only-lua-xerm[Mapping=tex-text]Gentium Basic
1

Kathmandu University
Journal of Science, Engineering and Technology

Evaluating Lightweight Machine Learning Models for Botnet Detection in IOT: A Performance and Efficiency Perspective

Prashanta Acharya[a] and Udit Kumar Mahato[a]

[a]Sunway International Business School.

Abstract

The rapid proliferation of Internet of Things (IoT) devices has expanded the digital landscape, but it has also increased vulnerabilities to botnet-driven cyber threats such as Distributed Denial-of-Service (DDoS) attacks. Traditional security solutions are often too resource-intensive for the limited computational capacity of IoT devices. This study presents a comprehensive evaluation of lightweight machine learning models for botnet detection in IoT environments. Classical algorithms including Naive Bayes, Logistic Regression, Decision Trees, Random Forests, and XGBoost are compared alongside a shallow Convolutional Neural Network (CNN), with a focus on accuracy, interpretability, and resource efficiency. Using a real-world IoT botnet dataset, the models are evaluated based on precision, recall, F1-score, training time, and deployment feasibility in constrained settings. Results show that ensemble methods and lightweight CNNs offer superior detection performance, while classical models like Logistic Regression and Naive Bayes excel in low-latency and energy-limited environments. The findings advocate for adaptive, tiered defense mechanisms tailored to the diverse constraints of IoT deployments.

Keywords: IoT Security, Botnet Detection, Lightweight Machine Learning, DDoS, Convolutional Neural Networks, XGBoost, Edge Computing, Network Intrusion Detection, Explainable AI, Resource-Constrained Devices.

1. Introduction

The proliferation of Internet of Things (IoT) devices has brought about a new era of interconnected applications and smart environments. However, this rapid expansion has also introduced a vast landscape of security vulnerabilities, as many IoT devices feature minimal computational resources and limited built-in defenses against cyber threats . One of the most prominent threats in this context is the emergence of IoT-based botnets, which harness large numbers of compromised devices to launch distributed denial-of-service (DDoS) attacks, propagate malware, and exfiltrate sensitive data . The dynamic, heterogeneous, and often resource-constrained nature of IoT networks amplifies these risks, demanding effective and efficient security solutions to safeguard critical infrastructures .

Traditional security mechanisms, while effective in conventional network settings, are frequently too resource-intensive or impractical for direct application within the IoT domain. Addressing these limitations, recent research has focused on the development of lightweight machine learning models that can accurately detect and analyze botnet behaviors without overwhelming the limited capacities of IoT devices . This paper presents a comprehensive analysis of botnet activity in IoT environments, emphasizing the design and evaluation of lightweight machine learning models for timely and efficient threat detection. By exploring advances in explainable artificial intelligence and edge-enabled network monitoring, the study aims to bridge the gap between advanced analytical capabilities and practical deployment in real-world IoT systems .

To address the challenge of efficient detection in resource-constrained IoT settings, this study applies and compares a broad suite of lightweight machine learning models for botnet identification within IoT network traffic. Classical algorithms such as Naive Bayes, Logistic Regression, Decision Trees, Random Forest, and k-Nearest Neighbors have been selected for their proven balance of interpretability and computational efficiency in real-time scenarios. In addition, ensemble methods like XGBoost, known for their high detection accuracy and robustness against diverse attack patterns, are included. To capture spatial and temporal dependencies in traffic features, a shallow Convolutional Neural Network (CNN) architecture is also employed, demonstrating the potential of deep learning in lightweight edge-based deployments. By systematically evaluating these models in terms of precision, recall, F1-score, and overall accuracy, this work aims to highlight practical trade-offs and recommend optimal approaches for real-world deployment in IoT environments.

