Zero-Day Attack Detection in 6G-Enabled IoT Networks: A Feasibility and Performance Analysis

José R. de Souza Silva*, Luiz Henrique B. A. da Silva*, Caio B. B. de Souza*†, and Andson M. Balieiro*
*Centro de Informática (CIn), Universidade Federal de Pernambuco (UFPE), Recife, Brazil.
† Universidade de Pernambuco (UPE), Surubim, Brazil.
{jrss, lhbas, cbbs, amb4}@cin.ufpe.br, caiobruno.souza@upe.br

***Abstract*—Sixth-generation (6G) networks will integrate Artificial Intelligence (AI) and billions of Internet of Things (IoT) devices, increasing exposure to *Zero-Day* attacks. This work evaluates six one-class anomaly detection models (Autoencoder, LSTM-Autoencoder, VAE, TinyML, Isolation Forest, and Gaussian Mixture Model) under a unified protocol using benign traffic from the CICIoT2023 dataset. Detection performance (overall and per category) and resource consumption are assessed on a real edge device (Khadas VIM4). Five models achieve an AUC-ROC of approximately 99.4% and a *Zero-Day* detection rate above 97%. However, per-category analysis shows that application-layer attacks remain difficult to detect regardless of the model, suggesting a limitation of the dataset feature space. On the edge device, five models achieve millisecond-scale inference, whereas *on-device* training ranges from approximately 1.6 s to 3.2 h, making the standard Autoencoder and TinyML the most practical choices for IoT gateways.**

***Index Terms*—Zero-Day, IoT, 6G, anomaly detection, edge computing, machine learning, TinyML.**

I. Introduction

The sixth generation of mobile networks (6G) is expected to deliver ultra-high data rates, near-zero latency, improved energy efficiency, autonomous operation, and support for advanced services such as digital twins and holographic communications [1]. Achieving these goals requires converging technologies such as cloud computing, network slicing, and programmability. However, this integration also increases architectural complexity, expands the cyberattack surface, and introduces new security challenges [2].

Artificial Intelligence (AI) is also a core enabler of 6G, enabling intelligent management, automation, and security across the network [1]. However, AI-driven 6G networks remain vulnerable to Zero-Day (ZD) attacks, which exploit previously unknown vulnerabilities and evade signature-based defenses [3]. The problem is exacerbated by the resource constraints of Internet of Things (IoT) devices, which often cannot support robust security mechanisms [4]. For instance, as billions of IoT devices become connected, compromised nodes can be recruited into *botnets*, enable lateral movement, and launch large-scale attacks [4].

Several AI-based solutions have been proposed to detect Zero-Day attacks, including graph neural networks with online learning [5], transformers [6], and fog-level autoencoders [7]. However, two key limitations remain. First, many studies evaluate only aggregate accuracy, not isolating the performance against unknown threats, where high success rates against volumetric attacks can obscure poor detection of ZD ones. Second, few analyze resource consumption on embedded hardware, despite its importance for practical deployment and 6G sustainability [3].

This work addresses these gaps by proposing a unified evaluation framework that isolates model effects, thereby enabling a fair comparison of six one-class detection solutions applied to the CICIoT2023 dataset [8], which contains traffic from 105 real IoT devices. Adopting the rigorous Level 1 Zero-Day protocol, all solutions are trained exclusively on benign traffic, making the dataset's 33 attack types, by construction, entirely unseen classes during training. The framework combines general, category-specific, and zero-day-specific detection metrics with resource consumption measurements on the Khadas VIM4 embedded platform, enabling a comprehensive analysis of performance and deployment viability. Experimental results reveal that five solutions achieve similar aggregate performance, with an AUC-ROC of about 99.4% and zero-day detection rates above 97%, yet their execution costs differ by orders of magnitude, with inference latency ranging from 0.03 to 25.6 ms per flow and on-device training from 1.6 s to 3.2 h, identifying the autoencoder and TinyML as the most viable for an IoT gateway.

The remainder of this article is organized as follows. Section II discusses related work. Section III presents the unified proposed methodology. Section IV discusses the results. Finally, Section V concludes this work and outlines future research directions.

II. Related Work

AI-based solutions for 6G security challenges have been widely explored. For instance, [3] systematically evaluates ten anomaly detection architectures, including standard autoencoders (AE), variational AEs (VAE), LSTM-AEs, Isolation Forest, and One-Class Support Vector Machines (OCSVM), trained exclusively on benign traffic to emulate Zero-Day (ZD) scenarios. However, their evaluation relies on simulated server hardware rather than physical IoT devices. Similarly, [9] utilizes AE reconstruction residuals as input features for one-class classifiers to improve unseen attack detection without training data from attacks, but the authors overlook the resource consumption and embedded hardware viability analyses.

