Vol. 5, No. 2 | July – December 2022

Evaluation of Machine Learning based Network Attack Detection

Muhammad Awais Rajput[1], Muhammad Umar[2], Adnan Ahmed[2], Ali Raza Bhangwar[3], Khadija Suhail Memon[2], Misbah[2]

Abstract:

The growth in the internet and communication technologies has driven tremendous developments in various application areas such as smart cities, cloud computing, internet-of-things, e-banking, e-commerce, and e-government. However, the advancements in networking infrastructure, hacking tools, and methodologies have enabled hackers to attempt newer and more complicated cyber-attacks. Consequently, cyber-security has now emerged as a vital research area to address security concerns. Traditional security mechanisms such as firewalls and anti-viruses are not enough to accurately detect intrusions. Therefore, an Intrusion Detection System (IDS) provides an additional layer of security to prevent intrusions through continuous surveillance of the network traffic. Machine Learning (ML) and Deep Learning (DL) techniques have been exploited to overcome the inherent deficiencies of IDS such as accurately detecting intrusions, countering zero-day cyber-attacks, and reducing false positive rates. Existing research has demonstrated that ML and DL-based techniques can efficiently detect patterns (features) from the network traffic and predict the behavior (normal or abnormal activity) based on these patterns. This research work first presents the concepts of IDS, followed by a comprehensive review of the recent ML and DL-based schemes. Later, a performance analysis of various ML algorithms such as Decision Trees (DT), Random Forest (RF), Gradient Booster (GB), and Deep Neural Networks (DNNs) is presented on a publicly available dataset. The performance is reported in terms of accuracy, F1-Score, cross-entropy loss, and training and testing times.

Keywords: *machine learning; intrusion detection systems; cyber security; anomaly detection*

1. Introduction

The recent advancement in emerging technologies such as healthcare, telecommunication, education, intelligent transportation systems, smart grids, e-commerce and e-government, manufacturing and infotainment provide ease and quality of services to the people [1], [2]. Despite the significant advantages, these technologies are exposed to numerous cyber threats and unauthorized access [3]. To mitigate and eliminate the impacts of these attacks, cybersecurity provides various methods and technologies, such as antivirus, encryption/decryption, firewalls, access control and IDS, to protect data and network infrastructures [4]–[6]. Although these methods can prevent various attacks, however, in-depth traffic analysis cannot be performed using these security techniques.

[1]Dept. of Artificial Intelligence, Quaid-e-Awam University of Engineering, Science and Technology, Nawabsah, Pakistan
[2]Dept. of Telecommunication, Quaid-e-Awam University of Engineering, Science and Technology, Nawabsah, Pakistan.
[3]Dept. of Computer Systems, Quaid-e-Awam University of Engineering, Science and Technology, Nawabsah, Pakistan.
Corresponding Author: adnan.ahmed03@quest.edu.pk

For continuous network traffic surveillance, IDS have been developed to perform deeper traffic analysis. However, the IDS still poses challenges in detecting zero-day attacks and minimizing false alarm rates [7]. To overcome the limitations of existing IDS schemes and to provide accurate, cost-effective and efficient IDS, machine learning and deep learning techniques have been integrated to learn network traffic's features and then predict and distinguish between the benign and normal traffic patterns. In this paper, we evaluate the performance of machine learning algorithms such as Decision Trees (DT), Random Forest (RF), Gradient Booster (GB) and Deep Neural Networks (DNNs) on various cyber security attacks: Denial of Service (DoS), Remote to Local (R2L), User to Root (U2R) and probe. The work was tested with the publicly available dataset KDD Cup99 and performance was evaluated in terms of accuracy and training time.

The contributions of this paper are highlighted in the following:

1) A machine learning-enabled attack detection flow is implemented.
2) The performance of various machine learning algorithms including deep neural networks is evaluated on a publicly available dataset.

The rest of the paper is organized as follows: Section 2 presents the relevant literature and discusses the ML and DL-based IDS schemes. Section 3 presents the methodology and overall flow of the work. Section 4 presents the results and discussion. Finally, section 5 concludes the paper.

