A Comparative Analysis of Black-Box and White-Box Models for IOT Botnet Detection

Richa Vivek Savant, Spoorthi M, Penumarty Krishna Mohan, Sreebha Bhaskaran and Shinu M. Rajagopal*
Department of Computer Science and Engineering
Amrita School of Computing, Bengaluru
Amrita Vishwa Vidyapeetham, India
*Corresponding Author: mr_shinu@blr.amrita.edu

***Abstract*—Botnets pose a significant threat to network security, especially to the potential server crashes they can cause. The proposed study focuses on methods for identifying the presence of and the type of botnet activity, by implementing machine learning techniques, and consecutively comparing the black-box and white-box approaches employed. Utilizing the CTU-13 and N-BaIoT datasets, the study involves data preprocessing and a comparative analysis of various models. The LGBM Classifier emerges as the top-performing model for detecting the presence of a botnet, while the XGBoost Classifier excels in identifying the specific type of botnet attack based on different performance evaluation metrics.**

***Index Terms*—botnets, network security, black-box, white-box, CTU-13, N-BaIoT, LGBM Classifier, XGBoost Classifier**

I. Introduction

The need for cybersecurity and protection against various cyberattacks has grown in recent years. The proliferation of apps used by people and businesses for personal or professional purposes, the IoT growing popularity, and the quick growth of computer networks have all contributed to the heightened attention that cybersecurity has received recently. A botnet is a collection of infected computers that is frequently managed remotely by a malevolent entity. These infected computers, referred to as ”bots,” unintentionally perform functions including distributing malware, initiating denial-of-service assaults, or pilfering confidential data. The zombie bot, also called a botnet, connects to an attacker-controlled command and control centre to get instructions for initiating other attacks.

The process of locating and reducing the threat of malicious botnets within a network is known as botnet detection. The act of protecting systems against unauthorized and potentially dangerous activity entails using a variety of approaches, including behavioural analysis, signature-based detection, and machine learning, to identify patterns and anomalies suggestive of botnet activity. Analyzing a botnet's behaviour and communication patterns within a network is necessary to determine its nature. Understanding the distinct characteristics and attack techniques of different botnet versions, such as Mirai or Gafgyt, is essential for distinguishing between them. The goal of the botnet can be ascertained by looking for particular traits in network traffic, such as the type of packets transmitted, and the packet features.

Machine learning concepts can be utilised to understand the botnet's behaviour and study the patterns within the network infected with botnets to detect the type of botnet attack and build an efficient system for the same. Algorithms improve decision-making processes by spotting patterns, predicting outcomes, and evolving with time. In machine learning, complicated algorithms that produce precise predictions but have opaque decision-making processes are referred to as ”black box” models. Users might not fully comprehend how these models arrive at particular results.

This study will work on the following:

- The proposed system of a two-staged architecture consisting of two efficient machine learning models one for each sub-problem of the problem statement: detection of presence of botnet, and type of botnet attack.
- A binary classification model to detect the presence of a botnet in a network.
- A multi-class classification model for further identifying the type of botnet and the type of attack performed by it.

This study is divided into multiple sections. Section I which is currently being discussed, provides an introduction regarding what botnets are, why their detection is important and which types of methods are being focused on in the proposed architecture to tackle the issue. Section II details the literature survey. Section III talks about the proposed methodology. Section IV highlights the results obtained by the proposed method and compares them with baseline models. Section V talks about the advantages, drawbacks and future enhancements that can be employed.

II. Literature Review

Javier et al. [2023] describe a model that operates on one-second time intervals designed to detect botnets as quickly as possible. A Decision Tree and four basic features—source and destination ports, packet count, and total bytes transmitted—were used to show a real-time method. It was discovered to be the fastest when put through rigorous testing with a 10 percent packet drop probability. When the procedure was parallelized, real-time detection for networks with speeds of 100 Mbps and 1 Gbps could be accomplished with four comparable CPU cores[1].

Mohammed et al. [2021] talk about the deployment stage that involves collecting network traffic using tcpdump, a lightweight tool for packet capturing. Raw network traffic is first directed to a feature extractor unit, Tshark, to extract relevant features from the packets. These features are then fed into a pre-trained classifier, which determines whether the packet is malicious or benign. Notably, the XGB classifier demonstrates superior accuracy and efficiency in terms of timing during this evaluation[2].

