An Explainable Approach for Network Intrusion Detection Using Machine Learning and Deep Neural Network

Explainable Network Intrusion Detection System

A study on synthetic data proportions and model performance in cybersecurity domain

Sajan K. Sarker*

Department of Electrical & Computer Engineering, North South University, sajan.sarker@northsouth.edu

Mahzabeen R. Meem

Department of Electrical & Computer Engineering, North South University, mahzabeen.rahman@northsouth.edu

Arifa A. Srity

Department of Electrical & Computer Engineering, North South University, arifa.srity@northsouth.edu

Web applications have become essential components in various sectors, including e-commerce, banking, and healthcare, making their security a crucial priority. Cyberattacks increasingly target these applications to exploit vulnerabilities and steal users' sensitive data. A robust network intrusion detection tool is essential to detect such threats. Many studies have proposed machine learning and dep learning-based approaches to build effective IDS solutions. A common challenge is these studies is the imbalance in datasets, which researchers often address by generating synthetic data to improve training quality. However, how much synthetic data can be effectively used to train accurate and reliable security models remains unclear. This research investigates the impact of different proportions of synthetic data on model performance. We used the widely recognized dataset CIC-IDS2017, applied the data balancing strategy using SMOTE, and produced three different versions of the dataset–small, medium, and large–to evaluate the model's performance under different levels of data complexities and sizes. Additionally, we applied XGBoost algorithm for feature selection purpose during the preprocessing phase. Our approach was evaluated using various machine learning algorithms, including Decision Tree, Random Forest, Naïve Bayes, Ada Boost, and Deep Neural Networks. Based on performance metrics, we developed an ensemble model that combines the two best-performing classifiers. To enhance transparency, we incorporated Explainable AI (XAI) techniques to make the model's predictions more interpretable and transparent. The result demonstrates that the ensemble model outperformed on the medium dataset, achieving an average accuracy of 97.20%, correctly classifying types of attacks with 97.30% accuracy, and producing a low false positive rate of 0.0048.

CCS CONCEPTS • Security and privacy–Intrusion detection and malware mitigation • Computing methodologies–Machine learning • Networks–Network types

Additional Keywords and Phrases: Network Intrusion Detection, Machine Learning, Synthetic Data, SMOTE, Ensemble Learning, Cybersecurity, Explainable AI, Deep Neural Network.

---

* Place the footnote text for the author (if applicable) here.

ACM Reference Format:
First Author's Name, Initials, and Last Name, Second Author's Name, Initials, and Last Name, and Third Author's Name, Initials, and Last Name. 2018. The Title of the Paper: ACM Conference Proceedings Manuscript Submission Template: This is the subtitle of the paper, this document both explains and embodies the submission format for authors using Word. In Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY. ACM, New York, NY, USA, 10 pages. NOTE: This block will be automatically generated when manuscripts are processed after acceptance.

1 INTRODUCTION

In today's digital age, web applications are integral to a wide range of activities. Worldwide, there are over a billion internet users and websites at the present time. A big reason for the success of the internet is the simplicity and the fact that you can access the applications from anywhere. Many organizations and individuals make their websites and provide online services. With the increasing rate of this website, the risk of cyberattacks are also increasing. So, the security of web applications has become increasingly important, and a secure web environment has become a high priority for any e-business community. Websites like e-commerce, banking and financial, healthcare, and government are more frequently targeted by cyberattacks due to the value of their data, the nature of their operation, or their popularity. Cyberattacks, such as various versions of DoS and DDoS, Port Scanning, Botnet, Heartbleed, SQL injection, cross-site scripting, path traversal, and command injection, are considered most the common and dangerous threats to websites, web applications, and web users [1]. These attacks mostly require only basic knowledge of vulnerabilities and access to readily available tools. A lack of security awareness among developers and organizations often leads to unpatched vulnerabilities and poor security practices, increasing susceptibility. These attacks can assist attackers to bypass the web systems authentication mechanisms, to carry out unauthorized modifications to web content and databases, to extract important data from web application databases, and to steal sensitive information of web servers.

