Parameter Optimization and Performance Analysis of State-of-the-Art Machine Learning Techniques for Intrusion Detection System (IDS)

Rashedun Nobi Chowdhury*, Maliha M. Chowdhury†, Sujoy Chowdhury‡,
Mohammed Rashedul Islam§, Md. Ahsan Ayub¶,
Anik Chowdhury‖, and Kazi A Kalpoma**
Department of Computer Science
Ahsanullah University of Science and Technology, Dhaka, Bangladesh
Tennessee Tech University, Cookeville, TN, USA¶
Emails: {160204039*, 160204043†, 160204048‡, 160204088§, kalpoma**}@aust.edu,
mayub42@tntech.edu¶, anikchy2140@gmail.com‖

2020 23rd International Conference on Computer and Information Technology (ICCIT) | 978-1-6654-2244-4/20/$31.00 ©2020 IEEE | DOI: 10.1109/ICCIT51783.2020.9392683

***Abstract*—The use of Intrusion Detection System (IDS) as one of the most trusted layers of security to an organization to defend against all sorts of cyber attacks is ubiquitous. The uniqueness as well as the severity of recent cyber threats is significant. The ability of state-of-the-art machine learning techniques to discover new types of malicious network packets has been explored in many research works in the past. In our study, we present an experimental analysis to empirically find the optimal parameter settings of 22 classification techniques from 9 machine learning families. To investigate the studied algorithms, we choose 4 widely used IDS datasets that resemble real-world network traffic for both benign and malicious activities. We conduct experiments on binary classification, i.e., given an IDS log, we predict whether the log is either benign or malicious, on such datasets. Moreover, we take a step further to analyze such algorithms' performances on multiclass classification; for example, we aim to detect the type of the attack from each dataset. As similar work in this domain is very limited to none, our study will play a vital role to select the optimal parameter settings for a certain studied classification algorithm.**

***Index Terms*—Intrusion Detection System, Machine Learning, Parameter Optimization**

I. Introduction

With the ever growing demand of secure network infrastructure, ensuring privacy from both external and inside entity is becoming a daunting task. Compromise of security policies, *i.e.*, *Confidentiality*, *Integrity* and *Availability* (CIA) in a small to large business organization is at a significant risk because of the cyber threats – Denial of Service (DoS) attacks, phishing, web attacks, and malware to name a few. In order to prevent such security risks as well as protect vulnerable assets from outside network, companies incorporate different types of security measures, such as, firewalls, two factor authentication, antivirus software, etc. However, we observe, Intrusion Detection Systems (IDS) have shown an effective network defensive mechanism as they provide network defenders alerts when a suspicious or malicious activity is observed [1], [2].

To further describe, intrusion detection is the process of monitoring the events that occur in a computer system or a network. It then performs a continuous analysis of such events to search the signs of intrusions. An intrusion detection system is a piece of software or hardware device that automates this process of intrusion detection as well as helps finding the intruder [3]. An IDS alerts the network administrator in the event of malicious activity both from the inside and the outside of the network. It is regarded as 'burglar alarms' of the computer security field [4]. IDS can mainly be divided into two categories based on the resources they monitor: (1) Host based Intrusion Detection System (HIDS) – it relies on events collected by the hosts they monitor [5]; and (2) Network based Intrusion Detection System (NIDS) – it monitors the activity in the whole network. NIDS can ensure the security of the whole network of the organization whereas HIDS is concerned with only the device it is installed on. Based on the detection approaches, IDS can be of two types - Signature based Intrusion Detection and Anomaly based Intrusion Detection [4]. Signature based IDS maintains a database of known intrusions and flags any events that are similar. They are only able to detect the attacks known to them. Anomaly based IDS, on the contrary, starts with a performance baseline of normal behavior and then compares network traffic against the baseline. IDS sends an alert when traffic differs significantly from the baseline [6]. Therefore, anomaly based IDS is able to detect new types of attacks when discovered.

