AI-Powered Botnet Detection: Deep and Machine Learning Strategies for Securing IoT

Maisha Maliha & Afsana Sharmin
Team 3

Abstract

In the rapidly growing Internet of Things (IoT) ecosystem, security remains a critical concern. This study focuses on strengthening IoT device security, particularly in detecting botnet threats. Using the NaiBot dataset, a labeled dataset for binary classification, we handled this supervised learning problem through machine learning and deep learning approaches. Our methodology involved training models such as Support Vector Machine (SVM), Decision Tree (DT), Random Forest (RF), Generative Adversarial Networks (GAN), Recurrent Neural Networks (RNN) and Convolution Neural Network (CNN). We employed multiple metrics to evaluate performance, including false alarm and missed detection rates. This research contributes to IoT security by offering a comprehensive assessment of machine learning and deep learning techniques for botnet detection.

Contents

1 Introduction

IoT (Internet of Things) security is a critical aspect of modern network security, focusing on the protection of interconnected systems across diverse sectors such as smart home automation, industrial control, healthcare, and transportation. With the rapid proliferation of IoT devices, ensuring their security has become paramount, as vulnerabilities within these systems can expose users to significant cyber risks. Unauthorized access to sensitive information, system breaches, and threats to personal safety are among the major concerns that need to be addressed.

One of the most pressing security challenges in the IoT ecosystem is the emergence of botnets. A botnet, short for "robot network," is a group of internet-connected devices that have been compromised by malware and are controlled remotely by an attacker. IoT botnets, in particular, are widely exploited for malicious purposes, including large-scale cyberattacks such as Distributed Denial-of-Service (DDoS) attacks [1]. In these attacks, a vast number of infected devices overwhelm a target network with excessive traffic, leading to service disruptions and potential operational failures. Given the increasing reliance on IoT technology, the threat posed by botnets is more alarming than ever.

To counteract these security risks, researchers and cybersecurity professionals have turned to advanced machine learning and behavioral analysis techniques. The application of various machine learning models has played a crucial role in enhancing botnet detection and fortifying network defenses. Notably, studies have employed algorithms such as Support Vector Machines (SVM), Random Forest, Decision Trees, and ensemble methods, each offering unique strengths and limitations in identifying botnet activity. By leveraging these models, cybersecurity experts aim to detect and mitigate cyber threats more efficiently.

Beyond traditional machine learning approaches, researchers are also exploring adaptive security frameworks that can dynamically respond to evolving communication technologies and deployment environments [2]. These frameworks emphasize the necessity of real-time threat detection and mitigation strategies. The continuous development of these security solutions underscores the importance of integrating cutting-edge technologies to strengthen network defenses against ever-evolving cyber threats.

In the following sections, we will discuss relevant research on IoT security and discuss our contributions to tackling these pressing challenges. Our study aims to provide a comprehensive evaluation of security strategies. Specifically, it focuses on the use of deep learning and machine learning techniques to enhance IoT cybersecurity.

2 Literatures Review

Botnets represent a significant threat in network security, capable of inducing server disruptions and posing substantial risks to network infrastructure. This research [3] contributes to the botnet detection through various machine learning models such as Naive Bayes, XGBoost, SVM(Support Vector Machine), and MLP(Multi-Layer Perceptron) models where SVM outperforms other models. A comparative analysis of these models highlights their respective strengths and limitations in botnet detection. The methodology is evaluated using real-world datasets from DGA, Alexa, and Secrepo repositories.

Similarly, supervised and unsupervised machine learning algorithms like SVM, Random Forest(RF), K-nearest neighbor(KNN), K-Means, Principal Component Analysis (PCA), and Recurrent Neural Networks (RNN) are used for botnet detection in this study [4]. The RF works better than any other model.

A Network Intrusion Detection System(NIDS) is developed to safeguard IoT environments against botnet attacks in [5]. The approach employed an ensemble-based methodology that uses machine learning algorithms, in order to detect attacks targeting the HTTP and DNS protocols within the network traffic. The performance of various classifiers such as Naive Bayes, Decision Tree and SVM were used and compared against the proposed ensemble method, primarily based on their detection rates and false positive rates.

Another study mainly focused on the selection of features to address botnet threats such as DDoS(Distributed Denial of Service) attacks and malware [6]. Support Vector Machine, Logistic Regression, Multi-layer Perceptron, and Random Forest classifier are used after applying feature selection. This approach achieves an impressive accuracy rate, a True Positive Rate (TPR), and a notably low False Positive Rate (FPR).

An innovative approach [7] was proposed by Maudoux et al. for identifying botnet threats by merging pre-processed Decision Trees. Using network flow features and merging Decision Trees and Random Forest algorithm, a combined Forest is introduced to detect attacks with precision.