In the federated learning (FL) domain, [10] combines online OCSVM with FL to autonomously isolate ZD candidates across

IoT nodes, reducing false positives compared to centralized approaches, although validation is limited to simulations. Similarly, [11] proposed Zero-Day Guardian, a 5G-enabled Industrial IoT (IIoT) FL framework combining dual AEs and OCSVMs per client, but does not report inference latency, CPU utilization, or energy consumption. Regarding adaptive methods, [12] applies Class Incremental Learning (CIL) with episodic memory to IoT intrusion detection, reducing retraining costs by 15-30%, but assumes labeled samples of new attacks and does not evaluate embedded hardware performance. Likewise, [13] proposes ZD-CSNN, a spiking neural network that outperforms deep learning baselines for ZD detection, yet omits computational or energy efficiency analysis.

Despite these advances, the literature still lacks a systematic evaluation that combines multiple ZD detection architectures, execution on real IoT hardware, and a unified analysis of resource consumption. This gap is particularly relevant for 6G, where energy efficiency is a key requirement and IoT devices face strict hardware constraints. Moreover, many studies rely solely on aggregate metrics instead of dedicated Zero-Day evaluation, while others validate IoT solutions using non-IoT datasets, such as the TinyML framework in [14], evaluated on CICIDS-2017. This work addresses these limitations, jointly evaluating detection performance, resource consumption, and Zero-Day Detection Rate (ZDR) on real IoT hardware and data.

III. Methodology

The methodology covers solution selection, dataset and preprocessing, the ZD protocol, solution configuration, evaluation metrics, and the experimental setup. Its rationale is the trade-off between security and computational cost: since IoT devices have scarce memory and energy, anomaly-based detection deployed on the edge gateway, between sensors/actuators and the cloud, is the balance evaluated in this study [1].

A. Literature Survey and Solution Selection

To identify one-class AI solutions for ZD attack detection under hardware constraints, a structured literature review was conducted across the ACM, IEEE, Scopus, and ScienceDirect databases (2021–2026), reducing 531 records to a corpus of 100 high-scoring papers. From this corpus, six representative one-class approaches were selected, all trained exclusively on benign traffic to detect anomalies without prior knowledge of attack signatures [3].

The reconstruction-based methods include the Autoencoder (S1) [3], which uses the Mean Squared Error (MSE) anomaly score $s(x) = \frac{1}{n}\sum_{i=1}^{n}(x_i - r_i)^2$; the LSTM-Autoencoder (S2) [3], which captures temporal dependencies over traffic windows of size W; and the Variational Autoencoder (S3) [15], trained with the loss $L = L_{\text{rec}} + \beta D_{KL}$, where $D_{KL} = \frac{1}{2}\sum_i(\mu_i^2 + \sigma_i^2 - \log\sigma_i^2 - 1)$. Lower-cost statistical methods include the Gaussian Mixture Model (S6) [3], which identifies anomalies through low log-likelihood under a mixture of Gaussians, and the Isolation Forest (S5) [3], which isolates anomalies using short paths in randomly partitioned trees. For resource-constrained devices, the TinyML framework (S4) [14] combines Principal Component Analysis (PCA) with an INT8-quantized Multilayer Perceptron (MLP) to reduce computational cost.

B. Dataset and Preprocessing

ZD evaluation in IoT requires a dataset with enough scale and attack diversity for one-class training and per-category analysis. This study uses CICIoT2023 [8], with about 46.8 million flows from 105 real IoT devices (Wi-Fi, Zigbee, Bluetooth, MQTT), more representative of the target environment than enterprise datasets. The dataset includes 33 attack types grouped into 7 categories (Table I), described by 39 statistical features. Additionally, the dataset is heavily imbalanced with 97.6% attack and 2.4% benign traffic.

Samples with missing or infinite values are removed, and features are standardized (zero mean, unit variance) with scikit-learn's StandardScaler fitted only on the benign training data to prevent leakage. While S1–S3, S5, and S6 use all 39 features, TinyML (S4) applies PCA retaining 95% of the variance (≈20 components).

TABLE I
THE SEVEN ATTACK CATEGORIES OF CICIOT2023 AND THEIR REPRESENTATIVE TYPES.