Hoafan et al. [2022] address the use of machine learning techniques by botnets to discover malicious domain names. It offers an approach that makes use of MLP, XGBoost, SVM, and Naive Bayes models. The DGA, Secrepo, and Alexa actual data sets are used in the paper's testing of these techniques. The findings demonstrate that the SVM model is best suited for detecting malicious domain names because this is the initial communication channel used by botnet-controlled mainframe systems to connect to command and control servers[3].

Nourhene et al. [2021] suggest a paradigm that emphasizes that performance and interpretation make use of both white-box and black-box models. For local interpretation, the framework makes use of Local Interpretable Model-agnostic Explanations (LIME), while for global interpretation, it makes use of permutation feature significance. The goal of LIME is to develop interpretable models, such as decision trees, logistic regression, and linear regression[4].

Hasan et al. [2021] propose a hybrid deep learning model using a convolutional neural network and long-term memory (CNN-LSTM) algorithm to detect botnet attacks on nine commercial IoT devices. This model achieved high accuracy in detecting botnet attacks from doorbells, thermostats, and security cameras with an accuracy of 90.88 percent and 88.61 percent respectively. However, it has low accuracy in detecting scans and TCP spoofing attacks. Research shows that the detection of botnet attacks depends on multiple training models rather than the type of IoT device. The CNN-LSTM model has shown superior results in detecting many botnet attacks[5].

Afnan et al. [2019] suggest a graph-based machine-learning model for botnet identification. The model makes use of five filter-based feature assessment metrics that are based on information, consistency, and correlation theories. Studies conducted on the CTU-13 and IoT-23 heterogeneous botnet datasets demonstrate that the use of features results in a reduction in training time and model complexity as well as high bot detection rates. The suggested detection methodology shows resilience against zero-day attacks and can identify various kinds of botnet families. When compared to cutting-edge methods, the model achieves competitive accuracy and greater precision. The results of the thorough feature assessment will help create a lightweight, effective botnet detection system[6].

Sneha et al. [2023] focus on how botnets pose a significant cybersecurity threat, and how traditional methods are ineffective. This work uses the CTU-13 dataset, a widely used cybersecurity dataset, to develop a machine learning-based method for botnet detection. This method uses real network traffic data from the botnet attack network environment to train various tree algorithms, including decision trees, regression, simple Bayes, and neural network models. The results show a high detection rate and a low false positive rate, making it possible to detect 99 percent of good traffic flows[7].

Farhad et al. [2023] present Harris Hawks binary multi-objective dynamic optimization (HHO) enhanced by mutation operator (MODHHO). This approach estimates possible features for intrusion detection using K-Nearest Neighbor, Support Vector Machine, Multilayer Perceptron and Decision Tree classifiers. According to the simulation findings, the MODHHO algorithm outperforms previous methods in IoT botnet detection with a lower error rate. Additionally, this approach outperforms competing classification algorithms on three datasets[8].

Aniket et al. [2023] provide a unique method of deep learning techniques based on Recurrent Neural Networks (RNNs) for regulating the prediction of botnet hosts. To improve recognition accuracy, the method makes use of the hybrid RNN model known as the Gated Recurrent Unit (GRU). To find possible risks, the model analyzes changing time series input from a network station. 85.4 percent total accuracy, 90.4 percent recall, and 81.6 percent precision are displayed in the findings[9].

Malak et al. [2022] discuss the recent developments in machine learning-based methods for bot identification and categorization on Facebook, Instagram, LinkedIn, Twitter, and Weibo are compared in this overview of the literature. It gives a summary of feature categories, datasets, and supervised, semi-supervised, and unsupervised techniques. To pinpoint knowledge gaps and carry out further in-depth research in the future, the study outlines possibilities, problems, and research objectives in this area. Bot detection was examined using a variety of ML and DL techniques[11].

Xiaoran et al. [2022] close a research gap in botnet identification by using grayscale photos from the CTU-13 and ISOT-2010 data sets for image recognition. With a 99.7 percent binary classification accuracy rate, the model lowers false alarm rates and has real-world applications[13].

III. Methodology

A. Dataset and Preprocessing

The first dataset used for identifying whether a network has a botnet or not is the CTU-13 dataset. It contains botnet attack traffic as well as background regular traffic that was recorded in 2011 at the CTU University in the Czech Republic. On concatenating and shuffling, the dataset comes to contain 92212 rows and 58 columns, out of which 53314 rows represent normal traffic and 38898 represent attack traffic. The attributes contained in the dataset include mostly packet-based features like packet lengths, flag counts, duration of flow, etc. On further analysis, no missing values or duplicates are found in the dataset. Most of the feature columns contain considerable outliers, hence Robust Scaling is used, which scales the data based on the median and interquartile range (IQR). To deal with the imbalanced dataset, the ADASYN (Adaptive

Synthetic Sampling) and SMOTE-Tomek (Synthetic Minority Over-sampling Technique with Tomek links) techniques are employed, between which SMOTE provides more satisfactory results, namely a balanced dataset having 42157 rows labelled benign and another 42157 rows labelled malignant.

