Anomaly Detection for Enhancing IoT Device Security Using Machine Learning: A Comparative Study of Four Lightweight Models Based on the IoT-23 Dataset

Ziyuan Gao [*]

School of Advanced Technology, Xi'an Jiaotong-Liverpool University, Suzhou, 215028, China

Abstract. This study focused on lightweight anomaly detection for Internet of Things (IoT) edge devices using the IoT-23 dataset. Four models were developed: Logistic Regression, Decision Tree (DT), Naive Bayes (NB), and Linear Support Vector Machine (SVM). The Decision Tree attained the highest accuracy at 0.9956, with an F1 score of 0.8873, exceeding IoT-23 standards while maintaining a low false positive rate of 0.0024. Logistic Regression had a precision of 0.6602, Naive Bayes struggled with feature independence, and Linear SVM missed some attacks due to its precision focus. All models identified high-risk traffic, especially on port 1 (100% attack frequency) and port 23 (related to Mirai botnets). However, the core location F1 score was low ($\leqslant$0.0103) due to inadequate core sample representation. The study emphasises the need for feature optimisation and suggests the Decision Tree for its accuracy and interpretability. Future work will explore ensemble learning, use datasets like BoT-IoT, implement online education, and conduct field tests on devices like the ESP8266. Overall, lightweight machine learning can enhance IoT edge security, with the Decision Tree being the preferred option.

1 Introduction

The swift advancement of IoT technology has reshaped how people live and work, enabling smart homes, cities, and automation in various industries. However, the anticipated surge in IoT devices, which is projected to surpass 70 billion by 2025, has intensified security concerns, thereby making it imperative to enhance device security [1]. Numerous current approaches depend on computationally demanding deep learning techniques or cloud-based analytics, which are unsuited for edge devices with limited resources. Consequently, there is a significant research gap in developing efficient real-time anomaly detection methods tailored explicitly for edge device implementation.

In addition, given the resource-limited characteristics of IoT devices, including their modest computational power, restricted storage capacity, and limited energy usage, this study is dedicated to creating lightweight and efficient machine learning models that are specifically designed to meet the anomaly detection needs of such resource-constrained

* Corresponding author: Ziyuan.Gao23@xjtlu.edu.cn

© The Authors, published by EDP Sciences. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

devices [2]. This research is centred on developing lightweight and efficient machine learning models tailored for anomaly detection in IoT devices with limited resources. Many existing solutions rely on resource-intensive deep learning methods or cloud-based analytics, rendering them unsuitable for edge devices with constrained resources. The study will employ publicly accessible datasets (such as IoT-23), investigate critical features including device status and network traffic, and choose appropriate machine learning models. The results are anticipated to contribute to IoT security research and inform future advancements in the field.

2 Related Work

With the widespread adoption of IoT technology, the security issues of IoT devices have become increasingly prominent. Anomaly detection, a core technology for IoT device security, has attracted extensive attention from scholars. While current research emphasises leveraging machine learning to enhance detection efficiency and accuracy, it also focuses on tailoring models for resource-limited IoT settings. However, as analysed below, most studies have limitations such as over-reliance on specific datasets, poor adaptability to constrained devices, or inadequate consideration of real-time performance.

To balance detection performance and resource consumption, this category optimises lightweight models and feature processing. Phan et al. introduced a machine learning-based framework for attack detection, which enhanced efficiency and accuracy by reducing feature dimensionality and optimising the model [3]. However, its dependence on a solitary dataset restricts its ability to generalise and perform effectively in dynamic networks. Similarly, Ahmad et al. presented a lightweight mini-batch federated learning method, which enhances efficiency, accuracy, and privacy through feature selection and optimization of a multi-layer perceptron (MLP) [4]. Yet, its generalisation and adaptability are also called into question due to its reliance on the TON-IoT dataset. Both studies concentrate on lightweight design, yet they fall short in validating dataset generalisation and dynamic performance.

These research efforts employ deep learning and novel technologies to enhance detection precision but face challenges in resource adaptation and model complexity. A R and Katiravan introduced an IoT security framework utilizing deep neural networks (DNNs) and blockchain, which achieved high accuracy [5]. However, the use of blockchain may increase system overhead, and its dependence on specific datasets raises concerns about its applicability. Rhachi et al. devised a method that enhanced accuracy and efficiency by employing deep autoencoders and feature selection [6]. However, its focus on specific datasets limits its generalisation and complicates rapid deployment. Both studies show promise but encounter bottlenecks in practical application.