Category	Representative Attack Types
DDoS	UDP/TCP/ICMP/HTTP/SYN Flood, Fragmentation, RSTFIN, PSHACK, SlowLoris
DoS	HTTP, UDP, TCP, SYN Flood
Mirai	greeth_flood, greip_flood, udpplain
Reconnaissance	HostDiscovery, PortScan, OSScan, PingSweep, VulnerabilityScan
Web	SqlInjection, XSS, CommandInjection, BrowserHijacking, Backdoor
Spoofing	ARP Spoofing (MITM), DNS Spoofing
Brute Force	DictionaryBruteForce

C. Zero-Day Protocol

The literature defines different rigor levels for Zero-Day evaluation. This study adopts the most stringent one (Level 1), in which models are trained exclusively on benign traffic and evaluated against entirely unknown attacks [12], so all 33 attack types serve as unseen classes. The benign traffic is split chronologically to preserve temporal dynamics and reflect potential concept drift, allocating 22% for testing and 78% for training and validation (subdivided into an 80/20 ratio). Hyperparameter optimization and threshold calibration occur solely on the benign validation subset, where the anomaly threshold T is established at the 99th percentile to target a 1% False Positive Rate (FPR). Each solution is trained across five random seeds (42–46) to report the mean and standard deviation, with final evaluations executed on the complete attack corpus to analyze both detection performance and edge hardware resource consumption.

D. Solution Configuration

To isolate performance variations strictly to model architecture, all solutions adopt the same unified evaluation framework. Neural models ($S1$–$S4$) use the Adam optimizer (learning rate 10^{-3}, weight decay 10^{-5}), MSE loss, gradient clipping (maximum norm 1.0), and early stopping (patience of 10 epochs, maximum 50). The LSTM-Autoencoder ($S2$) processes traffic windows of $W = 10$ with a stride of 10, the VAE ($S3$) uses a 16-dimensional latent space with regularization weight β. The TinyML framework ($S4$) operates on PCA features with 8-bit integer quantization. The Isolation Forest ($S5$) uses 100 trees with automatic contamination estimation, while the GMM ($S6$) employs $K = 5$ full-covariance Gaussian components. All hyperparameters were adopted directly from the original studies [3], [14], [15], ensuring reproducibility and attributing performance differences to architectural design rather than parameter tuning. The implementation of the six solutions and the evaluation pipeline is publicly available.[1]

E. Detection Metrics

Continuous anomaly scores are binarized by the threshold T, and performance is evaluated from the outcomes TP, FP, TN, FN via accuracy, precision, recall, F1-score, and the False Positive Rate (FPR, Eq. 1), i.e., the share of benign traffic flagged as malicious, which operators must handle as false alarms.

$$FPR = \frac{FP}{FP + TN} \tag{1}$$

Additionally, the Area Under the ROC Curve (AUC-ROC) evaluates score ranking quality independently of the threshold, making it highly suitable for imbalanced datasets. The Zero-Day Detection Rate (ZDR) is defined as the recall calculated specifically over the attack classes excluded from training, as expressed in Eq. 2. This metric is also reported per category to reveal fine-grained behavioral differences that might otherwise be masked by the aggregated score.

$$\mathrm{ZDR} = \frac{TP_{ZD}}{TP_{ZD} + FN_{ZD}}. \tag{2}$$

F. Resource Metrics and IoT Viability Index

IoT viability depends not only on detection accuracy but also on the computational cost of deploying models on resource-constrained devices, rarely analyzed in the literature. To evaluate deployment feasibility, we measure the following edge-device metrics: inference latency (ms/flow) with its P50, P95, and P99 percentiles, throughput (flows/s), peak RAM usage (MB), CPU utilization (%), model size (KB), temperature (°C), and on-device training time (s), which together determine real-time compliance, device compatibility, thermal headroom, and the practicality of local updates.

Without power instrumentation, energy is estimated as $E = \bar{P}t$ [16], where t is the wall-clock time measured on the VIM4 for each phase (full on-device training and full streaming evaluation) and $\bar{P} \approx 4.9$ W is the board power measured by Palma [17] on the VIM4 under sustained Python ML inference (0.96 A $\times$ 5.1 V). This first-order estimate ignores model-specific dynamic power, idle consumption, and phase differences; yet, even if power varied per model within the range in [17] (≈2.4–4.9 W), the three training-energy tiers (IF/GMM, AE/TinyML, VAE/LSTM-AE) would hold, as their times differ by at least $5.9\times$.

