An Explainable Anomaly Detection Benchmark of Gradient Boosting Algorithms for Network Intrusion Detection Systems

Merve Nur Yılmaz Batuhan Bardak
Cyber Security and Big Data Directorate, STM Trade Inc., Ankara, Turkey
{mnur.yilmaz, batuhan.bardak }@stm.com.tr

2022 Innovations in Intelligent Systems and Applications Conference (ASYU) | 978-1-6654-8894-5/22/$31.00 ©2022 IEEE | DOI: 10.1109/ASYU56188.2022.9925451

***Abstract*—Nowadays, protecting computer systems by preventing malicious network attacks is a vital topic. In recent years, machine learning-based network intrusion detection systems (NIDS) started showing effective results. While the task of classifying cyber attacks in NIDS has been studied extensively in the literature, there is no comprehensive benchmark study with gradient boosting algorithms on recent open-source datasets. This paper aims to evaluate different gradient boosting-based algorithm performances including XGBoost, CatBoost, and LightGBM on different open-source NIDS datasets such as CIC-IDS2017, CSE-CIC-IDS2018, and INSDN. Furthermore, the SHapley Additive exPlanations (SHAP) is applied to increase the interpretability of the models and investigate the relationship between cyber attacks and the network features. Our experimental results demonstrate that the XGBoost model consistently outperforms other comparative models in F1 score for all datasets. At the same time, we compare the training/inference time of different gradient boosting algorithms which is an important constraint for real-time intrusion detection systems. Moreover, the different important features between different datasets can help data scientists for designing better artificial intelligence-based intrusion detection algorithms.**

***Index Terms*—Intrusion detection system, ensemble learning, SHAP, Shapley value, machine learning**

I. Introduction

The rapid growth of computer networks worldwide has resulted in an increasing number of cybersecurity attacks. The total annual cost of all these cyber attacks on the global economy is 400 billion dollars [1]. Cyber attacks can be done in several different ways such as malware attack [2], unauthorized access [3], data breaches [4], denial of service (DoS) [5], or social engineering [6]. Intrusion Detection Systems (IDS) are presented as a solution to tackle various types of security challenges. The main goal of IDS systems is to detect suspicious activities by monitoring network traffic and system logs. IDS systems can be deployed in different network locations, and Network Intrusion Detection Systems (NIDS) is the one which is placed at network points including gateways and routers.

In addition to the increase in cyber security incidents in recent years, there have been remarkable advancements in the field of artificial intelligence. Especially, deep learning-based models such as convolutional neural networks, recurrent neural networks, and transformer networks achieve great success in many different domains including images [7], text [8], and audio [9]. Aside from these domains, deep learning approaches have shown effective results on a variety of challenges in the cyber security domain. Traditional NIDS can identify known attacks and anomalies using rule-based and pattern-matching algorithms. However, these systems are unable to keep up with the constant evolution of the cyber attack environment and the appearance of new cyber threats. To overcome this problem, there is an extensive number of research on intrusion detection models based on machine/deep learning techniques. Pande et al. [10] propose a Random Forest algorithm to detect DDoS attacks on the NSL-KDD dataset. They claim that their model can correctly classify 99.76% of the cyber attacks in their dataset. Suresh and Anitha [11] use different machine learning algorithms including Naive Bayes, C4.5, SVM, KNN, K-means, and Fuzzy c-means clustering to detect DDoS attacks. Before the training step, they make feature selections based on chi-square and information-gain methods. Yuan et al. [12] propose a deep learning-based method called DeepDefense to predict DDoS attacks. They use different variations of recurrent neural networks such as LSTM, GRU, and CNNLSTM. They compare these deep learning algorithms with random forest and claim that they can reduce the error rate from 7.5% to 2.1%. While deep learning generally gives successful results, especially with image, text, and audio data, the tabular data type is also one of the most used datasets in real-world applications. Deep learning algorithms pose many challenges to tabular data including handling missing values, mixed feature types, locality and so forth [13]. Since NIDS datasets are generally in tabular data form, we work with XGBoost [14], CatBoost [15], LightGBM [16] methods, which have proven to work successfully on tabular data, and compare them with Random Forest [17] algorithm.

