Nanotechnology Perceptions
ISSN 1660-6795
www.nano-ntp.com

Machine Learning Algorithms for Detection and Classification IoT Network Intrusion

V. Sree Ranganayaki[1], A. Ramesh Babu[2]

[1]Research Scholar (Ph.D), Department of Computer Science, Chaitanya Deemed to be University, India
[2]Department of Computer Science, Chaitanya Deemed to be University, India
Email: sreeranganayaki5@gmail.com

The exponential growth of Internet of Things (IoT) devices has led to an increase in cyber threats targeting these interconnected networks. Effective intrusion detection and classification systems are critical to safeguard IoT environments against potential attacks. In this study, we explore the application of machine learning algorithms for detecting and classifying IoT network intrusions using the widely-used UNSW-NB15 dataset, specifically designed for network intrusion detection. We investigate several machine learning models, including Decision Trees, Random Forest, Support Vector Machines (SVM), Neural Networks, and Ensemble Techniques. Through extensive experiments, we evaluate the performance of each algorithm in terms of accuracy, precision, recall, and F1-score. Our results demonstrate the effectiveness of machine learning-based approaches in accurately identifying diverse intrusion types within IoT networks, showcasing the potential of these algorithms to enhance the security posture of IoT ecosystems. The insights gained from this research contribute to the advancement of intelligent and proactive intrusion detection systems, enabling a safer and more resilient IoT landscape in the face of evolving cyber threats.
Keywords: Decision Trees, Support Vector Machines, Random Forest, UNSW-NB15 dataset.

1. Introduction

The Internet of Things (IoT) has witnessed explosive growth, revolutionizing industries and transforming everyday life with its vast network of interconnected devices. However, this proliferation of IoT devices has also opened new avenues for cyber threats and attacks, making the security of IoT networks a paramount concern. Intrusion detection and classification play a crucial role in identifying and mitigating potential threats to ensure the integrity and privacy

of IoT ecosystems. Traditional rulebased and signature-based intrusion detection systems often struggle to keep up with the dynamic and sophisticated nature of modern cyber attacks. As a result, the application of machine learning algorithms in IoT network intrusion detection and classification has gained significant attention in recent years. Machine learning algorithms offer the capability to automatically learn patterns and relationships from vast amounts of data, empowering intrusion detection systems to adapt and evolve as new threats emerge. By analyzing network traffic data, these algorithms can detect anomalies and classify them into different attack categories, providing real-time protection against malicious activities. In this research, we aim to explore and evaluate various machine learning algorithms for IoT network intrusion detection and classification. The objective is to identify the most effective models that can accurately detect and classify different types of intrusions within IoT environments. Leveraging the widely-used UNSW-NB15 dataset (Moustafa & Slay, 2016), a comprehensive network intrusion detection dataset, we will investigate the performance of popular machine learning techniques such as Decision Trees, Random Forest, Support Vector Machines (SVM), Neural Networks, and Ensemble Techniques. Through an empirical evaluation of these algorithms on the UNSW-NB15 dataset, we seek to identify the strengths and weaknesses of each approach. Performance metrics such as accuracy, precision, recall, and F1-score will be used to assess the models' effectiveness in detecting and classifying network intrusions accurately. The findings from this research will shed light on the potential of machine learning-based approaches to enhance the security and resilience of IoT networks against cyber threats.

The rest of this paper is organized as follows: Section 2 presents an overview of related work on IoT network security and intrusion detection using machine learning techniques. Section 3 outlines the methodology, including dataset description, data preprocessing, and the machine learning algorithms under investigation. Section 4 presents the experimental setup and performance metrics used in evaluating the algorithms. Section 5 presents the results and discusses the performance of each algorithm in detail. Finally, Section 6 concludes the research and highlights the implications of our findings for future developments in IoT network security.