To summarize the trade-off between detection performance and deployment cost, this work introduces the IoT Viability Index (IVI), defined as the ratio between normalized detection and cost terms, with all metrics scaled to the interval $[0, 1]$ across the evaluated solutions (Eq. 3). The weights reflect the study priorities: w_1=0.20 (AUC), w_2=0.30 (ZDR), w_3=0.15 (operability), w_4=0.15 (latency), w_5=0.10 (RAM), and w_6=0.10 (energy). Each cost term is normalized by the worst observed value in the set (maximum = 1), avoiding a zero denominator. The term En_n refers to the *on-device training* energy, since inference energy scales linearly with latency under the fixed-power model and would be redundant with Lat_n; the index thus captures both the recurring cost of inference and the one-time cost of local retraining. Because the assumed power cancels in the normalization ($\bar{P}t/\bar{P}t_{\max} = t/t_{\max}$), this term is invariant to the adopted power value and inherits the robustness of the measured time. As the weights are a design choice, a sensitivity analysis is reported in Section IV-D.

$$\mathrm{IVI} = \frac{w_1\,\mathrm{AUC} + w_2\,\mathrm{ZDR} + w_3\,(1 - \mathrm{FPR})}{w_4\,\mathrm{Lat}_n + w_5\,\mathrm{RAM}_n + w_6\,\mathrm{En}_n}. \tag{3}$$

G. Experimental Environment

Two distinct experimental environments were adopted, one for training and another for edge node performance evaluation. The solutions were trained on Google Colab using PyTorch and scikit-learn. The neural networks (S1–S4) were trained on an NVIDIA Tesla T4 GPU, while the classical models (S5, S6) were trained on the CPU. Feasibility evaluation was performed on the Khadas VIM4[2] acting as an IoT/edge gateway, featuring an Amlogic A311D2 SoC with a quad-core ARM Cortex-A73 @ 2.2 GHz and quad-core Cortex-A53 @ 2.0 GHz CPU, a 3.2 TOPS 8-bit integer NPU, 8 GB LPDDR4X, and Linux 5.15 (aarch64). Inference utilizes the Cortex-A73 cluster (four cores) with the Open Neural Network Exchange (ONNX) runtime [3].

The deployment of all solutions followed a unified pipeline. Model artifacts and the CICIoT2023 dataset were transferred to the VIM4 before deployment. The neural networks (S1–S4) are exported to the ONNX format and executed via ONNX Runtime while the classical models (S5, S6) are serialized and executed in scikit-learn. To respect the board's RAM, the dataset is processed in a streaming fashion, one attack class at a time. Latency is measured with Python's high-resolution clock around each inference call (ONNX Runtime for S1–S4, `score_samples` for S5–S6) on a single flow (batch size 1), emulating per-flow streaming; for S4, preprocessing is timed

[1] https://github.com/Dev-JoseRonaldo/Zero-Day-Attack-Detection-in-6G-Enabled-IoT-Networks-A-Feasibility-and-Performance-Analysis

[2] https://www.khadas.com/vim4

[3] Configured with `intra_op_num_threads=4`

TABLE II
OVERALL DETECTION PERFORMANCE OF THE EVALUATED SOLUTIONS.

Solution	AUC	ZDR	F1	FPR	Model	Training
S1 AE	99.53%	97.79%	98.88%	1.17%	36 KB	550 s
S2 LSTM-AE	99.50%	97.85%	98.91%	1.38%	245 KB	1,114 s
S3 VAE	99.38%	97.69%	98.83%	1.12%	57 KB	254 s
S4 TinyML	99.42%	97.77%	98.87%	1.14%	16 KB	458 s
S5 IF	94.87%	23.26%	36.50%	0.98%	232 KB	0.6 s
S6 GMM	99.40%	97.79%	98.88%	1.11%	96 KB	116 s

likewise and added. Up to 100 warmup calls are discarded to exclude one-time allocation and graph initialization, followed by up to 1000 timed calls, yielding the mean, P50, P95, and P99. Throughput is the inverse of the mean latency; a separate sweep measures batches of up to 1000 flows. A background thread samples CPU, RAM (Resident Set Size, RSS), and temperature (`/sys/class/thermal`) every 50–100 ms.

IV. RESULTS

The evaluation is guided by three research questions: (RQ1) under a unified protocol, do the evaluated one-class solutions achieve comparable Zero-Day attack detection performance? (RQ2) does the aggregate performance reflect the detection power across all attack categories, or are there systematic disparities among attack types? and (RQ3) what is the execution cost of the solutions on real IoT hardware, and how does it relate to detection performance?