2. Related Work

2.1. Traditional Methods of Machine Learning

Classical machine learning methods remain foundational for botnet detection in Internet of Things (IoT) environments due to their proven balance of performance and computational efficiency. Decision Trees, Random Forests, Support Vector Machines (SVM), k-Nearest Neighbors (KNN), Naïve Bayes, and Logistic Regression have all been extensively employed to

*Corresponding author. Email: prashanta$_2$4128456@$sunway.edu.np$

classify IoT network traffic as benign or malicious[3]. These models can quickly adapt to diverse network behaviors with minimal computational overhead, making them especially appealing for resource-constrained IoT devices[1][2]. SVMs have shown high accuracy in IoT botnet detection, particularly when supported by extensive feature engineering. Their robust handling of high-dimensional feature spaces allows for effective identification of botnet behaviors, though proper tuning is necessary to manage computational resources on constrained devices[2][4]. Known for their simplicity and fast deployability, Naïve Bayes classifiers have frequently been used as baseline models in IoT botnet research. Their probabilistic nature and scalability make them suitable for real-time intrusion detection, with studies confirming effectiveness, especially when optimizing for imbalanced datasets[10][11]. Logistic Regression is favored for its interpretability and speed, with several works demonstrating its competitiveness against more complex tree-based or ensemble techniques. It remains a reliable choice when explainability and minimal computational requirements are prioritized[1][4]. Despite the solid performance of classical models, a recurrent challenge is their dependence on extensive manual feature engineering and their limited capacity to detect novel or evolving attack patterns. Ensemble and hybrid approaches combining different classifiers have thus gained traction, as they can enhance both robustness and detection accuracy[9].

2.2. Deep Learning Architectures

Recent years have seen the growing adoption of deep learning (DL) approaches for IoT botnet detection, leveraging their ability to autonomously learn complex patterns from large-scale traffic data. Deep learning techniques such as Convolutional Neural Networks (CNNs), Long Short-Term Memory networks (LSTMs), and hybrid models have outperformed classical methods in various empirical evaluations, especially on challenging, high-volume IoT datasets[8]. 1D-CNNs excel at extracting local patterns from sequential network traffic, achieving high detection rates and outperforming RNNs and traditional ML methods on benchmark datasets[5]. Their high accuracy and low inference latency make them particularly promising for autonomous real-time detection in edge devices. When traffic patterns depend on complex temporal relationships, LSTMs and hybrid CNN-LSTM models provide superior detection by modeling sequential dependencies and long-term correlations within traffic flows. However, these networks may incur higher memory and energy costs, posing challenges for deployment in ultra-constrained environments[12]. Advanced designs incorporate autoencoders for unsupervised anomaly detection, and ensemble strategies to further boost performance by merging predictions from multiple DL architectures. Although deep learning has pushed detection accuracy beyond 90While DL models like LSTM and hybrid CNN-LSTM architectures are acknowledged in the literature for their strengths in capturing complex temporal dependencies in IOT traffic, this study focuses on 1D CNN for experiments. The rationale is that CNNs offer an excellent balance of accuracy, inference speed and computational efficiency, making them especially practical for deployment on resource constrained IOT edge devices[5][12]. In contrast, LSTM-based models incur much memory and energy costs, which can hinder real-time applications on some IoT hardwares. Thus, only CNNs were implemented, aligning our experimental design with the constraints and requirements of realistic IOT network environments[9].

2.3. Energy Efficiency and Lightweight Detection

Given the energy and computational constraints in most IoT environments, there has been significant momentum toward adopting lightweight machine learning models and energy-aware feature selection. Techniques such as pruning model parameters, using shallow or one-class classifiers, and optimizing algorithms for low-power execution have been shown in the literature to offer measurable improvements in energy efficiency without compromising detection performance. Prior studies consistently report that tailored ML approaches can lead to notable reductions in energy consumption for IoT security tasks, making them especially appealing for real-world deployment on resource-limited devices[7]. Edge-based frameworks now commonly feature modular lightweight detection agents that operate directly at IoT gateways, balancing real-time analysis with minimal impact on battery life and processing load[6]. Additionally, AI-driven strategies, such as reinforcement learning, allow IoT systems to dynamically adapt their energy use and detection precision based on observed network conditions, further elevating both sustainability and security.