While various machine learning and deep learning studies have reported impressive prediction performance on a variety of challenges, the Explainable Artificial Intelligence (XAI) [18] subject has gained a lot of interest in the AI field. The main goal of the XAI approaches is to make black-box deep learning algorithms more transparent, fair, and consistent. As in many domains, explainability is also very critical in the field of cyber security. To understand the

Table I
HYPERPARAMETER SPACE OF RANDOM SEARCH

Algorithm	Iterations	Max Depth	Learning Rate	Regularizer	Feature Fraction	Boosting	Criterion
Random Forest	80, 100, 120, 140	6, 8, 10, 12	-	-	sqrt, log2	-	gini, entropy
CatBoost	80, 100, 120, 140	6, 8, 10, 12	0.10, 0.20, 0.30	0.01, 1, 10, 100	-	-	-
LightGBM	80, 100, 120, 140	6, 8, 10, 12	0.10, 0.20, 0.30	0.01, 1, 10, 100	0.6, 0.8, 1	gbdt, goss	-
XGBoost	80, 100, 120, 140	6, 8, 10, 12	0.10, 0.20, 0.30	0.01, 1, 10, 100	0.6, 0.8, 1	-	-

prediction decisions of proposed machine learning models, we employ the SHapley Additive exPlanations (SHAP) [19] method which is an extension of the Local Interpretable model-agnostic explanations (LIME) [20]. The output of the SHAP method allows us to explain the important features for predicting cyber attacks in NIDS datasets. In the literature, there are few publications [21], [22] that utilized the SHAP framework on NIDs datasets, hence they work with a single dataset which is not suitable to compare important features across datasets. In this study, we apply the SHAP method to three different NIDS datasets and compare important features in different network environments.

In this paper, we use gradient boosting-based supervised algorithms on three well-known intrusion detection datasets: CIC-IDS 2017 [23] , CSE-CIC-IDS 2018 (which are shared by Canadian Institute for Cybersecurity[1][2]), and the InSDN dataset [24]. Furthermore, in order to indicate the relationship between the network features and the risk of cyber security attacks, SHAP values were introduced to evaluate important network variables in the predictor. There are two main motivations for this study. First, we make a benchmark study with GBDT algorithms on popular and recent NIDS datasets to compare the prediction successes of algorithms and training/inference times. Secondly, we use the SHAP framework to understand the important network features for NIDS datasets and check if these features are changed based on attack type and network environment.

II. DATASET

For this study, we select the CIC-IDS2017, CSE-CIC-IDS2018, and InSDN datasets since they are among the most recent and comprehensive datasets publicly available for the network intrusion detection problem. The details of these datasets, which are the sample sizes, number of features, and target classes are shown in Table II.

CIC-IDS2017. In 2017, the Canadian Institute of Cybersecurity (CIC) published this dataset with the purpose of creating an up-to-date, diverse and reliable intrusion detection dataset. This dataset is constructed by simulating benign traffic as well as several common attack types in a small network. More than 75 features are extracted from the generated network traffic using CICFlowMeter[3], which is an open-source network flow generator tool. The simulated attacks include DoS/DDoS, Botnet, Brute Force, Heartbleed, Infiltration, Web attacks, and subtypes of these attacks.

Table II
STATISTICS OF THE DATASET

Dataset	Sample Size	# of Features	# of Target Classes
CIC-IDS 2017	2,830,743	77	15
CSE-CIC-IDS 2018	16,233,002	84	15
InSDN	343,889	84	8

CSE-CIC-IDS2018. As a collaborative project between the Communications Security Establishment (CSE) and the CIC, this dataset was created in 2018. In terms of the network architecture and the implemented attack types, it is similar to the CIC-IDS2017 dataset. In addition, the CICFlowMeter tool is also used to generate more than 80 network flow features using the captured traffic. The main difference is that this dataset was constructed on AWS (Amazon Web Services) platform with a larger network topology. As a result, it contains around 16 million samples, considerably bigger than other selected datasets.

InSDN. Recently, SDN systems have been widely used to replace conventional networks in order to overcome their challenges and limitations. However, the most commonly used datasets in intrusion detection are created in conventional network architectures. InSDN is published in 2020 aiming to be used to evaluate intrusion detection systems generated for the SDN. It differs from other intrusion detection datasets in that it is one of the first datasets implemented in an SDN environment. The dataset is created similarly to the CIC-IDS2017 and CSE-CIC-IDS2018 datasets and the CICFlowMeter is used to extract more than 80 network flow features as well. The DoS/DDoS, Botnet, Probe, BFA, and Web attacks are among the simulated attack types.