A behavioural analysis is proposed to detect HTTP based botnet threats in this research [8]. After data pre-processing, they used Logistic Regression, SVM, Artificial Neural Network (ANN) and Decision Tree to detect the presence of these malwares. In the end, ANN performed better for their detection problem. Another paper [9] used classification algorithm such as KNN, Naïve Bayes, Decision Tree and Random Forest (RF) for identifying HTTP Botnet detection. The best classifier is KNN for this experiment.

De Neira et al. [10] introduces a ANTE system that employs autonomous machine learning to select the best machine learning pipeline for different scenarios, emphasizing the importance of tailored techniques for specific attack types and botnet variations. In [11], KNN, Naive Bayes, Decision Tree and SVM - four classifiers are used for the detection of botnet. In this experiment, Decision Tree classifier works better than the other classifiers.

Chi-square feature selection [12], [13] is used [14] to extract the best features from N-BaIoT dataset [15].Chi-square feature selection analyzes the data type of the features and calculates their relationship with the target variable. Different ML algorithms such as Logistic Regression, KNN, Naive Bayes, Decision Tree,RF, Multi-Layer Perceptron Network and Long Short-Term Memory(LSTM) are compared where Random Forest outperformed all the other methods.

Same N-BaIoT dataset is used in this study [16]. They have used hybrid feature selection method prior to ML algorithm where they found out KNN and RF achieved better result. [17] research used Naive Bayes, RF and ANN for multi-classification for the N-BaIoT dataset. For multi-classification the outcome of ANN algorithm is better. Another research [18] has been designed a neural network based NIDS that is Kitsune. It tracks the behavior of the network and employs ensemble auto-encoders for anomaly detection.

Our Contribution In this study, we enhance IoT security by leveraging machine learning (ML) and deep learning (DL) for botnet detection. Our key contributions include:

1. Comprehensive Model Evaluation – We analyze ML and DL models, including GAN, RNN, CNN and SVM, DT, RF for botnet detection.
2. Utilization of the NaiBot Dataset – A labeled dataset for binary classification of IoT traffic.
3. Integration of Deep Learning models– GAN, CNN, and RNN to observe the detection accuracy by capturing complex attack patterns.
4. We compared the deep learning methods with nonprobabilistic, and ensemble methods on the IoT botnet datasets.
5. Performance Analysis – We assess models using accuracy, F1-score, Precision, Recall as well as false alarm, and missed detection rates.

Our study provides a comprehensive AI-powered botnet detection framework to improve IoT cybersecurity.

3 Preliminary Data Examination

This entire section provides information on exploratory data analysis.

3.1 Overview of the Dataset

In this subsection, we discussed the process of data collection and preparation.

3.1.1 Data Collection

The dataset used in this study contains data from devices infected by Mirai and BASHLITE botnets, specifically focusing on the Provision PT 737E Security Camera. The raw traffic data was collected using port mirroring on a switch. For comparison, the dataset includes both benign (normal) traffic and nine types of attack traffic. It is multivariate and sequential, consisting of 115 numerical features, with no missing values. Feature extraction was done by capturing 23 features from a single time window. This was repeated across five different time windows: 100ms, 500ms, 1.5 seconds, 10 seconds, and 1 minute, resulting in a total of 115 extracted features. This dataset is well-suited for classification tasks and serves as a valuable resource for testing and evaluating deep & machine learning algorithms in IoT security, particularly to detect specific cyber attacks.

3.1.2 Data Organizing

We used five attack datasets (gafgyt.combo, gafgyt.junk, gafgyt.scan, gafgyt.tcp, and gafgyt.udp) along with a benign dataset for analysis. The details of these datasets, including their total instances and the number of selected instances, are shown in Table 1. To create a balanced dataset, we selected 2000 sequential instances from each attack dataset and whole benign dataset. Instead of random sampling, we opted for selecting the first 2000 samples from each attack dataset. The benign samples were labeled as '0', while attack samples were labeled as '1'.

Table 1: Dataset Preparation

Dataset Name	Total Instances	Instances Selected	Label
benign.csv	62154	62154	0
gafgyt.combo.csv	61380	2000	1
gafgyt.junk.csv	30898	2000	1
gafgyt.scan.csv	29297	2000	1
gafgyt.tcp.csv	104510	2000	1
gafgyt.udp.csv	104011	2000	1

The combined dataset was then split into three subsets. Table 2 shows the distribution of the datasets we used for our research.

Table 2: Dataset Distribution of Benign and Attack Samples

Dataset	Total Samples	Benign (%)	Attack (%)
Train	57,723	49,723 (86.14%)	8,000 (13.86%)
Validation (CV)	7,215	6,215 (86.14%)	1,000 (13.86%)
Test	7,216	6,216 (86.14%)	1,000 (13.86%)

Including both benign and attack samples in all three subsets ensures a balanced representation of real-world scenarios. Table 2 presents the distribution of benign and attack samples across the training, validation (CV), and test datasets. The training set consists of 57,723 samples, with 86.14% benign and 13.86% attack samples. The validation and test sets contain 7,215 and 7,216 samples, respectively, maintaining the same class distribution. This ensures a balanced evaluation of the model's performance across different phases of training and testing.This approach helps the models learn and generalize effectively, leading to robust and reliable botnet detection in practical IoT security applications.