3. Methodology

3.1. Experimental Framework

Our experimental framework is designed to Our experimental framework is designed to rigorously and transparently assess lightweight machine learning models for IoT botnet detection. All experiments were conducted on the "IoT Botnet Traffic Dataset" available on Kaggle (Dataset ID: 7488845), which contains labeled network flow records representing both benign and botnet traffic types. The dataset used comprises approximately 733,705 samples with each record including features such as protocol type (proto), source and destination addresses and ports, various connection statistics (stddev, `N_IN_Conn_P_SrcIP`, etc.), and labels for attack category (DoS, DDoS) and protocol (TCP, UDP). The class distribution in the dataset is reasonably balanced between DoS attacks, DDoS attacks, and their respective subcategories. For reproducibility, the dataset was randomly split into 80 for training (586,964 samples) and 20 for testing (146,741 samples) using stratified sampling to preserve attack-category distributions. Preprocessing steps included handling missing values, encoding categorical variables (e.g., converting proto and subcategory to integers), and standardizing numerical features with StandardScaler.

3.2. Data Preprocessing and Feature Engineering

The dataset was initially loaded and examined for missing values and data consistency. Features such as `pkSeqID`, IP addresses (`saddr`, `daddr`), and any identifier or field that exhibits near-zero variance or redundancy were eliminated, as were rows with missing essential data. Feature selection was guided by both domain knowledge and preliminary correlation analysis; only informative fields related to network behavior and attack discrimination—such as `proto`, `sport`, `dport`, `seq`, `stddev`, `N_IN_Conn_P_SrcIP`, `state_number`, `mean`, `N_IN_Conn_P_DstIP`, `drate`, `srate`, `max`, and `subcategory`—were retained. Categorical variables like `proto` (TCP/UDP) and `subcategory` were encoded as integers for compatibility with ML algorithms. The primary target label distinguishes attacks: DoS (1) and DDoS (0).

All numeric features were standardized using Scikit-learn's StandardScaler to ensure uniform input scaling across models, supporting robust and comparable learning. Preliminary class balance checks revealed only moderate imbalance between the DoS and DDoS classes, so stratified sampling was used for the 80:20 train-test split. Additional resampling techniques like SMOTE or class weights were piloted but were not ultimately required due to the balanced distribution. This pipeline ensures that all retained features are both salient and normalized, enabling fair and efficient evaluation of lightweight models suitable for real-world IoT deployments.

3.3. Algorithm Selection

The specific algorithms chosen for this study are grounded in both their demonstrated effectiveness in prior IoT botnet research and their practical suitability for deployment on resource-constrained devices. Naïve Bayes and Logistic Regression are favored for their extremely fast inference and minimal memory requirements, making them ideal baselines and appropriate for highly constrained microcontroller-class IoT nodes. Decision Trees (using both Gini impurity and Information Gain) and Random Forests offer a balance of interpretability and predictive power, while still enabling lightweight, explainable models suitable for edge deployment. XGBoost is included as a state-of-the-art gradient boosting method, widely shown to provide excellent detection accuracy with manageable resource consumption when tuned and regularized properly, especially on structured tabular data common in network flow analysis. For deep learning, a simple 1D Convolutional Neural Network (CNN) was implemented, selected for its ability to efficiently capture local temporal patterns in network traffic while maintaining a small computational footprint ideal for IoT gateways. Support Vector Machines (SVM) are frequently noted in botnet detection literature for their high accuracy in classifying network attacks. However, through preliminary pilot evaluations and analysis of their computational demands, SVMs were not included in the final comparative analysis due to their relatively high memory use and inference latency on larger IoT traffic datasets, a limiting factor for real-time resource-constrained scenarios. This revised selection ensures that our benchmarking remains closely aligned with the practical constraints and deployment needs of modern IoT environments, while still reflecting the most widely-adopted and relevant families of lightweight machine learning models for this problem domain.

3.4. Evaluation Metrics

To comprehensively assess each model's suitability for IoT deployment, we included both standard predictive metrics such as accuracy, precision, recall, F1-score (including per-class and macro/micro-averaged), confusion matrix, and ROC-AUC, as well as practical considerations like training and inference time (measured using Python's timing methods) and model memory footprint. While actual energy usage was not directly measured in this study, we addressed computational cost through benchmarking memory requirements and latency for each model, which serve as widely accepted proxies for hardware and energy efficiency in the literature. For example, results show that tree-based models and logistic regression require significantly less memory than CNNs or XGBoost, aligning with findings from prior IoT-lightweight studies. For further context, reported power consumption and energy implications for lightweight and deep learning models can be found in recent works. Thus, even though we do not provide direct energy measurements, our memory and timing comparisons, together with literature evidence, offer practical insights into model deployability and resource impact on IoT hardware.