2. Literature Review

The intrusion detection scheme based on linear discriminant analysis (LDA), classification and Regression trees CART has been used in [8]. For testing purpose, Random Forest and KDD Cup 99 dataset has been used in manner that it is divided on 80-20 rule. The performance of IDS schemes (LDA, RF and CART) is evaluated in terms of accuracy and Cappa. According to result analysis, the RF scheme performs better in accuracy (99.65%) than LDA (98.1%) and CART (98%).

In paper [9], a performance comparison of machine learning-based algorithms, KNN, decision tree and AdaBooost, is performed using TON-IoT data set. The 99.8% accuracy was achieved for AdaBoost scheme which was better than KNN and decision tress schemes.

The Intrusion Detection Tree (IntruDTree) machine-learning-based security model was proposed in [10], which predicts the unseen test cases accurately and also reduces the computational complexity by optimizing feature dimensions. The efficacy of proposed IntruDTree was examined with cybersecurity datasets and performance was measured in terms of ROC, accuracy, recall and precision. The performance of IntruDTree was compared with Naive Bayes (NB), Logistic Regression (LR), K-Nearest Neighbor (KNN), and Support Vector Machines (SVM). The IntruDTress achieved better results than their counterpart.

The paper [11] presented the variant of the FNN known as Self-normalizing Neural Networks (SSN) and compare its performance with the FNN. The evaluation was implemented on the BoT-IoT dataset. The SSN scheme is better than FNN based on multiple metrics such as accuracy, precision, and recall as well as multi-classification metrics such as Cohen Cappa's score.

The authors in [12] used Radial Based Function (RBF) for support vector machine for classifying DoS, Probe, R2L and U2R types of attack. Two datasets namely "Mixed" and "10% KDD Cup99" datasets have been used for evaluating performance of intrusion detection scheme. According to result analysis, validation accuracy was estimated to 89.85% and 99.9% for mixed and KDD respectively.

Table 1: Comparative analysis

PAPERS	***DATASET***	***ML METHODS***	***DL METHODS***	***PERFORMANCE METRICS***	***ATTACK DETECTION***
[7]	*KDD cup 99*	*LDA, RF, CART*	*--*	*Accuracy and Kappa*	*Normal, probe, U2R and R2L*
[8]	*TON IoT*	*KNN, Ada Boost*	*--*	*Precision, Recall, F1 score, Accuracy, and ROC*	*Normal, scanning, DDoS, DoS, password attacks*
[9]	*IntruDTree*	*KNN, SVM, LR, NB*	*--*	*precision, recDOI, F1-score, accuracy, and ROC*	*DoS Malware*
[10]	*BoT IoT*	*--*	*FNN, SNN*	*precision, recall and F1-score, Kappa Score and MC*	*Normal, DDoS, DoS, and Reconnaissance*
[11]	*KDD cup 99*	*SVM*	*--*	*Accuracy*	*Denial of service (Dos), Probe, User to Root (U2R), Remote to User (R2L)*
[12]	*DARPA 1999*	*KNN*	*--*	*Accuracy and F1-score*	*Snort based attack detection and minimizing false alarms*
[13]	*MalShare*	*Decision Tree*	*--*	*Accuracy*	*APT Attacks*
[14]	*KDD cup 99*	*--*	*DNN*	*Accuracy, Training and testing time*	*Denial of service (Dos), Probe, User to Root (U2R), Remote to User (R2L)*
[15]	*KDD Cup 99*	*LR*	*--*	*Accuracy and CPU time(s)*	*Normal. DOS, probe, U2L, R2L*
[16]	*NSL-KDD*	*--*	*RNN-IDS*	*Accuracy and Training time*	*Normal, DoS, probe, U2R and R2L*

Meng et al. [13] developed an intelligent knowledge-based alarm filter based on KNN-classifier with objective function to minimize the false alarm rate. The DARPA 1999 dataset was used to train the filter. A network environment was setup using snort and Wireshark where real-world web traffic was monitored. Snort detects various types of attacks and generated alerts. The generated alters were forwarded to KNN-based alarm filter for further analysis thereby filtering and minimizing the false alarm rate. The result analysis showed that accuracy of the design system was 85.2% and F-score was 0.82.