An important part of building an Anomaly based IDS is to determine how these systems should be trained, *i.e.*, how it should define a normal network behavior and how this behavior should be represented computationally [7]. One of the most unique features of Machine Learning (ML) is, its ability to detect variants of new cyber attacks, because of which ML techniques are widely applied in IDS to strengthen its detection capability. We notice, security researchers in both academic and industry have used different kinds of Machine Learning models for their analysis on IDS, such as, Decision Tree (DT) [8], K-Nearest Neighbors (KNN) [9], Support Vector Machine (SVM) [10], Ensemble Learning techniques, *e.g.*, Random Forests (RF) [11], Adaptive Boosting (AdaBoost)

978-1-6654-2244-4/20/$31.00 ©2020 IEEE

TABLE I
OVERVIEW OF THE STUDIED MACHINE LEARNING MODELS

Family	Family Description & Techniques	Tuned Parameter
Naive Bayes	Naive Bayes methods apply Bayes' theorem with the "naive" assumption of conditional independence between every pair of features given the value of the class variable. **Techniques:** Gaussian Naive Bayes, Bernoulli Naive Bayes, Multinomial Naive Bayes, Complement Naive Bayes.	• Additive (Laplace/Lidstone) smoothing parameter, alpha
Nearest Neighbor	Nearest Neighbor methods find a predefined number of training samples closest in distance to the new point, and predict the label from these. **Techniques:** K Nearest Neighbor (KNN), Nearest Centroid.	• Number of neighbors • Algorithm used to compute nearest neighbors
Discriminant Analysis	Discriminant Analysis applies different kernel functions to classify a set of observations. **Techniques:** Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA).	• Absolute threshold for a singular value of X to be considered significant
Tree Based	Tree Based models predict the value of a target variable by learning simple decision rules inferred from the data features. **Techniques:** Classification And Regression Trees (Decision Tree).	• Function to measure the quality of a split • Minimum no of samples to split an internal node
Support Vector Machine	Support Vector Machine (SVM) finds a hyperplane in an N-dimensional space (N number of features) to classify a data point. **Techniques:** Linear SVM, Kernel SVM.	• Regularization parameter • No of iterations • Type of kernel
Boosting	Boosting models create a strong classifier from a number of weak classifiers by adjusting weights, optimizing differentiable loss functions etc. **Techniques:** Adaptive Boosting (AdaBoost), Gradient Boosting, eXtreme Gradient Boosting (XGBoost), Histogram Based Gradient Boosting (HistGradBoost), Light Gradient Boosted Machine (LGBM), Category Gradient Boosting(CatBoost).	• Learning rate • No of estimators
Linear Model	Linear Model is specified as a linear combination of features, and based on training data, the learning process computes one weight for each feature to form a model that predicts the target value. **Techniques:** Stochastic Gradient Descent (SGD), Logistic Regression.	• No of epochs • Stopping criterion • Loss function
Bagging	Bagging considers homogeneous weak learners, learns them independently from each other in parallel, and combines them following some kind of deterministic averaging process. **Techniques:** Random Forest(RF), Bagging Ensemble	• No of base estimators • No of features
Rotation Forest	Rotation Forest generates classifier ensembles based on feature extraction. **Techniques:** Rotation Forest	• Function to measure the quality of a split • Minimum no of samples to split a node

