Mesopotamian journal of Cybersecurity
Vol.5,No.2, **pp**. 477–490
DOI: https://doi.org/10.58496/MJCS/2025/029; ISSN: 2958-6542
https://mesopotamian.press/journals/index.php/cybersecurity

Research Article

Threat Detection Based on Explainable AI (XAI) and Hybrid Learning

Shatha J. Mohammed [1], Bashar M. Nema [1,*]

[1] *Department of Computer Science, College of Science, Mustansiriyah University, Baghdad, Iraq.*

ARTICLE INFO

Article History

Received 04 Dec 2024
Revised 15 Mar 2025
Accepted 21 Mar 2025
Published 08 Jun 2025

Keywords

Explainable AI (XAI)

Machine Learning

Botnet IDS

IoT

Random forest

ABSTRACT

The proliferation of IoT networks has necessitated advanced botnet intrusion detection systems beyond conventional security measures. This research addresses the critical challenge of "black box" machine learning models in network security by integrating explainable AI (XAI) with hybrid learning approaches. We developed and evaluated three hybrid ML classifiers—XGBoost, decision tree, and random forest—using the UNSW-NB15 dataset to distinguish between benign and malicious network traffic patterns. The performance metrics demonstrated that our classifier-comparison methodology effectively enhanced botnet detection capabilities within organizational network streams. By implementing XAI techniques through Scikit-Learn, LIME, ELI5, and SHAP libraries, we transformed opaque ML models into interpretable systems with clear decision rationales. The results confirm that XAI integration is both feasible and beneficial, offering network security professionals transparent insights into threat detection processes while maintaining high performance. This research bridges the gap between advanced ML detection capabilities and the interpretability requirements essential for practical security implementations.

1. INTRODUCTION

Owing to the expanded measure of important client information contained in IoT organizations, they have turned into an undeniably significant objective for vindictive assaults. It is particularly important in basic administrations where digital assaults endeavour to sabotage the security standards of secrecy, uprightness, and accessibility [1]. Following initial advancements, researchers have developed Botnet Network Intrusion Detection Systems (Botnet-IDSs) and associated frameworks to monitor and detect cyberattack patterns within network management environments, utilizing artificial intelligence (AI) algorithms and models. These systems are designed to analyse the entirety of network traffic data, issuing alerts whenever they identify activities suggestive of an attack. Owing to shortcomings in the performance of current and conventional intrusion detection systems (IDSs), studies have increasingly explored machine learning (ML) algorithms as potential frameworks to improve IDS functionality.

- Owing to the unique structure and behavior of the IoT, conventional network security solutions might not be directly applicable.
- The power consumption is low, and the processing power is minimal.
- Conventional security measures, including authentication and encryption protocols, are inapplicable in this case.
- A single standard for IoT architecture, policies, and connectivity domains is lacking.

Current machine learning technologies facilitate the detection of botnets and malicious activities by extracting key features from IoT network inputs, irrespective of whether these features reflect normal or harmful behaviours.

2. METHODOLOGY AND PREVIOUS STUDIES

Optimizing data confidentiality in IoT ecosystems requires understanding ML-based intrusion detection systems (IDSs). Current ML-based IDSs exhibit algorithmic opacity, hindering the validation of model inferences by security analysts. As the IoT infrastructure expands, resilient security frameworks become imperative. This research mitigates model opacity in ML-based IDSs through explainable AI (XAI) integration, enhancing interpretability for practitioners. XAI methodologies demonstrate statistical significance in optimizing detection precision while minimizing false positives, thereby advancing operational IDS effectiveness in IoT security infrastructure. To optimize initial planning for privacy and data security while

*Corresponding author.Email: bmn774@uomustansiriyah.edu.iq

effectively allocating resources for IoT networks, it is critical to understand the methodology used by machine learning-based intrusion detection systems (IDSs) in predicting or classifying attacks. This can be outlined as follows:
Research shows that ML-based intrusion detection systems are often opaque 'black boxes' to cybersecurity users. The rise of IoT networks has increased demand for strong cybersecurity. A better understanding of these systems could help researchers and security professionals work more effectively and protect digital assets [2].
This work aims to expand the different model categorization, prediction logic, and predictability features. By doing so, we can increase the transparency of ML-based IDSs so that human analysts in the IoT cybersecurity domain can understand them better than before [3,4].
Owing to their varied uses, these tools have the potential for applying explanatory ability features to IoT network security intrusion detection system (NIDS) issues. To enhance the ability to explain, accuracy performance metrics have also been measured [5].
As the IoT is integrated into applications, cybersecurity specialists have focused on protecting IoT networks [6,7]. The creation of interruption discovery frameworks via AI approaches and other quantifiable component learning calculations has rarely been investigated and applied. In light of the proposed factual stream highlights for Web of Things network traffic protection, Nour Mustafa promoted an outfit-based AI interruption finding procedure [8]. His architecture started with organization highlights and built factual stream highlights. After that, AdaBoost outfit learning was used for the decision tree, Guileless Bayes (NB), and counterfeit brain organization AI computations. The UNSW-NB15 and NIMS datasets containing emulated IoT measurements were used to execute the algorithms to identify malicious events [9]. The experimental findings revealed high normal and harmful activity. Compared with three others common IoT network safety solutions, the outfit strategy has a greater identification rate and a lower false positive rate.
Kelton da Costa [3] presented several different AI-based interruption recognition techniques. His study examines AI technologies used in the Web of Things and interruption recognition for PC network security by online security experts. Numerous papers in machine learning subfields and IoT security were evaluated.
An article [4] on explainable artificial intelligence (XAI) summarized the core idea, taxonomy, potential, and all issues with AI responsibility. It may be used to examine the explainability of the ML approach. Since ML approaches such as ensembles and deep learning networks lack explainability, XAI has gained popularity. XAI is essential for ML model deployments, where researchers want openness, fairness, model explainability, and accountability. Shraddha Mane and Dattaraj Rao investigated the construction of a NIDS via an explainable AI tool [1].
ML models are more accurate but more complicated and less interpretable. The article introduced explainable AI to provide transparency of the model across the ML conduit and a deep learning network for the NIDS [10]. They used Protidic, SHAP, Contrastive-Explanations-Method (CEM), LIME, and Boolean-Decision-Rules via Column-Generation (BRCG) XAI algorithms to explain individual predictions on the NSL-KDD dataset to increase model flexibility [11].

3. INTERNET OF THINGS AND VULNERABILITIES

Between 2017 and 2023, the global population of Internet of Things (IoT) devices is projected to increase from 8 billion to 22 billion, representing a growth of more than twofold. However, a significant proportion of these IoT devices exhibit compromised security. A detailed investigation of ten commonly used devices identified 250 vulnerabilities, including open telnet ports, outdated firmware, and unencrypted transmission of sensitive data. This rising prevalence of insecure devices has been correlated with an escalation in botnet attacks, notably those perpetrated by the parcel botnet, which target internet infrastructure. In October 2016, the Mirai botnet used ~100,000 IoT devices, mostly security cameras, for a DDoS attack on Dyn DNS, briefly knocking out sites such as GitHub and Amazon. After its source code was released in January 2017, IoT botnet DDoS attacks surged, driving new detection methods. Recent anomaly detection research has shown the promise of machine learning in spotting malicious traffic [12][13].
In light of this perception, we create an AI pipeline for IoT traffic DDoS location that incorporates information assortment, highlight extraction, and double grouping. The highlights are expected to use network stream attributes such as parcel length, bundle spans, and conventions while additionally exploiting IoT-explicit organization ways of behaving. For the assault examination, we look at various classifiers. We produce classifiers that prepare information by recreating a purchaser IoT gadget network since there are no freely accessible datasets of customer IoT assault traffic. We set up a nearby organization that incorporated a switch, some famous shopper IoT gadgets for harmless traffic, and some ill-disposed gadgets that performed DoS assaults. Ideally, our classifiers will want to recognize assault traffic with an exactness high ratio. We intend to use datasets from Kaggle or the UCI Libraries [15].