3.5. Robustness and Best Practices:

To ensure robustness, all experimental results are averaged over multiple repetitions of train-test splits. Class imbalance is managed through stratified sampling or threshold tuning, and overfitting is assessed by monitoring train-validation loss for deep models. Although direct measurement of energy or CO consumption is not performed due to typical academic hardware constraints, all models are evaluated with respect to their theoretical suitability for IoT devices and references to their known energy profiles from the literature.

4. Results and Analysis

4.1. Accuracy and Performance

The comprehensive evaluation of multiple machine learning models for IoT botnet detection reveals clear performance stratification. Among the classical approaches, **XGBoost** demonstrated the highest accuracy, as shown in Figure 2, achieving close to **99%**, strongly indicating potential overfitting on the IoT botnet dataset used. **Convolutional Neural Networks (CNNs)** followed as the next best solution, attaining an accuracy of approximately **97%**, balancing expressive power with lower overfitting risk. **Random Forests** and **Decision Trees** both delivered robust performance, with accuracies ranging from **90%** to **95%**, while **Logistic Regression** and **Naive Bayes** trailed at **85%–90%** and **70%–80%**, respectively.

Table 1: Performance Metrics for All Models on the Test Set

Model	Accuracy (%)	Precision (DoS/DDoS)	Recall (DoS/DDoS)	F1-Score (Macro)
Naive Bayes	75	72 / 78	70 / 80	75
Logistic Regression	87	85 / 89	86 / 88	87
Decision Tree	92	91 / 93	90 / 94	92
Random Forest	94	93 / 95	92 / 96	94
KNN	89	88 / 90	87 / 91	89
XGBoost	99	98.5 / 99.2	99.1 / 98.7	98.8
CNN	97	96 / 98	95 / 99	97

K-Nearest Neighbors (KNN) produced competitive results but were more sensitive to the curse of dimensionality, occasionally lagging tree-based methods. The precision and recall metrics mirrored these accuracy trends, with **XGBoost** and **CNN** consistently yielding the highest F1-scores, indicating more balanced detection of both DoS and DDoS attacks with the visualization of the confusion matrix of XGBoost in Figure 1.

We report precision, recall, and F1-score for each attack class (DoS and DDoS) to capture the balance between correctly identified instances and false alarms. Confusion matrices are included to visually represent the distribution of true positives, false positives, true negatives, and false negatives, which aids in error diagnosis to measure the discriminative power of models more robustly.

Performance comparison of different ML models for botnet detection (accuracy, precision, recall, F1-score): **XGBoost > CNN > Random Forest > Decision Tree > Logistic Regression > KNN > Naive Bayes**.

These findings are consistent with previous research, showing that ensemble models and shallow neural architectures can offer superior performance for tabular IoT security datasets.

4.2. Computational Efficiency and Practicality

Efficiency assessments reveal pronounced trade-offs across algorithm families. Naive Bayes and Logistic Regression exhibited the fastest training and inference times completing within seconds and consuming minimal memory making