III. METHODS

To make benchmark on the selected NIDs datasets, we apply gradient boosting decision tree (GBDT) based XGBoost [14],

[1]https://www.unb.ca/cic/datasets/ids-2017.html
[2]https://www.unb.ca/cic/datasets/ids-2018.html
[3]https://github.com/ahlashkari/CICFlowMeter

Table III
PERFORMANCE COMPARISON OF MODELS. FOR ALL THREE DATASET, WE REPORT BOTH PREDICTIVE AND TRAINING/TESTING TIME PERFORMANCE.

Dataset	Algorithm	Precision Macro	Recall Macro	F1 Macro	False Positive Rate	Training Time	Testing Time
CIC-IDS 2017	Random Forest	0.93	0.71	0.74	0.00137	1438.39s	17.68s
	CatBoost	0.89	0.83	0.83	0.00017	715.74s	2.44s
	LightGBM	0.87	0.84	0.85	0.00014	456.11s	3.23s
	XGBoost	0.89	0.85	0.86	0.00015	2597.57s	2.59s
CSE-CIC-IDS 2018	Random Forest	0.94	0.76	0.79	0.00530	6920.62s	68.60s
	CatBoost	0.91	0.80	0.83	0.00426	4401.09s	14.42s
	LightGBM	0.91	0.81	0.83	0.00419	841.41s	17.49s
	XGBoost	0.89	0.83	0.84	0.00423	20424.06s	11.30s
InSDN	Random Forest	0.98	0.97	0.97	0.00041	20.50s	0.59s
	CatBoost	0.99	0.97	0.98	0.00036	23.41s	0.32s
	LightGBM	0.94	0.98	0.95	0.00031	4.54s	0.27s
	XGBoost	0.99	0.98	0.99	0.00033	53.64s	0.25s

LightGBM [16], CatBoost [15], and bagging based Random Forest [17] models. Below, we give brief key ideas of each model.

Extreme Gradient Boosting (XGBoost). XGBoost is a tree-based boosting algorithm that is very popular and effective in the machine learning community. Especially, in the Kaggle platform, XGBoost dominates many competitions for tabular data. The main differences between the XGBoost and the traditional gradient boosting approach are utilizing advanced regularization methods and parallelizing across resources easily.

LightGBM. LightGBM is another variation of GBDT algorithms. While XGBoost has demonstrated remarkable effectiveness on a variety of challenges, there can be issues with efficiency and scalability when the feature dimension is high and the data quantity is large. To address this issue, LightGBM proposes two different methods: Gradient-based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). According to the experimental results, LightGBM can reduce training time by up to 20 times compared to regular GBDT algorithms while maintaining the same accuracy performance.

CatBoost. CatBoost is another GBDT algorithm that was developed and proposed by Yandex researchers in 2017. There are two main algorithmic advances proposed in this method: supporting categorical features effectively and implementing ordered boosting. The authors of this algorithm also claim that CatBoost provides great results with the default parameters which makes this method easy to use.

Random Forest. Random Forest is a popular ensemble learning method based on the bagging technique. In this approach, each ensemble model trains with different random initial parameters and training data. Additionally, unlike the boosting algorithms, the base learners can be trained simultaneously. The final decision is made by the majority vote of a group of trees.

IV. EXPERIMENTAL SETUP

Data Preprocessing. The publicly available datasets that are used in this study need some data preprocessing steps such as feature elimination, and missing value imputation before using them in machine learning algorithms. To achieve this, the following steps are applied to all the datasets.

- **Feature Elimination**: The network-identifier features which are Flow ID, Source and Destination IP, Source and Destination Port, Protocol, and Timestamp are removed, leaving the datasets with 77 statistical features.
- **Data Cleaning**: The datasets contain some anomalies such as duplicate columns and rows with inconvenient values which are removed to obtain a clean dataset.
- **Data Transformation**: Many features contain missing or infinite values that should be handled before training. The missing values are filled with the mean of each feature. The infinite values are replaced with the max/min value of the features.

Training. CatBoost, LightGBM, XGBoost, and Random Forest methods are used to build different multiclass classifiers for each dataset. The number of target classes for each dataset is shown in Table II, as well as the sample sizes. Each dataset is split into train, validation, and test sets with 80%, 20%, and 20% ratios, respectively. The parameters of the models are optimized by using the random search approach. For each method, hyperparameter optimization is made with 30 different parameter settings. The parameter space for the random search is shared in Table I which is inspired from [25].