4. PROPOSED SYSTEM

IoT devices can be used as bots in DDoS attacks. There has been a rise in IoT-based botnet attacks due to the rapid proliferation of IoT devices, which are easier to hack than PCs. In a botnet attack, the attacker overwhelms a target by coordinating the actions of many IoT devices. This type is difficult to detect since the device continues to function normally, and the client will not notice whether his device is a victim of an attack; at times, the device may experience a decrease in utility.

5. UNSW-NB15 DATASET

UNSW-NB15 is an IoT-based network traffic informational index that classifies typical organizational action and malignant botnet assault behavior (Fussers, Examination, Indirect Accesses, DoS, Exploits, Conventional, Surveillance, Shellcode, and Worms). The UNSW-NB 15 dataset's crude organization bundles were created by the IXIA Perfect Storm device in the Digital Reach Lab of the Australian Community for Digital Protection (ACCS) to produce a cross breed of authentic, modern-day exercises and engineered, modern-day assault behaviours on IoT-based networks. Fig. 1 depicts the UNSW-NB15 testbed configuration dataset and feature creation method [16].
UNSW-NB15 can be divided into two sets, training set.csv and testing set.csv, so that the model can be trained and tested accordingly. There are only 175,341 records in the training set and 82,332 in the testing set, with normal, attack, and target responses for traffic behavior for each record. Three score numerical characteristics make up the dataset. UNSW-NB15 features.csv contains a list of features along with descriptions of each. The experiments are conducted via a binary classification system with "Normal" and "Attack" designated as the target features. Fig. 2 shows the breakdown and value distribution for each attack class in the subsets of data; 0 indicates normal behavior, and 1 indicates attack behavior. The dataset demonstrates a balanced distribution concerning the binary response variable related to activity behavior.

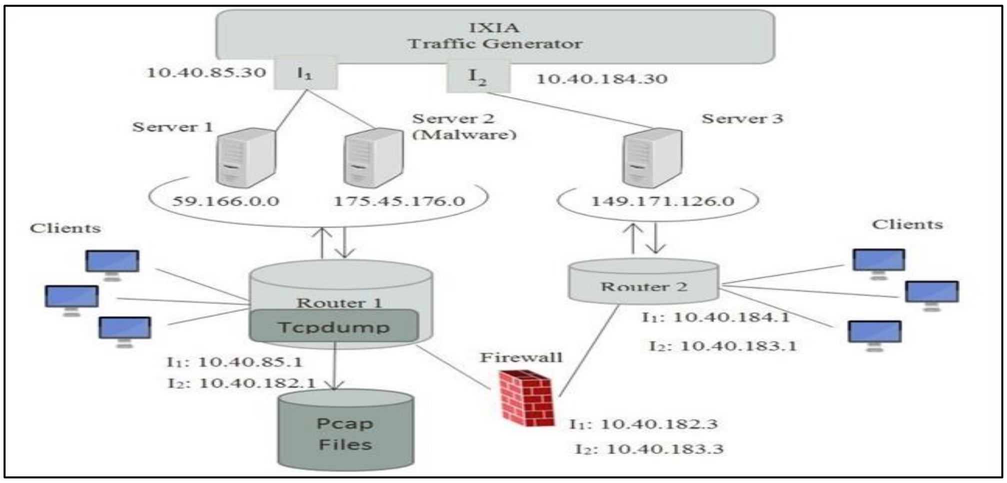

Fig. 1. IXIA Traffic Generator Overview [16]

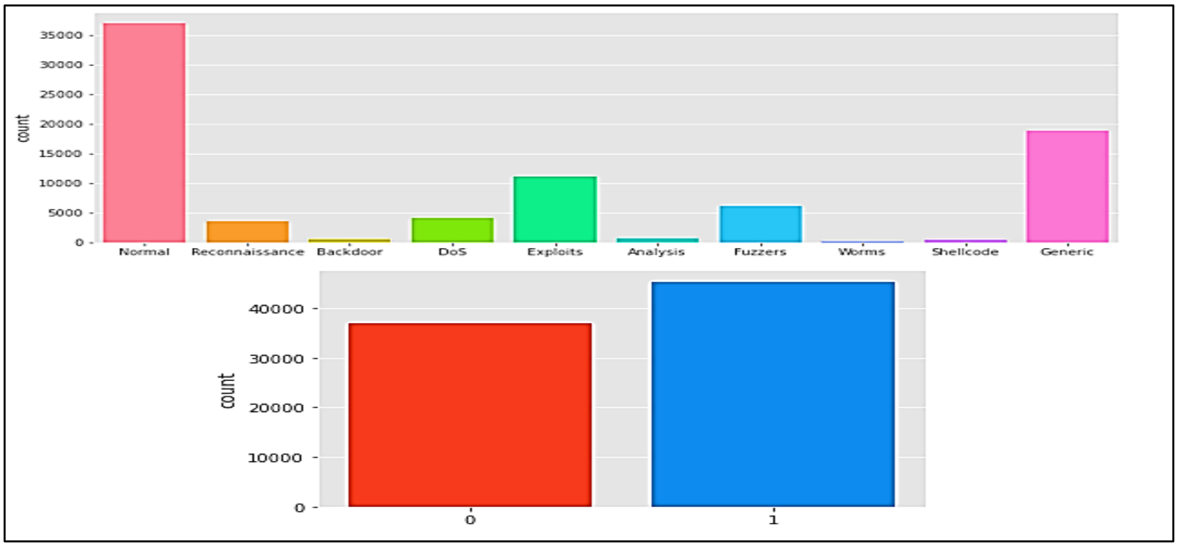

Fig. 2. Training dataset distribution and counts

6. ML METHODS

Decision trees, random forests, and XGBoost are integrated to create a binary classification classifier, which is a hybrid of three supervised ML approaches. In terms of explainability, the ML algorithms listed in that sequence provide progressively worse results (XAIs) [18,19, 20, 21].
Alternatively, research has utilized the primal formulation of max-margin multilabel classification (M3L) to train a classifier. This approach incorporates additive Laplace smoothing and leverages previous knowledge of tightly correlated labels. After the reduced set of attributes is obtained, the M3L method is applied with Laplace smoothing to categorize the instances as either benign or botnet. The M3L algorithm computes posterior probabilities associated with each class on the basis of feature values and assigns instances to the class with the highest probability. The exponential number of restrictions may be decreased to a linear number while simultaneously achieving extended 1-vs.-all multilabel classification [17]:

$$P1 = \frac{\min}{\mathrm{f}} \left(\frac{1}{2} \parallel f \parallel^2 + C \sum_{i=0}^{N} \xi i \right) \tag{1}$$

which is subject to:

$$\mathrm{f(xi,yi)} \geq \mathrm{f(xi,y)} + \Delta\mathrm{(yi,y)} - \xi\mathrm{i}, \forall \mathrm{i,y} \in \{\pm 1\}\mathrm{L} \backslash \{\mathrm{yi}\};\ \xi\mathrm{i} \geq 0, \forall \mathrm{I} \tag{2}$$

Ultimately, the output results of the two proposed classifiers are evaluated, and the maximum classification ratio is selected for botnet detection and classification. In this proposed classification work, the output of the DT classification result with dataset features is passed to the M3L classifier as f. When a new point x comes along, names are given to the following random forest classifier:

$$y^{*} = \arg \frac{\min}{y} \qquad f(x,y) \tag{3}$$

As an improvement in the bagging technique, the random forest algorithm builds a forest of decision trees that are not connected to each other by mixing bagging and feature chance. Feature bagging, "the random subspace method" (link resides outside ibm.com), and feature randomness all refer to the same thing: generating a randomly selected subset of features. This guarantees that decision trees will have low correlation. Decision trees and random forests differ significantly in this regard. Decision trees consider all possible feature splits, whereas random forests pick and choose which features to use.