2. Literature Review

This comprehensive survey provides an overview of IoT technologies and applications. It highlights the importance of security in IoT systems and lays the foundation for the need to employ machine learning algorithms for intrusion detection and classification [1]. This paper presents a comprehensive review of various machine learning algorithms used for IoT security, including intrusion detection. It assesses the strengths and weaknesses of different algorithms to aid researchers in selecting appropriate models for IoT network intrusion detection and classification [2]. This review focuses on IoT security specifically in the context of 802.15.4-based networks. It discusses the unique challenges in intrusion detection for such networks and provides insights into potential machine learning-based solutions [3]. This research introduces a new intrusion detection dataset, UNSW-NB15, and characterizes intrusion traffic. It provides researchers with a benchmark dataset for evaluating machine learning algorithms for IoT network intrusion detection and classification [4]. This paper introduces another widely used dataset, UNSW-NB15, for network intrusion detection systems. It offers valuable

insights into the characteristics of network intrusions and serves as a benchmark for evaluating machine learning algorithms [5]. This survey paper reviews various machine learning techniques for IoT network intrusion detection. It provides an in-depth analysis of different algorithms and their applicability in securing IoT networks [6]. Although focused on phishing detection, this survey paper discusses the application of machine learning techniques in cybersecurity. It offers insights into the potential transferability of these techniques to IoT network intrusion detection and classification [7]. This survey paper explores various machine learning approaches for IoT security, emphasizing intrusion detection. It discusses the challenges and opportunities in securing IoT networks and the role of machine learning algorithms [8].

3. Overview & Benefits of Machine Learning

Advanced and state of the art ML algorithms and models offer valuable applications in establishing better IoT network security. The ML techniques, learn from input features generated in network traffic, and offer support to cybersecurity personnel in making critical threat detection decisions. However, these techniques are based on advanced models that are too complex to be interpreted by human analysts; hence, may they turn to traditional tools that may not be as viable but offer more explainability or inherent trust by the human involved. In many cases, it is nearly impossible to get a feeling for its inner workings of a ML system for Intrusion Detection. This may further decrease trust that a certain prediction from the model is correct even though performance results may indicate otherwise. Having an intuitive explanation of the rationale behind individual predictions or model decision-making framework will better position cybersecurity experts to trust prediction or the classifiers itself, especially, in understanding how it behaves in particular cases. Explainable AI (XAI) offers a variety of explanation or feature importance tools for generating explanations about the knowledge captured by trained ML models to aid in increasing overall trust.

4. Methodology

4.1 UNSW-NB15 Dataset UNSW-NB15 is an IoT-based network traffic record with different categories of normal activity and malicious botnet attack behavior (Fuzzer, Analysis, Backdoor, DoS, Exploit, Generic, Reconnaissance, Shellcode, Worm by classifying attack types such as Raw network packets of the UNSW-NB 15 dataset were created using his IXIA Perfect Storm tool from the Australian Cyber Security Center (ACCS) Cyber Range Lab and synthesized with realworld normal activity and contemporary attacks I captured a combination of movements. IoT Base generates the network. Figure 1 shows how the configuration records and functions of the UNSW-NB15 testbed were created.

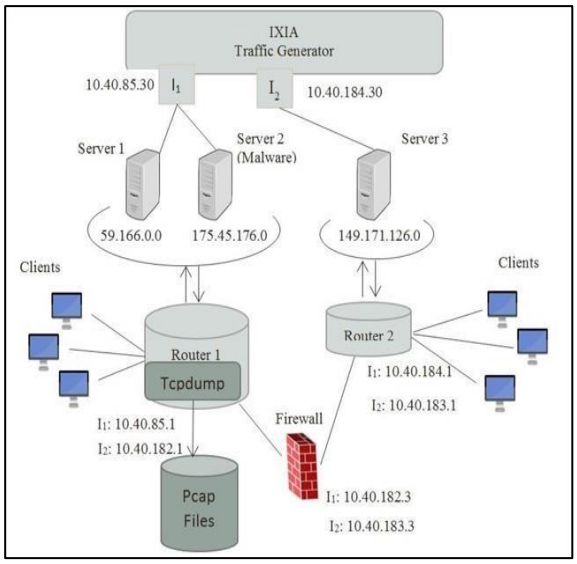

Figure 1: IXIA Traffic Generator Overview

UNSW-NB15 is pre-split by the developer so that it can be configured into a training set for model training and a testing set for model performance, namely UNSW_NB15_trainingset.csv and UNSW_NB15_testing-set.csv respectively. The number of records in the training set is 175,341 records, and the test set is 82,332 records, containing traffic behavior target responses for each record, attack, and normal behavior. The dataset consists of 39 features that are numeric in nature. Features and their descriptions are listed in the UNSWNB15_features.csv file. To complement the experimental process, the target trait will be a binary classification of normal and aggressive behavior. Figure 2 shows the details and score distribution for each attack class within the data subset. 0 represents normal and 1 represents aggressive behavior. We can see that the dataset for the activity behavior binary response variables is well balanced.

