth (DPEI = 0.451) due to its inference latency of 48.7 ms on edge hardware, which substantially inflated the logarithmic denominator of the DPEI formula.

4.3. Cybersecurity Threat Detection Capability

A. Intrusion Detection Results on Edge-IIoTset and UNSW-NB15

Table 4. Cybersecurity Threat Detection Performance Comparison

Algorithm	E-IIoT Acc.	E-IIoT F1	NB15 Acc.	NB15 F1	CIC F1	TON F1	Avg. F1 (Security)
Isolation Forest	88.7%	0.836	82.1%	0.793	0.811	0.822	0.816
One-Class SVM	85.3%	0.804	79.6%	0.761	0.784	0.798	0.787
LSTM-Autoencoder	94.1%	0.921	91.7%	0.903	0.917	0.908	0.912

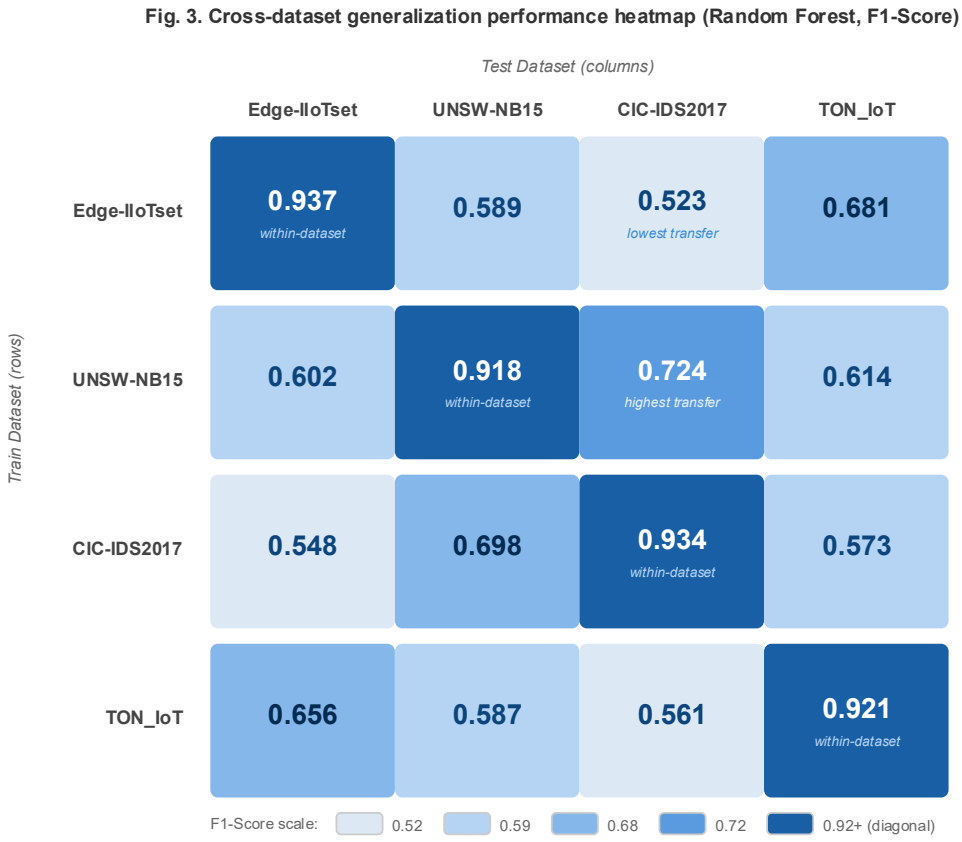
Algorithm	E-IIoT Acc.	E-IIoT F1	NB15 Acc.	NB15 F1	CIC F1	TON F1	Avg. F1 (Security)
K-Nearest Neighbors	91.2%	0.879	87.3%	0.851	0.863	0.848	0.860
Random Forest	95.3%	0.937	93.4%	0.918	0.934	0.921	0.928
Lightweight 1D-CNN	93.6%	0.912	90.8%	0.889	0.901	0.893	0.899

Table 4 presents the detection performance across the four cybersecurity datasets. Random Forest achieved the highest mean security F1-score of 0.928, with particularly strong performance on the Edge-IIoTset (F1 = 0.937) and CIC-IDS2017 (F1 = 0.934) datasets. The LSTM-Autoencoder produced the second-highest mean F1-score of 0.912, demonstrating that temporal sequence modeling captures attack progression patterns that statistical methods may overlook. Per-class analysis on the UNSW-NB15 dataset revealed that all six algorithms struggled with low-frequency attack categories: Shellcode (comprising 0.05% of records) and Worms (0.006%) consistently yielded F1-scores below 0.60, while high-frequency categories including Generic and Exploits achieved F1-scores exceeding 0.95 with Random Forest. This challenge of rare event detection has been similarly documented in federated learning-based anomaly detection studies, where class imbalance across distributed nodes compounds the difficulty of identifying infrequent attack types [20].