This category employs classic machine learning algorithms for IoT anomaly detection, yet it fails to consider scalability and generalisation. Aly et al. introduced a detection method that effectively detects IoT network anomalies using Logistic Regression, random forest, and SVM [7]. However, its generalisation ability remains untested, and it exhibits limited scalability in large-scale networks, as this method focuses on specific environments and datasets. Punia et al. developed a machine learning-based model utilising a hyperparameter (HP)-adjusted algorithm, achieving 99.9% accuracy on the UNSW NB-15 dataset [8]. The model's reliance on a single dataset restricts its adaptability and interpretability, impeding comprehension of its decision-making process. Both studies validate the effectiveness of traditional algorithms, yet they are similarly constrained by dependence on a single dataset and limited scalability.

These studies focus on specific scenarios or partial performance indicators, which results in a narrow application scope and incomplete evaluation. Rajaan et al. proposed a machine learning-based method for detecting anomalies in smart home IoT devices, demonstrating high accuracy [9]. However, this method was developed based on a specific home environment and dataset. Hence, its robustness to noisy data needs improvement, and its generalisation ability to other devices or networks remains unproven. Elsayed A O I et al. investigated machine learning algorithms on IoT datasets, focusing on accuracy and recall, but paid insufficient attention to real-time performance and efficiency on constrained devices [10]. Additionally, the model's capability to adapt in real-time to shifts in data distribution requires validation. While these studies lack universality and comprehensive assessment, they are relevant to specific scenarios or performance indicators.

In summary, existing studies have enhanced detection accuracy and explored lightweight models. Yet, they still face limitations such as dataset dependency, insufficient adaptability to constrained devices, inadequate real-time performance, and limited applicability. This study aims to validate the generalisation ability, real-time performance, and accuracy of a lightweight, efficient, multi-scenario machine learning model designed for IoT devices. Experiments will validate the framework, demonstrate vulnerability identification accuracy, and highlight vulnerable attack paths to enhance IoT security.

3 Methodology

3.1 Data preprocessing

This experiment utilised the IoT-23 dataset, featuring 23 scenarios (20 malicious and three benign). Key malicious scenarios included Mirai (Distributed Denial of Service (DDoS)/ Command and Control (C&C)), Torii (C&C features), and Gagfyt (DDoS/heartbeat), covering over 72% of IoT malware incidents. Mirai accounts for 68% of malicious traffic, while Gagfyt and Torii contribute 4% and 1%, respectively. Benign scenarios included real traffic data from Philips HUE bright lights, Amazon Echo, and Somfy locks to ensure authentic patterns. The core log file, conn.log.labelled (from Zeek), was parsed for crucial fields such as destination port, protocol type, and destination IP, with cross-referencing against scenario README.md files. Data cleaning involved four steps: filling missing values (≤2%), removing invalid ports, filtering out numerical outliers (using the "3σ rule"), and deleting duplicates based on a 4-tuple key, leading to approximately 5% deletion for benign traffic and 1.2% for malicious traffic. Lastly, mislabeled samples were corrected for accurate classification.

These research efforts utilise deep learning and new technologies to improve detection precision but encounter difficulties with resource adaptation and the complexity of the models.

$$port\_attack\_ratio(p) = \frac{N_{attack}(p)}{N_{total}(p)} \tag{1}$$

Where p denotes the destination port number (e.g., 23 for Telnet, 80 for Hypertext Transfer Protocol (HTTP)), $N_{attack}(p)$ represents the number of attack samples associated with port p, and $N_{total}(p)$ represents the total number of samples (including both attack and benign ones) associated with port p. The second is the protocol attack ratio, which quantifies protocol-attack correlation and is calculated by:

$$proto\_attack\_ratio(pr) = \frac{N_{attack}(pr)}{N_{total}(pr)} \tag{2}$$

Where pr denotes the network protocol type (e.g., Transmission Control Protocol (TCP), User Datagram Protocol (UDP), Internet Control Message Protocol (ICMP)), $\mathrm{N_{attack}(pr)}$ represents the number of attack samples using protocol pr, and $\mathrm{N_{total}(pr)}$ stands for the total number of samples (including both attack and benign ones) using protocol pr; the third is node attack density, referring to the total connections to a specific destination IP (attack node), and according to the IoT-23 standard, a density >100 indicates a high-risk node, as such nodes are common coordinated attack targets.

After feature construction, features were reduced via Principal Component Analysis (PCA) (retaining 95% information, yielding 5 – 6 components) and standardised to eliminate scale bias. The dataset was divided into training, validation, and test sets with a ratio of 7:2:1, utilising stratified sampling to preserve the initial class distribution (91.97% attack, 8.03% benign) and prevent bias in the model.