Lasso regression hyperparameter tuning is performed using GridSearchCV with 5-fold cross-validation. It searches over a range of alpha values for the best hyperparameter and prints the optimal alpha value found, which happens to be 0.00001. The lasso model is then instantiated with the previously identified optimal hyperparameter. On calculating the lasso coefficients, a bar plot is generated to visualize the importance of each feature based on their corresponding coefficients. Features with an importance greater than 0.001 are selected. Through the identification and retention of the most pertinent aspects of the model, this technique helps with feature selection and may enhance the model's performance and interpretive capacity. Out of the original 58 features, 33 features are selected. A subset of the original DataFrame is constructed using the previously obtained relevant features, and SMOTE-Tomek is applied once again to this dataset to deal with possible oversampling. This dataset is then studied via various models.

N-BaIoT is the second dataset that was utilised to determine the kind of botnet attack. This dataset allows for practical assessment using actual traffic data, obtained from nine commercial IoT devices that were compromised by authentic botnets in two households on distinct networks. It enables the examination of two prevalent and demonstrated risky IoT-related botnets, namely Mirai and BASHLITE. Every file has 115 separate features plus a class label that can be found in the corresponding filename. Each feature header based on stream aggregation is described as follows:

1) H: The recent traffic statistics from the source IP of the packet.

2) MI: A summary of recent traffic statistics from the host (IP + MAC) of this packet, corresponding to the ”Source MAC-IP”.

3) HH: Statistics summarizing recent traffic from the source IP of this packet to the destination host, identified as ”Channel”.

4) ”Channel jitter” (HH jit): Traffic jitter statistics from the source IP of this packet to the destination host.

5) HpHp: Statistics detailing the recent traffic from the source host and port (IP) of this packet to the destination host and port, denoted as ”Socket”.

There are 8 classes in this dataset target variable as shown below and in Fig. 1 a detailed class distribution plot of the same is displayed:

0 - benign
1 - mirai udp attack
2 - mirai syn attack
3 - mirai scan attack
4 - mirai ack attack
5 - gafgyt udp attack
6 - gafgyt tcp attack
7 - gafgyt scan attack

Preprocessing is performed on the dataset, which includes removing null values and removing duplicate values. A Random Forest classifier is used for feature selection, and a subset of the original DataFrame is constructed, using the obtained relevant features having a feature score greater than a threshold of 0.03.

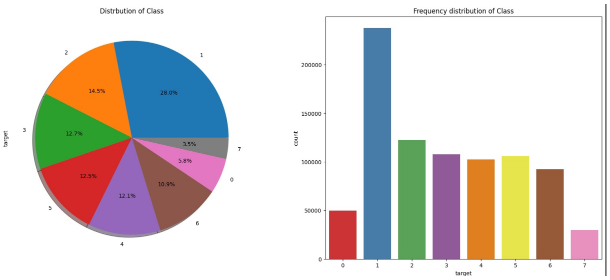


Fig. 1. Frequency distribution of various classes in N-BaIoT dataset

B. Black-Box Models

The black-box models used are the XGBoost Classifier, the Random Forest Classifier, the Support Vector Classifier, the k-Nearest Neighbors Classifier and the LGBM Classifier. eXtreme Gradient Boosting, or XGBoost, works incredibly well for regression and classification applications and is well-known for its exceptional predictive performance. Random Forest (RF) is a flexible and strong machine-learning classifier which enhances generalization by including randomness in data sampling and feature selection. Support Vector Classifier (SVC) performs well in high-dimensional spaces and is especially useful for handling complex data distributions. the k-Nearest Neighbors (KNN) classifier is a flexible and user-friendly machine learning technique. As part of the LightGBM framework, the LGBMClassifier is a scalable and effective machine learning model for gradient boosting. Its main advantages are its ability to increase prediction accuracy through iterative error correction and the use of elements like distributed and parallel computing for increased efficiency.