An intrusion detection system (IDS) is a cybersecurity tool that monitors and analyzes network traffic for suspicious behavior or potential threats. It generates alerts when abnormal patterns or known malicious activities are detected [2]. IDS can generally be categorized into two types: signature-based and anomaly-based. Signature-based IDS relies on pre-defined patterns of known attacks to identify threats, while anomaly-based IDS identifies unusual behaviours that deviate from the normal profile using statistical or machine learning methods [3].

This research aims to explore the impact of synthetic data on model performance by generating three versions of the dataset using SMOTE. Each version contains varying degrees of synthetic samples in the minority classes. We also apply XGBoost for feature selection to enhance model efficiency. We consider the small dataset as our primary dataset for feature selection purposes. A range of classifiers, including decision trees, random forests, naïve bayes, ada boost, and deep neural networks, are trained and evaluated. Furthermore, we propose an ensemble model combining the two best-performing classifiers based on evaluation metrics. To promote model transparency and trust, we incorporate XAI techniques LIME to interpret the predictions. Although this approach, we aim to provide insights into how synthetic data affects model accuracy and generalizability in intrusion detection while offering a practical and interpretable solution for real-world deployment.

2 RELATED WORKS

A novel intrusion detection presented by Callegari et al. [4] that is capable of processing raw network traffic in real-time using deep learning techniques. Performed feature extraction directly from packet-level traces using efficient probabilistic

data structure. Different types of deep learning networks–MLP, CNN, RNN, LSTM, and GRU–are evaluated and compared based on detection accuracy, false alarm rate, and execution time. The CNN model achieved detection rate of 92.6% and an accuracy of 89.9% in binary classification, and around 86% for multiclass detection to identify specific attack types. The author highlights the importance of training on realistic, well-balanced datasets to avoid overly optimistic results.

Ahmed et al. [5] proposed a novel intrusion detection model HAEnID (Hybrid Adaptive Ensemble for Intrusion Detection), which integrates three ensemble methods–Stacking Ensemble (SEM), Bayesian Model Averaging (BMA), and Conditional Ensemble Method (CEM)–to enhance detection accuracy, reduce false positives, and adapt to evolving cyber threats. The model was enhanced with explainable AI techniques SHAP and LIME to make the decision transparent and interpretable. The Wednesday Attack Record subset of the CIC-IDS2017 dataset contains various types of DoS, and safe records used in this work that. The Bayesian Model Averaging (BMA) approach achieved the best 98% of accuracy on binary classification and 97.94% accuracy for multi-class classifications.

In a work of et Sajid al. [6] present a hybrid intrusion detection system that integrates XGBoost, CNN, and LSTM to enhance cybersecurity threat detection. It focuses on improving feature selection and classification using several benchmark datasets such as: CIC-IDS2017, UNSW-NB15, NSL-KDD, and WSN-DS. They combine the XGBoost and CNN for feature extraction and utilize LSTM for sequential learning, leading to better detection accuracy and reducing false positives. The results demonstrate high detection rates, with CNN-LSTM achieving 98.55% accuracy on CIC-IDS2017. The approach demonstrates robustness, scalability, and the ability to detect both known and unknown threats across diverse network environments.

A signature-based intrusion detection system was presented by Ahmed et al. [7] that integrates machine learning, deep learning, and fuzzy clustering techniques to enhance network security. They used the UNSW-NB15 dataset for binary classification of normal and malicious traffic. They integrate the data fusion, adaptive modeling, and signature-based detection with intelligent techniques. The system focuses on both knowns and unknown threats. Fuzzy clustering was applied to improve data grouping and pattern recognition. They emphasize the need for scalable, real-time IDS solutions capable of adapting of dynamic nature of cyber threats.