To understand the important features of different datasets, the SHAP framework is applied to XGBoost models sep-

Table IV
XGBOOST MODEL PERFORMANCE COMPARISON ON THREE NIDS DATASETS. WE REPORT THE TRAIN/VALIDATION/TEST SIZE AND F1 SCORES FOR EACH CLASS.

CIC-IDS 2017				CSE-CIC-IDS 2018				InSDN			
Target Class	**Train+Val**	**Test**	**F1**	**Target Class**	**Train+Val**	**Test**	**F1**	**Target Class**	**Train+Val**	**Test**	**F1**
BENIGN	2,045,787	227,310	0.99	Benign	12,135,985	1,348,443	0.99	DDoS	109,747	12,195	1.0
DoS Hulk	207,965	23,108	0.99	DDOS attack-HOIC	617,372	68,597	1.0	Probe	88,316	9,813	0.99
PortScan	143,037	15,893	0.99	DDoS attacks-LOIC-HTTP	518,521	57,614	0.99	Normal	61,581	6,843	0.99
DDoS	115,224	12,803	0.99	DoS attacks-Hulk	415,703	46,189	1	DoS	48,254	5,362	0.99
DoS GoldenEye	9,263	1,030	0.99	Bot	257,539	28,616	0.99	BFA	1,265	140	0.90
FTP-Patator	7,144	794	0.99	FTP-BruteForce	174,001	19,334	0.79	Web-Attack	173	19	1.0
SSH-Patator	5,307	590	0.99	SSH-Bruteforce	168,809	18,757	0.99	BOTNET	148	16	1.0
DoS slowloris	5,216	580	0.99	Infiltration	145,737	16,193	0.07	U2R	16	1	1.0
DoS Slowhttptest	4,949	550	0.96	DoS attacks-SlowHTTPTest	125,889	13,987	0.61				
Bot	1,770	196	0.83	DoS attacks-GoldenEye	37,354	4,150	0.99				
Web Attack-Brute Force	1,357	150	0.84	DoS attacks-Slowloris	9,889	1,099	0.99				
Web Attack-XSS	587	65	0.47	DDOS attack-LOIC-UDP	1,556	173	0.85				
Infiltration	33	3	0.8	Brute Force -Web	548	61	0.79				
Web Attack-Sql Injection	19	2	0	Brute Force -XSS	207	23	0.78				
Heartbleed	10	1	1	SQL Injection	79	8	0.79				

arately. The TreeSHAP method, which is a faster variant of the original SHAP framework designed for tree-based algorithms, is chosen to obtain the global explanations from the training data of each dataset. The most important features are visualized using bar charts and shared in Figure 1, 2, and 3. These figures show the top 20 important features and their impact on the target classes for each dataset.

The experiments are performed on several computers with the following specifications: NVIDIA Tesla K80 with 24GB of VRAM, 378 GB of RAM, Intel Xeon E5 2683 processor, and Ubuntu 16.04 operation system.

Evaluation. For quantifying the quality of the classification models, macro averages of the Precision, Recall, F1 score, and False Positive Rate metrics are used. The macro average is produced by calculating the metrics for each class and averaging them. Additionally, F1 score per class will also be provided. Since all datasets have a high class imbalance, it will be significant to discuss the results class-wise, especially for classes with low sample sizes. Furthermore, training and inference times of all ensemble methods are computed to be able to compare them in terms of time efficiency.

V. RESULTS AND DISCUSSION

A. Comparison of Algorithms

Table III presents the performance of different ensemble methods in terms of the macro average of Precision, Recall, F1 score, and False Negative Rate metrics. In the literature, micro-based metrics are generally used for evaluation. For this work, both the macro and micro averages of the metrics are calculated and macro average scores are chosen over micro average scores. The reason for this is that the algorithms perform similarly on micro average metrics, yet the macro average metrics provide more suitable results to evaluate the performance of the algorithms. The table also includes the training and inference times of each method.