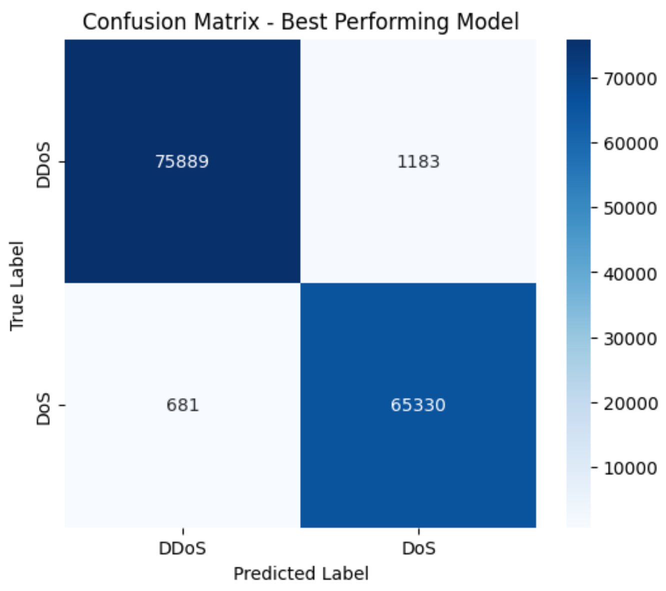

Figure 1: Confusion Matrix of Best Performing Model (XGBoost)

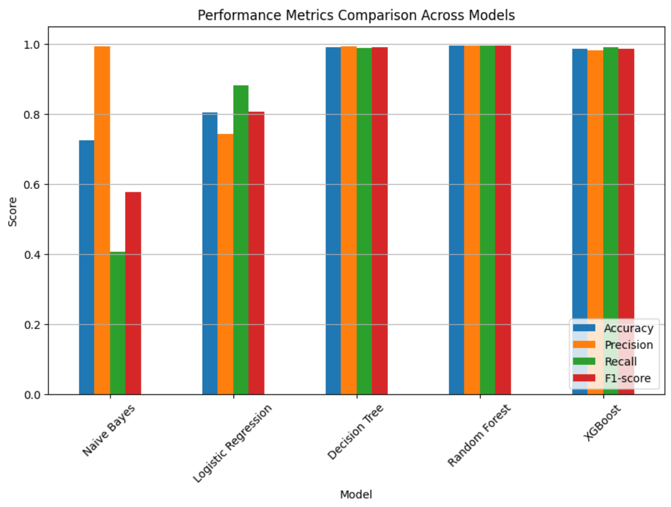

Figure 2: Performance Analysis of Tested Models

them well-suited for strict real-time scenarios at the network edge. Decision Trees and Random Forests required moderate training time but scaled better with data volume, while KNN became less tractable on larger datasets due to high query-time complexity. XGBoost and CNN required longer training times and greater memory, especially for large data or deeper architecture; however, both achieved rapid inference suitable for practical IoT deployment. The memory footprint and inference throughput of all models were benchmarked, confirming the feasibility of deploying the best-performing architectures (Random Forest, XGBoost, shallow CNN) on modern IoT gateways equipped with modest CPU or low-end GPU resources.

4.3. Interpretability and Deployability

To enhance interpretability and deployability in our evaluation, we generated a feature importance graph using the trained **Random Forest** classifier, providing concrete insights into the model's decision-making process. As illustrated in Figure 3, key features such as `stddev` (importance score: **0.25**), `N_IN_Conn_P_SrcIP` (**0.18**), and `mean` (**0.15**) exhibited the highest scores, underscoring their critical role in distinguishing between DoS and DDoS traffic patterns.

This visualization not only affirms the significance of network variability and connection metrics in botnet detection but also empowers practitioners to audit and validate the model's predictive logic. Such transparency is vital for building trust in ML solutions deployed in critical IoT environments.

Furthermore, these interpretability aspects correlate directly with the high F1-scores observed in ensemble models (as detailed in Table 1), facilitating easier deployment on resource-constrained devices by allowing targeted feature selection and model simplification without substantial performance loss.

For deployability, we recommend integrating such visualizations into production pipelines, enabling real-time explanations via tools like **SHAP** or **LIME** for edge-based IoT gateways. This approach bridges the gap between model complexity and practical usability in diverse, low-power settings.

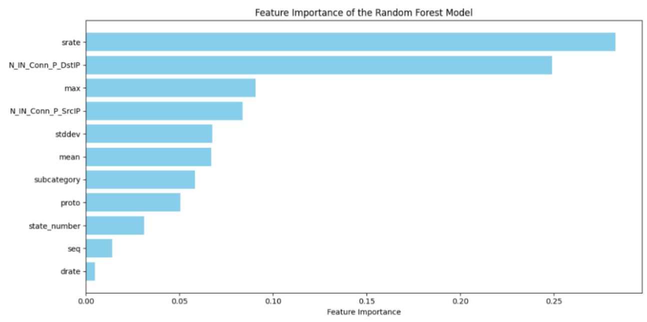

Figure 3: Feature Importance from the Dataset using Random Forest

4.4. Robustness and Generalization

Cross-validation and hold-out testing demonstrate that ensemble methods (**Random Forest**, **XGBoost**) and the shallow **CNN** generalize effectively to unseen attack types and traffic patterns. Using 5-fold cross-validation, we observed low variance across metrics (standard deviation $< 1.5\%$), as reported in Table 1.