Talukder et al. [8] address the key challenges in IDS, including class imbalance, high dimensionality, and detection rate. Their method uses SMOTE for data balancing and XGBoost for feature selection to improve classification performance. The approach was tested on KDDCUP99 and CIC-MalMem-2022 datasets, achieving accuracy rates up to 99.99% with minimal false positives and false negatives. The model includes a pipeline of preprocessing, feature ranking, and classification using various machine learning and deep learning models.

Nirvikar et al. [9] explored how Artificial Intelligence and Machine Learning techniques can improve cybersecurity, concentrating on threat identification and response. The methodology covered various machine learning technique such as anomaly detection, malware classification, and network intrusion detection. They used real-world cybersecurity datasets, including network traffic logs and malware samples. They provide an overview of AI and ML applications in cybersecurity, including key techniques, case studies, and an examination of limitations and challenges, with notable issues being the need for large labeled datasets and the interpretability of black-box models.

Saleem et al. [10] proposes a machine learning-based intrusion detection model that analyzes web server logs to identify and classify these attacks. This technique improves the identification of unidentified attacks. Here, the author utilizes two machine learning models- simple feature extraction and text-based classification, on a dataset generated using a private network. The text- based model, which uses a larger set of features, outperforms the simple model in accuracy. The model has lacking on capability to handle multiclass attack detection effectively.

In research Liang et al. [11] proposed a Deep Learning approached Anomaly-Based Web Attack Detection focus on Cloud Technology and Web Attacks. There are some limitations of Current Approaches: Signature-based methods, and Rule-based methods. The model has few advantages, such as, no feature selection is needed, making it adaptable and easy to apply, and the model was accurate in identifying web-based attacks and outperforms traditional Web Application Firewalls (WAFs) like MoD Security, especially when it comes to dynamic attacks and anomaly detection. It achieves 98% accuracy with high sensitivity of 97.56% and specificity of 99.21% on the CSIC dataset. The authors describe how well LSTM/GRU RNNs work in conjunction with neural network classifiers and offer possible uses for this combination in other domains where anomaly detection is needed.

3 METHODOLOGY

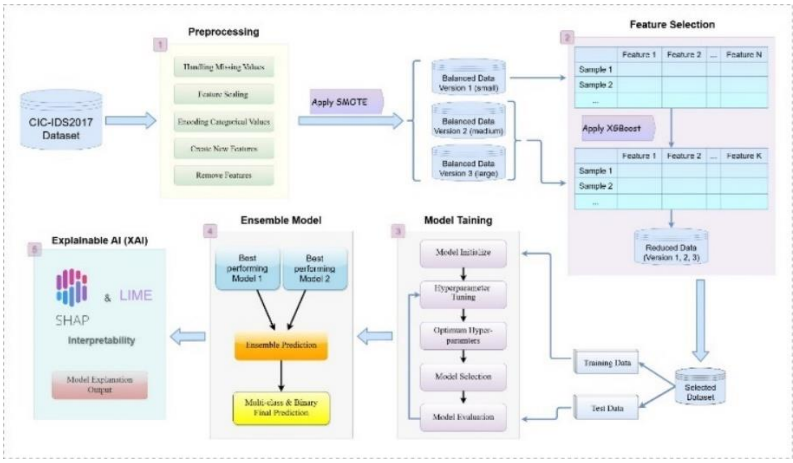

Figure 1: Network Attack Detection and Explanation System Design Process.

This study aims to analyze how much synthetic data can be effectively used to train accurate and reliable security models and develop an Intrusion Detection System (IDS) framework that can analyze network traffic data to detect and classify various cyberattacks using machine learning and deep neural networks. To improve the accuracy, we balance the dataset using SMOTE and produce three different version of dataset–small, medium, and large, and select important features with XGBoost. We also use Explainable AI (XAI) techniques to make the prediction easier to understand.