A. Overall and Zero-Day Detection (RQ1)

Five of the six solutions exhibit comparable performance, with AUC-ROC ranging from 99.38% to 99.53% and ZDR from 97.69% to 97.85%, as shown in Table II. Although the Isolation Forest (IF) provides an AUC of 94.87%, its ZDR drops to 23.26%, making it an outlier. In practical terms, a gateway running any solution from the homogeneous group would detect about 98 out of every 100 unseen attacks at the cost of 1% false alarms, whereas one based on the IF would fail to detect more than three-quarters of Zero-Day attacks.

The remaining results (Table II) reinforce this contrast: F1 stays between 98.83% and 98.91% for the five high-performing solutions but drops to 36.50% for the IF, reflecting its lower recall, while training time spans more than three orders of magnitude (0.6 s for the IF to 1,114 s for the LSTM-AE), with the GMM offering the lowest cost among the strong detectors (116 s) at a ZDR of 97.79%. The effective FPR remains close to the 1% target (0.98%–1.38%), indicating consistent calibration, and model sizes (16–245 KB) confirm compatibility with resource-constrained edge devices.

The results also reproduce the benchmark in [3], with AUC-ROC and ZDR differing by less than 0.2 and 0.1 percentage points for the five high-performing detectors. The Isolation Forest reproduces the same low and unstable recall, varying ± 11 percentage points across seeds against ± 8 in [3]. Notably, the GMM matches the autoencoders while requiring no GPU acceleration and substantially less training time. In contrast, the axis-aligned partitioning employed by the IF fails to capture the non-linear structure of IoT traffic, demonstrating that computational efficiency alone is insufficient for solution feasibility.



Fig. 1. Heatmap of the mean ZDR (1% FPR) per category and solution.

TABLE III
ZDR (1% FPR) PER ATTACK TYPE FOR THE SIX SOLUTIONS.

Attack type	S1	S2	S3	S4	S5	S6
DDoS/DoS/Mirai (19 types)	100%	100%	100%	100%	57.6%*	100%
Recon-HostDiscovery	54.1%	58.6%	52.1%	57.8%	45.2%	53.7%
Recon-PortScan	41.3%	51.6%	36.9%	39.6%	21.2%	37.3%
Recon-OSScan	29.8%	40.2%	25.8%	28.1%	13.7%	26.2%
Recon-PingSweep	14.8%	23.9%	11.5%	16.3%	13.6%	13.3%
VulnerabilityScan	7.3%	9.4%	5.6%	9.5%	9.7%	6.2%
DictionaryBruteForce	30.0%	60.0%	23.4%	28.1%	8.0%	23.2%
DNS_Spoofing	9.3%	6.4%	2.2%	7.6%	15.8%	9.7%
MITM-ArpSpoofing	7.0%	5.6%	1.5%	4.8%	12.2%	9.0%
Backdoor_Malware	5.7%	15.6%	3.7%	14.2%	9.9%	5.0%
XSS	4.9%	14.1%	3.8%	12.9%	9.1%	5.1%
CommandInjection	3.9%	11.5%	2.5%	9.9%	7.2%	4.0%
BrowserHijacking	11.7%	5.6%	1.7%	5.6%	5.6%	3.5%
SqlInjection	2.7%	2.7%	1.3%	6.0%	10.6%	1.7%

*Mean over the 19 volumetric types (per-type range 0.1%–99.9%).

B. Per-Category Analysis (RQ2)

Since DDoS, DoS, and Mirai constitute about 90% of the attack traffic and are simple to detect, all solutions except the IF reach 100% ZDR in these categories and dominate the aggregate metric. The solutions differ in the less frequent categories, such as web (under 1% of the dataset) and reconnaissance. Fig. 1 consolidates the per-category ZDR in which the volumetric block saturates at 100% for all neural solutions and the GMM, while reconnaissance attacks are partially detectable (21–37%), and web attacks remain nearly invisible (3–10% on average, 1–16% per web type).

Table III details the discriminant attack types for all six solutions. Dictionary brute force highlights the gain of the LSTM-AE's temporal structure (60.0%, against 30.0% for the AE and 23.2% for the GMM), while spoofing remains low for every solution (below 16%). Among the web types, the LSTM-AE and the TinyML achieve the best values (15.6% and 14.2% on Backdoor), the VAE is consistently the weakest (1.3–3.8%), and the IF, curiously, is the best on SqlInjection (10.6%), albeit at the cost of its collapse on the volumetric block, where it is the only solution below 100%.