C. White-Box Models

The white-box models used are the Naive Bayes Classifier, Logistic Regression and the Decision Tree Classifier. The Naive Bayes Classifier is a good fit for high-dimensional datasets with discrete features. Logistic regression is very good at managing interpretable correlations in data and predicts distinct outcomes. The technique helps identify important variables by estimating the coefficients for each feature. For jobs involving both regression and classification, the Decision Tree classifier provides an adaptable and simple technique. Decision trees are useful for comprehending intricate data patterns because they may capture non-linear relationships.

D. Proposed Methodology

In the proposed study, both the CTU-13 dataset and the N-BaIoT dataset, after preprocessing are studied via the aforementioned white box models and black box models for binary classification and multi-class classification respectively, which are comparatively analyzed to decipher which model has the best performance. The effectiveness of the models is evaluated using metrics like precision, recall, F1-score, accuracy, AUC-ROC curve and confusion matrices.

The methodology showcases a balanced approach to handling high-dimensional data and ensures accuracy in botnet detection and classification.

E. Workflow

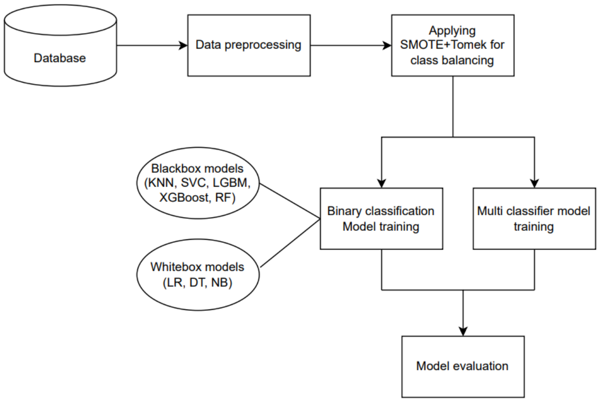

Database
Data preprocessing
Applying SMOTE+Tomek for class balancing
Blackbox models (KNN, SVC, LGBM, XGBoost, RF)
Whitebox models (LR, DT, NB)
Binary classification Model training
Multi classifier model training
Model evaluation

Fig. 2. Architecture of the proposed method

Fig. 2 represents the detailed workflow of the proposed method, which is a two-stage process, wherein the first stage deals with a simple binary classification model to detect the presence of a botnet and the second stage deals with the multi-class classification of the type of botnet and the type of botnet attack.

1) Binary Classification - CTU13: The initial step of the analysis is to pre-process the chosen dataset. The data is cleaned to remove noise and irregularity in data points; this includes removing those data points with missing values, removing duplicates, checking for outliers within the DataFrame using the z-score method, and changing the data types of particular features to more appropriate ones. The data is then passed through robust scaling. This method normalizes the data by scaling features to the interquartile range (IQR) after removing the median. Compared to traditional scaling, it is less susceptible to outliers, which makes it appropriate for datasets containing extreme values. A subset DataFrame from the existing DataFrame is then extracted based on certain selected features through lasso regression and elastic net models. These embedded feature selection models are passed through GridSearchCV functions to extract the best hyperparameters to give the best parameters to build the model with.

A feature set (X) and a target variable (y) are then made on both the datasets (full dataset and the subset dataset). Now the datasets are further split into corresponding sets of training data and testing data for further operations. The datasets are then passed through various white box models, namely, naive Bayes, logistic regression and decision trees; and black box models, namely, XGBoost, LGBM, Random forest models, SVC and kNN which are comparatively analyzed to decipher which model has the best performance. Metrics such as precision, recall, F1-score, accuracy, AUC-ROC curve, and confusion matrices are used to assess the performance of the models. The classification report gives a comprehensive look at how the model works by analyzing its performance across all the classes it recognizes. In addition, this gives a confusion matrix to know how well the model identifies each class.

2) Multi-Class Classification - N-BaIoT: The first phase of the analysis involves preprocessing the selected dataset. This entails eliminating noise and inconsistencies in data points, which encompass removing instances with missing values and duplicates. Additionally, the DataFrame undergoes outlier detection using the z-score method, and the count of outliers is computed. Column data types are adjusted, ensuring the target variable, represented by class indices, is converted to the appropriate integer type. The data is then subjected to robust scaling. Next, a heatmap of the correlation matrix is generated. The correlation matrix quantifies the linear relationship between pairs of variables in the dataset. This visualization aids in identifying patterns, dependencies, and potential multicollinearity within the dataset. For the feature selection step, the Random Forest Classifier is used and a threshold of 0.03 is set for feature importance. The features are first sorted according to their importance in the dataset and how they will affect the target variable and out of the 115 features a total of 15 more significant features are extracted into the subset dataset.