3.1 Dataset collection and preprocessing

We used the CIC-IDS2017 comprehensive dataset [12], that is widely-used dataset for intrusion detection system (IDS) research. It contains network traffic data, that captured in real-world conditions, including various malicious traffic data and normal traffic data. However, the dataset was imbalanced as figure 2, with the majority class being safe traffic data, and minority class belong to the malicious traffic. To address this issue, we used data balancing strategies and produced three different version of the dataset–small, medium, and large. The original dataset contains total of 1,854,979, where the new versions small - 75,000, medium – 1,522,380, and large contains 2,602,112 total samples. These contains different types of network attack traffic such as Safe traffic, DoS Hulk, Dos Golden Ey, DoS Slowloris, DoS Slowhttptest, SSH-

Patator, DDoS, Brute Force Attacks, Web Attacks XSS, Web Attacks SQLi, Port Scans, Botnet, Infiltration, and Heartbleed. To processes these data for our model training, we've followed several steps:

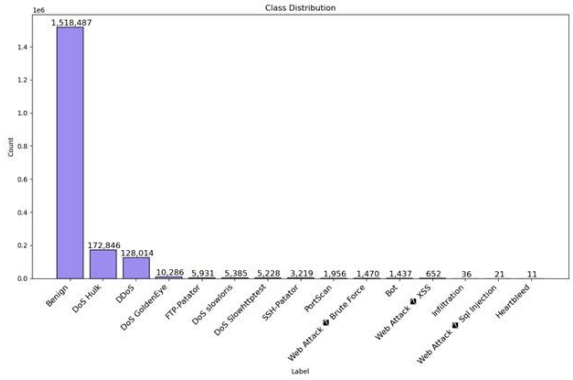

Figure 2: Original datasets class distribution.

During the preprocessing steps, we handle missing values, and remove duplicate unwanted samples, and eliminate identical or low-variance features that do not contribute meaningfully to model learning. Analyze the feature correlation to determine how strongly two features are related. Normalize the data to ensure uniform scaling across all features, which is especially crucial for our different range of features. Apply the SMOTE techniques to balance the class distribution using both oversampling and undersampling methods. Using these techniques, we produced three different versions of the dataset with varying levels of data complexity and size.

3.1.1 Small (Version 1)

This version contains a total number of 75,000 samples, with 15,000 samples for each class in the multi-class target. This dataset was treated as the primary dataset, used for feature selection and model hyperparameters. During processing, we removed attack categories that contained fewer than 5,000 samples, as they were considered insufficient for effective learning. The remaining categories were balanced using SMOTE to ensure that each class had an equal number of samples. Similar attack types were grouped into broader categories illustrated in Figure 3 to simplify the classification task and improve model generalization. That mapping helped in reducing the complexity of the classification task while maintaining the diversity of attack types.

3.1.2 Medium (Version 2)

This version contains a total of 1,522,380 samples, with the benign class comprising 359,910 and each malicious class initially having approximately 193,745. To achieve a balanced dataset, the benign class was undersampled to 25% of the total samples, while each malicious class was oversampled to match the size of the largest malicious category. As with the small version, attack types were grouped into broader categories to simplify the classification task and improve the model's generalization capabilities. The distribution is shown in Figure 3.

3.1.3 Large (Version 3)

This dataset version includes the entire set of benign samples, preserved without any modification to maintain the original distribution of normal network traffic after preprocessing. The malicious classes were balanced using oversampling techniques to match the size of the largest malicious category. Similar to the medium version, the attack types were grouped

into broader categories to simplify the classification task. The final class mapping used in this version is consistent with the one applied in the medium version, shown in Figure 3.