3.2 Domain Specific Challenges

In addressing domain-specific challenges, the observation of non-linear patterns in scatter plots, shown in Figure 1 between features indicates complex relationships or dependencies among variables. The dataset shows a non-linear relationship between features. This means the values of one feature do not increase or decrease in a straight-line pattern with another feature. The scatter plot shows how $HH_L0.1_weight$ and HH_L3_mean are related.

The data points are spread in a curved and scattered way, not in a straight line. This tells us that simple models like linear regression will not work well. We need more advanced methods like decision trees or other deep learning models that can understand complex patterns. Later, in Section 4 such deep and machine learning models are discussed which we used to classify benign and malicious data in our dataset.

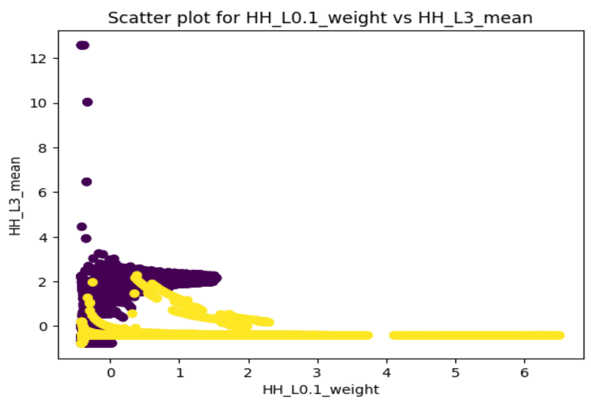


Figure 1: Non-linear relationship between features

4 Methodology

In this section, we will discuss about the machine learning and deep learning techniques that we have applied for our botnet security dataset. As it is a supervised binary classification problem, we have used common deep learning methods such as CNN, RNN and GAN. Also, we applied well known machine learning algorithms such as SVM, Random Forest & Decision Tree classifier.

4.1 Proposed Deep Learning Techniques

Our three deep learning approaches including CNN, GAN, and RNN are used for detecting botnet traffic in IoT networks. Each model is designed to address different aspects of the problem: spatial feature extraction, data imbalance handling, and temporal pattern analysis. Their effectiveness is evaluated and compared for robust anomaly detection.

4.1.1 *Convolution Neural Network (CNN)*

Convolutional Neural Network (CNN) is a powerful deep learning model widely used in feature extraction and pattern recognition tasks. It processes data with a grid-like topology, making it particularly effective for analyzing structured information in sequential datasets. CNNs utilize convolutional, pooling, and fully connected layers to learn hierarchical feature representations automatically.

Although CNNs are traditionally applied to image data, they are also highly effective for sequential and multivariate tabular data, such as network traffic. In this study, the dataset consists of 115 numerical features, extracted across five different time windows. The sequential nature of the data, combined with its multivariate structure, aligns well with CNNs.

CNNs are particularly suited for detecting anomalies or attacks in network traffic, such as those caused by botnet activity. The convolutional layers can automatically identify patterns and relationships between different features of the traffic, such as packet size, frequency, and other network behaviors, which may signal malicious activity. This ability to detect subtle and complex relationships in the data makes CNNs a valuable tool for improving classification performance in identifying botnet traffic from normal benign traffic.

4.1.2 *Generative Adversarial Network (GAN)*

Generative Adversarial Networks (GANs) consist of two main components: a generator and a discriminator. The generator creates synthetic data, while the discriminator evaluates it to decide whether it's real or fake. Through this adversarial training process, the generator gradually learns to produce highly realistic data that mimics real-world patterns.

In our approach, IoT botnet datasets are typically imbalanced, with far fewer attack instances compared to benign traffic. To tackle this, we focus the generator solely on creating attack data. By training only on real attack data, the generator learns to replicate the patterns of malicious traffic. Meanwhile, the discriminator is trained to distinguish between actual attack data and the synthetic attack data, which helps the generator improve and create even more realistic attack traffic.

By generating only attack traffic, the GAN helps augment the dataset with synthetic attack data, improving the performance of detection models. This approach effectively addresses the class imbalance issue, making the detection models more robust and capable of identifying adversarial attacks, even when attack traffic is underrepresented in the original dataset.

4.1.3 *Recurrent Neural Network (RNN)*

Recurrent Neural Networks (RNNs) are designed to process sequential data by retaining information from previous inputs, making them ideal for capturing time-dependent patterns. This is particularly useful for analyzing IoT network traffic, where botnet attacks like Mirai and BASHLITE evolve over time.