Next, for both datasets (the whole dataset and the subset dataset), a feature set (X) and a target variable (y) are created. For additional processes, the datasets are now further divided into matching sets of training and testing data. Next, the datasets are examined using the Random Forest, KNN, and Gradient Boosting classifiers; the models' respective performances are compared to determine which performs best. Metrics like precision, recall, F1-score, accuracy, AUC-ROC curve, and confusion matrices are used to assess how effective the models are. The classification report examines the model's performance over all the recognized classes to provide a thorough understanding of how it operates. Furthermore, this provides a confusion matrix that indicates how accurately the model recognizes each class.

IV Results and Analysis

F. Binary Classification - CTU13

The proposed method compares various white-box and black-box models in terms of their performance parameters. The white box models used are naive Bayes, logistic regression and decision trees. The black box models used are XGBoost, Random forest, SVC, KNN and LGBM. It is found that the LGBM Classifier has the best performance on the full dataset, with an accuracy of 0.9975058287697229 and a precision of 0.9967507148427346. Combining accuracy and precision, it gets a perfect F1-score of 0.9970098803952159, which shows how well it differentiates normal networks from botnet networks. Additionally, it has a 99 percent recall rate, so it identifies all botnets in the data set correctly.

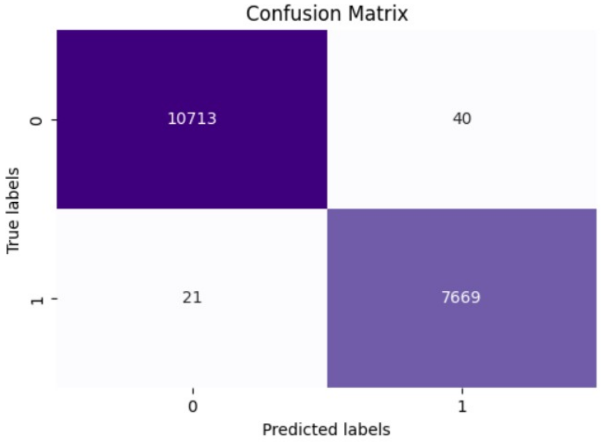


Fig. 3. Confusion matrix for LGBM Classifier

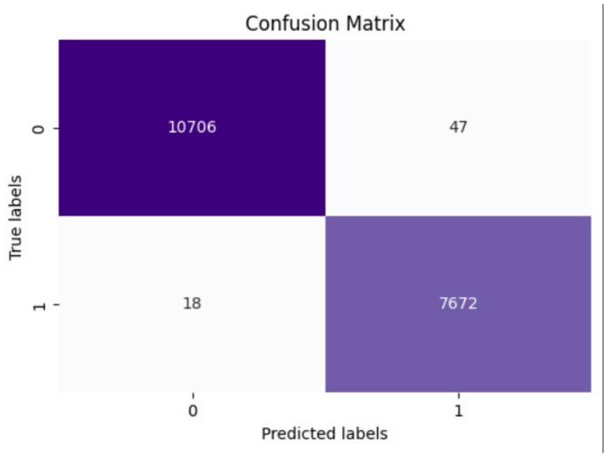


Fig. 4. Confusion matrix for XGBoost Classifier

Fig. 3 is that of the confusion matrix of the LGBM Classifier (blackbox model) which shows how well the model predicts classes 0 and 1 by the proposed model. On further analysis, it is observed that the Black-Box models, specifically the Boosting Technique Classifiers including the LGBM Classifier and XGBoost Classifier have the best performance. Fig. 4 shows the confusion matrix for XGBoost classifier (blackbox model), Fig. 5 shows the confusion matrix for Random Forest classifier (blackbox model), Fig. 6 shows the confusion matrix for KNN classifier (blackbox model).

Sophisticated detection systems that can recognize unusual network-wide communication and behaviour patterns are necessary for breaking into botnets. A robust and secure digital environment also helps to preserve important network resources and protects against potential cyber threats.

Fig. 7 gives a detailed comparative analysis of various performance metrics for all the different white box and blackbox models used in the experimentation. It can be observed that Boosting models outperform the other algorithms.