To verify the sensitivity of the metrics to test-set composition, an equal-weighting evaluation with a per-category sample cap was also performed: the autoencoder's AUC drops from 99.5% to 95.5% and its ZDR from 97.8% to 79.6%, confirming

that the ∼99% aggregate mostly reflects volumetric traffic and reinforcing the need to report the per-category analysis alongside the real composition. In response to RQ2, there is a systematic disparity: web attacks reach an AUC of only ≈ 0.80 even for the best solutions. The central point is that three independent mechanisms, reconstruction (S1–S4), axis-aligned partitioning (S5), and density (S6), fail at the same place. When such distinct families converge to the same ceiling, a plausible hypothesis is that the limitation lies less in the architectures than in the 39-feature flow space of CICIoT2023: these features are header and flow statistics without payload information, and web attacks such as XSS or SQL injection produce HTTP flows statistically similar to benign ones. This remains a hypothesis, since richer feature sets or other datasets were not evaluated, but the shared ceiling, revealed only by the per-category analysis, is one of the main findings of this work.

C. Execution Cost on the Edge Device (RQ3)

All six solutions were deployed on the Khadas VIM4. Because numerical precision (FP32) is preserved, detection performance is hardware-independent, so Table IV focuses on computational cost. Latency per sample spans nearly three orders of magnitude: the autoencoders (S1–S3) operate in the sub-millisecond range, sustaining 9.4–29 thousand flows/s and exceeding one million flows/s in batches of 1000; TinyML (1.36 ms) and the GMM (1.70 ms) remain within a few milliseconds; and the IF is a clear outlier at 25.6 ms, due to the non-vectorized traversal of its 100 trees. The trend holds at P99, where the autoencoders stay below 1 ms against 37.7 ms for the IF. Peak RAM varies only slightly (927–1,205 MB), being dominated by the Python runtime and the CSV streaming pipeline rather than the models, whose serialized size (16–245 KB, Table II) is below 0.01% of the board's 8 GB; isolating the per-model memory footprint is left for future work. CPU utilization differs substantially, from 87% (LSTM-AE) to 14% (IF, whose sequential traversal limits parallelism). The temperature never exceeded 60.3 °C, about 15 °C above idle and well below the SoC throttling threshold of 95 °C.

Decomposing the TinyML (S4) latency reveals that preprocessing (StandardScaler, Pearson filter, and PCA projection) consumes 1.280 ms per sample against only 0.075 ms for the ONNX model, i.e., $\approx 94\%$ of the total latency and about 17 times slower than the network it feeds. Optimizing the network alone thus brings little end-to-end gain. Since standardization, feature selection, and PCA are linear, they can be folded offline into one affine layer inside the ONNX graph, removing the per-sample Python/scikit-learn overhead; batched vectorization, fixed-point projection, and NPU offloading are complementary options. In addition to inference, Table V presents the training cost of each solution from scratch on the VIM4, which measures cost and convergence only. The LSTM-AE took ∼3.2 h (10.3× the Colab time) and the VAE ∼2.0 h (28.1×), while the AE and the TinyML trained in ∼18–20 min and the classical models in seconds to minutes, with modest penalties (1.5–2.6×) since they never depended on a GPU. This favors the train-in-the-cloud, infer-at-the-edge paradigm, with on-device training practical only for the classical models or for infrequent retraining. In terms of estimated energy (Table V), training the LSTM-AE requires about 15.6 Wh, roughly 11 times the AE (1.5 Wh) and 70 times the GMM (0.23 Wh). The energy consumed during evaluation, dominated by the streaming processing of the dataset, varies slightly across solutions (0.11–0.38 Wh).

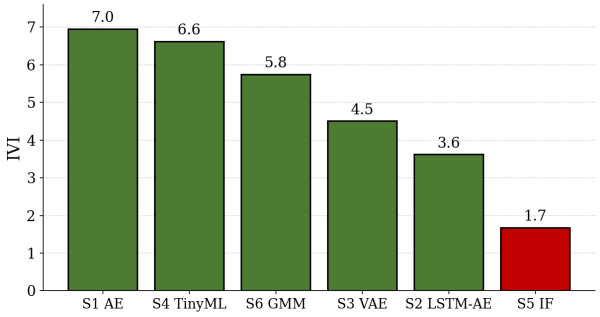


Fig. 2. IoT Viability Index (IVI) per solution.

D. Performance Cost Trade-off and Discussion

Combining detection, inference cost, and training energy, Fig. 2 presents the IVI (higher is better). The standard Autoencoder leads (7.0), followed by TinyML (6.6) and the GMM (5.8), while the VAE (4.5) and the LSTM-AE (3.6) are penalized by their on-device training cost despite their strong detection and low inference latency. The IF ranks last (1.7) due to its low ZDR and high latency. The index therefore rewards the models that are simultaneously viable for inference and for local retraining. Regarding weight sensitivity, equal, detection-oriented, and cost-oriented weights preserve this order, and under 10,000 random ±50% perturbations of each weight the AE and TinyML hold the top two positions in 99.8% of cases and the IF stays last in 99.9%. The order changes only if the training-energy term is removed ($w_6 = 0$), lifting the VAE and LSTM-AE, or made dominant, favoring the GMM; the IVI is thus conditional on local retraining being required.