Our dataset, which contains 115 features collected across multiple time windows from IoT devices infected by botnets, is well-suited for RNNs. The traffic data includes both normal and malicious activity, with botnet patterns spread across time, requiring the model to capture the sequence of events.

To improve the RNN's ability to handle long-term dependencies, we use Long Short-Term Memory (LSTM) layers. LSTMs are designed to remember information over longer sequences, which is crucial for detecting botnet attacks that unfold over time. By retaining past information, LSTMs help the model better understand network traffic flow and identify potential attacks.Using LSTMs allows the model to learn from past traffic patterns and predict future attacks, making them especially effective for detecting the complex, time-based behavior of IoT botnet traffic.

4.2 Proposed Machine Learning Techniques

Our proposed machine learning techniques involve cross-validation which allow us to accurately classify anomalies and significantly increase the accuracy of the models avoiding overfitting.

4.2.1 *Support Vector Machine (SVM)*

Support Vector Machine (SVM) is a powerful non-probabilistic supervised learning algorithms. This algorithm with support vectors and margin in terms of separating the datapoints to their classes. SVM generally works great in classification problem that leads us to try this technique to find a hyperplane that best separates the two classes: benign and malicious.

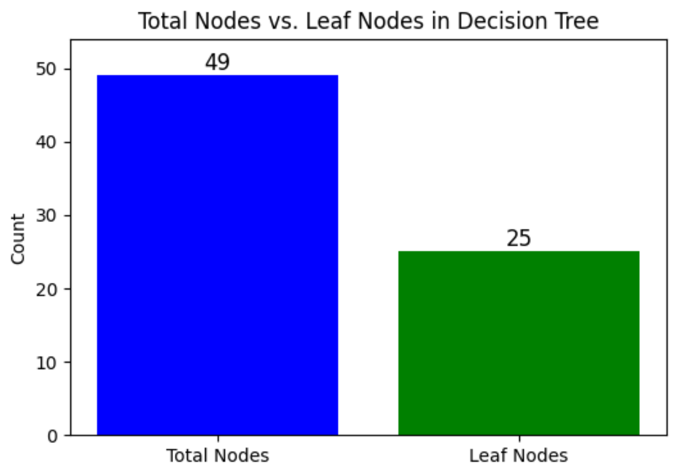
Given that our IoT botnet dataset contains 115 features, SVM can be a strong choice because it performs well in high-dimensional spaces. With so many features, simpler models like decision trees may struggle to find optimal splits without overfitting, especially if many features are correlated or noisy. In contrast, SVM focuses on finding the best boundary between classes by using only the most relevant data points (support vectors), which helps reduce the impact of irrelevant or redundant features. Additionally, the botnet traffic may have subtle patterns that are not easily captured by linear models; SVM's ability to use kernel functions allows it to model complex relationships in the data. This makes it a good fit for our dataset, where detecting small differences between normal and malicious traffic is critical for accurate classification.

4.2.2 *Decision Tree Classifier (DT)*

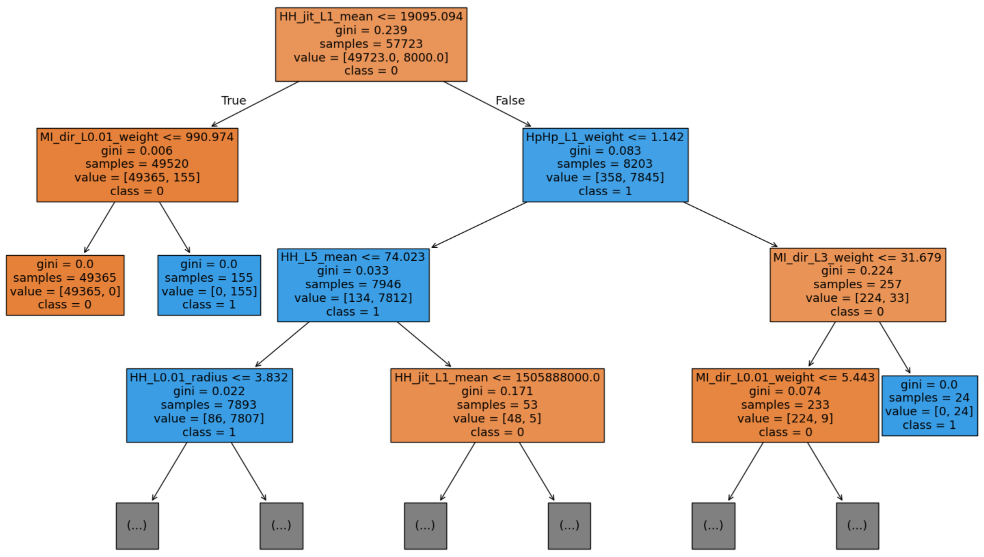
Decision tree is another efficient supervised machine learning classifier that is used for classification problem. This achieves a threshold value and then splits the datapoints into branches. The first split starts from the root. In the end, the outcome of this classifier is predicted by the terminal nodes, shown in Figure 2.