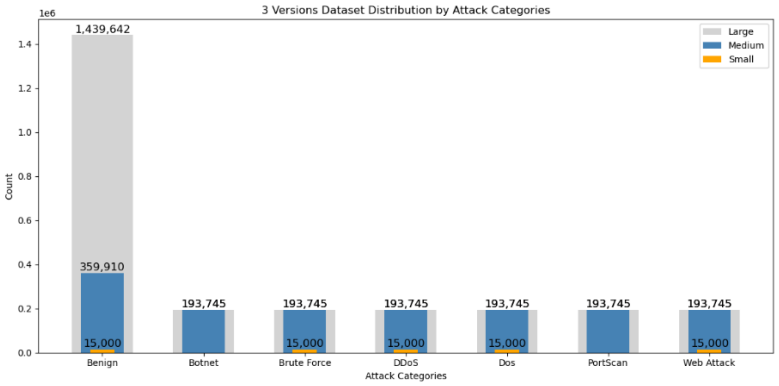

Figure 3: Distribution of new datasets.

3.2 Feature Selection with XGBoost

XGBoost, a powerful gradient-boosting algorithm, is utilized for the feature selection. The XGBoost model is trained on the small version of the dataset to evaluate the importance of each feature in distinguishing between safe traffic and malicious network traffic. These feature importance scores provided by the XGBoost indicate the contribution of each feature to the model's decision-making process. In particular, it reflects the gain, which measures the improvement in accuracy bought by a feature to the decision tree branches it appears on–higher gain implies higher importance. We also considered how frequently each feature appeared across the decision trees, which highlighted its importance. This approach helped us to printout the most impactful features, allowing us to streamline the model, boost its performance, and lower chance of overfitting. To finalize the selection, we applied a threshold of 0.0030 to the feature importance scores and kept only the top-ranking ones.

3.3 Model Training:

Initially, all five base models were trained on each dataset version to establish baseline performance. This step was crucial to identify how well each algorithm could handle the data without any optimizationns. The initial performance also served as a reference point for further improvements through hyperparameter tuning. Hyperparameter tuning was performed to enhance the model performance and generalization. This process was performed exclusively on the small dataset, which served as the primary dataset fro model development and tuning. We employed GridSearchCV and Optuna as the primary optimization techniques. Each model was tuned independently using a validation split of the small dataset to identify the best-performing hyperparameter configurations. Once the optimal hyperparameters for each model were identified using the small dataset, we reused this same configuration when training and evaluating the models on the medium and large datasets. This approach ensured a consistent baseline for comparison across all dataset sizes, isolating the effect of dataset scale on model performance while avoiding bias introduced by re-tuning on large datasets.

Each model was evaluated on a validation set (a subset of the training set) to observe the overfitting and underfitting issues. Models exhibiting poor generalization were re-tuned interactively to ensure optimal performance. This phase ensured that only well-regularized and robust models advanced to the testing phase. Once we found the best-performing model of each dataset, we evaluated the model on the test set to get the model performance and generalization ability.

3.4 Ensemble Model Implementation:

The ensemble model was designed to combine predictions from multiple machine learning models, specifically for multi-label classification task (which involves both multi-class & binary classification). Best two performing models was selected based on their test evaluation scores. The ensemble checks how well each model performs using a scoring method called log loss (Equation 1), which tells us how confident and correct each model's predictions are. If a model makes better

$$Logloss(y,p) = -(y\log(p) + (1-y)\log(1-p)) \quad (1)$$

predictions, it gets a higher weighted value, and if a model performs poorly, it gets a lower weighted value. The for each

$$w_i = \frac{1}{logloss_m} \quad (2)$$

$$p_{final} = \sum_{i=1}^{n} w_i * P_i \quad (3)$$

model was calculated as the inverse of the log loss (Equitation 2). During the prediction process, each model makes predictions and provide probabilistic values. These predictions are then combined. Each model's prediction is given a weight, which reflects how good that model is. Then for every class, we multiply each model's predicted probability by its weights (Equation 3), and sum up the weighted probabilities across all models. Next, we pick the class with the highest probability as the final predictions.

3.5 Explainable AI (XAI) for Model Interpretation

An explainable AI technique was applied to enhance the interpretability of the chosen ensemble model's prediction. This is important in the domain of cybersecurity, where understanding the reason behind a mode's classification–whether network traffic is safe or malicious–is essential for building trust, supporting decision-making, and providing actionable insights to security analytics. Two widely used XAI methods were employed in this study.