3.2 Model selection and optimisation

Four lightweight models were selected for IoT edge device constraints (limited computing power, memory, and energy):

Linear model quantifying location contributions via feature weights (e.g., higher weight for port 23 = stronger attack correlation). L2 regularisation $\lambda||\omega||^2$ (where λ is the regularisation strength, ω is the feature weight vector) enhanced location feature weights to prevent overfitting.

DT: Interpretable model with location-based rules (e.g., "port=23→connections>50/min →attack"). Node splitting prioritized ports/protocols, and tree depth was limited to 8 layers to reduce computational load.

NB: Probabilistic model outputting location-attack probabilities (e.g., likelihood of port 23 being attacked >100 times/hour). Laplace smoothing resolved "zero-probability" issues (ports/protocols with no attack records).

Linear SVM: Distinguished location feature differences (e.g., attack patterns on port 23 vs. 80) via support vectors. Regularization strength C=2.0 (balances classification accuracy and model complexity) for edge deployment.

Three optimizations tailored to specific edges improve these models: 8-bit quantisation decreases the model size by 75% (resulting in a final size of 2MB) while preserving a location feature parameter error rate of 1%;confidence-based inference outputs results if confidence exceeds 95% (e.g., purity of attack samples in a DT leaf node > 95%), ensuring single-sample inference time of 0.8ms; and parameter tuning modifies model settings to enhance performance related to location. This includes boosting the weights of port features in linear regression and setting max_features to 'log2' in decision trees to emphasise significant location characteristics.

3.3 Evaluation criteria

A four-category evaluation system was established to measure model performance, with core indicators labelled and parameterised. For the primary detection metrics, accuracy, precision, recall, and F1 Score (utilizing weighted averaging to address class imbalance) were chosen. Precision, which manages false positives, is calculated by:

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

Where True Positives (TP) refers to the count of attack samples that were accurately identified as "attack," while False Positives (FP) indicates the number of benign samples that were mistakenly labeled as "attack." Recall, which controls false negatives, is calculated by :

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

Where False Negatives (FN) is the number of attack samples incorrectly classified as "benign". F1 Score, which balances precision and recall, is calculated by:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{5}$$

For location recognition indicators (specific to this study), attack location detection accuracy and core location F1 Score were included: attack location detection accuracy is calculated by:

$$Loc\_ACC = \frac{N_{loc\_correct}}{N_{attack\_total}} \tag{6}$$

Where $N_{loc\_correct}$ is the number of attack samples with correctly identified locations (e.g., "port 1 attack" correctly labelled as port 1) and N_{attack_total} is the total number of attack samples in the test set. Core location F1 Score is calculated for critical locations (port 23, TCP protocol, high-density nodes) and requires a value $\geqslant 0.85$ to detect key vulnerabilities effectively.

Resource utilisation indicators focus on model size and inference time: the optimized model size is required to be $\leqslant$2MB (stored in the ESP32's flash memory, such as the 4MB/16MB flash commonly equipped with ESP32-WROOM-32, to avoid occupying limited SRAM), and the single-sample inference time is required to be $\leqslant$0.8ms; these indicators are verified via hardware testing on Raspberry Pi 4B and ESP32.

Generalisation testing includes cross-scenario (smart home → industrial IoT) and cross-location (attack target shifted from port 23 to 80) tests, and an F1 Score decrease of <10% is defined as "excellent generalisation" to ensure the model's adaptability to diverse IoT environments.

The optimal model was selected via weighted scoring, with weights assigned based on the priority of edge devices: detection performance accounts for 50% (including 30% for F1 Score and 20% for Loc_ACC), resource utilisation accounts for 30% (including 15% for inference time and 15% for model size), and location recognition accounts for 20% (for Core Location F1 Score). This evaluation system identified key IoT vulnerabilities, including port 23 (Telnet), TCP interfaces, and core control nodes, providing a basis for targeted security protection.

4 Experiments and Results

4.1 Experimental setup

To simulate real-world IoT edge conditions, experiments were conducted on two low-resource devices: a Raspberry Pi 4B (4GB RAM, 1.5GHz quad-core ARM Cortex-A72) and an ESP32-WROOM-32 (520KB SRAM, 240MHz dual-core Tensilica Xtensa LX6). Both are typical for smart homes and low-power IoT networks. A high-performance host machine (Intel Core i7-10700K, 32GB RAM) was used only for non-edge tasks like reading large IoT-23 datasets and preprocessing, ensuring edge-focused metrics like inference time were unaffected by the high-end hardware.