6.1 Decision Tree Classifier

To classify the behavior as either normal or malicious via the 39 input features, a supervised ML algorithm called the decision tree (DT) classifier is employed. A decision-making operation resembling a tree-like model with branches or nodes is developed by the DT algorithm that follows. Prior to running the decision tree, the maximum depth can be set. By visualizing the resulting trees, decision trees are already machine learning algorithms that can be explained.

6.2 Random Forest classifier

The random forest (RF) algorithm is a powerful supervised learning algorithm that has been successfully applied to botnet detection. RF seeks to find an optimal hyperplane that separates different classes in the feature space, maximizing the margin between them. RFs are effective in handling high-dimensional data and can capture nonlinear relationships through the use of kernel functions. RFs have demonstrated the best performance in botnet detection, particularly in scenarios with a clear separation between botnet and nonbotnet instances.

6.3 XGBoost Classifier

XGBoost (eXtreme gradient boosting) is the most effective option for fast and efficient gradient-boosted decision trees. It is an open-source programming library in which computations are executed to ensure the effectiveness of the process time and memory assets. The plan design considers the best utilization of available resources to develop the model. The XGBoost library executes a choice tree calculation given that the slope helps. Supporting is an outfit method that adds new models to rectify the errors present in current models. Slope is a strategy wherein new models are developed that predict the residuals or errors of previous models, which are then joined to make the last expectation. Moreover, the slope plunge calculation limits misfortune during the incorporation of new models. This helps model normal or attack behavior classification and prediction.

7. PROPOSED HYBRID APPROACH

Three decision trees, random forest-supervised ML binary classifiers, and XGBoost, are trained on the UNSW-NB15 training dataset following pre-processing techniques for data cleaning, normalization, and transformation. The feature that will be targeted is a binary classification of behavior, either normal (0) or aggressive (1). After that, we put the trained model through its paces on the UNSW-NB15 testing dataset that has been processed. Model performance is assessed in terms of accuracy. No model or classifier hyperparameters are changed for the procedure. We used trained classifiers to identify testing set network traffic trends. We need interpretable graphs, feature importance charts, and visualizations to provide categorization and predictions. To better understand how ML classifiers work, the following Python packages can be used:

1) ELI5 is a library for visualizing operations that aids in debugging machine learning models and elucidating their predictions.
2) LIME (Local Interpretable Model-Agnostic Explanations) facilitates machine learning algorithm prediction explanations.
3) SHAP is a game-theoretic technique used to explain machine learning model output. SHAP helps in understanding how characteristics affect model output.

8. RESULTS & DISCUSSION

8.1 Assessment Using a Decision Tree

A classification model using decision trees (for both normal and attack behavior) was built using the training set and the decision tree classifier. Figure 4 shows that the model's accuracy on the testing set was 89.4%. The dataset has a slight class imbalance, with Class 1 having more samples (45,332) than Class 0 (37,000). The model achieves an accuracy of 0.8866 (88.66%), which means that it correctly predicts the class for approximately 88.66% of the samples. Table I contains a final classification report for a decision tree classifier and logistic regression (RegLog) model.

TABLE I. COMPARISON OF CLASSIFICATION FOR A DECISION TREE CLASSIFIER AND LOGISTIC REGRESSION (REGLOG) MODEL.

Metric	**Class 0**	**Class 1**	**Macro Average**	**Weighted Average**
Precision	0.90	0.88	0.89	0.89
Recall	0.85	0.92	0.88	0.89
F1-Score	0.87	0.90	0.88	0.89
Dataset Samples	37,000	45,332		
Average Support	82,332	82,332		
Accuracy	**88.66%**	**88.66%**		

Using ELI5's permutation importance toolkit, we plotted the feature importance for the top 10 features, as shown in Table II. The highlight significance is determined by the reduction in hub contamination multiplied by the likelihood of reaching that hub. In the generated tree-like impressions, the primary features are the front and center, and the decision tree feature importance (ELI5) permutation importance, as shown in Table III and Fig. 3. Key observations can be described as follows**:**

1. Most Important Feature:
   - ct_dst_src_ltm has the highest relative importance (**0.30**), meaning that it contributes the most to the model's predictions.
2. Least Important Feature:
   - ct_src_dport_ltm has the lowest relative importance (**0.02**), meaning that it contributes the least to the model's predictions.
3. Moderately important features:
   - Features such as tcprtt (0.25), styles (0.20), and dbytes (0.18) also play significant roles in the model's decision-making process.

TABLE II. ELI5 DECISION TREE TOP FEATURES.

Feature	**Weigth**		**Feature**	**Weigth**
Sttl	0.2145 ± 0.0020		Dloss	0 ± 0.0000
Sbytes	0.1289 ± 0.0011		Stcpb	0 ± 0.0000
ct_dst_src_ltm	0.1155 ± 0.0019		Swin	0 ± 0.0000
Sloss	0.0720 ± 0.0020		Dpkts	0 ± 0.0000
Smean	0.0358 ± 0.0016		Sload	0 ± 0.0000
ct_dst_sport_ltm	0.0076 ± 0.0006		Dload	0 ± 0.0000
Sinpkt	0.0008 ± 0.0001		Dttl	0 ± 0.0000
ct_src_dport_ltm	0.0008 ± 0.0002		Djit	0 ± 0.0000
Sjit	0 ± 0.0000		is_sm_ips_ports	0 ± 0.0000
Dinpkt	0 ± 0.0000		***… 19 more …***	
Rate	0 ± 0.0000			

TABLE III. THE DECISION TREE FEATURE IMPORTANCE ELI5 PERMUTATION IMPORTANCE MODEL.

Feature	Relative Importance	Feature	Relative Importance
ct_dst_src_ltm	0.3	Rate	0.12
Tcprtt	0.25	ct_srv_src	0.1
Styles	0.2	Smean	0.08
Dbytes	0.18	Sinpkt	0.06
ct_srv_dst	0.15	ct_dst_ltm	0.05