[12] and eXtreme Gradient Boosting (XGB) [13]. During our literature survey, we find, research work that studied different ML algorithms that the mentioned ones here as well as presented survey work [14], [15]. Popular datasets among the IDS researchers include KDD-99 [16], NSL-KDD [17], DARPA [18], CIC-IDS2017 [19], etc. Hindy et al. [20] surveyed prominent IDS research papers from 2008 to 2018 and found out that 63.8% of them used KDD 99 and 11.6% used NSL-KDD. As datasets play a vital role in building and training a ML classifier, we notice, there are issues with older datasets as they often do not reflect current network trends. Additionally, the use of unbalanced dataset results in biased training. Once the datasets are acquired by any means, researchers can choose to use a suite of the ML algorithms that are widely available through open source software, such as, WEKA [21], scikit-learn [22], etc. It is to note that the offered ML models offer configurable hyperparameters, and the default parameter setting may not always be optimal [23]. Fine tuning parameter setting to find an optimal setting is one of the most important tasks to gain better performances of such stated algorithms. Tantithamthavorn et al. [24] conducts a study on hyperparameter settings for defect predictive models and finds that optimized hyperparameters improve performance of algorithms by up to 40 percent. To the best of our knowledge, there is no prior work similar to this in IDS.

The major contributions of our paper are as follows:

- We perform a detailed experimental analysis on 9 machine learning families algorithms suite (see Table I) to empirically find the optimal parameter settings for each studied classifier in order to detect malicious traffic;
- We first perform binary classification tasks on such algorithms, *i.e.*, given a network log, we predict whether the log is benign or malicious. Then, we take a step further and perform multiclass classification on the studied IDS datasets, *i.e.*, given a malicious network log, we aim to detect what type of attack it represents. This strengthens the effectiveness of our study as it presents the capabilities of some of the state-of-art machine learning

techniques; and
- We analyze every built machine learning model's performance on 4 widely used IDS datasets to ensure we comprise a diverse example sets of network traffic and hence we provide a comparison chart which will help researchers in this realm select which classification techniques work better after certain parameter tuning.

The rest of the paper is organized as follows: Section 2 discusses different machine learning algorithms. Section 3 contains information about the datasets we used while section 4 demonstrates our working methodologies and experiment process. In section 5, we illustrate the results of our studies and discuss our research findings, followed by section 6, where we mention some of the key relevant work. Finally, section 7 summarizes the paper, its contributions and future work to improve further upon this research.

II. Background

In this section, we describe 22 different Machine Learning models that we have studied in order to present a comparative analysis on their performances. Our aim is to perform experimentations on every studied algorithm's default parameter setting and explore parameters it offers to tune. Thus, we illustrate the background information of our experimented classification techniques as well as configurable settings we account to perform tuning on in Table I.

III. Studied Datasets

Finding the right datasets to address the problem using machine learning techniques is one of the major tasks as this ensures the model(s) we train and test, is not biased to perform a certain set of tasks. We seek to find supervised datasets to train and evaluate Anomaly based IDS. Over the course of last decade, there have been many IDS datasets generated by researchers. When selecting a dataset for IDS, properties such as year of creation, public availability, and how much the dataset reflects on real-world network behavior are considered. Ring et al. [25] performed a comprehensive overview of 34 IDS datasets, many of which are publicly available. They mention that IDS datasets appear in 3 formats: (1) Packet based network traffic contains network traffic with payload; (2) Flow-based network traffic contains nothing but the meta information about network connections; and (3) Others, which may be flow-based datasets that have been enriched with additional information from packet based data. For our experiment, we select 4 publicly available datasets with at least 1 from each of the mentioned domains. The following is the descriptions of the datasets:

CIC-IDS 2017 [19]: CIC-IDS 2017 is one of the recommended datasets in [25]. The dataset contains benign and mostly common cyber-attack vectors that resemble real-world network applications. It has more than 80 features and contains all common protocols such as HTTP, HTTPS, FTP, SSH, and email protocols. The dataset also covers all 11 necessary criteria for a dataset according to the IDS dataset evaluation framework published in 2016 [26]. The dataset can be placed in both category 1 and 2 as it contains network traffic in both packets based and bidirectional flow based format [25].