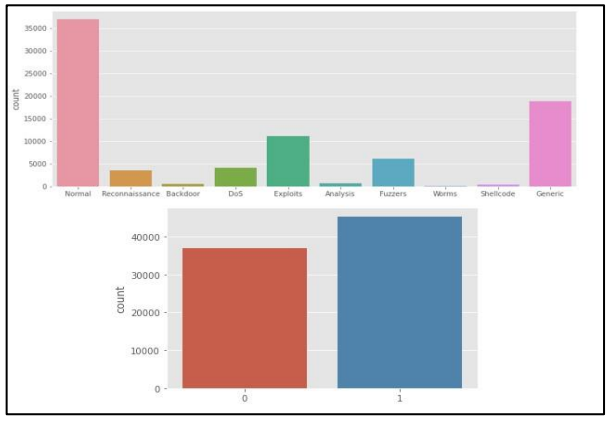

Figure 2: Training Dataset Distribution and Counts

4.2 Overview of ML Methods

The three supervised ML approaches that will be used develop binary classification classifiers are Decision Trees, Neural Network based on Multilayer Perceptron, and XGBoost. The ML algorithms in the mentioned order offer decreasing capabilities for explainability (XAI).

Decision Tree Classifier

The decision tree (DT) classifier is a supervised ML algorithm that will be utilized for the classification task of Normal or Attack behavior based on the 39-input feature. The resulting DT algorithm develops a decision-making process based on a tree-like model with nodes or branches. The max depth of the decision tree can be defined beforehand. A decision tree is already an explainable machine learning algorithm through visualizations of the resulting trees.

Multi-layer Perceptron Classifier

The Multi-layer Perceptron (MLP) is a type of artificial neural network that belongs to the family of feed forward neural networks. It is one of the fundamental architectures used in machine learning and has been successfully applied in various applications, including classification, regression, and pattern recognition. The input layer is responsible for receiving the features or data points of the input samples. Each node (neuron) in the input layer corresponds to a specific feature in the input data. The hidden layers are intermediate layers between the input and output layers. Each hidden layer contains multiple neurons that process the input data using weighted connections and activation functions. The number of hidden layers and the number of neurons in each layer can be adjusted based on the complexity of the problem. The output layer produces the final predictions or classifications based on the processed information from the hidden layers. The number of neurons in the output layer is determined by the number of classes in the classification problem.

During the forward propagation phase, the input data is fed into the input layer. The input values are multiplied by weights and passed through activation functions in the hidden layers to compute intermediate outputs. This process is repeated through the hidden layers until the final output layer produces the predicted values or classifications. Activation functions introduce non-linearity to the model, allowing it to learn complex relationships in the data. Common activation functions include ReLU (Rectified Linear Unit), Sigmoid, Tanh (Hyperbolic Tangent), and Softmax (for multiclass classification). The MLP's learning process involves updating the weights to minimize the difference between the predicted outputs and the true labels. Backpropagation is used to calculate the gradients of the loss function with respect to the model's weights. These gradients are then used to update the weights through optimization algorithms.

Neural networks like MLP Classifiers for the most part, lack sufficient model explainability and interpretability. In the tradeoff between the explainability/interpretability of an algorithm and its accuracy in application, neural networks heavily lean more toward the prediction performance. Neural networks contain visible layers and hidden layers of neural units, which hidden layers and its unknown interaction post training significantly causes neural networks to act as “black-box” algorithms instead.

XGBoost Classifier

XGBoost is a gradient boosted decision tree implementation designed for speed and performance. XGBoost stands for "eXtreme Gradient Boosting". XGBoost is provided as an open-source software library with algorithm implementations designed for efficiency in computation time and memory resources. A design architecture allows you to optimally use available resources to train your model. The XGBoost library implements a decision tree algorithm for gradient boosting. Boosting is an ensemble technique that adds new models 'to correct errors in existing models. Gradient boosting is the approach of creating a new model that predicts the residuals or errors of previous models and adding them together to get the final prediction. Moreover, the implemented gradient descent algorithm minimizes losses when adding new models. This approach supports classification predictive modeling of normal or aggressive behavior.