The Figure 2a shows a part of a decision tree extracted from the Random Forest model. Each box in the tree represents a decision node that splits the data based on a certain feature and threshold. The color of the boxes indicates the predicted class: orange for class 0 and blue for class 1. The tree uses features like $HH_jit_L1_mean$, $MI_dir_L0.01_weight$, and $HpHp_L1_weight$ to make decisions. At each step, the model tries to reduce impurity (measured by the Gini index) and split the data in a way that improves classification accuracy.

From this tree, we can observe that the majority of samples are classified as class 0, but the model is still able to separate out class 1 with clear splits. For example, when the condition $HH_jit_L1_mean <= 19095.094$ is false, the tree continues down the right side, making further splits to isolate class 1 samples.



(a) Total Nodes vs. Leaf Nodes



(b) Partial Decision Tree (Top 3 Levels)

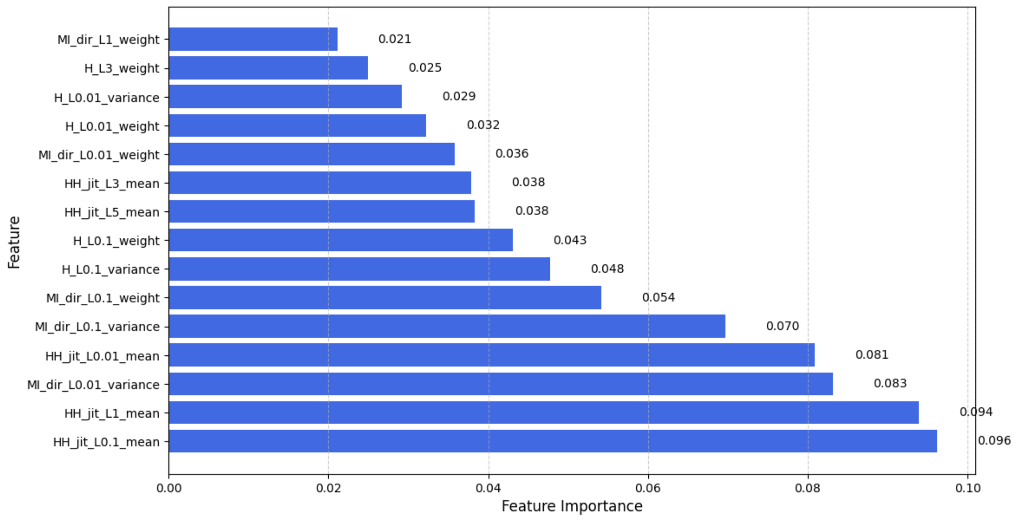
Figure 2: Decision Tree

The Figure 2b shows a simple bar chart comparing the total number of nodes and leaf nodes in the decision tree. The tree has 49 total nodes, out of which 25 are leaf nodes—these are the endpoints where final predictions are made. This balance suggests that the tree is complex enough to capture patterns in the data, but not too deep to cause overfitting. A good number of leaf nodes means the model has learned enough decision rules to make accurate predictions without becoming overly complicated.

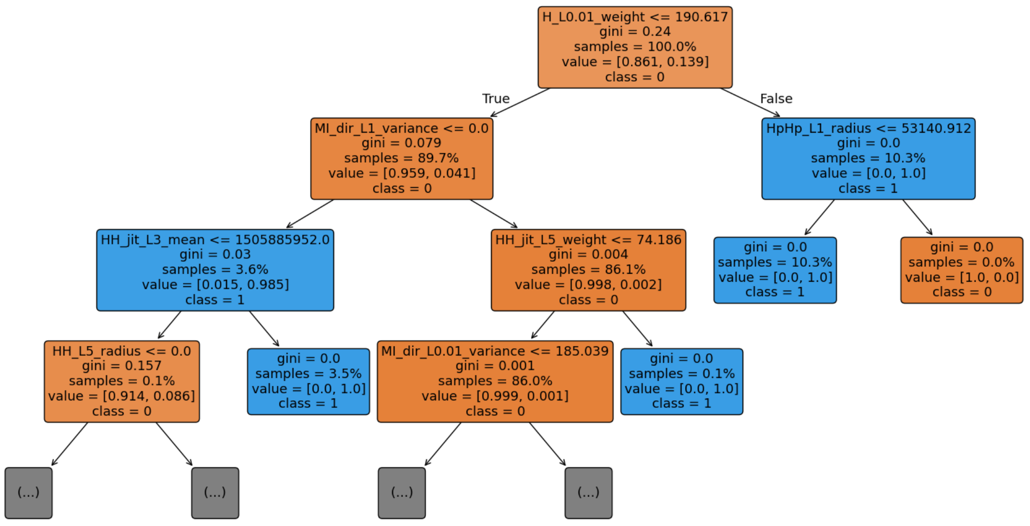
4.2.3 *Random Forest Classifier (RF)*

Random Forest is another ensemble method that is used for supervised machine learning algorithm. As we know, decision tree goes through by every attributes in the dataset one at a time and make predictions. On the other hand, we can see in figure 3b, random forest technique uses a collection of decision trees.