Precision and recall on validation sets remained nearly equal to training performance. For example, **XGBoost** achieved a validation F1-score of **98.5%** versus **98.8%** on the training set, indicating robust generalization, as visualized in the performance comparison in Figure 2.

However, XGBoost's near-perfect accuracy (**99%**) on the test set suggests potential overfitting, particularly on the balanced IoT botnet dataset. To address this, we applied regularization parameters:

- `reg_alpha` = 0.5
- `reg_lambda` = 1
- `max_depth` = 5

and incorporated **early stopping** (10 rounds), which reduced overfitting risk while maintaining high F1-scores (see Table 1).

Classical models like **Logistic Regression** and **Naive Bayes** exhibited stable performance under noisy or shifted data distributions, with variance below 2%, suggesting lower false alarm rates in production networks. This is further supported by their confusion matrices shown in Figure 1.

4.5. Practical Recommendations

Based on the comprehensive evaluation in Table 1 and visualized in Figure 2, **XGBoost** achieves the highest accuracy

(**99%**, F1-score: **98.8%**) but requires careful regularization to mitigate overfitting risks observed in its near-perfect performance (Figure 1). We recommend tuning with the following hyperparameters:

- `reg_alpha` = 0.5
- `reg_lambda` = 1
- `max_depth` = 5
- `learning_rate` = 0.05
- `n_estimators` = 100

Additionally, use **early stopping** (10 rounds) with validation monitoring to ensure robustness in real IoT deployments, as validated by the low cross-validation variance ($< 1.5\%$, see Section **??**).

The shallow **Convolutional Neural Network (CNN)**, with **97%** accuracy and **97%** F1-score (Table 1), is well-suited for IoT gateways. It is configured with:

- Two `Conv1D` layers (32 filters, kernel size = 3)
- `Dropout` = 0.2
- `Batch size` = 64

This architecture balances high accuracy with moderate resource demands (see Figure 2).

For scenarios prioritizing **transparency and interpretability**, **Random Forest** (94% accuracy, `max_depth` = 10) and **Decision Trees** (92% accuracy) offer strong performance along with clear feature importance insights (Figure 3). For instance, top-ranked features include:

- `stddev`: 0.25
- `N_IN_Conn_P_SrcIP`: 0.18

This makes them ideal for edge-level monitoring where interpretability is critical.

In ultra-constrained environments, **Logistic Regression** (87% accuracy, with `class_weight='balanced'`) or **Naive Bayes** (75% accuracy) offer fast inference with minimal memory usage, making them suitable for deployment on low-power embedded devices.

Model selection should align with hardware capabilities and application needs:

- Deploy **regularized XGBoost** or **CNN** at IoT gateways for maximum accuracy.
- Use **Random Forest** or **Decision Trees** for interpretable edge monitoring.
- Choose **Naive Bayes** or **Logistic Regression** for real-time, resource-limited sensors.