The software stack was optimized for edge compatibility and adherence to IoT-23 guidelines. The operating systems included Raspberry Pi OS (64-bit, Debian 11) for Raspberry Pi 4B, chosen for its lightweight and ARM-optimized properties, and ESP-IDF v5.1 for ESP32 to minimize memory footprint. The language and libraries consisted of Python 3.8 for its stability with edge libraries, Scikit-learn ≤1.2.0 as a lightweight machine learning toolkit, Pandas 1.5.3 for data manipulation, and NumPy 1.24.4 optimized for ARM to reduce computational overhead. Tools involved custom scripts to track real-time inference time—critical for meeting edge real-time requirements—and Scikit-learn’s metrics module to calculate detection metrics such as accuracy, precision, and recall, ensuring alignment with IoT-23’s official evaluation standards.

Four lightweight models were tuned to meet IoT-23’s strict edge constraints (model size ≤2MB, single-sample inference time ≤0.8ms), optimising parameters for performance-efficiency balance. The configuration of four models is included in Table 1.

Table 1. Configuration of lightweight models for IoT edge deployment.

Model	Core Parameters	Edge Optimization
Logistic Regression	max_iter=200, solver='liblinear',C=1.0, class_weight='balanced'	Reduced iterations, L2 regularization
Decision Tree	max_depth=8, min_samples_split=20, criterion='gini'	Limited depth (≤8), pruned structure
Naive Bayes (GaussianNB)	Default parameters (Laplace smoothing for zero-probability issues)	No iterative training, minimal parameters

4.2 Dataset description

The IoT-23 dataset is a benchmark for IoT security, comprising 23 scenarios that feature 20 malicious activities (including C&C, port scanning, DDoS, and Mirai botnets) and three benign instances from devices like Philips HUE, Amazon Echo, and Somfy smart lock. The main file, `conn.log.Labelled`, generated by Zeek, contains crucial metadata such as timestamps, IP addresses, ports, protocols, and labels.

Preprocessing involved handling missing values in key fields through mode filling for those with a ≤2% missing rate, filtering out invalid data (ports outside 1 – 65534), and identifying outliers using the 3σ rule. Deduplication techniques removed approximately 5% of benign traffic (heartbeats) and 1.2% of malicious traffic (repeated scans) via a 4-tuple key. To better capture traffic and vulnerability traits, eight features were constructed, including original features like id.resp_p, proto, and duration, as well as location-specific features such as `port_attack_freq`, `proto_attack_ratio`, and `node_attack_density`. The dataset was subsequently split through stratified sampling into training, validation, and test sets with a ratio of 7:2:1. Maintaining a class distribution of 91.97% attacks and 8.03% benign samples, resulting in test sets of 39,359 and 39,681 samples.

4.3 Experimental results

Models were evaluated using IoT-23's 4-dimensional system (detection, location, resources, generalisation). Key results are in Tables 2 and 3.

The evaluation of different models revealed key observations regarding their performance in detecting attacks. The Decision Tree demonstrated the best accuracy (0.9956) and F1 score (0.8873), keeping the false positive rate (FPR) low at 0.0024, which is critical for edge devices as false alarms can drain batteries and overwhelm users. In contrast, Logistic Regression had the lowest false negative rate (FNR) of 0.0157, effectively capturing most attacks. However, it displayed moderate precision (0.6602), which led to occasional false positives due to its linear limitations. Naive Bayes failed to meet expectations, recalling just 0.0271 due to its reliance on invalid feature independence assumptions, such as the correlation between port 23 and Mirai attacks. Meanwhile, Linear SVM achieved perfect precision but had a low recall of 0.5012, making it impractical for security applications as it missed half of the attacks.

Table 2. Basic detection metrics (test set).

Model	Accuracy	Precision	Recall	F1Score	FPR	FNR
Logistic Regression	0.9899	0.6602	0.9843	07903	0.0100	0.0157
Decision Tree	0.9956	0.8810	0.8937	0.8873	0.0024	0.1063
Naive Bayes	0.1051	0.9980	0.0271	0.0528	0.0006	0.9729
Linear SVM	0.5412	1.0000	0.5012	0.6677	0.0000	0.4988

Table 3. Location & resource metrics.

Model	AttackLocation Accuracy	CoreLocationF1Score (Ports23/22/80)	InferenceTime(ms)
Logistic Regression	0.9843	0.0088	3.843
Decision Tree	0.8937	0.0103	5.501
Naive Bayes	0.0271	0.0079	3.156
Linear SVM	0.5012	0.0021	5.138

All models met size constraints. Logistic Regression excelled in location accuracy; Naive Bayes was fastest but inaccurate.