3.5.1 SHAP (SHapley Additive exPlanations)

SHAP was used to provide a global and local interpretation of model predictions by quantifying the contribution of each feature to the output. It assigns each feature a value that represents its impact on the prediction, making it easier to understand how the model arrives at its decision across different data instances.

3.5.2 LIME (Local Interpretable Model-Agnostic Explanaton)

Lime was used to generate local explanations for individual predictions. It works by creating a simple model just around one prediction to see how each input feature affects the results. This helps to understand why a specific sample was classified in a certain way, making the model's behavior easier to understand one case at a time.

4 RESULTS AND DISCUSSION

Each of these model's result was analyzed based on various performance metrics to find out the impact of syntactic data and find out the best performing model. We analyzed the performance of our models based on the selected features. It shows us from Decision Trees, Random Forests, Naïve Bayes, AdaBoost, and Deep Neural Network, in every version of

dataset the Decision Trees, and Random Forest Performed Better. The Ensemble model which was created with these best two performing model was performed more better compare to a single model.

Table 1: Classification Result of Small Dataset

	Binary Classification Results					***Multiclass Classification Results***				
	Acc	**Prec.**	**Rec.**	**F1**	**FPR**	**Acc**	**Prec.**	**Rec.**	**F1**	**FPR**
Decision Trees	0.9613	0.9564	0.9972	0. 9763	0.1822	0.9707	0.9709	0.9707	0.9707	0.0073
Random Forests	0.9779	0.9749	0.9980	0.9863	0.1028	0.9744	0.9750	0.9744	0.9746	0.0064
Naïve Bayes	0.7368	0.9397	0.7171	0.8134	0.1842	0.8472	0.8608	0.8472	0.8427	0.0383
AdaBoost	0.9021	0.8911	0.9999	0.9424	0.4895	0.3602	0.1645	0.3602	0.2171	0.1597
Multi-Layer Perceptron	0.9596	0.9851	0.9641	0.9745	0.0584	0.9560	0.9584	0.9560	0.9564	0.0110
Ensemble Model (DT + RF)	**0.9695**	**0.9652**	**0.9978**	**0.9812**	**0.1441**	**0.9729**	**0.9731**	**0.9729**	**0.9729**	**0.0068**

This small version contains fewer samples, which made it challenging for the model to effectively learn the patterns associated with different types of malicious activity. As a result on Table 1, achieved 96.95% accuracy for binary classification, and 97.29% for multiclass classification. While the model showed lower false positive rate of 0.0068 for multi-class classification, and it had a higher false negative rate for binary classification with 0.1441. Which indicates that the model was struggled more to correctly identify malicious instances in binary classification task.

In this version of the dataset, synthetic data was generated for two types of malicious classes, while the remaining three classes contain the original real-world data. Overall, this small version dataset struggled to provided enough diverse and balanced information, making it difficult for the model to learn effectively and generalize well.

Table 2: Classification Result of Medium Dataset

	Binary Classification Results					***Multiclass Classification Results***				
	Acc	**Prec.**	**Rec.**	**F1**	**FPR**	**Acc**	**Prec.**	**Rec.**	**F1**	**FPR**
Decision Trees	0.9682	0.9547	0.9948	0. 9795	0.1187	0.9791	0.9597	0.9691	0.9692	0.0054
Random Forests	0.9766	0.9725	0.9975	0.9849	0.0919	0.9734	0.9741	0.9734	0.9735	0.0047
Naïve Bayes	0.8649	0.8554	0.9909	0.9182	0.5458	0.7000	0.8169	0.7000	0.6805	0.0491
AdaBoost	0.8826	0.8671	0.9999	0.9288	0.4995	0.3402	0.1379	0.3402	0.1897	0.1212
Multi-Layer Perceptron	0.9382	0.9836	0.9348	0.9586	0.0509	0.9172	0.9276	0.9165	0.9165	0.0148
Ensemble Model (DT + RF)	**0.9719**	**0.9671**	**0.9972**	**0.9819**	**0.1107**	**0.9730**	**0.9731**	**0.9726**	**0.9726**	**0.0048**