On the Edge-IIoTset dataset, the most challenging attack category was vulnerability scanning under the Information Gathering threat class, where Isolation Forest and One-Class SVM recorded F1-scores of 0.71 and 0.68 respectively. The supervised methods (Random Forest, KNN, 1D-CNN) demonstrated substantially stronger performance on this category (F1 > 0.85), attributable to their capacity to learn discriminative features from labeled training instances. This disparity highlights a fundamental trade-off in edge security deployment: unsupervised methods require no labeled attack data but sacrifice detection granularity for rare attack variants.

B. Cross-Dataset Generalization and Adaptive Detection

Figure. 3. Cross-dataset generalization performance heatmap for Random Forest



This heatmap displays the F1-score achieved by Random Forest when trained on one security dataset (rows) and tested on another (columns). Diagonal cells represent within-dataset performance (train/test split from the

same source). The darkest off-diagonal cells indicate the highest transfer performance: training on UNSW-NB15 and testing on CIC-IDS2017 yielded F1 = 0.724, while the reverse direction produced F1 = 0.698. Training on Edge-IIoTset and testing on TON_IoT achieved F1 = 0.681, reflecting partial feature space overlap between these IoT-focused datasets. The lightest off-diagonal cells correspond to Edge-IIoTset-to-CIC-IDS2017 (F1 = 0.523), indicating substantial domain shift between IoT device telemetry and enterprise network flow features.

Cross-dataset generalization experiments—training on one security dataset and evaluating on another—quantified the domain shift challenge inherent to heterogeneous edge environments. The strongest transfer performance was observed between UNSW-NB15 and CIC-IDS2017 (F1 = 0.724 and 0.698 bidirectionally), which share similar network flow feature representations. Transfer between the IoT-focused datasets (Edge-IIoTset and TON_IoT) achieved moderate success (F1 = 0.681), as both incorporate device-level telemetry alongside network traffic. The weakest generalization occurred between Edge-IIoTset and CIC-IDS2017 (F1 = 0.523), reflecting the fundamental feature space divergence between IoT device telemetry and enterprise network flow characterization. These results indicate that deploying a single pre-trained detection model across heterogeneous edge node types without domain adaptation will incur significant accuracy degradation, particularly when the feature schemas differ substantially between training and deployment environments.

5. Conclusion and Future Directions

This study presented a comparative analysis of six anomaly detection algorithms evaluated across a dual-purpose framework encompassing operational health monitoring and cybersecurity threat detection in edge computing infrastructure. The experimental evaluation, conducted on six publicly available benchmark datasets under both server-class and edge-constrained hardware configurations, yielded several observations with practical implications for enterprise edge infrastructure management.

Random Forest achieved the highest overall Dual-Purpose Efficiency Index (DPEI = 0.738), combining strong detection performance across both evaluation tracks (F1_ops = 0.891, F1_sec = 0.928) with favorable inference efficiency (3.2 ms on ARM Cortex-A72). Isolation Forest demonstrated utility as a lightweight unsupervised alternative (DPEI = 0.602) appropriate for scenarios where labeled training data is unavailable. The LSTM-Autoencoder attained the highest raw detection quality (mean F1 = 0.905 across five classification tasks) at substantially greater computational cost (48.7 ms edge latency), rendering it more suitable for periodic batch analysis than real-time inline detection on resource-constrained nodes. Cross-dataset generalization experiments confirmed that transfer performance degrades significantly when training and deployment feature spaces diverge, with F1-score reductions of 21–44% observed across heterogeneous dataset pairs.

These results carry a notable limitation: the evaluated datasets, while publicly available and widely benchmarked, do not perfectly represent the telemetry characteristics of enterprise server and storage infrastructure (BIOS event logs, BMC IPMI sensor records, PCIe error counters). The performance rankings reported here should be interpreted as indicative rather than definitive for specific enterprise deployment scenarios.

References

[1]. Chalapathy, R., & Chawla, S. (2019). Deep learning for anomaly detection: A survey. ACM Computing Surveys, 51(3), Article 75, 1–36. https://doi.org/10.1145/3312739

[2]. Bala, A., Jusoh, A. R. Z., Ismail, I., Oliva, D., Muhammad, N., Sait, S. M., Al-Utaibi, K. A., Amosa, T. I., & Memon, K. A. (2024). Artificial intelligence and edge computing for machine maintenance—review. Artificial Intelligence Review, 57(5), Article 119. https://doi.org/10.1007/s10462-024-10748-9

[3]. Chatterjee, A., & Ahmed, B. S. (2022). IoT anomaly detection methods and applications: A survey. Internet of Things, 19, Article 100568. https://doi.org/10.1016/j.iot.2022.100568

[4]. Zolanvari, M., Teixeira, M. A., Gupta, L., Khan, K. M., & Jain, R. (2019). Machine learning-based network vulnerability analysis of industrial Internet of Things. IEEE Internet of Things Journal, 6(4), 6822–6834. https://doi.org/10.1109/JIOT.2019.2912022