The results demonstrate that the XGBoost method performs the best for all datasets based on the F1 macro scores. Additionally, the LightGBM and CatBoost algorithms provide competitive results to the XGBoost. On the contrary, the Random Forest performs successfully in terms of the Precision metric, yet considerably lower on Recall. Since creating false alarms in the intrusion detection domain can be costly, we also monitor the False Positive Rate. According to the results, the LightGBM method shows the lowest False Positive Rate for all datasets, even though it is not the best on other metrics. Additionally, it is clear that LightGBM is the fastest algorithm in terms of training time while XGBoost is significantly slower than all the other algorithms. On the other hand, all methods are close to each other on inference time, except for Random Forest which is notably slow, especially on larger datasets.

As per Table III, the performance of ensemble algorithms varies in different datasets. The reason for this can be explained using Table IV which demonstrates the F1 score from XGBoost models, as well as train and test sizes of each class for all datasets. According to the results, all subclasses of the InSDN are classified successfully with more than 90% F1 score. Conversely, F1 scores of several classes of the CIC-IDS2017 and CSE-CIC-IDS2018 datasets are significantly lower. More importantly, the performances vary for the same attack types in these datasets. For example, the F1 score of the Web Attack in the InSDN dataset is 1. However, the F1 score of Web Attack-Brute Force, Web Attack-XSS, and, Web Attack-SQL Injection are 0.84, 0.47, and 0, respectively in the CIC-IDS2017 dataset. Additionally, in the CSE-CIC-IDS2018 dataset, the results of the web-based attacks such as Brute Force -Web, Brute Force -XSS, and, SQL Injection are around 0.78.

We also compare the ensemble models on different datasets to understand the models' behavior on various networks. We select these datasets based on their up-to-dateness and

collection from different network testbeds such as traditional networks (CIC-IDS2017), cloud networks (CSE-CIC-IDS2018), and software-defined networks (INSDN). When the experimental results are examined, the same attacks can be detected with different success. For example, while the success of finding web-based attacks is quite low in the CIC-IDS2017 dataset, in the InSDN dataset, the XGBoost model can identify web-based attacks perfectly. It can be argued that this performance difference is due to two reasons. Firstly, while the datasets are collected under different testbed architectures, the feature extraction strategy is the same. Secondly, while the attack flows are generated, the differences between the attack tools and the procedures followed during these attacks cause these different results between the datasets.

It is clear from Table IV that the classification performance of several classes is noticeably low, which required further investigation. The confusion matrix of each classifier is examined to see the distribution of the false negatives for the classes with low F1 scores. This investigation showed that the model for the CSE-CIC-2018 dataset classified the Infiltration class samples mostly as Benign. Interestingly, FTP-BruteForce and DoS attacks-SlowHTTPTest instances are often classified as each other. The confusion matrix of the CIC-IDS2017 dataset demonstrates that all false negatives of the Bot class are classified as Benign. It also indicates that the classifier mistake Web Attack-Brute Force and Web Attack-XSS samples for each other.

B. Analysis of SHAP output

SHAP framework is used for improving the transparency of the models, and understanding the relationship between model features and the predictions. For each dataset, the SHAP method is applied separately, and the top 20 most important features are extracted which are illustrated in Figure 1, 2, and 3. The top 5 important features for each dataset can be found in Table V.

Table V
TOP 5 FEATURES FOR EACH DATASET

IDS 2017	IDS 2018	InSDN
Init_Win_bytes_backward	Fwd Seg Size Min	Init Bwd Win Byts
Init_Win_bytes_forward	Init Fwd Win Byts	Bwd Header Len
min_seg_size_forward	Tot Bwd Pkts	Pkt Len Max
Total Length of Fwd Packets	Fwd Pkts/s	Fwd Header Len
Fwd Packet Length Max	Fwd Header Len	Bwd IAT Min

For all of these datasets, statistical time-related flow features are extracted using CICFlowMeter tool [26]. The extracted features can be divided into eight groups such as network-identifiers, byte-based, packet-based, interarrival times, flow timers, flag-based, flow-based, and subflow-based. For the explanations and details of this grouping process, the reader may refer to [24]. The output of the SHAP framework on different datasets reveals that packet-based (total length of fwd packets, fwd packet length max, tot bwd pkts, etc..) and flow-based (init fwd/bwd win byts, fwd seg size, min_seg_size_forward, etc..)