Moon et al. [14] proposed a DTB-IDS (Decision Tree-based IDS) for detection and preventing Advance Persistent Threat (APT). The proposed system executes the malicious code on the virtual environment and then analyze the behavior to detect APT. the result analysis depicts the accuracy of proposed system was 84.7%.

The author in [15] proposed an IDS that classify the non-labeled data using ladder network and then classify the non-labeled data using Deep belief Network (DBN). Moreover, the proposed scheme also integrates the semi-supervise learning with neural network with aim to achieve high accuracy with small number of labeled samples using KDD Cup99 dataset. The detection accuracy was 99.18%.

An IDS based on Deep Belief Network (DBN) using logistic regression was proposed in [16]. In order to improve the overall performance of IDS system, the multi-class logistic regression was trained on 10 epochs with pre-trained data of the 10% KDD Cup99 dataset. A low false rate of 2.47% was measured and detection rate of 97.9% was achieved for the proposed scheme.

A Recurrent Neural Network (RNN) based IDS was proposed by Ying et al. [17] using NSL-KDD data set. The performance of proposed model was evaluated in binary classification, influence of number of neurons, learning rates and multi-class classification. The result analysis showed that multi-class classification achieved the training accuracy 99.53% and test accuracy 81.29%. The binary classification model achieves the training accuracy 99.81% and test accuracy 83.28%.

A categorical comparison of recent ML and DL based works is provided in Table -1 in an attempt to provide further insight on ML/DL methods, various attack types and datasets that have been recently adopted in this domain.

3. Methodology

This section explains our ML-based intrusion detection flow. First, we provide an overview of the main steps involved in the methodology. Later, the key steps are explained in the subsequent subsections.

Figure 1 visualizes the overall flow of the intrusion detection using ML. The flow starts with the dataset available as the input. Initially, a preprocessing step performs tasks to prepare the dataset for the next steps. The preprocessing is an essential step which when overlooked, could affect the accuracy of the later steps of the flow. An in-depth understanding of the dataset is made via visualization and correlation mapping of the features. Figure 2 shows the correlation map of the dataset with only those features for which the correlation is higher than 0.6.

The cleaned dataset is forwarded to the training step where essentially the dataset is divided into train and test sets. The train data is provided to the selected ML model for learning the patterns. The test part of the dataset is kept for prediction purposes for the later step. In the testing step, the trained model from the last step is evaluated by applying unseen data (that was initially kept in the test part of the dataset).

3.1. Preprocessing

The capability of quickly learning patterns in the data is what makes ML models preferrable over the traditional approaches [18]. However, for many applications areas, the datasets are not in a standard form. Moreover, the data obtained from repetitive readings requires excessive steps to bring it in a form suited for further processing in the ML pipeline.

The tasks to be performed in preprocessing step vary depending on the application and the type of data that is being dealt with. Nevertheless, there is a set of techniques that have been commonly employed in most of the ML applications. In case of intrusion detection, we apply a sequence of common preprocessing tasks along with some specific tasks that favor the IDS.

Next important task involves understanding of the feature set. For this purpose, correlation map could provide useful insight to check which features could be eliminated due to redundancy. This is really important in many cases including IDS since the dataset (*KDDCUP99*) contains 41 features out of which many could be dropped. Other visualizations could also be used to undertake dataset formatting such as class balancing. In certain supervised ML methods such as SVM, feature mapping is required to represent features of data in suitable space. Finally, suitable and relevant features are kept only for the next step.