The Table 2 results demonstrate that the ensemble model, built using Decision Tree and Random Forest classifiers, achieved the most balanced and effective performance across both binary and multiclass classification tasks. It achieved an accuracy of 97.19% for binary classification and 97.30% for multiclass classification, maintaining high precision, recall, and F1 scores while keeping a low false positive rate. Among the individual models, Random Forest showed strong performance with consistent metrics across both tasks. The

Decision Tree and Multi-Layer Perceptron models offered good results compared to the other two models. The other two models failed to provide competitive results.

In this medium dataset, synthetic data was generated for four different malicious the classes in order to address class imbalance and improve model learning. Meanwhile, the benign class and the other two malicious classes contain the original samples. Although the synthetic data helped balance the dataset, the model faced difficulties in accurately classifying the artificially generated samples. The synthetic data failed to capture the complex patterns of different kinds of malicious activity. As a result, the model showed reduced performance when destringing between certain attack types.

Table 3: Classification Result of Large Dataset

	Binary Classification Results					***Multiclass Classification Results***				
	Acc	**Prec.**	**Rec.**	**F1**	**FPR**	**Acc**	**Prec.**	**Rec.**	**F1**	**FPR**
Decision Trees	0.9667	0.9711	0.9539	0. 9624	0.0230	0.9662	0.9665	0.9660	0.9656	0.0094
Random Forests	0.9716	0.9849	0.9511	0.9677	0.0118	0.9716	0.9721	0.9716	0.9713	0.0081
Naïve Bayes	0.6998	0.5998	0.9883	0.7465	0.5338	0.6520	0.8160	0.6502	0.6627	0.0541
AdaBoost	0.7224	0.6171	0.9999	0.7632	0.5021	0.5534	0.3055	0.5527	0.3935	0.1429
Multi-Layer Perceptron	0.8850	0.9488	0.7852	0.8593	0.0343	0.8728	0.8873	0.8724	0.8469	0.0373
Ensemble Model (DT + RF)	**0.9692**	**0.9774**	**0.9533**	**0.9652**	**0.0179**	**0.9691**	**0.9697**	**0.9691**	**0.9688**	**0.0088**

The result of the larger Dataset of Table 3 show that the Ensemble Model performed best overall achieving high accuracy, precision, and F1 scores in both binary and multiclass classification with low false positive rates. However, the large dataset contains a large number of benign (safe) samples, which influenced the model to focus more on those patterns, making it more challenging to accurately learn the complex characteristics of malicious samples. As a result, the model exhibited some bias in its predictions.

In Figure 4, the SHAP summary plot shows the most important features influencing the model's predictions for multiclass classifications. Features like **Bwd Packet Length Std**, **Fwd Seg Size Min**, and **Avg Packet Size** had the highest impact across multiple classes. Each color shows whether a low or high feature value pushes the model toward a particular class. This highlights how specific network traffic characteristics contribute significantly to classifying different types of malicious activities.