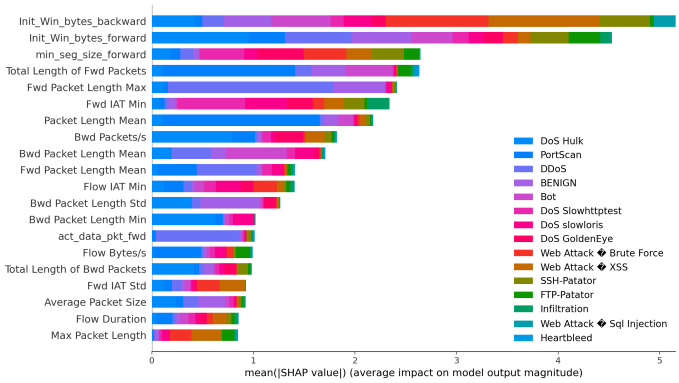

Figure 1. Top 20 important features of the CIC-IDS2017 dataset

features dominate the most important features. An important exception is that the CSE-CIC-IDS2018 dataset has RST Flag Count and ACK Flag Count features as important features, whereas the other datasets don't have any flag-based features. In addition, the features which are related to the interarrival times (IAT) between two packets appear six times in the most important features of InSDN, while appearing less number of times for the other datasets.

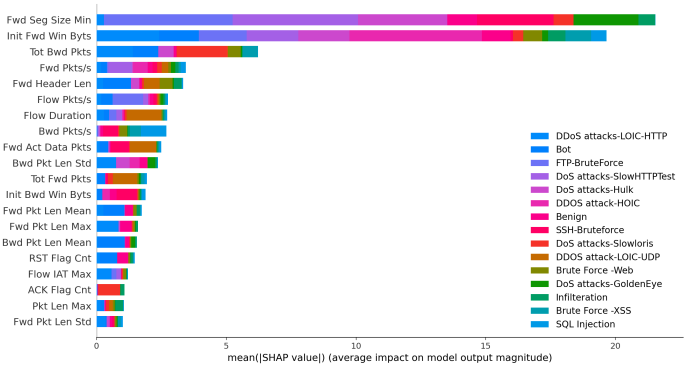

Figure 2. Top 20 important features of the CSE-CIC-IDS2018 dataset

As can be observed from the results, some of the most valuable features are common for all datasets, even though the namings are slightly different. Packet Length Max and Forward Packet Length Max are two of these features which contain the maximum length of a packet and the total size of the packets in forward direction. Init Backward Window Bytes is especially interesting since it is in the top 20 most important features for all datasets and even the most important, for both CIC-IDS2017 and InSDN datasets. Flow Duration and Backward Packets/s are the other two features that are in the top 20 for all datasets.

The SHAP figures also present that there are many impactful features in the CIC-IDS2017 dataset based on the mean SHAP values. Conversely, for the CSE-CIC-IDS2018 and InSDN datasets, there are two features that are by far more impactful

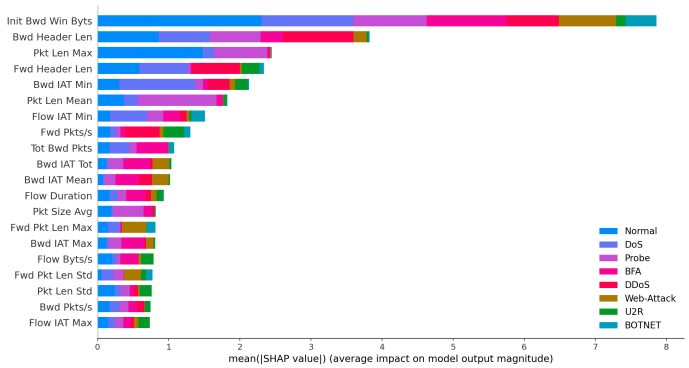

Figure 3. Top 20 important features of the InSDN dataset

than the rest of the features.

VI. Conclusion

Over the last decade, there has been growing attention to predicting anomalies in network intrusion detection systems. In this study, we aim to show the predictive performance of gradient boosting-based models and compare their results on different NIDS datasets. The experimental results show that the XGBoost is the best method based on predictive performance, while if the training time is critical for the user, the LightGBM method should be preferred over the XGBoost. Moreover, we apply the SHAP framework which is used widely by the machine learning community to explain model outputs and investigate how important features are changing based on different NID datasets.

This work can be extended in many ways. First, we can extend the number of experiments with deep learning-based algorithms such as fully connected neural networks, convolutional neural networks, or TabNet [27]. Secondly, we can make deeper analyses of SHAP outputs and compare the important features based on target classes. Finally, another line of research could be on domain adaptation for NID datasets. Based on the SHAP framework results, the important features and the feature space can vary for different networks. This makes it difficult to train a common model for different NIDS datasets. Learning domain-invariant feature representations can provide accurate classification results for both the source and the target datasets.