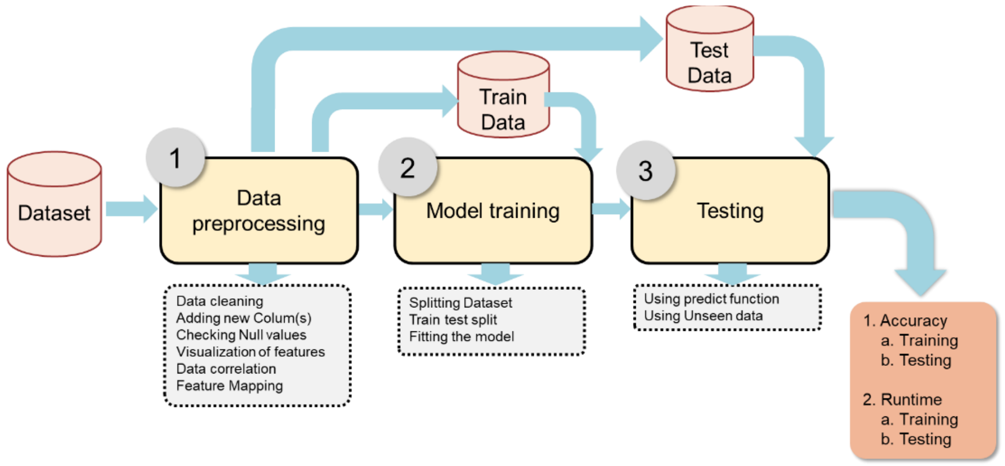

Figure 1: Overall flow

3.2. Training and Testing

The training step starts by first splitting train/test data. This is done to avoid over/under fitting of the model. The overfitting effect refers a scenario where the model learns the patterns of a dataset during the train and when tested on the same data, provides extremely accurate results since it has already seen the data. In underfitting, the model suffers to finds patterns in the training data and when tested on the new data, the model accuracy is very low.

The overfitting effect can be minimized by breaking the dataset with appropriate proportions of the training data to let the model learn the relationship of the data and enough samples in the test data to evaluate the model. The underfitting effect can be tackled by using well suited ML model and careful selection of the features to capture essential input/output pattern of the data.

We divide the processed dataset with a proportion of *70%* and *30%* for training and testing of the model. This is done by invoking *train_test_split* function from *sklearn* library. Then for the available training data, we utilize the most common seven ML methods from different python's ML libraries e.g., *sklearn* and *keras*, to train the models. The testing data is prepared by stripping the class label column. The accuracy score of the model is computed by applying the test data to the trained model.

4. Results

In this section, we provide details on our experimental setup and obtained results for various ML techniques on the intrusion detection.

4.1. Setup of experiment

As discussed in the previous section, we use a standard dataset i.e., *KDDCUP99*, to evaluate ML techniques for accurate prediction of anomalies in the network traffic data. The dataset manipulation is performed mainly via python library *pandas* and *numpy*. For visualization purposes, we use *matplotlib*. For ML models except for DNN, python's *sklearn* library is utilized, whereas for DNN model we use *keras*. The model parameters for all the ML models used in our evaluation are shown in Table-2.