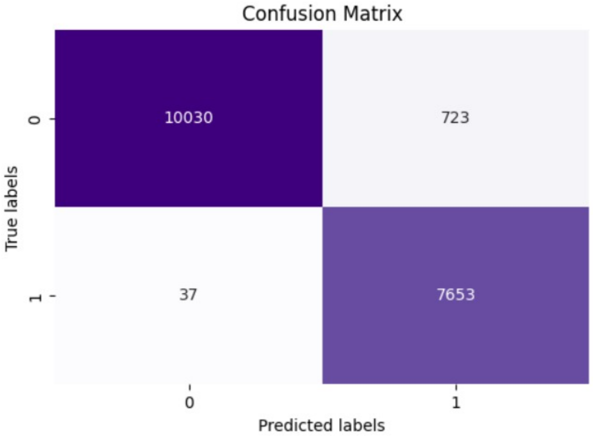


Fig. 5. Confusion matrix for Random Forest Classifier

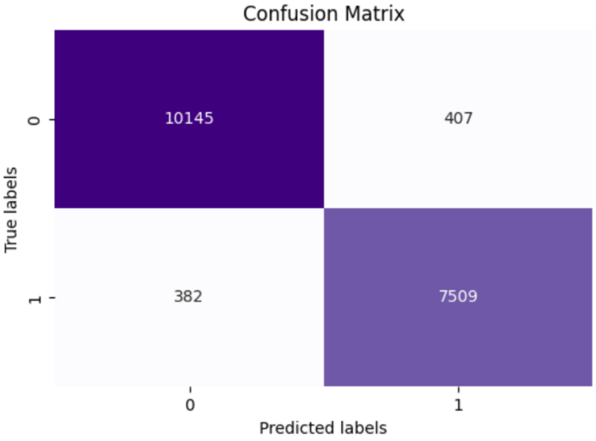
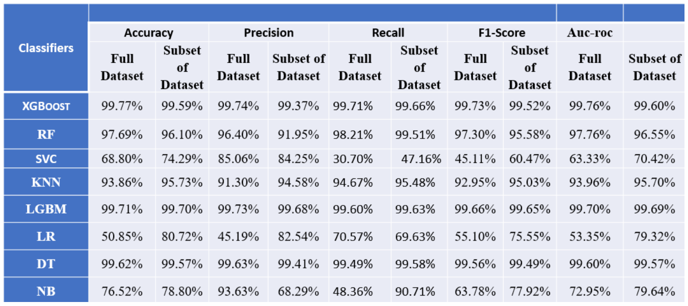


Fig. 6. Confusion matrix for KNN Classifier



Classifiers	Accuracy		Precision		Recall		F1-Score		Auc-roc	
	Full Dataset	Subset of Dataset	Full Dataset	Subset of Dataset	Full Dataset	Subset of Dataset	Full Dataset	Subset of Dataset	Full Dataset	Subset of Dataset
XGBoost	99.77%	99.59%	99.74%	99.37%	99.71%	99.66%	99.73%	99.52%	99.76%	99.60%
RF	97.69%	96.10%	96.40%	91.95%	98.21%	99.51%	97.30%	95.58%	97.76%	96.55%
SVC	68.80%	74.29%	85.06%	84.25%	30.70%	47.16%	45.11%	60.47%	63.33%	70.42%
KNN	93.86%	95.73%	91.30%	94.58%	94.67%	95.48%	92.95%	95.03%	93.96%	95.70%
LGBM	99.71%	99.70%	99.73%	99.68%	99.60%	99.63%	99.66%	99.65%	99.70%	99.69%
LR	50.85%	80.72%	45.19%	82.54%	70.57%	69.63%	55.10%	75.55%	53.35%	79.32%
DT	99.62%	99.57%	99.63%	99.41%	99.49%	99.58%	99.56%	99.49%	99.60%	99.57%
NB	76.52%	78.80%	93.63%	68.29%	48.36%	90.71%	63.78%	77.92%	72.95%	79.64%

Fig. 7. Comparison of performance of various blackbox and whitebox methods

G. *Multiclass Classification - N-BaIoT*

The proposed method compares various multi-class classification models based on their performance parameters. The classification models used for this purpose are the Random forest classifier, KNN classifier and Gradient Boosting classifier. With 0.9989 accuracy and 0.9984 precision on the entire dataset, the Gradient Boosting classifier is shown to perform the best. It achieves a flawless F1-score of 0.9972 by combining accuracy and precision. Furthermore, it accurately detects every botnet in the data collection with a 99 percent recall rate.