References

[1] I. H. Sarker, A. Kayes, S. Badsha, H. Alqahtani, P. Watters, and A. Ng, "Cybersecurity data science: an overview from machine learning perspective," *Journal of Big data*, vol. 7, no. 1, pp. 1–29, 2020.

[2] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, and S. Venkatraman, "Robust intelligent malware detection using deep learning," *IEEE Access*, vol. 7, pp. 46 717–46 738, 2019.

[3] P. Dixit and S. Silakari, "Deep learning algorithms for cybersecurity applications: A technological and status review," *Computer Science Review*, vol. 39, p. 100317, 2021.

[4] A. Ibrahim, D. Thiruvady, J.-G. Schneider, and M. Abdelrazek, "The challenges of leveraging threat intelligence to stop data breaches," *Frontiers in Computer Science*, vol. 2, p. 36, 2020.

[5] M. Premkumar and T. Sundararajan, "Dldm: Deep learning-based defense mechanism for denial of service attacks in wireless sensor networks," *Microprocessors and Microsystems*, vol. 79, p. 103278, 2020.

[6] M. Lansley, N. Polatidis, S. Kapetanakis, K. Amin, G. Samakovitis, and M. Petridis, "Seen the villains: Detecting social engineering attacks using case-based reasoning and deep learning." in *ICCBR Workshops*, 2019, pp. 39–48.

[7] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions," *Journal of big Data*, vol. 8, no. 1, pp. 1–74, 2021.

[8] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[9] H. Zhu, M.-D. Luo, R. Wang, A.-H. Zheng, and R. He, "Deep audio-visual learning: A survey," *International Journal of Automation and Computing*, vol. 18, no. 3, pp. 351–376, 2021.

[10] S. Pande, A. Khamparia, D. Gupta, and D. N. Thanh, "Ddos detection using machine learning technique," in *Recent Studies on Computational Intelligence*. Springer, 2021, pp. 59–68.

[11] M. Suresh and R. Anitha, "Evaluating machine learning algorithms for detecting ddos attacks," in *International Conference on Network Security and Applications*. Springer, 2011, pp. 441–452.

[12] X. Yuan, C. Li, and X. Li, "Deepdefense: identifying ddos attack via deep learning," in *2017 IEEE International Conference on Smart Computing (SMARTCOMP)*. IEEE, 2017, pp. 1–8.

[13] R. Shwartz-Ziv and A. Armon, "Tabular data: Deep learning is not all you need," *Information Fusion*, vol. 81, pp. 84–90, 2022.

[14] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

[15] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," *Advances in neural information processing systems*, vol. 31, 2018.

[16] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, 2017.

[17] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[18] D. Gunning, M. Stefik, J. Choi, T. Miller, S. Stumpf, and G.-Z. Yang, "Xai—explainable artificial intelligence," *Science Robotics*, vol. 4, no. 37, p. eaay7120, 2019.

[19] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[20] M. T. Ribeiro, S. Singh, and C. Guestrin, "" why should i trust you?" explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.

[21] S. Mane and D. Rao, "Explaining network intrusion detection system using explainable ai framework," *arXiv preprint arXiv:2103.07110*, 2021.

[22] M. Wang, K. Zheng, Y. Yang, and X. Wang, "An explainable machine learning framework for intrusion detection systems," *IEEE Access*, vol. 8, pp. 73 127–73 141, 2020.

[23] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." *ICISSp*, vol. 1, pp. 108–116, 2018.

[24] M. S. Elsayed, N.-A. Le-Khac, and A. D. Jurcut, "Insdn: A novel sdn intrusion dataset," *IEEE Access*, vol. 8, pp. 165 263–165 284, 2020.

[25] A. Anghel, N. Papandreou, T. Parnell, A. De Palma, and H. Pozidis, "Benchmarking and optimization of gradient boosting decision tree algorithms," *arXiv preprint arXiv:1809.04559*, 2018.

[26] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, and A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related," in *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*, 2016, pp. 407–414.

[27] S. O. Arık and T. Pfister, "Tabnet: Attentive interpretable tabular learning," in *AAAI*, vol. 35, 2021, pp. 6679–6687.