TABLE II
Overview of the Studied Datasets

Dataset	*Attack Class*	*Types of Attacks*
CICIDS-2017	**Brute Force**	FTP-Patator, SSH-Patator
	DoS	DoS slowloris, DoS Slowhttptest, DoS Hulk, DoS GoldenEye, Heartbleed
CSE-CIC-IDS2018	**Web Attack**	Web Attack-Brute Force, Web Attack-XSS, Web Attack-Sql Injection, Infiltration, Botnet, Port Scan
	DDoS	(DDoS LOIT, DDoS-LOIC-UDP, DDoS-LOIC-HTTP, DDoS-HOIC
	Infiltration, Botnet, Port Scan	
TRAbID-2017	**DoS**	
UNSW-NB15	**Analysis, Backdoor, DoS, Exploits, Fuzzers, Reconnaissance, Generic, Shellcode, Worm**	

CIC-IDS 2018 [19]: CIC-IDS 2018 dataset is quite similar to CIC-IDS 2017. It contains a few more types of attacks compared to CIC-IDS2017.

TRAbID [27]: The TRAbID dataset was released in 2017 for public use. The dataset contains network traffic data in packet based format. Network services include HTTP, SMTP, SSH, SNMP, NTP, and DNS. TRAbID contains two types of attack categories, Probing and DoS attacks. We work with only DoS attacks for this dataset. The dataset falls into category 1.

UNSW-NB15 [28]: UNSW-NB15 is another recommended dataset in [25]. The dataset contains 49 features that comprise the flow based between two hosts and packet headers. It has a predefined train set and test set. It falls into category 3.

In Table II, we present an overview of the types of attack mapped to their respective classes for all four datasets.

IV. Experimental Methodology

In this section, we explain the methodologies we apply to conduct our experimentations. The steps we take to effectively perform a comparative study of a set of state-of-the-art machine learning algorithms are as follows:

Dataset Sampling: All the four datasets we use for our experiments are imbalanced. He and Gracia [29] showed that using imbalanced data to train a model generated poor performance results. Thus, we randomly sample all 4 datasets in order to make them balanced. Before sampling and following the common practice of ML developers, we drop all the observations that contain null and infinite values. Then, we remove the IP address columns from the datasets if present, as we do not aim to detect any analysis on the device from which a network packet might be arrived or sent.

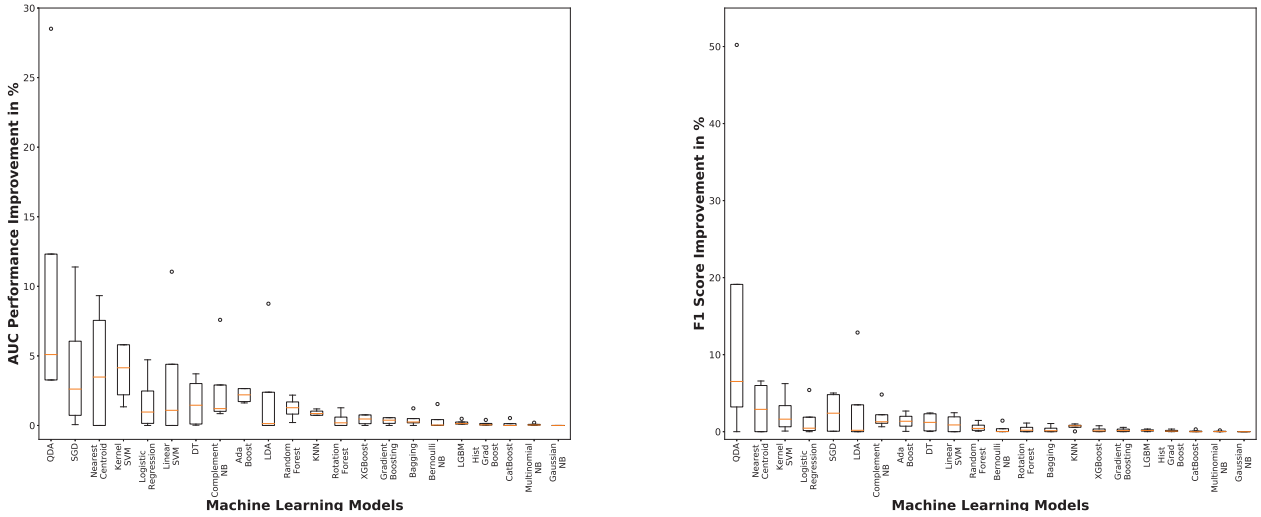

Fig. 1. Distribution of AUC score improvement (left side) and F_1 score improvement (right side) for binary classification