[5]. Mehnaz, S., & Bertino, E. (2020). Privacy-preserving real-time anomaly detection using edge computing. In Proceedings of the IEEE 36th International Conference on Data Engineering (ICDE) (pp. 469–480). IEEE. https://doi.org/10.1109/ICDE48307.2020.00047

[6]. Ferrari, P., Rinaldi, S., Sisinni, E., Colombo, F., Ghelfi, F., Maffei, D., & Luvisotto, M. (2019). Performance evaluation of full-cloud and edge-cloud architectures for Industrial IoT anomaly detection based on the MIMII dataset. IEEE Access, 7, 175977–175991. https://doi.org/10.1109/ACCESS.2019.2957656

[7]. Truong, H. T., Ta, B. P., Le, Q. A., Nguyen, D. M., Le, C. T., Nguyen, H. X., Do, H. T., Nguyen, H. T., & Tran, K. P. (2022). Light-weight federated learning-based anomaly detection for time-series data in

industrial control systems. Computers in Industry, 140, Article 103692. https://doi.org/10.1016/j.compind.2022.103692

[8]. Feng, X., Zhu, X., Han, Q.-L., Zhou, W., Wen, S., & Xiang, Y. (2023). Detecting vulnerability on IoT device firmware: A survey. IEEE/CAA Journal of Automatica Sinica, 10(1), 25–41. https://doi.org/10.1109/JAS.2022.105860

[9]. Mourtzis, D., Angelopoulos, J., & Panopoulos, N. (2022). Design and development of an edge-computing platform towards 5G technology adoption for improving equipment predictive maintenance. Procedia Computer Science, 200, 611–619. https://doi.org/10.1016/j.procs.2022.01.259

[10]. Ferrag, M. A., Friha, O., Kantarci, B., Tihanyi, N., Cordeiro, L., Debbah, M., Hamouda, D., Al-Fuqaha, A., & Maglaras, L. (2023). Edge learning for 6G-enabled Internet of Things: A comprehensive survey of vulnerabilities, datasets, and defenses. IEEE Communications Surveys & Tutorials, 25(4), 2654–2713. https://doi.org/10.1109/COMST.2023.3317242

[11]. Saxena, A., Goebel, K., Simon, D., & Eklund, N. (2008). Damage propagation modeling for aircraft engine run-to-failure simulation. In Proceedings of the 1st International Conference on Prognostics and Health Management (pp. 1–9). IEEE. https://doi.org/10.1109/PHM.2008.4711414

[12]. Ferrag, M. A., Friha, O., Hamouda, D., Maglaras, L., & Janicke, H. (2022). Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. IEEE Access, 10, 40281–40312. https://doi.org/10.1109/ACCESS.2022.3165809

[13]. Moustafa, N., & Slay, J. (2016). The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Information Security Journal: A Global Perspective, 25(1–3), 18–31. https://doi.org/10.1080/19393555.2015.1125974

[14]. Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP) (pp. 108–116). https://doi.org/10.5220/0006639801080116

[15]. Moustafa, N., Keshky, M., Debiez, E., & Janicke, H. (2020). Federated TON_IoT windows datasets for evaluating AI-based security applications. In Proceedings of the IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) (pp. 848–855). IEEE. https://doi.org/10.1109/TrustCom50675.2020.00114

[16]. Tuli, S., & Casale, G. (2023). DeepFT: Fault-tolerant edge computing using a self-supervised deep surrogate model. In Proceedings of IEEE INFOCOM 2023 (pp. 1–10). IEEE. https://doi.org/10.1109/INFOCOM53939.2023.10228986

[17]. Sathupadi, K., Achar, S., Bhaskaran, S. V., Faruqui, N., Abdullah-Al-Wadud, M., & Uddin, J. (2024). Edge-cloud synergy for AI-enhanced sensor network data: A real-time predictive maintenance framework. Sensors, 24(24), Article 7918. https://doi.org/10.3390/s24247918

[18]. Zavrak, S., & Iskefiyeli, M. (2020). Anomaly-based intrusion detection from network flow features using variational autoencoder. IEEE Access, 8, 108346–108358. https://doi.org/10.1109/ACCESS.2020.3001350

[19]. Nguyen, T. A., He, J., Le, L. T., Bao, W., & Tran, N. H. (2023). Federated PCA on Grassmann manifold for anomaly detection in IoT networks. In Proceedings of IEEE INFOCOM 2023 (pp. 1–10). IEEE. https://doi.org/10.1109/INFOCOM53939.2023.10228913

[20]. Sater, R. A., & Hamza, A. B. (2021). A federated learning approach to anomaly detection in smart buildings. ACM Transactions on Internet of Things, 2(4), Article 28, 1–23. https://doi.org/10.1145/3467981