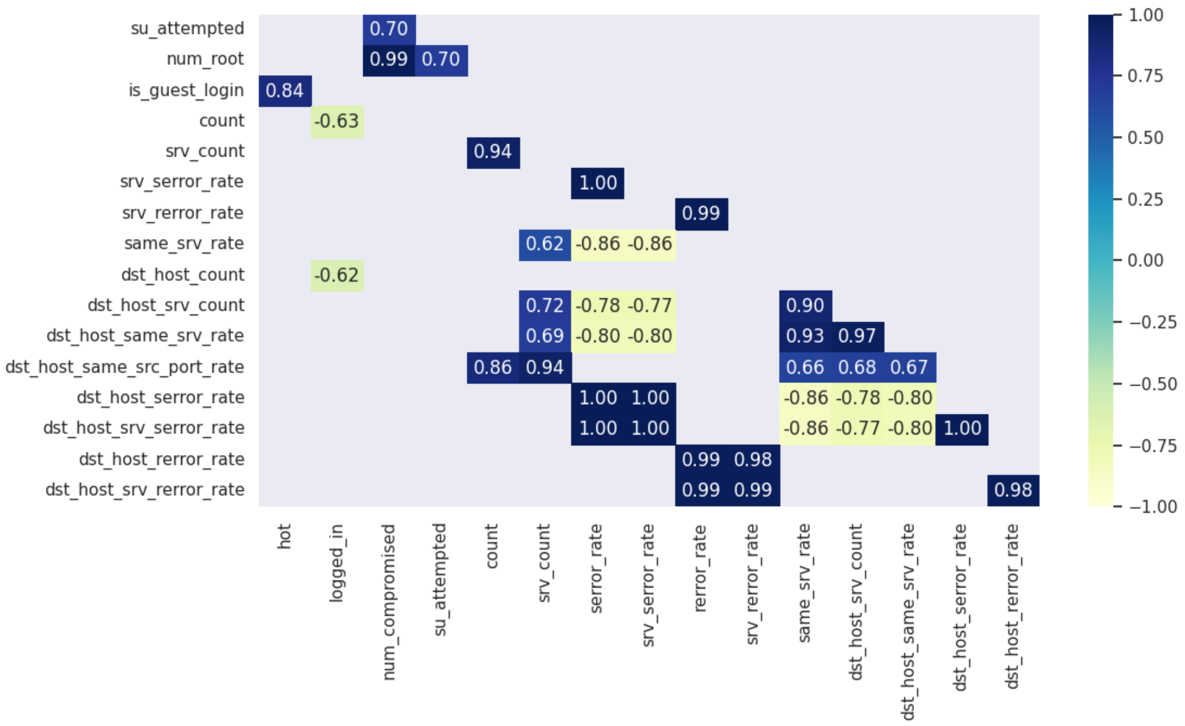

Figure 2: Correlation matrix for attributes of the dataset

4.2. Results and Discussion

For each ML classification method from Table-2, we evaluate the performance along two key parameters i.e., time and accuracy. For time, we report training and testing times separately.

Figure 3 and Figure 4 show the training and testing times, respectively, for all ML methods on the dataset. As can be seen from Figure 1, simpler ML models tend to have smaller training times whereas models that involves more complex learning structure could spend more time in learning the relationship of data. In this case, the DNN took the largest training time on the dataset followed by GB, SVC and LR. NB took the smallest training time but as we will see that costs us performance penalty on the accuracy front.

Table-2: ML algorithm parameters

S. #	Model	Parameter
1	Naïve Bayesian (NB)	*default*
2	Decisions tree (DT)	*criterion="entropy max_depth = 4*
3	Random Forest (RF)	*n_estimators=30*
4	Support Vector Machine (SVM)	*gamma = 'scale'*
5	Linear Regression (LR)	*max_iter =1200000*
6	Gradient Boosting (GB)	*random_state=0*
7	Deep Neural Network (DNN)	*epochs=100 batch_size=64 activation=relu optimizer=adam*

For testing, all algorithms perform equally well except SVC which took significantly larger time on the prediction for the testing part of the dataset. We believe that this increase might be the result of an increase in the support vectors during the prediction step.