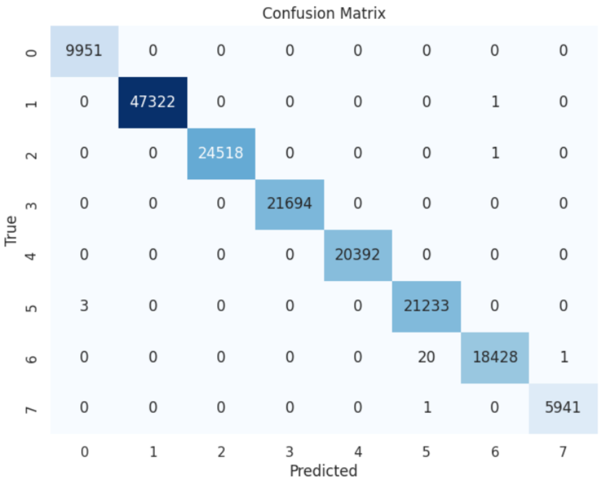


Fig. 8. Confusion matrix for Gradient Boosting classifier

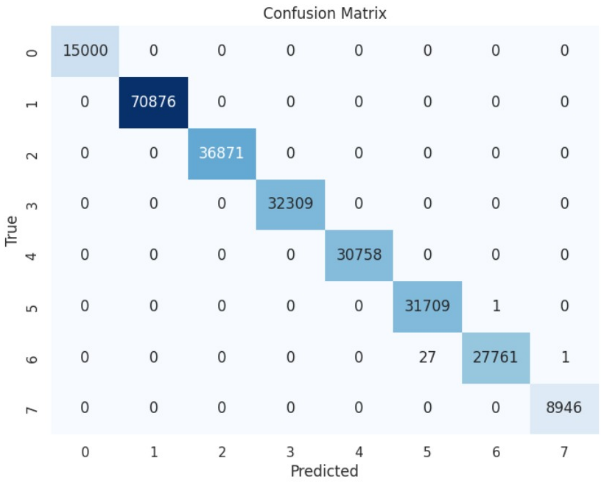


Fig. 9. Confusion matrix for Random forest classifier

The following figures represent the confusion matrices for multi-class classification models. Fig. 8 shows the confusion matrix for Gradient Boost classifier, Fig. 9 shows the confusion matrix for Random Forest classifier, and Fig. 10 shows the confusion matrix for KNN classifier. It can be observed from the above plots that, the gradient boost classifier performs the best out of all.

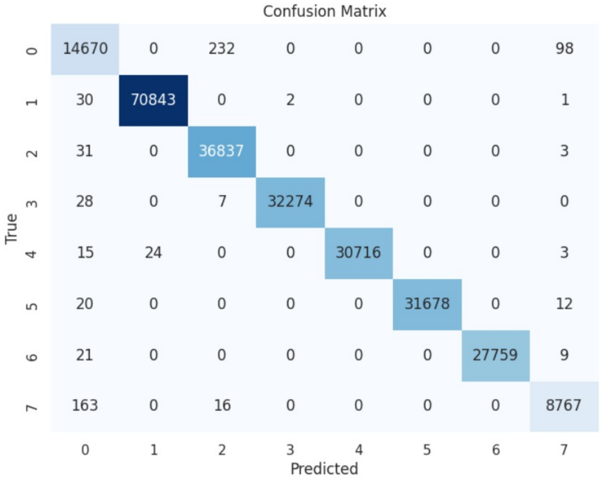


Fig. 10. Confusion matrix for KNN classifier

V Conclusion and Future Work

In this study, two datasets have been considered for building a two-staged efficient system for botnet detection. The CTU-13 and N-BaIoT datasets have been utilised to visualise patterns of botnet attacks and build a machine-learning model to tackle the issue at hand. For the initial stage of detecting the presence of a botnet, various white box models have been compared for their performances with black box models. The analysis of each of their evaluation metrics has been considered to conclude that boosting models perform well. For the final stage of detecting the type of botnet attack (UDP attacks, TCP attacks, SYN attacks, ACK attacks, etc.), machine learning models like Gradient boosting, random forest classifier and KNN classifiers have been made use of, to build a multi-classifier system for the N-BaIoT dataset containing information about two botnets namely 'Mirai' and 'Gafgyt'. From the evaluation metrics, it can be concluded that boosting models give the best performance for this stage as well. Overall, the application of boosting methods, to build the system enhances the efficiency and accuracy of botnet detection, making it a crucial component in modern cybersecurity strategies enabling the detection of previously unseen botnet behaviours.

As the amount of data being transferred keeps on increasing, it would be tough for a single system or a single server to identify different botnets or malware entering the system. To prevent heavy load on the server, distributed systems (or) distributed servers could help by dividing the incoming packet traffic and identifying and scrutinising the botnet at each sub-server. Hence, installing botnet detectors between nodes would be more cost-effective than at the nodes of the server and receiver. This kind of establishment could also prevent packet loss. These propositions can be worked upon for future research.