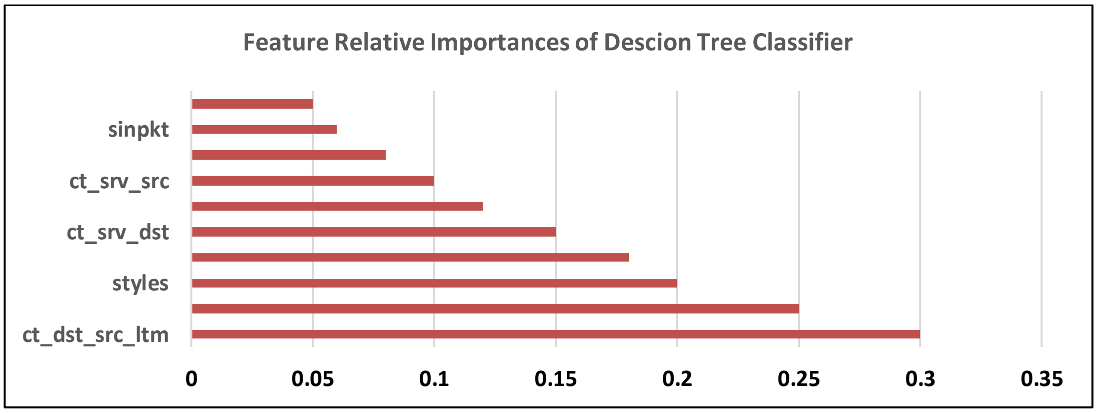

Fig. 3. Feature relative importance of the decision tree classifier

The feature 'sttl', which stands for (source-to-destination time-to-live value) in network traffic analysis, was found to be the most important feature for classification prediction, according to both feature importance outputs. By examining the feature and splitting value for each condition and each decision level, the decision tree visualizations allow the model to explain itself. A network traffic sample is directed to the left branch or node if it meets the condition; otherwise, it is directed to the right branch. Furthermore, the classification prediction result is shown in each class line according to the maximum depth of the tree that was selected. Machine learning (ML) botnets that use decision trees to classify IoT network traffic demonstrate strong threat detection capabilities. Additionally, human analysts can gain a better grasp of the model with the aid of the DT algorithm's explainability features, which are based on decision tree plotting. The cybersecurity landscape surrounding IoT networks can then be better understood. As part of this comprehension, you can compare expectations or speculate on the Botnet machine's feature-based learning. To further assist the machine learning process, human analysts can use their domain knowledge to add features or engineer features. This will be an enormous boon for analysts trying to determine how to improve the model decision framework and whether it is correct.

8.2 M3L Classifier

The M3L classifier was trained and tested on matched datasets. Before feature selection, the model had 89.83% accuracy throughout the testing set. Excellent score for predicting regular or malicious IoT traffic. The local interpretability model-agnostic explanations (LIME) package may build an M3L classifier model prediction visual for each training set prediction. The original data characteristics and forecasts are disturbed to input into LIME's internal classification model. Then, it monitors outputs. The library then prioritizes additional data outputs by distance from the initial location. The database is used to construct a surrogate linear regression model with sample weights to adjust for variances. Finally, the newly trained explanation model can explain the original data points. The Lime Tabular Explainer output in Fig. 4 displays the top 5 features and ELI5 permutation importance.

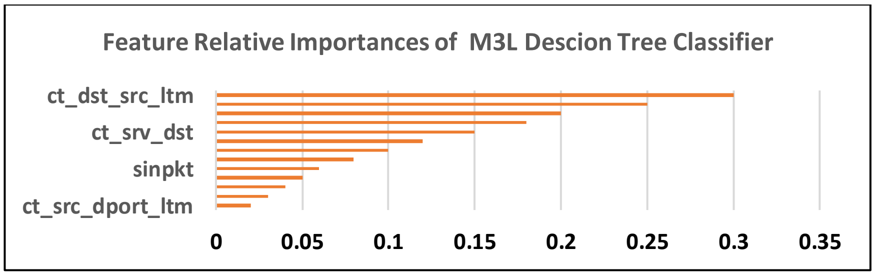

Fig. 4. M3L classifier single prediction.

The visual dashboard shows the features that were used to classify the overall behavior of that network traffic record as normal, along with their weights. The inspection confirms that this categorization is accurate since the actual category is 'Normal'. Individually, the visual dashboard's predicted classifications are very explainable. Leveraging the high-performance advantages of an M3L classifier essentially "black boxes", this tool provides a more transparent capability of predictions, which can be utilized for future cybersecurity research. Fold 1, Fold 2, etc., refer to the results from different iterations of cross-validation. Accuracy results after Eli5 optimized weights:

TABLE IV. ACCURACY RESULTS AFTER ELI5 OPTIMIZED WEIGHTS.

Metric	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Mean
Fit Time	0.8511798	0.9096644	0.89364886	0.8793206	0.8617952	0.8791218
Score Time	0.0781703	0.0625029	0.06248999	0.0781148	0.0624924	0.0687541
Accuracy	0.9734346	0.9766199	0.96503948	0.9623232	0.9701011	0.973862
Precision (Micro)	0.9720046	0.9766167	0.96833948	0.9689898	0.8744322	0.9738619
Recall (Micro)	0.9744446	0.9766576	0.97111111	0.9712134	0.9532616	0.973862
F1 (Micro)	0.9625046	0.9723232	0.96523457	0.9577711	0.9548777	0.973862
ROC AUC (OVR)	0.9945932	0.9954326	0.99480833	0.9940172	0.9952749	0.994825

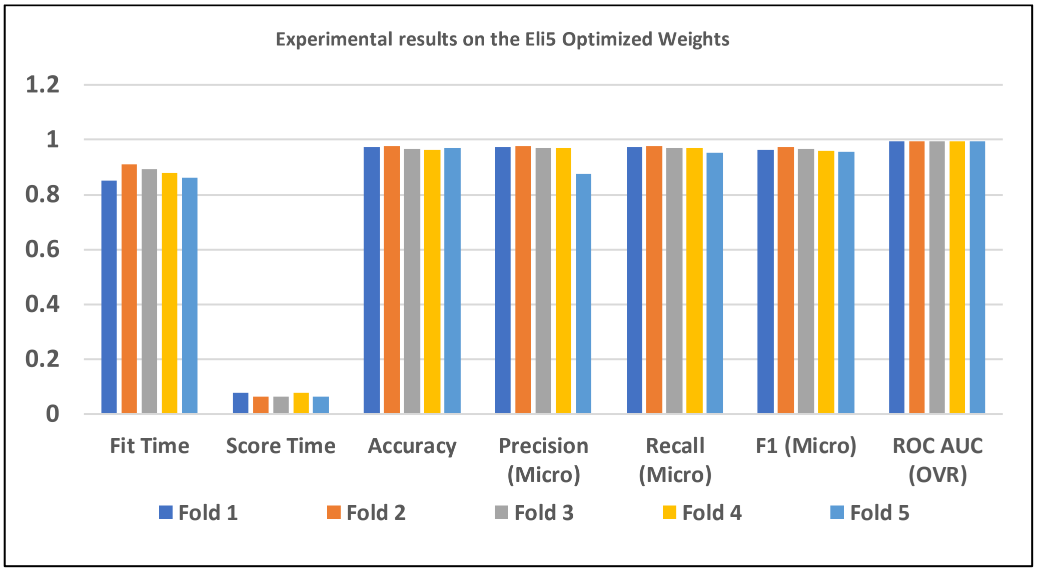

Fig. 5. Experimental results on the Eli5 optimized weights.

8.3 Random forest classifier

Both the training and testing runs of the random forest (RF) classifier were conducted on the matching datasets. Before feature selection, the model attained an accuracy of 89.61% with respect to the entire testing set. In terms of determining the legitimacy of IoT traffic, this is a very encouraging sign. To visualize the RF classifier's model predictions, a local interpretability model–agnetics explanations (LIME) library can be used for every prediction in the training set. Disrupting the initial data features and prediction is the first step in feeding LIME's internal classification model. After that, it observes the results. When deciding how significant the new data outputs are, the library takes the distance from the initial point into account. Next, a surrogate linear regression model is fitted using the database, taking into consideration the sample weights to account for variations. The last step is to apply the freshly trained explanation model to the initial data points described in Table V to bring them to light. The results of the Lime Tabular Explainer are displayed in Fig. 6. The accuracy increased after Eli5 selected features to 95.54%.

TABLE V. RF FRESHLY TRAINED EXPLANATION MODEL INITIAL DATA POINT FEATURES.