In this study, a Random Forest classifier random forest detect malicious activity in an IoT botnet dataset containing 115 features. Random Forest works well for this type of data because it builds multiple decision trees and combines their results, which helps improve accuracy and reduce overfitting. It is especially effective when dealing with high-dimensional data, as it automatically identifies the most important features for classification. The feature importance plot in



(a) Feature Importance



(b) Partial Random Forest Tree

Figure 3: Random Forest Tree

figure 3a highlights that only a subset of features—such as $HH_{j}it_{L}0.1_{m}ean$, $HH_{j}it_{L}1_{m}ean$, and $MI_{d}ir_{L}0.01_{v}ariance$ —contribute the most to the model's decision-making process. Additionally, the decision tree visualization shows how the model splits the data based on key feature values to distinguish between benign and malicious behavior. This confirms that Random Forest is a suitable and interpretable choice for intrusion detection in IoT environments.

5 Experimental Evaluation

In this section, a thorough analysis is done to compare our models' performance, taking into account metrics like accuracy, precision, recall, and F1 Score, shown in Table 3. To determine the precise error types in the model, the analysis also included a false alarm and missed detection rate evaluation.

5.1 Performance Metrics for Model Evaluation

We evaluate our models' performance using standard classification metrics. These metrics help in understanding the model's ability to distinguish between benign and malicious traffic. The primary performance indicators used in this study include Accuracy, Precision, Recall, and F1 Score. Additionally, we analyze error types through false alarm rate and missed detection rate to ensure a robust evaluation.

5.1.1 Accuracy

Accuracy is one of the fundamental evaluation metrics used to measure the proportion of correctly classified instances out of the total instances. It is calculated as:

Accuracy $= \frac{TP+TN}{TP+TN+FP+FN}$

where:

- TP (True Positive): Correctly classified attack instances.
- TN (True Negative): Correctly classified benign instances.
- FP (False Positive): Benign instances incorrectly classified as attacks.
- FN (False Negative): Attack instances incorrectly classified as benign.

A higher accuracy indicates that the model effectively distinguishes between benign and malicious traffic. However, accuracy alone may not be sufficient when dealing with imbalanced datasets, which is why additional metrics are necessary.

5.1.2 Precision

Precision evaluates the correctness of the model when predicting positive (attack) instances. It is given by:

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

Precision is particularly important in cybersecurity applications, as a high false positive rate (FP) can lead to unnecessary alerts and disruptions. A higher precision score implies fewer false positives, improving the reliability of attack detection.

5.1.3 Recall

Recall measures the model's ability to identify all actual attack instances. It is defined as:

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

A higher recall indicates that the model correctly identifies most attack instances while minimizing false negatives (FN). In cybersecurity, a low recall could mean that many real attacks go undetected, posing serious security risks.

5.1.4 F1 Score

The F1 Score provides a harmonic mean of precision and recall, balancing both metrics in cases where one might be more dominant than the other. It is computed as:

F1 Score $= 2 \times \frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$

The F1 Score is particularly useful when the dataset is imbalanced, as it ensures that neither precision nor recall is disproportionately prioritized.

5.2 Evaluation Metrics

From Table 3, we can see that **CNN, DT, and RF** performed the best. It achieved almost perfect accuracy, precision, recall, and F1-scores. **RNN** also showed high performance, with a slightly lower precision compared to RF but maintaining excellent recall and F1-score.

On the other hand, **GAN and SVM** performed poorly. GAN had an accuracy of only 20%, meaning it failed to classify most of the instances correctly. SVM had an even worse performance, with 0.00 in precision, recall, and F1-score, meaning it completely failed to make any correct predictions. This indicates that CNN, DT, and RF are highly reliable models, while GAN and SVM struggle in this particular task.

Table 3: Results

Models	Accuracy	Precision	Recall	F1-Score
RNN	0.9985	0.9891	1.0000	0.9945
GAN	0.20	0.0039	0.0039	0.0039
CNN	0.9993	0.9950	1.00	0.9975
DT	0.9994	0.9960	1.000	0.9980
RF	0.9999	0.9999	0.9999	0.9999
SVM	0.8614	0.000	0.000	0.000

5.3 False Alarm Rate & Missed Detection Rate

Table 4 shows how well the models balance false alarms and missed detections. CNN, DT, and RF again performed the best, achieving almost zero false alarms and no missed detections. This means these models make very few wrong alerts and correctly detect all positive cases. RNN also performed well, with a very low false alarm rate of 0.0018 and no missed detections.