4.3 Proposed Approach with Scikit-learn, XGBoost, and XAI Libraries

UNSW-NB15 training dataset after applying data processing techniques for data cleaning, normalization, and transformation will be used to train each of the three supervised ML binary classifiers: Decision Trees, Neural Network based on Multi-layer Perceptron, and XGBoost. The target feature will be a binary classification of Normal (0) or Attack (1) behavior. Thereafter, the next process will be to test the trained model using the data processed UNSW-NB15 testing dataset. The model performance will be evaluated using the accuracy score. The procedure described above is will not be tuned using model or classifier hyperparameters. Scikit-Learn implementation of the Decision Trees Classifier and Multi-layer Perceptron Classifier will be utilized, while the XGBoost library will be utilized for the XGBoost Classifier. After classifiers are trained and tested, the next process is to develop interpretable diagrams, feature importance plots, and classification/prediction explanation visuals based on the trained classifiers used to detect network traffic behavior in the testing set. The following Python packages are and investigate to modify the ML classifiers for explainability:

- ELI5 is a visualization library that helps you debug machine learning models and explain the predictions they produce.
- LIME (Local Interpretable Model-Agnostic Exploitations) is a package for demonstrating predictions in machine learning algorithms.
- SHAP (Shapley Additive exPlanations) is a game-theoretic approach to explaining the output of machine learning models. SHAP helps us better understand the impact of features on model output.

5. Results & Discussion

Decision Tree Classifier

Using the Scikit-learn library's tree. Decision Tree Classifier(), the training set was used to build a Decision Tree classification model for Normal or Attack behavior. The model performance accuracy against the testing set was 85% as indicated in Figure 3.

Accuracy: 0.8510901614568184
Reporting for ['Decision Tree Classifer', 'RegLog']:

	precision	recall	f1-score	support
0	0.69	0.98	0.81	56000
1	0.99	0.79	0.88	119341
accuracy			0.85	175341
macro avg	0.84	0.88	0.84	175341
weighted avg	0.89	0.85	0.86	175341

Figure 3: Decision Tree Classifier Report

The feature importance for the top 10 features was graphed with both the scikit-learn library and ELI5's Permutation Importance toolkit. Feature importance is computed as the reduction in node contamination weighted by the probability of reaching that node. The most important properties are higher up in the tree structure -like visualization generated.

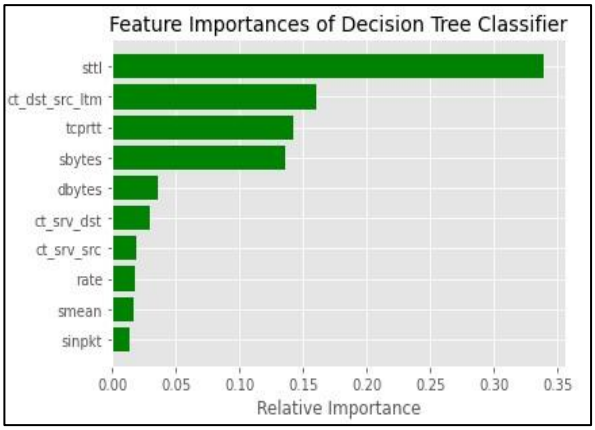

Figure 4: Decision Tree Feature Importance: Scikit Learn

Weight	Feature
0.2755 ± 0.0003	sttl
0.2411 ± 0.0007	ct_dst_sport_ltm
0.1359 ± 0.0014	sbytes
0.0707 ± 0.0012	ct_dst_src_ltm
0.0354 ± 0.0002	sloss
0.0288 ± 0.0009	smean
0.0108 ± 0.0005	dbytes
0.0011 ± 0.0001	sinpkt
0.0005 ± 0.0001	ct_dst_ltm
0 ± 0.0000	sjit
0 ± 0.0000	dinpkt
0 ± 0.0000	dpkts
0 ± 0.0000	dloss
0 ± 0.0000	stcpb
0 ± 0.0000	swin
0 ± 0.0000	sload
0 ± 0.0000	rate
0 ± 0.0000	dload
0 ± 0.0000	djit
0 ± 0.0000	is_sm_ips_ports
… 19 more …	

Figure 5: Decision Tree Feature Importance: ELI5 Permutation Importance

Both of the feature importance outputs indicate very similar results with feature ‘sttl’ or “source to destination time to live value” in the network traffic analysis being indicated as the most important to classification prediction. The most important features can be visualized in the upper layers of the decision tree visualization in Figures 6 through 8.