Feature	Relative Importance	Feature	Relative Importance
ct_srv_dst	0.026	Sbytes	0.05
Synack	0.033	Sload	0.059
Sinpkt	0.034	Smean	0.067
Ct_state_ttl	0.035	ct_dst_sport_ltm	0.077
Dur	0.037	ct_dst_src_ltm	0.11
Dinpkt	0.039	Sttl	0.125

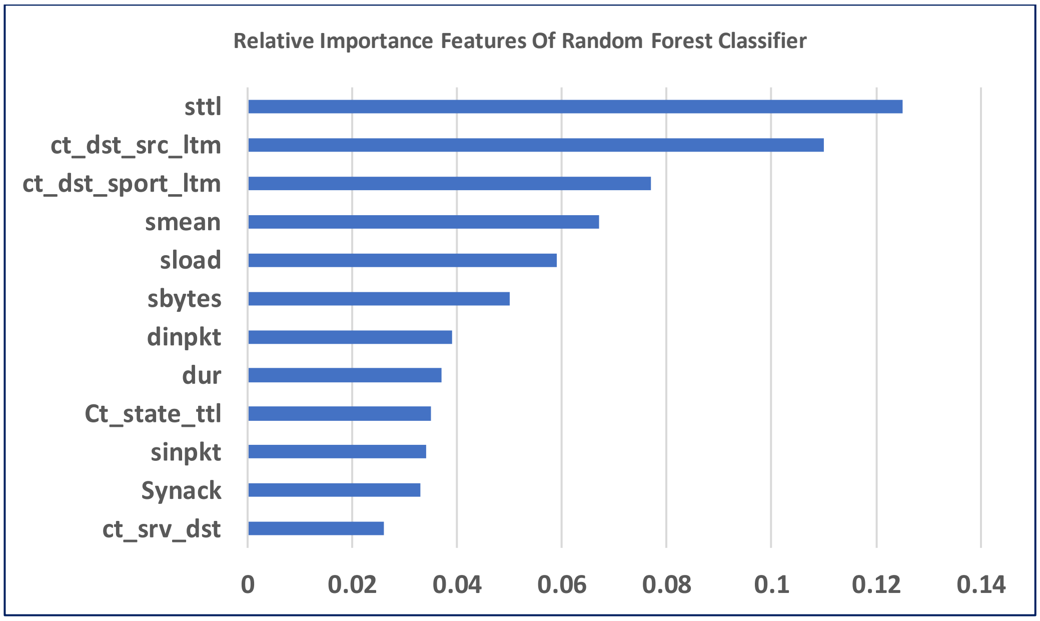

Fig. 6. Relative importance features of the random forest classifier.

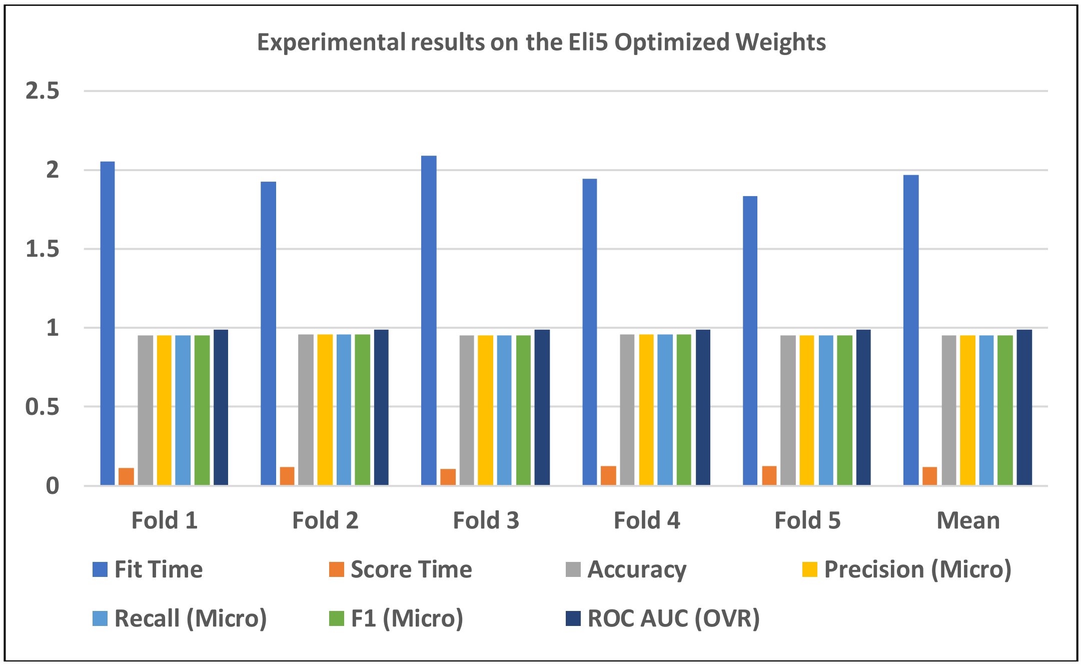

Fig. 7. Experimental results on the Eli5 optimized weights in the RF classifier.

Fig. 7 and Table VI contain a final classification for a random forest classifier model with the following key observations:

1. Mean Test Accuracy:
   - The average accuracy across all the folds is 0.95541261 (95.54%), indicating strong model performance.
2. Fit Time:
   - The time taken to fit the model ranges from 1.83 s to 2.09 s across the folds.
3. Score Time:
   - The time taken to score the model ranges from 0.11 s to 0.12 s across folds.
4. Consistency across Folds:
   - The model shows consistent performance across all folds, with accuracy, precision, recall, and F1-scores being very close in value.

TABLE VI. COMPARISON OF CLASSIFICATION RESULTS FOR THE RANDOM FOREST CLASSIFIER.

Metric	Class 0	Class 1	Macro Average	Weighted Average
Precision	0.76	0.99	0.88	0.92
Recall	0.98	0.86	0.92	0.90
F1-Score	0.86	0.92	0.88	0.89
Dataset Samples	175341	175341		
Average Support	56000	119341		
Accuracy	**89.61%**	**88.66%**		

8.4 XGBoost Classifier

After training on the classification task, the XGBoost classifier underwent testing on the testing set, similar to the other two classifiers. With an overall accuracy of 89.05%, the XGBoost classifier was highly capable of classifying network behavior. Compared with those of the M3L classifier, the results are quite close.
SHAP was used to include explainability features in this classifier. The key benefits of SHAP are consistency in tree-based model structures such as XGBoost and local explanations. The outputs of tree-based models can be better understood with the help of the SHAP values. The method relies on value calculations from game theory and offers a wealth of information about the significance of features by analysing their marginal contribution to the model's output.
The testing set classification predictions can be explained via the Tree SHAP implementation, which is integrated into XGBoost. Figure 11 depicts a multiprediction explanation via feature comparison or output classification values, whereas Figure 10 visualizes an explanation of a single prediction. In the network traffic record, f(x) values closer to 0 indicate normal activity, and values closer to 1 indicate attack behavior. For ['XGBoost Classifier', 'RegLog'], Table VII summarizes the performance metrics for the binary classes:

TABLE VII. XGBOOST PERFORMANCE METRICS FOR THE BINARY CLASSES WITH SHAP.

Class	Precision	Recall	F1-Score
XGBoost Classifier			
0	0.81	0.93	0.86
1	0.96	0.90	0.93
XGBoost Classifier with SHAP			
Macro avg	0.89	0.91	0.90
Weighted avg	0.91	0.91	0.91

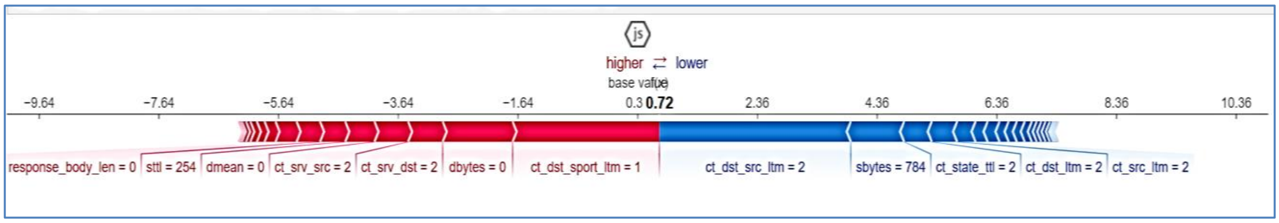

Fig. 8. Visualization of a single prediction via SHAP with XGBoost.