References

[1] Javier Velasco Mata, Víctor González Castro, Eduardo Fidalgo and Enrique Alegre, 2023 "Real time botnet detection on large network bandwidths using machine learning"

[2] Mohammed M. Alani, 2022 “BotStop : Packet-based efficient and explainable IoT botnet detection using machine learning”

[3] Haofan Wang, 2022 “Botnet Detection via Machine Learning Techniques”

[4] Nourhene, Bene dicte, Manuele, 2021 “IoT Botnet detection using Blackbox machine learning models”

[5] Hasan Alkahtani1, Theyazn H. H. Aldhyani, 2021 ”Botnet Attack Detection by Using CNN LSTM Model for Internet of Things Applications”

[6] Afnan Alharbi, Khalid Alsubhi, 2021 ”Botnet Detection Approach Using Graph-Based Machine Learning”

[7] Sneha Padhiar, Ritesh Patel, 2023 “Performance evaluation of botnet detection using machine learning techniques”

[8] Farhad Soleimanian Gharehchopogh, Benyamin Abdollahzadeh, Saeid Barshandeh, Bahman Arasteh, 2023 “A multi-objective mutation-based dynamic Harris Hawks optimization for botnet detection in IoT”

[9] Aniket Mishra, Indira Bharathi, 2023, “ Data driven approach to identify a flow-based Botnet Host using Deep Learning”

[10] Oliver Kornyo, Michael Asante, Richard Opoku, Kwabena Owusu-Agyemang, Benjamin Tei Partey, Emmanuel Kwesi Baah, Nkrumah Boad, 2023, “Botnet attacks classification in AMI networks with recursive feature elimination (RFE) and machine learning algorithms”

[11] Malak Aljabri, Rachid Zagrouba, Afrah Shaahid, Fatima Alnasser, Asalah Saleh, Dorieh M. Alomar, 2022 “Machine learning based social media bot detection: a comprehensive literature review”

[12] Rizwan Hamid Randhawa, Nauman Aslam, Mohammad Alauthman, Muhammad Khalid, Husnain Rafiq, 2023, “Deep reinforcement learning based Evasion Generative Adversarial Network for botnet detection”

[13] Xiaoran Yang, Zhen Guo, Zetian Mai, 2022, “Botnet Detection Based on Machine Learning”

[14] N. B. Gokul and Sriram Sankaran, “Identity Based Security Framework For Smart Cities”, 2020 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS). 2020.

[15] J. Sunny, Sriram Sankaran, and Saraswat, V., “A Hybrid Approach for Fast Anomaly Detection in Controller Area Networks”, 2020 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS). 2020.

[16] R. S. Ramachandruni and Poornachandran, P., “Detecting the network attack vectors on SCADA systems”, in 2015 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2015, 2015, pp. 707-712.

[17] Nimmy, K., Sankaran, S. and Achuthan, K., “A Novel Lightweight PUF based Authentication Protocol for IoT without Explicit CRPs in Verifier Database”, Journal of Ambient Intelligence and Humanized Computing, 14, 6227–6242 (2023), DOI: https://doi.org/10.1007/s12652-021-03421-4.

[18] P. Poornachandran, Sreeram, R., Krishnan, M. R., Pal, S., Sankar, A. U. Prem, and Ashok, A., “Internet of Vulnerable Things (IoVT): Detecting Vulnerable SOHO Routers”, in Proceedings - 2015 14th International Conference on Information Technology, ICIT 2015, 2015, pp. 119-123.

[19] R. S. Ramachandruni and Poornachandran, P., “Detecting the network attack vectors on SCADA systems”, in 2015 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2015, 2015, pp. 707-712.

[20] L. Mohan, M. K. Jinesh, Bipin, K., Harikrishnan, P., and Shiju Sathyadevan, “Implementation of Scatternet in an Intelligent IoT Gateway”, in Emerging ICT for Bridging the Future - Proceedings of the 49th Annual Convention of the Computer Society of India CSI Volume 2, Hyderabad, 2015, vol. 338, pp. 275–287.

[21] N. Manmadhan, Hari, N. N., Jayaraj Poroor, and Achuthan, K., “Design for Prevention of Intranet Information Leakage via Emails”, Security in Computing and Communications: Second International Symposium, SSCC 2014, Proceedings of Communications in Computer and Information Science, vol. 467. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 136–148, 2014