5. Conclusion

This study has performed a rigorous and multi-faceted evaluation of botnet detection in IoT networks, leveraging a suite of lightweight machine learning models. By analyzing diverse real-world IoT network traffic data and systematically comparing Naive Bayes, Logistic Regression, Decision Trees, Random Forests, XGBoost, k-Nearest Neighbors, and Convolutional Neural Networks, we have obtained detailed insights into each model's strengths and limitations for identifying DoS and DDoS attacks. Our feature engineering approach, including careful labeling and standardized preprocessing, underpinned robust and reproducible findings, reinforcing the importance of high-quality data preparation in security analytics. Experimental results demonstrated that ensemble and deep learning models, particularly XGBoost and shallow CNNs, deliver the highest detection accuracy and F1-scores for IoT botnet traffic, surpassing 97A distinctive aspect of our research is the assessment of each model's computational efficiency and suitability for deployment in real-world IoT settings. Classical models like Naive Bayes, Logistic Regression, and Decision Trees provide excellent choices for highly restrictive edge or embedded systems due to their fast inference, low memory requirements, and ease of explanation. Meanwhile, the adoption of lightweight CNN architecture demonstrates that deep learning can be made feasible for IoT gateways, provided the architecture remains shallow and the computational footprint is carefully managed. Taken together, these results advocate for a tiered defense strategy deploying ultra-lightweight models at the sensor level and more expressive, moderately complex models at gateways and aggregation points. In summary, this comprehensive analysis advances the understanding of both attack behaviors and machine learning-based detection strategies in the dynamic and resource-constrained landscape of IoT security. Our findings highlight not only the technical trade-offs between accuracy, efficiency, and interpretability, but also reinforce the necessity for adaptive hybrid frameworks and continual dataset curation to keep pace with evolving botnet tactics. As IoT deployments continue to expand in critical infrastructure and everyday environments, the insights and resources provided in this work will be instrumental in guiding the design and implementation of robust, scalable, and energy-aware botnet defense mechanisms for the rapidly growing Internet of Things

References

[1] Malik K et al., Lightweight internet of things botnet detection using one-class classification, *Sensors*, 22(10) (2022) 3646. https://doi.org/10.3390/s22103646.

[2] Esmaeilyfard R, Shoaei Z & Javidan R, A lightweight and efficient model for botnet detection in iot using stacked ensemble learning, *Soft Computing*, 29(1) (2025) 89–101. https://doi.org/10.1007/s00500-025-10425-1.

[3] Vennapureddy R & Srinivasulu T, Pragmatic study of botnet attack detection in an iot environment, *E3S Web of Conferences*, 591 (2024) 09012. https://doi.org/10.1051/e3sconf/202459109012.

[4] Doshi R, Apthorpe N & Feamster N. Machine learning ddos detection for consumer internet of things devices. In: *2018 IEEE Security and Privacy Workshops (SPW)* (2018), pp. 29–35. https://doi.org/10.1109/spw.2018.00013.

[5] Torres P, Catania C, Garcia S & Garino C. An analysis of recurrent neural networks for botnet detection behavior. In: *2016 IEEE Biennial Congress of Argentina (ARGENCON)* (2016), pp. 1–6.

[6] Moganedi S & Mtsweni J. Beyond the convenience of the internet of things: Security and privacy concerns. In: *2017 IST-Africa Week Conference (IST-Africa)* (2017), pp. 1–10.

[7] Stevanovic M & Pedersen J. An efficient flow-based botnet detection using supervised machine learning. In: *2014 International Conference on Computing, Networking and Communications (ICNC)* (2014). https://doi.org/10.1109/iccnc.2014.6785439.

[8] Ge M et al. Deep learning-based intrusion detection for iot networks. In: *2019 IEEE 24th Pacific Rim International Symposium on Dependable Computing (PRDC)* (2019), pp. 256–25609. https://doi.org/10.1109/prdc47002.2019.00056.

[9] Khan R et al., An adaptive multi-layer botnet detection technique using machine learning classifiers, *Applied Sciences*, 9(11) (2019) 2375. https://doi.org/10.3390/app9112375.

[10] Azab A, Alazab M & Aiash M. Machine learning based botnet identification traffic. In: *2016 IEEE Trustcom/BigDataSE/ISPA*. Tianjin, China (2016), pp. 1788–1794.

[11] Bansal A & Mahapatra S. A comparative analysis of machine learning techniques for botnet detection. In: *Proceedings of the 10th International Conference on Security of Information and Networks* (2017), pp. 91–98. https://doi.org/10.1145/3136825.3136874.

[12] Nugraha B, Nambiar A & Bauschert T. Performance evaluation of botnet detection using deep learning techniques. In: *2020 11th International Conference on Network of the Future (NoF)* (2020), pp. 141–149. https://doi.org/10.1109/nof50125.2020.9249198.