Figure 6: Decision Tree Classifier (Depth = 3 Nodes) Explainable AI Visualization

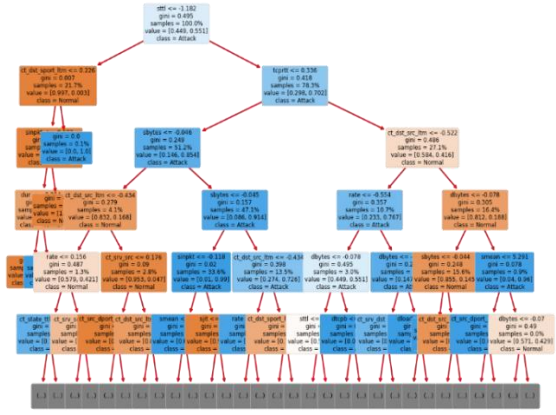

Figure 7: Decision Tree Classifier (Depth = 5 Nodes) Explainable AI Visualization

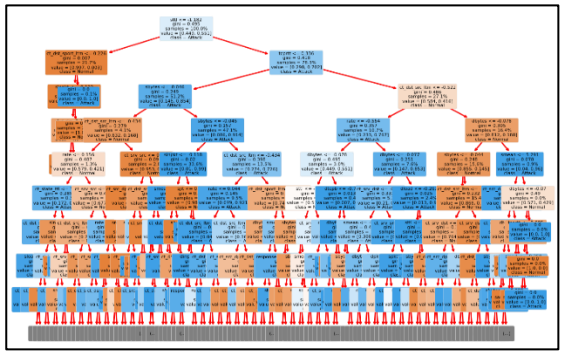

Figure 8: Decision Tree Classifier (Depth = 8 Nodes) Explainable AI Visualization

The decision tree visualizations enables model explainability through inspection of each decision level and its associated feature and splitting value for each condition. If a certain network traffic sample satisfies the condition, it goes to the left branch or node, otherwise it goes to the right branch. Additionally, in each class line, the classification prediction result is depicted depending on the max depth of the tree selected. Utilizing decision trees for IoT network traffic ML-based IDSs provide high accuracy classification results, indicating robust detection of malicious threats. Furthermore, the explainability features of the DT algorithm based on plotting decision trees can help human analysts understand the model. This will allow for greater understanding of the cybersecurity landscape around IoT networks. This understanding includes theorizing what the IDS machine learned from the features or comparing expectations. Human analyst may further aid the machine in learning though adding features or feature engineering using domain knowledge.

This will significantly help analysts assess the correctness of the model decision framework and improve upon it. Multi-layer Perceptron (MLP) Classifier For the MLP classifier, the model was trained and tested using the corresponding dataset. The overall performance accuracy of the model compared to the test set was 89.83%. This shows very exceptional classification predictive value in detecting normal or attack behavior in IoT traffic. The library LIME – Local Interpretable Model-Agnostic Explains can be used to generate model-predictive visualizations of MLP classifiers for individual predictions in the training set. LIME perturbs the original data features and predictions to feed into the developed internal classification model and observe the results. The library then weights the new data output as a function of its proximity to the original point. We then use the sample weights to determine the variation and fit a linear permutation regression to the data set. Finally, the original data points can be explained by the newly trained explanation model. Figure 9 displays an example of the Lime Tabular Explainer output with the top 5 features indicated.

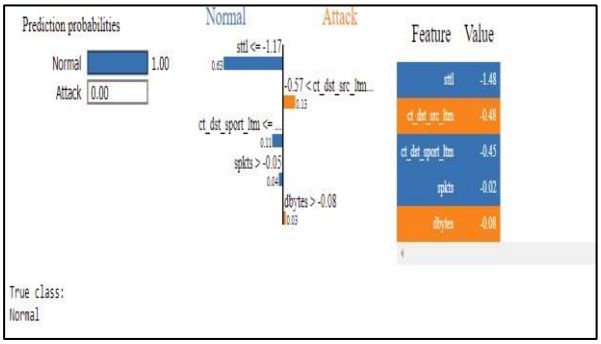

Figure 9: Single Classification Prediction using the MLP Classifier Explanation

The visual dashboard indicate which features and their weights brought the overall behavior classification to be predicted as Normal for that network traffic record. This classification is inspected to be correct as the true class is 'Normal'. This visual dashboard offers robust individual explainability of predicted classifications. Human analysts can conduct indepth analysis for cybersecurity research or follow-up assessment on why certain network traffic was classified in which they were by the model. This tools offers increased transparency capability of predictions that can be exploited for future cybersecurity research, while utilizing

the high-performance benefits of a neural net MLP classifier, which are functionally 'black boxes.