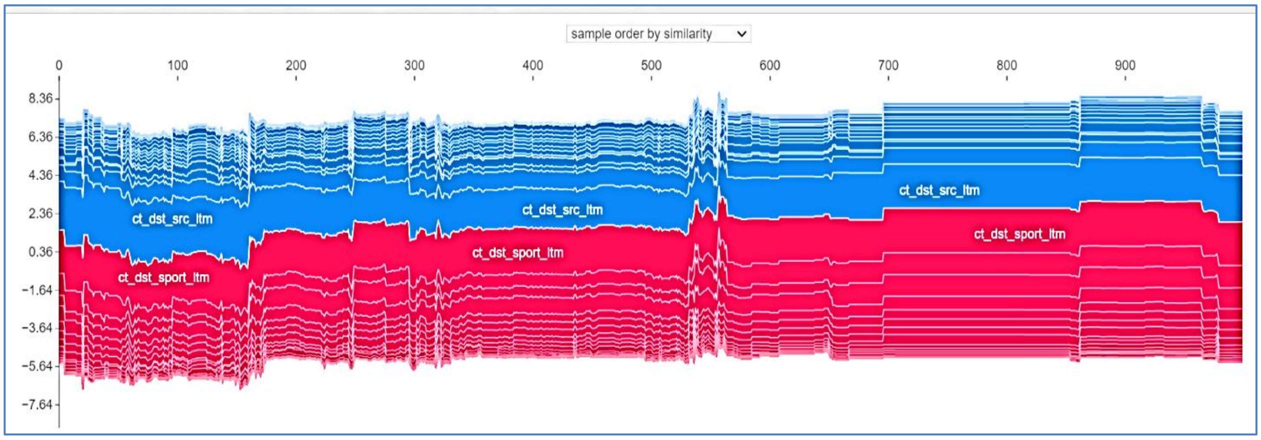

Fig. 9. Visualization of multiple predictions via SHAP with XGBoost.

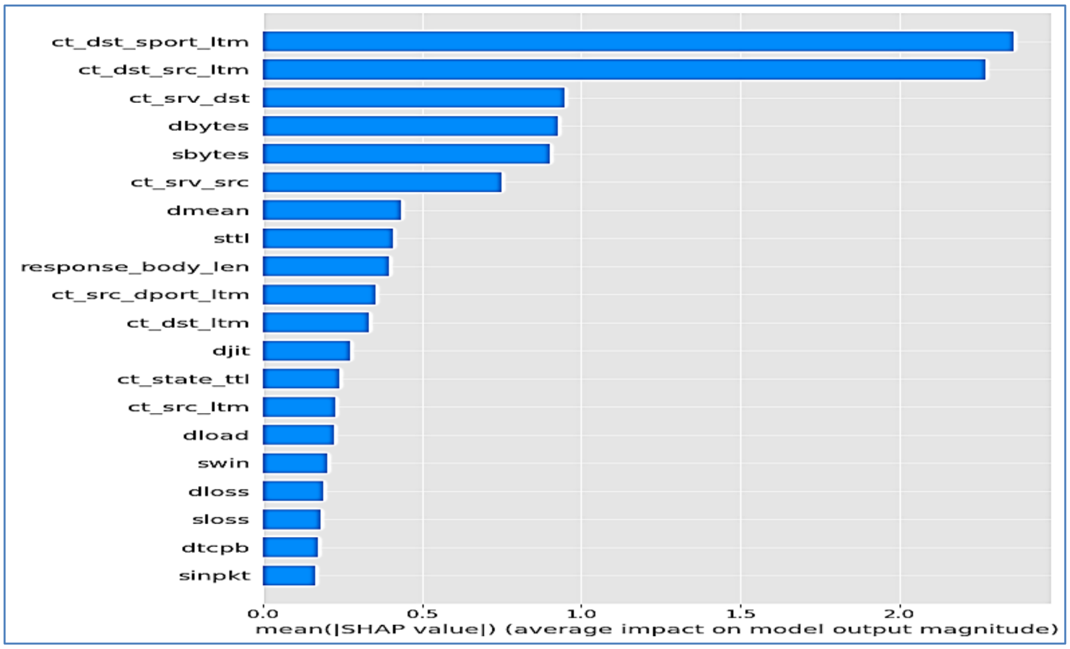

Fig. 10. SHAP feature importance analysis for the XGBoost classifier

Fig. 10 displays a feature importance plot generated by SHAP, which helps to ascertain the average significance of the input training features for classification prediction. The outcomes resemble those of the DT classifier. Visual indicators of the effects of element values on characterization predictions are shown in the SHAP outline plot, which also displays the most common mixes. Red indicates a higher element value in Fig. 11, whereas blue indicates a lower highlight value. Assault conduct is more likely when SHAP values are lower to the left, and higher expectations are associated with lower SHAP values when they are higher to the right (typical action).

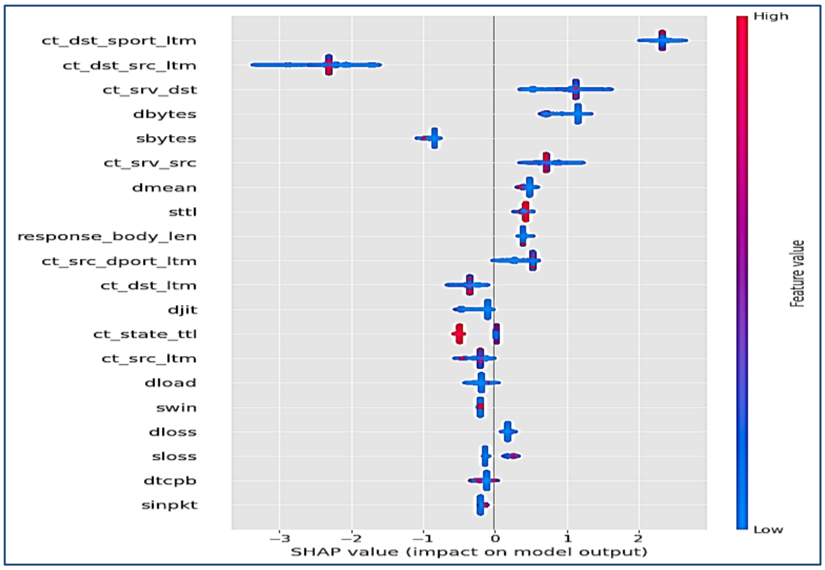

Fig. 11. XGBoost summarized via SHAP,

In addition, one can use SHAP values, and dependency charts show how one feature affects the whole dataset. Display the link between the feature value and SHAP for several samples and examine feature interactions. Fig. 12 shows the SHAP interaction value summary plot and the SHAP dependence plot for the 'sttl' feature.

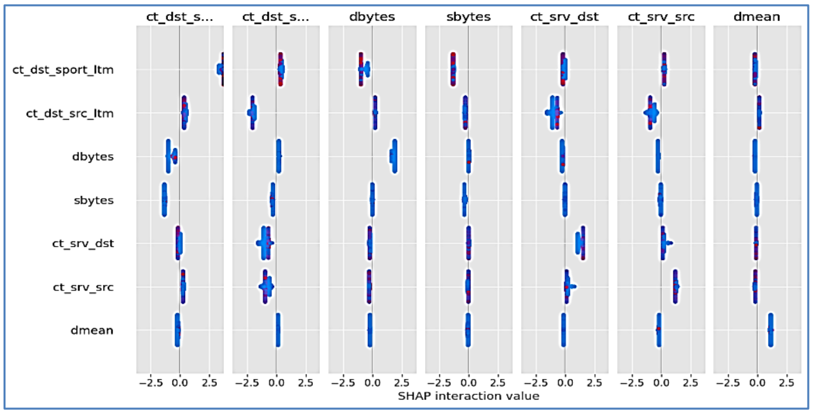

Fig. 12. SHAP interaction values.