Followed by, we take only one instance of duplicate data and drop the others. Finally, we use random sampling from the pandas library[1] to process the same percentage of data for each class and prepare one sampled dataset for binary class classification as well as one for multi class classification.

Data Preprocessing: We employ dummy variables from pandas to encode categorical variables. To encode dependent variables, we utilize LabelEncoder from scikit learn [22]. Finally, we use the MinMax scaler to scale the datasets so that all the feature values range between 0 and 1 [22].

Building Machine Learning Models: After processing the datasets successfully, we split the dataset into 75% training set and 25% testing set. Once we prepare the training and testing set, with the incorporation of five-fold cross validation, we run all the 22 classification algorithms (mentioned in Table 1) one by one for compilation. After successful compilation, we record the models' prediction performance with the testing set. For every case, we analyze the performance of the models using 5 evaluation metrics: (1) Accuracy, which defines the overall effectiveness of a classifier; (2) Precision, which is a ratio of true positive instances to the positively labeled instances; (3) Recall, which identifies the effectiveness of a classifier to identify positive labels; (4) F_1 Score, the harmonic mean of recall and precision scores; and (5) AUC score, which means the classifier's ability to avoid false classification (only for binary classification) [30].

$$Precision\ Score = \frac{\sum True\ Positive}{\sum True\ Positive + \sum False\ Positive}$$

$$Recall\ Score = \frac{\sum True\ Positive}{\sum True\ Positive + \sum False\ Negative}$$

$$F1\ Score = 2 \cdot \frac{Precision\ Score \times Recall\ Score}{Precision\ Score + Recall\ Score}$$

[1]https://pandas.pydata.org/

$$AUC\ Score = 1 - \frac{False\ Positive + False\ Negative}{2}$$

Hyperparameter Tuning: For tuning the hyperparameters for each of the models, we use GridSearchCV from scikit-learn with cross validation value of 5 and *accuracy* scoring metric [22].

V. Results

In this section, we describe our experimental findings in the following two categories to best illustrate our results.

A. Binary Classification

We begin our experiments by analyzing all the studied machine learning families for binary classification. For this experiment phase, we aim to detect two types of classes: benign and malicious for given a set of independent data observations – in other words, network traffic. After performing each classification technique from the families, we observe the classifiers from the *Boosting Family* outperform others for all the datasets we used. We primarily focus on AUC performance as well as F_1 scores to evaluate our built binary classifiers. We notice, the Light Gradient Boosted Machine (LGBM), Gradient Boosting, and Histogram based Gradient Boosting classifiers achieve an AUC score higher than 88% across all the datasets. Apart from the Boosting Family, we observe Random Forests classifier shows a really high AUC score, that is, more than 87%. We illustrate Fig. 1 to present the distribution of AUC and F_1 scores improvement that will enable readers to visualize the overall performance comparison.

B. Multiclass Classification

We repeat the similar approach of our experimentations to generate the results for multiclass classification. As mentioned, we used 3 datasets out of 4 datasets as TRAbID dataset offers only DoS attacks and Benign data. For other datasets, the number of classes / attack types varies (as shown in Table III). We consider Precision, Recall and F_1 scores to evaluate the classifiers' performance. Interestingly, we see