XGBoost Classifier

Similarly, to the other two classifiers, the XGBoost Classifier was trained for the classification task and tested on the testing set. The overall model performance accuracy was 89.89%, demonstrating high capability of the XGBoost classifier to classify network behavior. The performance is approximately similar to the MLP Classifier.

To utilize explainability capabilities with this classifier, the SHAP (SHapley Additive exPlanations) library was utilized. The SHAP library offers the ability to analyze which training samples and features offer the highest impact on model or classifier output. SHAP's main advantages are local explanation and consistency in tree-based model structures such as XGBoost. SHAP creates values that interpret results from tree-based models. It is based on value calculations from game theory and provides extensive feature importance using by 'marginal contribution to the model outcome'.

To explain the predictions, we can use XGBoost's built-in Tree SHAP implementation to explain the classification predictions on the test set. Figure 10 provides a visualization into explaining single prediction, while Figure 11 captures an explanation into many predictions through feature comparison or output classification values. The f(x) values provides a classification value, where closer to 1 indicates Attack behavior, while closer to 0 indicates Normal Activity by a network traffic record.

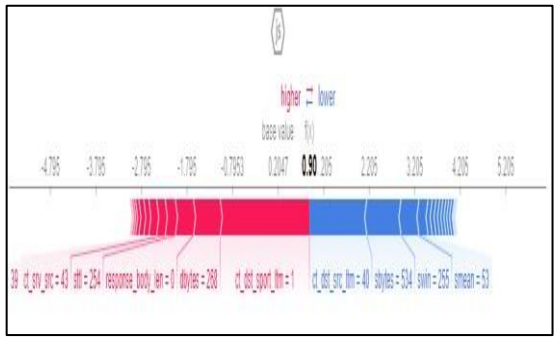

Figure 10: XGBoost SHAP- Visualize a single prediction

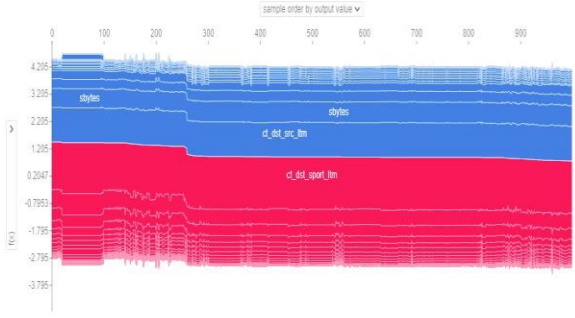

Figure 11: XGBoost SHAP - Visualize many predictions

A feature importance plot through SHAP is conveyed in Figure 12 to determine the mean importance of input training features to predict classification. The results are similar to the DT Classifier.

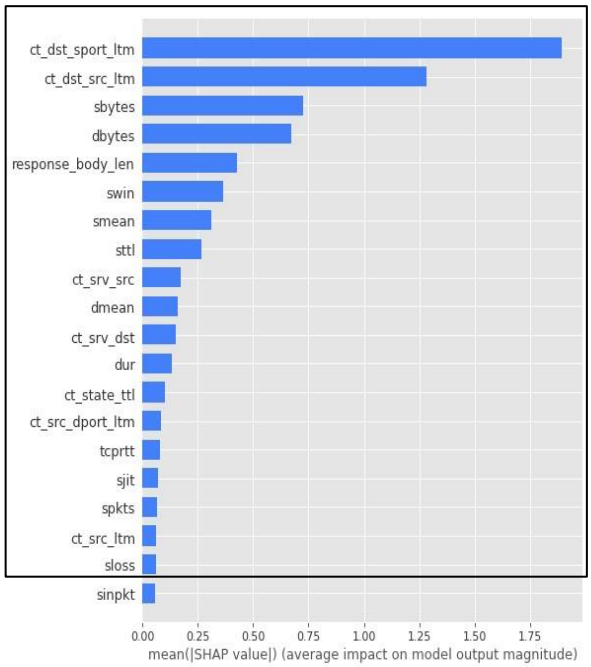

Figure 12: SHAP Fearure Importance on XGBoost Classifier

The SHAP summary graph displays key feature combinations and provides visual indicators of how feature values affect classification predictions. In Figure 13, red indicates high feature scores and blue indicates low feature scores. On the x-axis, high SHAP values on the right correspond to predictive values (aggressive behavior) and low SHAP values on the left correspond to low predictive values (normal activity).