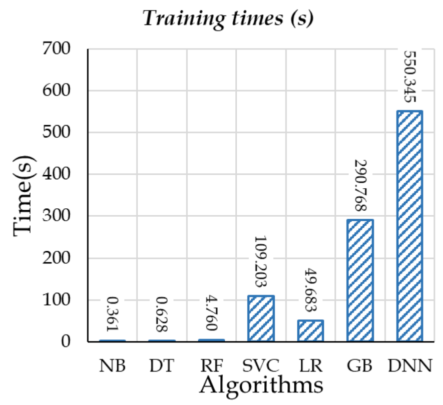

Figure 3: Training times for all ML methods

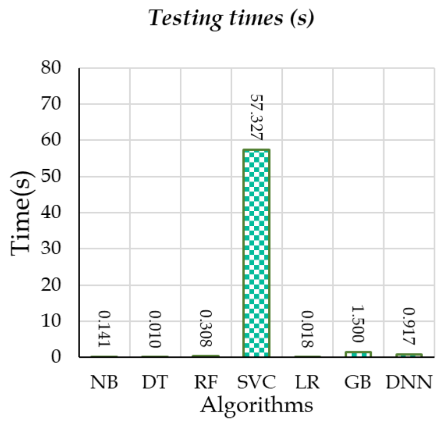

Figure 4: Testing times for all ML methods

In Figure 5 and 6, we report the accuracy scores for all algorithms. In case of accuracy results, we see a similar trend for training and testing accuracy. All algorithms except NB could achieve accuracy higher than 90%. The highest accuracy is achieved by the RF algorithm (1.0). DT, SVC, LR, GB and DNN achieve 0.990, 0.999, 0.993, 0.997, and 0.998 accuracy respectively. For testing accuracy, a similar trend can be observed. Overall RF algorithm turns out to be the most accurate classifier with a very low training and testing time.

To further evaluate the performance of the ML models, we employ a commonly used loss function i.e., cross-entropy loss and report the values for all algorithms in Table-3. It is important to mention that other most common metrics for loss such as means squared error or mean arithmetic error are not applicable in our case since the problem at hand is classification rather than regression and thus these metrics cannot capture the loss function efficiently. Cross-entropy loss function can better represent the training and testing performance in multi-class classification problems [19]. Results of cross-entropy represent a general trend of slight increase in loss for testing except in case of NB classifier where testing loss is a bit lower.

Table-3: Cross-entropy values for all algorithms

Alg.	***Cross-Entropy***	
	Training	**Testing**
NB	2.777	2.776
DT	0.025	0.026
RF	0.000	0.002
SVC	0.005	0.005
LR	0.017	0.017
GB	0.048	0.052
DNN	0.041	0.042

In addition to accuracy which might not be sufficient to indicate a model's performance, we attempt get further insight into the performance of the ML algorithms by computing another important quality metric i.e., F1-score that combines precision and recall into a single composite quality indicator. For F1-scores, we observe that the RF archives the highest F1-score (see Figure-7) followed by SVC and LR. The NB classifier could only obtain 0.45 making it the lowest in terms of F1-score.

5. Conclusion

This paper presents a work on classification of network traffic for intrusion detection using machine learning techniques.

Intrusion detection has become a challenging task due to data explosion in the recent years driving novel attacks and penetration techniques from the hackers. Manual techniques based on signatures of malicious activity could miss certain attack types or even could not respond to unknown threats. ML techniques are capable of learning complex patterns in the data and can classify unseen data with high accuracy. The experimental results provided in this paper show that ML techniques could separate normal and bad connections with great accuracy (up to 99.9%) on a standard dataset with computationally light-weight inference models.