TABLE III
COMPARISON CHART OF THE STUDIED MACHINE LEARNING MODELS' PERFORMANCE FOR MULTICLASS CLASSIFICATION

Family	Algorithm	CIC-IDS2017				CIC-IDS2018				UNSW-NB15			
		Default		Optimized		Default		Optimized		Default		Optimized	
		Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
Naive Bayes	Gaussian Naive Bayes	94.27	94.97	94.27	94.97	64.59	66.13	64.59	66.13	24.35	26.28	24.35	26.28
Nearest Neighbor	KNN	95.00	94.98	97.49	99.01	89.81	90.11	90.03	90.35	38.24	45.91	40.18	48.47
Discriminant Analysis	QDA	67.97	73.59	98.51	87.91	79.19	73.51	87.27	86.18	27.13	28.65	40.06	39.11
Tree Based	Decision Tree	99.43	99.41	99.50	99.50	88.24	87.83	88.57	87.95	43.62	57.91	47.93	58.43
Support Vector Machine	SVC	89.59	88.89	99.14	93.86	82.36	85.39	88.07	87.38	41.60	43.92	38.36	46.98
Boosting	LGBM Classifier	99.70	97.93	99.72	99.72	92.47	92.09	92.53	92.12	49.96	64.12	49.96	64.12
Linear Model	Logistic Regression	99.45	97.68	99.72	99.72	88.51	87.63	89.49	89.01	42.54	53.57	45.49	55.83
Bagging	Random Forest	99.37	99.37	99.70	99.69	88.95	88.79	89.43	89.15	46.13	58.86	46.16	59.73
Rotation Forest	Rotation Forest	91.79	89.26	95.59	98.26	83.22	84.36	83.22	84.36	38.50	45.75	38.16	44.25

similar results as binary classifiers as classifiers from Boosting Family outperform others here as well. Models from most of the family show precision and recall score above 80% for CIC-IDS2017 and CIC-IDS2018 with KNN and LGBM classifier achieving scores above 90% after proper parameter setting. However, we find that models from all families perform poorly on the UNSW-NB15 dataset. LGBM classifier is the best performer with a precision score of 49.96% and recall score of 64.12%. We illustrate Fig. 2 to present the distribution of F_1 score improvements.

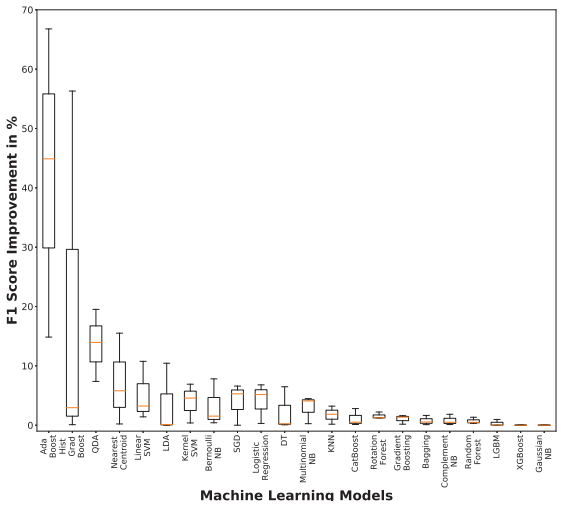

Fig. 2. Distribution of F_1 score improvement for multiclass classification

VI. RELEVANT WORK

Supervised machine learning has been widely used in the world of Intrusion Detection System. This is large because Machine Learning has the ability to detect new cyber attacks thus strengthening the detection capacity of IDS. Tsai et al. [14] studied 55 papers from 2000 to 2007 and found that most of the papers used either single classifiers or hybrid classifiers. KDD-99 [16] is the most used dataset in this timeframe. Among the many criticisms faced by the dataset, the primary one is that it is not an authentic simulation of real network traffic [31]. Different ML models show varying performances across IDS. Haq et al. [15] performed a statistical comparison of classifier design type, chosen algorithms, and results of papers between 2009 and 2014. Based on these survey works, we find that most researchers focus on one dataset and try to propose new algorithms to achieve better performances. One important aspect of training a machine learning model is setting hyperparameters. As Bergstra et al. [23] points out, all ML models have configurable hyperparameters and actual learning algorithms can be obtained after proper hyperparameter setting. Song et al. [32] study the impact of parameter tuning for Software Effort Estimation (SEE). They point out that different ML algorithms have different sensitivity to their parameter settings. For example, bagging classifiers are quite sensitive to parameter settings while others like Regression Trees (RT) are not.