GAN had the worst performance, with a false alarm rate of 1.00 and a missed detection rate of 0.9961. This means GAN classified almost everything incorrectly. SVM completely failed, having a missed detection rate of 1.00, meaning it did not detect any positive cases correctly. These results confirm that **CNN, DT, and RF** are the most reliable models for minimizing false alarms and missed detections.

Table 4: Performance Evaluation

Models	False Alarm Rate	Missed Detection Rate
RNN	0.0018	0.000
GAN	1.0000	0.9961
CNN	0.0008	0.000
DT	0.0006	0.000
RF	0.0002	0.000
SVM	0.000	1.000

5.4 Execution Time

Table 5 provides details about how long each model takes to train and test. SVM took the longest time to train (2318.07 seconds) and also had a very high testing time (70.97 seconds). Despite this, its performance was very poor, meaning it is not an efficient model for this task. CNN and RF also had long training times (469.67 seconds for CNN and 1077.93 seconds for RF), but they provided excellent results, making them worth the computational cost.

Table 5: Execution Time for Various Models

Model	Training Time (seconds)	Testing Time (seconds)
RNN	47.10	0.69
GAN	591.3	0.82
CNN	469.67	2.63
DT	1.98	0.0037
RF	1077.9301	0.025513
SVM	2318.07	70.9728

RNN had a moderate training time (47.10 seconds) and a very fast testing time (0.69 seconds), showing that it is efficient and performs well. DT was the fastest model, training in just 1.98 seconds and testing in 0.0037 seconds. This makes it an excellent choice when speed is a priority. GAN had a long training time (591.3 seconds) and a quick testing time (0.82 seconds), but its performance was very poor, making it an unreliable model. These results suggest that DT and RF are the best choices when considering both performance and execution time, while SVM is not a good option due to its high time cost and poor accuracy.

5.5 Comparison with Previous Works

The Table 6 compares accuracy of various ML models employed for the NaiBot dataset and the last two columns in the table compare the performance of these models on two additional datasets used for botnet detection: CTU-13 [4] and UNSWNB15 [5] dataset.

Our models showcase strong performance across different algorithms, notably achieving perfect accuracy (1.0) in RF and high accuracy in Decision Tree (DT), Recurrent Neural Network(RNN) and Convolution Neural Network (CNN). While our study uses PCA, other studies utilized different feature selection techniques such as Chi-square feature selection [14] in one case and a combination of filter and wrapper methods [16] in another. In this study [17], they use deep learning methods along with NB and RF. Another research [19] highlights the effective detection mechanisms by using six machine learning models (SVM) Support Vector Machine, (LR) Logistic Regression, (KNN) K-Nearest Neighbors, (NB) Naive Bayes, (DT) Decision Trees, and (RF) Random Forest which were evaluated on 09 (nine) IoT devices network traffic datasets. Our models demonstrate robust performance, especially in RF, DT, and CNN, showcasing their efficacy in botnet detection.

Table 6: Comparison with Previous Works

Models	**Accuracy**						
	Our Approach	**[17]**	**[14]**	**[16]**	**[19]**	**[4]**	**[5]**
RNN	0.9985	N/A	0.999	N/A	N/A	0.8994	N/A
GAN	0.20	N/A	N/A	N/A	N/A	N/A	N/A
CNN	0.9993	N/A	N/A	N/A	N/A	N/A	N/A
SVM	0.8614	N/A	N/A	N/A	0.79	0.9298	0.984
DT	0.996	N/A	0.999	N/A	0.91	N/A	0.967
RF	1.000	0.91	0.999	0.9992	0.99	0.999	N/A

6 Conclusion

In this project, we used deep and machine learning models to detect malicious activity in IoT networks using the NaiBot dataset. Among the models tested, Random Forest (RF) performed the best overall, achieving perfect accuracy (1.0), very low false alarms, and no missed detections. Decision Tree (DT) and Convolutional Neural Network (CNN) also gave excellent results with high accuracy and fast response times.

We also compared the models based on their training and testing times. DT was the fastest, while SVM and GAN were the slowest and least reliable. This shows that while some models are accurate, they may not be practical due to high computational cost. When comparing with previous research, our models showed strong performance, especially RF, DT, and CNN, proving their effectiveness for botnet detection in IoT systems. This work highlights the potential of machine learning in improving network security and can be extended in the future with more datasets and optimization techniques.

References

[1] Hoque, Nazrul, Dhruba K. Bhattacharyya, and Jugal K. Kalita. "Botnet in DDoS attacks: trends and challenges." IEEE Communications Surveys & Tutorials 17, no. 4 (2015): 2242-2270.