Fig. 13 also shows how the XGBoost classifier's individual predictions can be explained via the same LIME package.

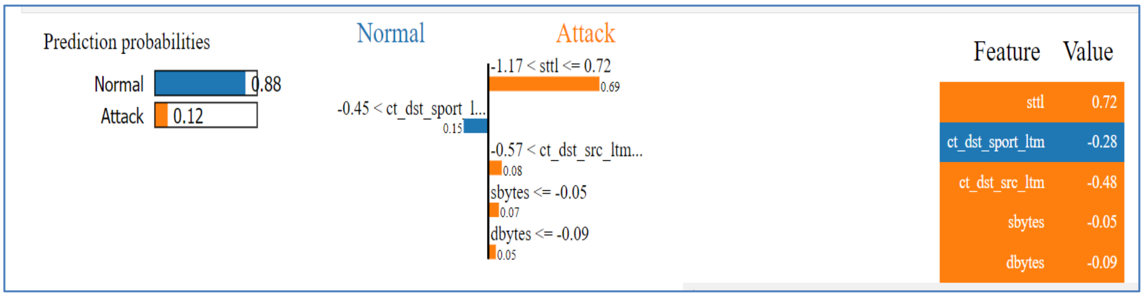

Fig. 13. XGBoost classifier's single-classification prediction and its explanation.

Table VIII summarizes the final results of the XGBoost classifier performance, with a mean accuracy of 90.60:

TABLE VIII. XGBOOST PERFORMANCE METRICS FOR THE BINARY CLASSES WITH SHAP.

Metric	**Fold 1**	**Fold 2**	**Fold 3**	**Fold 4**	**Fold 5**	**Mean Value**
Fit Time (s)	1.8621	1.7370	1.7384	1.6795	1.6607	-
Score Time (s)	0.1406	0.1100	0.1407	0.1096	0.1099	-
Accuracy	0.9064	0.9063	0.9065	0.9059	0.9049	**0.9060**
Precision (Micro)	0.9064	0.9063	0.9065	0.9059	0.9049	0.9060
Recall (Micro)	0.9064	0.9063	0.9065	0.9059	0.9049	0.9060
F1-Score (Micro)	0.9064	0.9063	0.9065	0.9059	0.9049	0.9060
ROC AUC (OvR)	0.9398	0.9393	0.9404	0.9384	0.9390	-

8.5 Performance of the Hybrid Proposed Classifier

A high-performing classifier is available in the proposed hybrid model, which is described in detail in Fig. 13 and is both efficient and flexible. When combined with the SHAP and LIME libraries, it offers robust explainability features. To evaluate ML-based Botnets effectively for IoT network security, the credibility of sophisticated black-box algorithms can be increased. Finally, Table IX displays the outcomes of the voting operation among the three hybrid ML classifiers:

TABLE IX. COMPARATIVE PERFORMANCE OF THE HYBRID PROPOSED CLASSIFIER

Algorithm	**Accuracy**	**XAI Accuracy**
Naïve Bayes	79.00%	88.00%
Decision Tree DT	88.60%	89.80%
Random Forest	89.50%	95.50%
Support Vector Machine	88.71%	95.38%
M3L	89.80%	97.30%
XGBoost	89.89%	90.09%
DT+M3L	89.83%	94.22%
Voting (*XGBoost+DT+M3L*)	98.97%	98.97%

The combination of the XGBoost, DT, RF, and M3L algorithms outperformed traditional and state-of-the-art machine learning algorithms in terms of accuracy (98.97%), minimum systematic error (0.003), and voting results. DT and M3L also achieved the second-highest ratios. The results show that the methods were successful in detecting botnet activities and reducing the associated security risks.

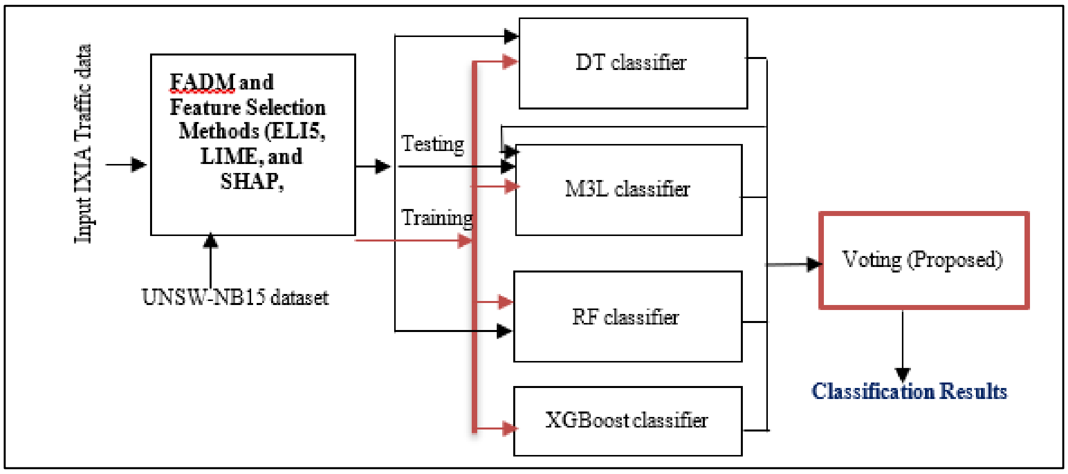

Fig. 14. The general structure of the proposed NIDS.

9. COMPARATIVE STUDY WITH RECOMMENDATIONS

In our study, we present a comparative analysis and recommendations, formatted as a table, juxtaposed with findings from references [18,20], which assess the performance of various feature extraction techniques using the NSL-KDD and CICIDS2017 datasets. These results are detailed in our manuscript and can be reviewed in Table X. The comparison encompasses two models: the first model employs a decision tree (DT) classifier integrated with the Eli5 library, whereas the second model utilizes a random forest (RF) classifier, also enhanced with Eli5, to optimize feature weights. The proposed model outperforms the baseline across multiple metrics. It trains 55% faster, scores 42% quicker, and achieves a 1.9% improvement in accuracy, precision, recall, and F1 score. Additionally, it demonstrates superior discrimination with a 0.55% higher ROC AUC, making it a more efficient and effective solution, as described in Table X.

TABLE X. COMPARATIVE CHECKLIST: FIRST MODEL VS. SECOND MODEL

Metric	**First Model (Mean)**	**Second Model (Mean)**	**Comparison**
Fit Time (seconds)	0.8791218	1.969118072	First Model is **faster** (takes ~55% less time to train than Second Model).
Score Time (seconds)	0.0687541	0.119285678	First Model is **faster** (scores ~42% quicker than Second Model).
Accuracy	0.973862	0.955412612	First Model is **more accurate** (1.9% higher than Second Model).
Precision (Micro)	0.9738619	0.955412612	First Model has **higher precision** (1.9% better than Second Model).
Recall (Micro)	0.973862	0.955412612	First Model has **higher recall** (1.9% better than Second Model).
F1 (Micro)	0.973862	0.955412612	First Model has **higher F1 score** (1.9% better than Second Model).
ROC AUC (OVR)	0.994825	0.989286902	First Model has **better ROC AUC** (0.55% higher, indicating better discrimination).