We demonstrate the impact of hyperparameter optimization for Anomaly based IDS; however, there are other fields in computer science which has seen similar works, such as, defect predictive model [24], software effort estimation [32], image recognition [33], natural language processing [34]. This research adds IDS to the list where research on hyperparameter optimization has been very limited.

VII. CONCLUSION

Incorporation of the Intrusion Detection System (IDS) to the network infrastructure of small to large business organization has been playing a pivotal role to defend against various types of cyber attacks. As adversaries adapt intuitive techniques to

bypass different defense mechanisms, *i.e.*, crafting legitimate network packet(s) using adversarial machine learning technique, further intelligence on top of IDS has become much desired in modern days. To combat this situation, researchers attempted utilizing a variety set of state-of-the-art machine learning techniques; however, we notice, the study of exploring optimal parameter settings of such configurable machine learning algorithms is very limited to none. With experiments conducted for 22 ML algorithms from 9 machine learning families over 4 widely used IDS datasets, we find:

- Light Gradient Boosted Machine (LGBM) Classifier is one of the best algorithms to detect malicious traffic through IDS logs;
- 17 algorithms (77%) out of 22 show noticeable performance improvement after proper parameter settings; and
- The difference in performance before and after parameter tuning can be upto 28% for Binary classification and 66% for multiclass classifiers.

We are highly optimistic, our work will be helpful for anyone who wants to build a machine learning based approach to effectively detect network traffic through Anomaly based Intrusion Detection System. We acknowledge that the adversaries will still be finding new ways to bypass such defensive schemes as our approaches do not guarantee a flawless detection rate. With that keeping in mind, our future work includes gathering more classification techniques in our algorithm suite, leveraging deep learning approaches to learn from the dataset without performing feature extractions, and evaluating the models' performances in a real-time setting.

REFERENCES

[1] H. Debar, M. Dacier, and A. Wespi, "Towards a taxonomy of intrusion-detection systems," *Computer networks*, vol. 31, no. 8, pp. 805–822, 1999.

[2] M. A. Ayub, W. A. Johnson, D. A. Talbert, and A. Siraj, "Model evasion attack on intrusion detection systems using adversarial machine learning," in *2020 54th Annual Conference on Information Sciences and Systems (CISS)*, pp. 1–6, IEEE, 2020.

[3] R. G. Bace and P. Mell, "Intrusion detection systems," 2001.

[4] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," tech. rep., Technical report, 2000.

[5] H. Bidgoli, *Handbook of Information Security, Threats, Vulnerabilities, Prevention, Detection, and Management*, vol. 3. John Wiley & Sons, 2006.

[6] D. Gibson, "Comptia security+ get certified get ahead: Sys0-301 study guide," *North Charleston, SC: CreateSpace*, 2011.

[7] S. K. Wagh, V. K. Pachghare, and S. R. Kolhe, "Survey on intrusion detection system using machine learning techniques," *International Journal of Computer Applications*, vol. 78, no. 16, 2013.

[8] O. Depren, M. Topallar, E. Anarim, and M. K. Ciliz, "An intelligent intrusion detection system (ids) for anomaly and misuse detection in computer networks," *Expert systems with Applications*, vol. 29, no. 4, pp. 713–722, 2005.

[9] Y. Liao and V. R. Vemuri, "Use of k-nearest neighbor classifier for intrusion detection," *Computers & security*, vol. 21, no. 5, pp. 439–448, 2002.