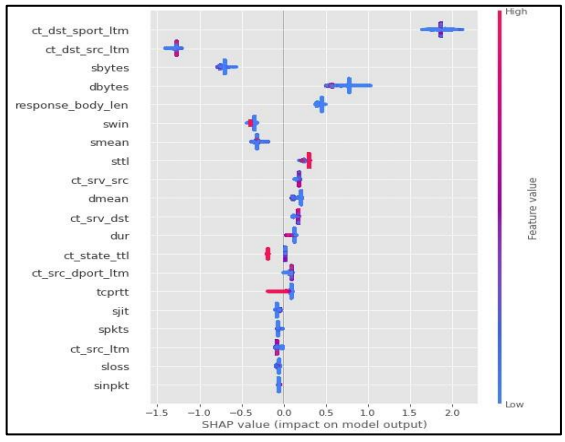

Figure 13: SHAP Summary Plot for XGBoost

Additionally, SHAP values can create SHAP dependency graphs that show the influence of a single feature on the entire data set. They plot the value of a feature across many samples against the SHAP value of that feature and take into account interaction effects present in the feature. In addition, the SHAP Interaction Score Matrix Summary Graph displays a matrix of summary graphs with main effects on the diagonal and interaction effects off the diagonal.

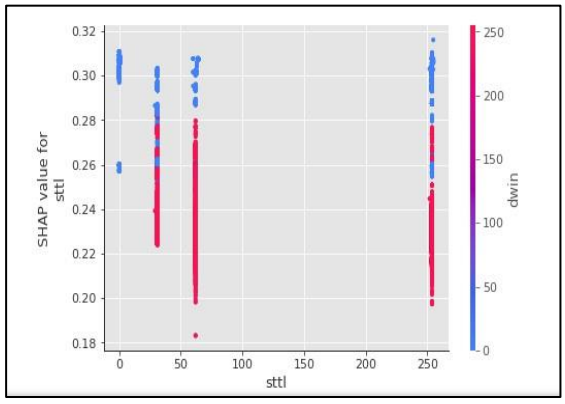

Figure 13: SHAP Dependence Plots for 'sttl' feature

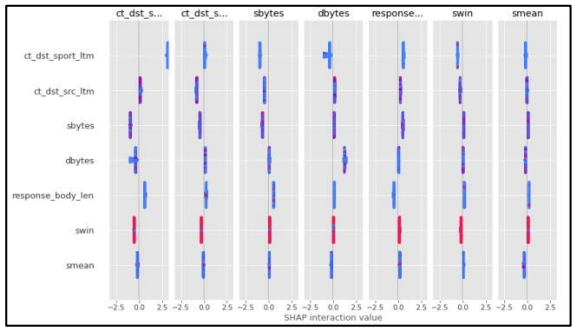

Figure 14: SHAP Interaction Value Summary Plot

Furthermore, using the LIME package as used for the MLP Classifier, individual predictions of the XGBoost Classifier can be explained.

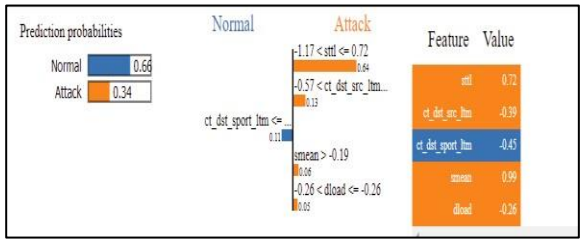

Figure 15: Single Classification Prediction using the XGBoost Classifier Explanation

The model offers a highly efficient and flexible, while high-performing classifier that can be paired with the SHAP and LIME libraries offer robust explainability features. This will increase trustworthiness of advanced black-box algorithms for effective evaluations of ML-

based IDSs for IoT network security.