The key observations can be discussed as follows:

1. Training efficiency:
   - The first model is trained significantly faster (0.879 vs. 1.969 s) and scores faster (0.069 vs. 0.119 s), making it more computationally efficient.
2. Performance Metrics:
   - The first model consistently outperforms the second model across all the classification metrics (accuracy, precision, recall, F1), with an ~1.9% improvement in each metric.
3. Discriminative power:
   - The first model's ROC AUC (0.9948) is greater than that of the second model (0.9893), suggesting a better overall ability to distinguish between classes.
4. Consistency:
   - The first model shows slightly more variation across folds (e.g., the accuracy ranges from 0.9623--0.9766) than does the second model (0.9528--0.9572), but its mean performance is superior.

The recommendations of the proposed system are as follows:

1. The first model is chosen if the priority is higher accuracy, better classification performance, and faster training/scoring times.
2. The second model is chosen if the computational resources are less constrained and a slightly more stable (less variable) performance across folds is preferred, although it sacrifices ~1.9% accuracy.

10. CONCLUSION

As the complexity of ML models used by IoT traffic security IDSs continues to rise, human analysts will be essential in analysing results using their innate domain knowledge to allocate resources and develop cybersecurity strategies. Many people use ML algorithms without understanding the reasoning or logic that goes into making predictions. Using decision tree classifier training, random forest, the M3L classifier, and XGBoost on the UNSW-NB15 dataset, we achieve high-performance accuracy in examining attack or typical client behavior in an IoT network. The trained classifiers were then subjected to explainable AI (XAI) evaluations and decision-making by using pre-existing libraries and methods for ML classifier performance analysis. This transparency will immediately demonstrate that ML systems in the IoT cybersecurity domain are more reliable. By making it easier to understand how various factors affect cyber-attack prediction, increased explainability will pave the way for a plethora of new capabilities in IoT cybersecurity, owing to the insights gleaned from advanced machine learning models. We evaluated decision trees, random forests, M3L, and XGBoost on the UNSW-NB15 dataset and achieved high accuracy in distinguishing attacks from normal IoT traffic. XGBoost performed best, whereas the random forest algorithm achieved strong recall and precision. Explainable AI (XAI) techniques, such as SHAP values, identify key features influencing attack predictions, with packet-based statistical features being the most critical. This transparency enhances trust in ML-driven IDSs, allowing analysts to refine cybersecurity strategies. Our findings suggest that combining high-performing models with interpretable techniques can improve both detection accuracy and real-world adoption in IoT security frameworks.

Conflicts of interest

The authors declare that they have no conflicts of interest.

Funding

There is no funding for the paper.

Acknowledgement

The authors would like to thank Mustansiriyah University for their support in conducting the work published in this paper.

References

[1] Y. Meidan, M. Bohadana, Y. Mathov, Y. Mirsky, D. Breitenbacher, A. Shabtai, and Y. Elovici, "N-BaIoT: Network-based detection of IoT botnet attacks using deep autoencoders", *IEEE Pervasive Computing*, vol. 17, no. 3, pp. 12–22, Jul.-Sep. 2018.

[2] S. Garcia, M. Grill, H. Stiborek, and A. Zunino, "An empirical comparison of botnet detection methods", *Computers &. Security*, vol. 45, pp. 100–123, Sep. 2014.

[3] W. Ren, X. Tong, J. Du et al., "Privacy-preserving using homomorphic encryption in mobile IoT systems", *Computer Communications*, vol. 165, pp. 105–111, Jan. 2021.

[4] A. A. Abd El-Latif, B. Abd-El-Atty, S. E. Venegas-Andraca et al., "Providing end-to-end security using quantum walks in IoT networks", *IEEE Access*, vol. 8, pp. 92687–92696, 2020.

[5] M. M. Ogonji, G. Okeyo, and J. M. Wafula, "A survey on privacy and security of Internet of Things" *Computer Science Review*, vol. 38, Art. no. 100312, Nov. 2020.

[6] G. De La Torre Parra, P. Rad, K.-K. R. Choo, and N. Beebe, "Detecting Internet of Things attacks using distributed deep learning", *Journal of Network and Computer Applications*, vol. 163, Aug. 2020.

[7] I. Ali, A. I. A. Ahmed, A. Almogren et al., "Systematic literature review on IoT based botnet attack", *IEEE Access*, vol. 8, pp. 212220–212232, 2020.

[8] A. Marzano, D. Alexander, O. Fonseca et al., "The Evolution of Bashlite and Mirai IoT Botnets", *IEEE Symposium on Computers and Communications (ISCC),* Natal, Brazil, pp. 813–818, Jun. 2018.

[9] R. Kour, "Cybersecurity issues and challenges in Industry 4.0", *Applications and Challenges of Maintenance and Safety Engineering in Industry 4.0*, Hershey, PA, USA: IGI Global, pp. 84–101, 2020.

[10] J. Prinsloo, S. Sinha, and B. von Solms, "A review of Industry 4.0 manufacturing process security risks", *Applied Sciences*, vol. 9, no. 23, Dec. 2019.

[11] S. Dange and M. Chatterjee, "IoT botnet: The largest threat to the IoT network", *Data Communication and Networks: Advances in Intelligent Systems and Computing*, Springer, Singapore, vol. 1049, pp. 137–157, 2020.

[12] L. Gupta, T. Salman, A. Ghubaish, D. Unal, A. K. Al-Ali, and R. Jain, "Cybersecurity of multi-cloud healthcare systems: A hierarchical deep learning approach", *Applied. Soft Computing*, vol. 118, Mar. 2022.

[13] F. Hussain, R. Hussain, S. A. Hassan, and E. Hossain, "Machine learning in IoT security: Current solutions and future challenges", *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1686–1721, 3rd Quart. 2020.

[14] D. Krishnan and P. Babu, "Imbalanced classification for botnet detection in Internet of Things", *Next Generation of Internet of Things: Lecture Notes in Networks and Systems*, Springer, vol. 201, pp. 595–605, 2021.

[15] S. Bagui and K. Li, "Resampling imbalanced data for network intrusion detection datasets", *Journal of Big Data*, vol. 8, no. 1, Jan. 2021.

[16] T. N. Ghorsad and A. V. Zade, "Optimal feature picking for intrusion detection on the basis of explainable artificial intelligence", *Advances in Nonlinear Variational Inequalities*, vol. 12, no. 11, pp. 1377–1385, 2024.

[17] B. Hariharan, S. V. N. Vishwanathan, and M. Varma, "Efficient max-margin multilabel classification with applications to zero-shot learning", *Machine Learning*, vol. 88, no. 1-2, pp. 127–155, Jul. 2012.

[18] W. K. Mohammed, M. A. Taha, and S. M. Mohammed, "A novel hybrid fusion model for intrusion detection systems using benchmark checklist comparisons", *Mesopotamian Journal of CyberSecurity*, vol. 4, no. 3, pp. 216–232, Dec. 2024.

[19] A. M. Mahmood and İ. Avcı, "Cybersecurity defence mechanism against DDoS attack with explainability", *Mesopotamian Journal of CyberSecurity*, vol. 4, no. 3, pp. 278–290, Dec. 2024.

[20] T. Ali Abdalkareem, K. A. Zidan, and A. S. Albahri, "A systematic review of adversarial machine learning and deep learning applications", *Al-Iraqia Journal for Scientific Engineering Research*, vol. 3, no. 4, pp. 14–40, Dec. 2024.

[21] B. M. Nema and S. J. Mohammed, "Secure location privacy transmitting information on cellular networks", *Iraqi Journal of Science*, vol. 63, no. 11, pp. 5004–5014, Nov. 2022.