[2] Tawalbeh, Lo'ai, Fadi Muheidat, Mais Tawalbeh, and Muhannad Quwaider. "IoT Privacy and security: Challenges and solutions." Applied Sciences 10, no. 12 (2020): 4102.

[3] Wang, Haofan. "Botnet Detection via Machine Learning Techniques." In 2022 International Conference on Big Data, Information and Computer Network (BDICN), pp. 831-836. IEEE, 2022.

[4] K. B. Aswathi, S. Jayadev, N. Krishna, R. Krishnan and G. Sarath, "Botnet Detection using Machine Learning," 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), Kharagpur, India, 2021, pp. 1-7.

[5] Sivasankari, N., and S. Kamalakkannan. "Building NIDS for IoT Network using Ensemble Approach." In 2022 7th International Conference on Communication and Electronics Systems (ICCES), pp. 328-333. IEEE, 2022.

[6] Muhammad, Ali, Muhammad Asad, and Abdul Rehman Javed. "Robust early stage botnet detection using machine learning." In 2020 International Conference on Cyber Warfare and Security (ICCWS), pp. 1-6. IEEE, 2020.

[7] Maudoux, Christophe, Selma Boumerdassi, Alex Barcello, and Eric Renault. "Combined Forest: A New Supervised Approach for a Machine-Learning-Based Botnets Detection." In 2021 IEEE Global Communications Conference (GLOBECOM), pp. 01-06. IEEE, 2021.

[8] Goyal, Mohit, Ipsit Sahoo, and G. Geethakumari. "Http botnet detection in iot devices using network traffic analysis." In 2019 International Conference on Recent Advances in Energy-efficient Computing and Communication (ICRAECC), pp. 1-6. IEEE, 2019.

[9] Dollah, Rudy Fadhlee Mohd, M. A. Faizal, Fahmi Arif, Mohd Zaki Mas'ud, and Lee Kher Xin. "Machine learning for HTTP botnet detection using classifier algorithms." Journal of Telecommunication, Electronic and Computer Engineering (JTEC) 10, no. 1-7 (2018): 27-30.

[10] de Neira, Anderson Bergamini, Alex Medeiros Araujo, and Michele Nogueira. "Early botnet detection for the internet and the internet of things by autonomous machine learning." In 2020 16th International Conference on Mobility, Sensing and Networking (MSN), pp. 516-523. IEEE, 2020.

[11] Alshamkhany, Mustafa, Wisam Alshamkhany, Mohamed Mansour, Mueez Khan, Salam Dhou, and Fadi Aloul. "Botnet attack detection using machine learning." In 2020 14th International Conference on Innovations in Information Technology (IIT), pp. 203-208. IEEE, 2020.

[12] Thaseen, Ikram Sumaiya, and Cherukuri Aswani Kumar. "Intrusion detection model using fusion of chi-square feature selection and multi class SVM." Journal of King Saud University-Computer and Information Sciences 29, no. 4 (2017): 462-472.

[13] Thaseen, I. Sumaiya, Ch Aswani Kumar, and Amir Ahmad. "Integrated intrusion detection model using chi-square feature selection and ensemble of classifiers." Arabian Journal for Science and Engineering 44 (2019): 3357-3368.

[14] Gandhi, Rishabh, and Yanyan Li. "Comparing machine learning and deep learning for IoT botnet detection." In 2021 IEEE International Conference on Smart Computing (SMARTCOMP), pp. 234-239. IEEE, 2021.

[15] Al-Garadi, Mohammed Ali, Amr Mohamed, Abdulla Khalid Al-Ali, Xiaojiang Du, Ihsan Ali, and Mohsen Guizani. "A survey of machine and deep learning methods for internet of things (IoT) security." IEEE Communications Surveys & Tutorials 22, no. 3 (2020): 1646-1685.

[16] Guerra-Manzanares, Alejandro, Hayretdin Bahsi, and Sven Nõmm. "Hybrid feature selection models for machine learning based botnet detection in IoT networks." In 2019 International Conference on Cyberworlds (CW), pp. 324-327. IEEE, 2019.

[17] Tran, Thanh Cong, and Tran Khanh Dang. "Machine Learning for Multi-Classification of Botnets Attacks." In 2022 16th International Conference on Ubiquitous Information Management and Communication (IMCOM), pp. 1-8. IEEE, 2022.

[18] Mirsky, Yisroel, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. "Kitsune: an ensemble of autoencoders for online network intrusion detection." arXiv preprint arXiv:1802.09089 (2018).

[19] Hossain, Mahnaz, Rifat Al Mamun Rudro, Rupom Razzaque, and Kamruddin Nur. "Machine learning approaches for detecting IoT botnet attacks: A comparative study of N-BaloT dataset." In 2024 International Conference on Decision Aid Sciences and Applications (DASA), IEEE, 2024.