Taken together, aggregate metrics alone are insufficient, since the ∼99% AUC is largely driven by volumetric attacks. On the VIM4, the standard Autoencoder and TinyML offer the best trade-off, the GMM is a competitive non-neural alternative, and the LSTM-AE provides only a modest gain for reconnaissance attacks at the cost of ∼3.2 h of training.

These results, however, assume that the benign traffic used for training is free of attacks, which may not hold for a gateway collecting real traffic. Benign behavior also drifts over time, requiring periodic recalibration of T. Even a 1% FPR yields a high absolute number of alerts at gateway scale. Finally, although the Level 1 protocol treats the 33 CICIoT2023 attack types as unseen, they are known attack families captured in a single testbed rather than truly novel exploits, so whether the results, and the feature-space hypothesis for application-layer attacks, generalize to other datasets and to real Zero-Day scenarios remains to be verified.

TABLE IV
INFERENCE COST ON THE VIM4. PEAK RAM INCLUDES THE EVALUATION PIPELINE (CSV STREAMING).

Solution	Latency/sample (ms)	P95/P99 (ms)	Throughput (flows/s)	Avg. CPU (%)	Peak RAM (MB)	Temp. (°C)
S1 AE	0.073	0.10 / 0.28	13,671	23	992	57.7
S2 LSTM-AE	0.034*	0.39 / 0.98	29,255	87	927	57.3
S3 VAE	0.106	0.29 / 0.47	9,432	61	954	57.1
S4 TinyML	1.355**	0.11 / 0.25†	738	53	943	60.3
S5 IF	25.59	33.0 / 37.7	39	14	1,123	52.7
S6 GMM	1.703	2.55 / 3.32	587	45	1,205	58.0

* per flow (the native unit of S2 is the 10-flow window: 0.342 ms/window). ** total = preprocessing 1.280 ms + ONNX model 0.075 ms. † percentiles refer to the ONNX model only.

TABLE V
ON-DEVICE TRAINING COST ON THE VIM4 VERSUS THE CLOUD (COLAB) AND ESTIMATED ENERGY ON THE VIM4.

	Training time (s)			Peak	Energy (Wh)	
Solution	VIM4	Colab	Ratio	RAM (MB)	Train	Eval.
S1 AE	1,079	550	2.0×	728	1.47	0.11
S2 LSTM-AE	11,450	1,114	10.3×	1,417	15.58	0.38
S3 VAE	7,120	254	28.1×	908	9.69	0.12
S4 TinyML	1,202	458	2.6×	1,195	1.64	0.12
S5 IF	1.6	0.6	2.6×	969	0.002	0.13
S6 GMM	171	116	1.5×	1,568	0.23	0.14

Seeds: 1 (S1–S4) and 5 (S5, S6). The ratio shows the penalty of the ARM CPU without GPU acceleration. Energy: $E = \bar{P}t$ with $\bar{P} \approx 4.9$ W.

V. CONCLUSION

This work evaluated six one-class Zero-Day detectors for 6G IoT, trained only on benign traffic under a unified protocol and deployed on a real edge device. Five models, four reconstruction-based and the GMM, reached an AUC of about 99.4% and a ZDR above 97% (RQ1), whereas the IF reached only 23% ZDR, showing that computational efficiency alone does not guarantee viability. Per-category analysis (RQ2) showed that web and reconnaissance attacks are poorly detected by all model families, suggesting a limitation of the CICIoT2023 feature space rather than of the detection models. Regarding deployment viability (RQ3), five models achieved inference latencies below a few milliseconds (except the IF, at 25.6 ms) with stable thermal behavior, while on-device training remained feasible but highly architecture-dependent, ranging from 1.6 s to 3.2 h. Future work includes direct per-workload power measurement and per-model memory profiling, an optimized end-to-end preprocessing pipeline, NPU-accelerated INT8 inference and training, evaluation on additional IoT datasets without retraining, richer feature representations (e.g., payload-aware) for application-layer attacks, and assessment of Zero-Day detection solutions considering different rigor levels.