6. Conclusion

ML learning models utilized for IoT network traffic security through IDSs are increasing becoming more complex, but the need for human analysts to analyze outcomes through inherent domain knowledge for resource allocation and cybersecurity strategy development is a critical role. ML algorithms are often considered “black boxes”, in which the logic or explanation behind the output predictions are not interpretable. Through utilizing the UNSW-NB15 dataset and training a Decision Tree Classifier, MLP Classifier, and XGBoost Classifier, the accuracy results conveyed high-performance for analyzing network behavior of Attack or Normal Activity between connected clients in a IoT network. After, analyzing the performance of ML classifiers, established libraries and techniques for enabling explainability or Explainable AI (XAI) were applied to the trained classifiers to explain its decisions and evaluate feature importance. In the immediate term, this increased transparency will increase trust with ML systems in the IoT cybersecurity domain. Ultimately, it will enable a new range of capabilities of IoT cybersecurity trough extracting insights from sophisticated machine learning models as more explainability conveys the influence of the influence the prediction of a cyber-attack and to what degree.

References

1. Mane, Shraddha, and Dattaraj Rao. "Explaining Network Intrusion Detection System Using Explainable AI Framework." arXiv preprint arXiv:2103.07110 (2021).
2. Moustafa, Nour, and Jill Slay. "UNSWNB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)." Military Communications and Information Systems Conference (MilCIS), 2015. IEEE, 2015.
3. da Costa, Kelton AP, et al. "Internet of Things: A survey on machine learningbased intrusion detection approaches." Computer Networks 151 (2019): 147-157.
4. Arrieta, Alejandro Barredo, et al. "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI." Information Fusion 58 (2020): 82115.
5. Zoghi, Zeinab, and Gursel Serpen. "UNSW-NB15 Computer Security Dataset: Analysis through Visualization." arXiv preprint arXiv:2101.05067 (2021).
6. García-Magariño, R. Muttukrishnan and J. Lloret, "Human-Centric AI for Trustworthy IoT Systems With Explainable Multilayer Perceptrons," in IEEE Access, vol. 7, pp. 125562-125574,2019, doi: 10.1109/ACCESS.2019.2937521.
7. Wang Z: Deep learning-based intrusion detection with adversaries. IEEE Access. 2018;6:38367–384.
8. Moustafa, Nour, et al. "An Ensemble Intrusion Detection Technique based on proposed Statistical Flow Features for Protecting Network Traffic of Internet of Things." IEEE Internet of Things Journal (2018).
9. C. S. W. M. M. Daniel L. Marino, "An Adversarial Approach for Explainable AI in Intrusion Detection Systems," in IECON 2018 - 44th Annual Conference of the IEEE Industrial
10. K. Z. Y. Y. X. W. Maonan Wang, "An Explainable Machine Learning Framework for Intrusion Detection System," IEEE Access , vol. 8, pp. 73127 - 73141, 16 April 2020.

11. Koroniotis, Nickolaos, Moustafa, Nour, et al. "Towards Developing Network Forensic Mechanism for Botnet Activities in the IoT Based on Machine Learning Techniques." International Conference on Mobile Networks and Management. Springer, Cham, 2017.
12. Moustafa, Nour, et al. "A New Threat Intelligence Scheme for Safeguarding Industry 4.0 Systems." IEEE Access (2018).
13. Cohen, "Explainable AI (XAI) with a Decision Tree," Medium, 20-Apr-2021. [Online].Available: https://towardsdatascience.com/explain able-ai-xai-with-a-decision-tree960d60b240bd. [Accessed: 30-Apr-2021].
14. Kasongo, S.M., Sun, Y. Performance Analysis of Intrusion Detection Systems Using a Feature Selection Method on the UNSW-NB15 Dataset. J Big Data 7, 105 (2020). https://doi.org/10.1186/s40537-020-00379-6
15. "LIME: Local Interpretable ModelAgnostic Explanations," C3 AI, 19-Oct-2020. [Online].Available: https://c3.ai/glossary/datascience/lime-local-interpretable-modelagnostic-explanations/. [Accessed: 30Apr-2021].
16. M. Ribeiro, Tutorial - continuous and categorical features. [Online]. Available: https://marcotcr.github.io/lime/tutorials/Tutorial%20%20continuous%20and%20categorical %20features.html. [Accessed: 30-Apr2021].
17. S. Slundberg, "SHAP (SHapley Additive exPlanations)," slundberg/shap. [Online]. Available:https://github.com/slundberg/shap. [Accessed: 30-Apr-2021].