[10] C. A. Catania, F. Bromberg, and C. G. Garino, "An autonomous labeling approach to support vector machines algorithms for network traffic anomaly detection," *Expert Systems with Applications*, vol. 39, no. 2, pp. 1822–1829, 2012.

[11] N. Farnaaz and M. Jabbar, "Random forest modeling for network intrusion detection system," *Procedia Computer Science*, vol. 89, no. 1, pp. 213–217, 2016.

[12] T. P. Tran, L. Cao, D. Tran, and C. D. Nguyen, "Novel intrusion detection using probabilistic neural network and adaptive boosting," *arXiv preprint arXiv:0911.0485*, 2009.

[13] S. S. Dhaliwal, A.-A. Nahid, and R. Abbas, "Effective intrusion detection system using xgboost," *Information*, vol. 9, no. 7, p. 149, 2018.

[14] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, "Intrusion detection by machine learning: A review," *expert systems with applications*, vol. 36, no. 10, pp. 11994–12000, 2009.

[15] N. F. Haq, A. R. Onik, M. A. K. Hridoy, M. Rafni, F. M. Shah, and D. M. Farid, "Application of machine learning approaches in intrusion detection system: a survey," *IJARAI-International Journal of Advanced Research in Artificial Intelligence*, vol. 4, no. 3, pp. 9–18, 2015.

[16] "Uci kdd archive.." [Online; accessed 25-July-2020].

[17] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE symposium on computational intelligence for security and defense applications*, pp. 1–6, IEEE, 2009.

[18] "Darpa intrusion detection evaluation - mit lincoln laboratory.." [Online; accessed 25-July-2020].

[19] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization.," in *ICISSP*, pp. 108–116, 2018.

[20] H. Hindy, D. Brosset, E. Bayne, A. Seeam, C. Tachtatzis, R. Atkinson, and X. Bellekens, "A taxonomy and survey of intrusion detection system design techniques, network threats and datasets," *arXiv preprint arXiv:1806.03517*, 2018.

[21] "Weka 3 - data mining with open source machine learning software in java.." [Online; accessed 25-July-2020].

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[23] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Advances in neural information processing systems*, pp. 2546–2554, 2011.

[24] C. Tantithamthavorn, S. McIntosh, A. E. Hassan, and K. Matsumoto, "Automated parameter optimization of classification techniques for defect prediction models," in *Proceedings of the 38th International Conference on Software Engineering*, pp. 321–332, 2016.

[25] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Computers & Security*, vol. 86, pp. 147–167, 2019.

[26] A. Gharib, I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "An evaluation framework for intrusion detection dataset," in *2016 International Conference on Information Science and Security (ICISS)*, pp. 1–6, IEEE, 2016.

[27] E. K. Viegas, A. O. Santin, and L. S. Oliveira, "Toward a reliable anomaly-based intrusion detection in real-world environments," *Computer Networks*, vol. 127, pp. 200–216, 2017.

[28] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 military communications and information systems conference (MilCIS)*, pp. 1–6, IEEE, 2015.

[29] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[30] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[31] J. McHugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory," *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, no. 4, pp. 262–294, 2000.

[32] L. Song, L. L. Minku, and X. Yao, "The impact of parameter tuning on software effort estimation using learning machines," in *Proceedings of the 9th international conference on predictive models in software engineering*, pp. 1–10, 2013.

[33] A. O. Restrepo Rodríguez, D. E. Casas Mateus, P. A. Gaona García, C. E. Montenegro Marín, and R. González Crespo, "Hyperparameter optimization for image recognition over an ar-sandbox based on convolutional neural networks applying a previous phase of segmentation by color–space," *Symmetry*, vol. 10, no. 12, p. 743, 2018.

[34] H. Caselles-Dupré, F. Lesaint, and J. Royo-Letelier, "Word2vec applied to recommendation: Hyperparameters matter," in *Proceedings of the 12th ACM Conference on Recommender Systems*, pp. 352–356, 2018.