REFERENCES

[1] V. Ziegler, H. Viswanathan, H. Flinck, M. Hoffmann, V. Räisänen, and K. Hätönen, "6G architecture to connect the worlds," *IEEE Access*, vol. 8, pp. 173 508–173 520, 2020.

[2] Z. Pan, Z. Ying, X. Jiang, J. Cai, and W. Zhou, "FedGAD: Real-time anomaly detection and adaptive defense with federated GANs in 6G," *IEEE Network*, vol. 39, no. 5, 2025.

[3] R. Lee and Y. Zhang, "Performance–efficiency trade-offs in anomaly detection architectures for zero-day IoT attack detection: A systematic benchmark on CICIoT2023," in *2025 Annual Computer Security Applications Conference Workshops (ACSAC Workshops)*, Honolulu, HI, USA, 2025, pp. 467–472.

[4] D. M. Dhanvijay, M. M. Dhanvijay, and V. H. Kamble, "Cyber intrusion detection using ensemble of deep learning with prediction scoring based optimized feature sets for IoT networks," *Cyber Security and Applications*, vol. 3, p. 100088, 2025.

[5] T. Dhanush, Y. Shadharan, N. Abishek, and R. Saranya, "Zero-day attack detection in IoT networks using graph neural networks and online learning," in *Proc. 10th International Conference on Communication and Electronics Systems (ICCES)*, 2025.

[6] Z. Rizq and P. Sukarno, "Zero-day attack detection using transformer encoder," in *Proc. 3rd International Conference on Software Engineering and Information Technology (ICoSEIT)*, 2025.

[7] M. R. Minhas, Q. M. Shafi, S. A. Khan, T. Ahmad, S. Ullah, A. Buriro, and M. A. Yaqub, "F-OSFA: A fog level generalizable solution for zero-day DDoS attacks detection," *IEEE Access*, vol. 13, 2025.

[8] E. C. P. Neto, S. Dadkhah, R. Ferreira, A. Zohourian, R. Lu, and A. A. Ghorbani, "CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment," *Sensors*, vol. 23, no. 13, 2023.

[9] B. Lewandowski and R. Paffenroth, "One-class classification using autoencoder feature residuals for improved IoT network intrusion detection," in *2023 32nd International Conference on Computer Communications and Networks (ICCCN)*, Honolulu, HI, USA, 2023, pp. 1–7.

[10] T. Ohtani, R. Yamamoto, and S. Ohzahata, "Detecting zero-day attack with federated learning using autonomously extracted anomalies in IoT," in *2024 IEEE 21st Consumer Communications & Networking Conference (CCNC)*, Las Vegas, NV, USA, 2024, pp. 356–359.

[11] P. Verma, N. Bharot, J. G. Breslin, D. O'Shea, A. Vidyarthi, and D. Gupta, "Zero-day guardian: A dual model enabled federated learning framework for handling zero-day attacks in 5G enabled IIoT," *IEEE Trans. on Consumer Electronics*, vol. 70, no. 1, pp. 3856–3866, 2024.

[12] F. Cerasuolo, G. Bovenzi, C. Marescalco, F. Cirillo, D. Ciuonzo, and A. Pescapè, "Adaptive intrusion detection systems: Class incremental learning for IoT emerging threats," in *2023 IEEE International Conference on Big Data (BigData)*, Sorrento, Italy, 2023, pp. 3547–3555.

[13] N. A. Shiltagh Al-Jamali, A. R. Zarzoor, and H. S. Al-Raweshidy, "An effective technique of zero-day attack detection in the Internet of Things network based on the conventional spike neural network learning method," *IET Networks*, p. e70019, 2025.

[14] R. Rajput, S. Gulati, P. Shah, and S. Gautam, "TinyML in network security: A lightweight and real-time intrusion detection framework," in *IEEE Conference on Artificial Intelligence and Machine Vision*, 2025.

[15] S. Gond, B. Chhetry, R. K. Dutta, R. Matam, and F. A. Barbhuiya, "VAE-BiLSTM-IDS: A two-phase deep-learning framework for enhanced IoT security," in *TENCON 2025 – IEEE Region 10 Conference*, 2025.

[16] L. Bouza, A. Bugeau, and L. Lannelongue, "How to estimate carbon footprint when training deep learning models? a guide and review," *Environmental Research Communications*, vol. 5, no. 11, 2023.

[17] G. F. Palma, "Análise de desempenho de plataformas de hardware embarcado para inferência de redes neurais convolucionais em aplicações de visão computacional," Dissertação de Mestrado, Universidade Tecnológica Federal do Paraná (UTFPR), Cornélio Procópio, 2025.