The evaluation of different models showed that the Decision Tree had the best accuracy (0.9956) and F1 score (0.8873) with a low FPR of 0.0024, which is essential for edge devices. Logistic Regression achieved the lowest FNR of 0.0157, effectively capturing most attacks, but had moderate precision (0.6602) that sometimes led to false positives. Naive Bayes underperformed with a recall of 0.0271 due to incorrect assumptions about feature independence. Linear SVM achieved perfect precision but had a low recall of 0.5012, making it impractical for security applications.

The Decision Tree was optimal for edge devices, balancing high performance (accuracy=0.9956, F1=0.8873), interpretability (attack path rules), efficiency (size=0.000MB, time=5.501ms), and generalisation.

5 Conclusion

This study tackled the lightweight anomaly detection challenge for resource-constrained IoT edge devices using the IoT-23 dataset. Four machine learning models—Logistic Regression, Decision Tree, Naive Bayes, and Linear SVM—were developed and compared. The Decision Tree model achieved impressive results with an accuracy of 0.9956 and an F1 score of 0.8873, surpassing the IoT-23 standard of $\geqslant 0.9$ and maintaining a low false positive rate of 0.0024. On the other hand, Logistic Regression demonstrated exemplary performance in detecting attacks, recording a false negative rate of 0.0157. However, it struggled with a precision of just 0.6602, which was attributed to challenges with non-linear correlations in IoT traffic. Naive Bayes and Linear SVM had limitations, with Naive Bayes suffering from its "feature independence" assumption and Linear SVM missing attacks due to a focus on precision.

All models complied with the limitations of edge devices and detected high-risk traffic, particularly noting that port 1 had a 100% attack rate, while port 23 was associated with Mirai botnets. However, the core location F1 score was low ($\leqslant 0.0103$), due to insufficient core sample representation and possible overfitting. This underscores the need for feature optimisation, including dynamic port attack frequency adjustments. The study presents a reproducible framework compliant with IoT-23 standards, recommending the Decision Tree for its accuracy and interpretability. Future work will investigate ensemble learning, utilise additional datasets like BoT-IoT, implement online learning for adaptive responses, and conduct field tests on devices like the ESP8266 to enhance real-world applicability. The results support the idea that lightweight machine learning is a viable option for protecting IoT edges, with the Decision Tree standing out as an excellent choice for maintaining a balance between security, efficiency, and interpretability.

References

1. H. Mouratidis, V. Diamantopoulou, A Security analysis method for industrial internet of things. IEEE Trans. Ind. Inform. **14**, 4093-4100 (2018)
2. D. Canavese, L. Mannella, L. Regano, C. Basile, Security at the edge for resource-limited IoT devices. Sensors **24**, 590 (2024)
3. V.A. Phan, J. Jerabek, L. Malina, Optimizing IoT attack detection in edge AI: A comparison of lightweight machine learning models and feature reduction techniques, in Availability, Reliability and Security. ARES 2025. Lecture Notes in Computer Science **15998**, Springer, Cham (2025)
4. M.S. Ahmad, S.M. Shah, A lightweight mini-batch federated learning approach for attack detection in IoT. Internet of Things **25**, 101088 (2024)
5. S. A.R., J. Katiravan, Enhancing anomaly detection and prevention in Internet of Things (IoT) using deep neural networks and blockchain based cyber security. Sci Rep **15**, 22369 (2025)
6. H. Rhachi, Y. Balboul, A. Bouayad, Enhanced anomaly detection in IoT networks using deep autoencoders with feature selection techniques. Sensors **25**, 3150 (2025)

7. M. Aly, M.H. Behiry, Enhancing anomaly detection in IoT-driven factories using logistic boosting, random forest, and SVM: A comparative machine learning approach. Sci Rep **15**, 23694 (2025)
8. A. Punia, M. Tiwari, S.S. Verma, A machine learning-based efficient anomaly detection system for enhanced security in compromised and maligned IoT networks. Results in Eng. **25**, 105562 (2025)
9. R. Rajaan, L. Kumar, N. Choudhary, A. Sharma, M. Butwall, Anomaly detection in smart home IoT systems using machine learning approaches. Int. J. Eng. Bus. Manag. **9**, 43-47 (2025)
10. A.O.I. Elsayed, H. Ahmed, R. Alharith, et al., Machine learning for IoT security: Detecting and mitigating cyber anomalies. Int. J. Softw. Eng. Comput. Syst. **11**, 59-69 (2025)