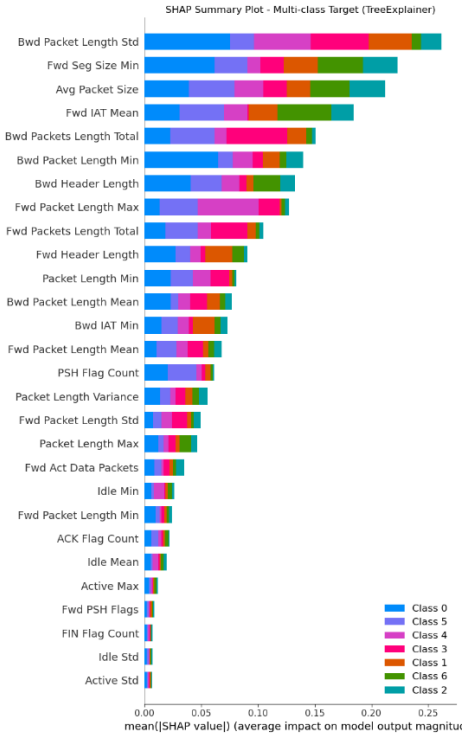

Figure 4: SHAP Result Analysis

5 CONCLUSION

Overall, the medium dataset performed well in both binary and multiclass classification task, containing a balanced of synthetic and real data across classification task. The model's capacity to recognize complex patterns, particularly for malicious classes, was enhanced by the synthetic data, leading to increased by the synthetic data, leading to increased accuracy and decreased the false positive rates. The ensemble model, which included Decision Tree and Random Forest, consistently outperformed individual classifiers by maximizing their strengths and reducing their shortcomings out of all the models that were assessed. The ensemble approach was a dependable solution for accurate and interpretable for network intrusion detection since it was shown to be both robust and effective.

ACKNOWLEDGMENTS

The authors would like to thank the Department of Electrical & Computer Engineering at North South University for their guidance and support throughout the completion of this project. Special thanks to our supervisor for valuable feedback and encouragement.

6 HISTORY DATES

In case of submissions being prepared for Journals or PACMs, please add history dates after References as (*please note revised date is optional*):

Received November 2019; revised August 2020; accepted December 2020

REFERENCES

[1] K. Thompson, "Most Common Website Security Attacks and How to Protect Yourself," 2024. [Online]. Available: https://www.tripwire.com/state-of-security/most-common-website-security-attacks-and-how-to-protect-yourself

[2] M. H. K. a. R. T. Patil, "A Review of Intrusion Detection System Using Machine Learning Approach," *Journal of Information Security,* vol. 6, pp. 131-137, 2015

[3] J. D.-V. G. M.-F. a. E. V. P. Garcia-Teodoro, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers & Security,* vol. 28, pp. 18-28, 2009

[4] S. G. a. M. P. C. Callegari, "A real-time deep learning based approach for detecting network attacks," *Big Data Research,* vol. 36, 2024

[5] Z. J. A. A. S. K. M. T. S. A. A. a. A. U. R. U. Ahmed, "Explainable AI-based innovative hybrid ensemble model for intrusion detection," *Journal of Cloud Computing: Advances, Systems and Applications,* vol. 13, 2024

[6] K. R. M. A. A. T. S. M. A. H. K. J. T. a. A. U. R. M. Sajid, "Enhancing intrusion detection: A hybrid machine and deep learning approach," *Journal of Cloud Computing: Advances, Systems and Applications,* vol. 13, 2024

[7] M. N. A. S. T. A. E.-H. M. A. T. S. a. M. A. K. U. Ahmed, "Signature-based intrusion detection using machine learning and deep learning approaches empowered with fuzzy clustering," *Scientific Reports,* vol. 15, 2025

[8] K. F. H. M. M. I. M. A. U. A. A. M. A. Y. F. A. a. M. A. M. M. A. Talukder, "A dependable hybrid machine learning model for network intrusion detection," *arXiv,* vol. 72, p. 103405, 2023

[9] S. T. P. K. S. V. A. K. S. a. S. S. N. Katiyar, "AI and cyber-security: Enhancing threat detection and response with machine learning," *Educational Administration: Theory and Practice,* vol. 30, no. 4, pp. 6273-6282, 2024

[10] M. S. M. H. a. U. F. S. Saleem, "Web Server Attack Detection using Machine Learning," in *2020 International Conference on Cyber Warfare and Security (ICCWS)*, Islamabad, Pakistan, 2020

[11] W. Z. a. W. Y. J. Liang, "Anomaly-based web attack detection: A deep learning approach," in *The 2017 VI International Conference*, Beijing, China, 2017

[12] A. H. L. a. A. A. G. I. Sharafaldin, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018)*, 2018