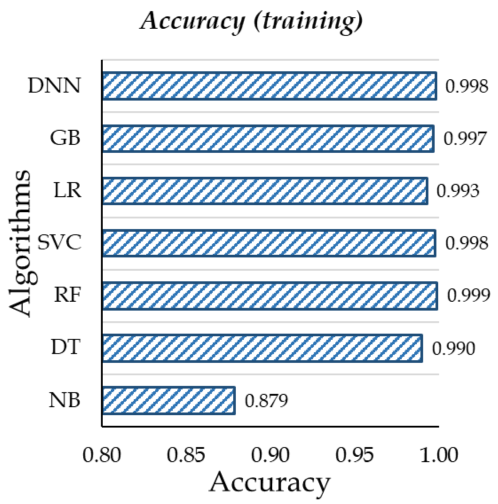

Figure 5: Training accuracy for all ML methods

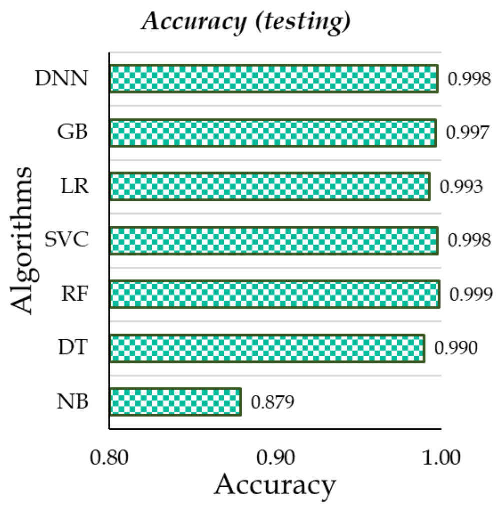

Figure 6: Testing accuracy for all ML methods

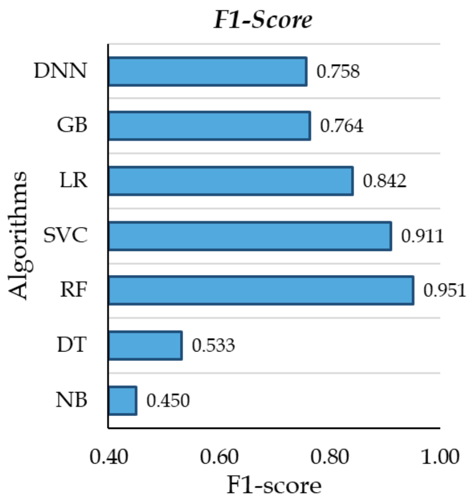

Figure 7: F1-score for all algorithms

REFERENCES

[1] Z. Zhang *et al.*, "Artificial intelligence in cyber security: research advances, challenges, and opportunities," *Artif Intell Rev*, vol. 55, no. 2, pp. 1029–1053, 2022, doi: 10.1007/s10462-021-09976-0.

[2] Z. Tang *et al.*, "Data Augmentation for Graph Convolutional Network on Semi-supervised Classification," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12859 LNCS, pp. 33–48, 2021, doi: 10.1007/978-3-030-85899-5_3.

[3] T. T. Nguyen and V. J. Reddi, "Deep Reinforcement Learning for Cyber Security," *IEEE Trans Neural Netw*

Learn Syst, 2021, doi: 10.1109/TNNLS.2021.3121870.

[4] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, “Network Intrusion Detection for IoT Security Based on Learning Techniques,” *IEEE Communications Surveys and Tutorials*, vol. 21, no. 3, pp. 2671–2701, Jul. 2019, doi: 10.1109/COMST.2019.2896380.

[5] P. Parkar and A. Bilimoria, “A survey on cyber security IDS using ML methods,” in *Proceedings - 5th International Conference on Intelligent Computing and Control Systems, ICICCS 2021*, May 2021, pp. 352–360. doi: 10.1109/ICICCS51141.2021.9432210.

[6] F. Tao, M. Akhtar, and Z. Jiayuan, “The future of Artificial Intelligence in Cybersecurity: A Comprehensive Survey,” *EAI Endorsed Transactions on Creative Technologies*, vol. 8, no. 28, p. 170285, 2021, doi: 10.4108/eai.7-7-2021.170285.

[7] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, “Network intrusion detection system: A systematic study of machine learning and deep learning approaches,” *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, Jan. 2021, doi: 10.1002/ett.4150.

[8] T. Saranya, S. Sridevi, C. Deisy, T. D. Chung, and M. K. A. A. Khan, “Performance Analysis of Machine Learning Algorithms in Intrusion Detection System: A Review,” in *Procedia Computer Science*, 2020, vol. 171, pp. 1251–1260. doi: 10.1016/j.procs.2020.04.133.

[9] I. ; Hidayat, M. Ali, ; Zulfiqar, and A. Arshad, “Machine learning based intrusion detection system: an experimental comparison.” [Online]. Available: https://edshare.gcu.ac.uk/id/eprint/5179

[10] I. H. Sarker, Y. B. Abushark, F. Alsolami, and A. I. Khan, “IntruDTree: A machine learning based cyber security intrusion detection model,” *Symmetry (Basel)*, vol. 12, no. 5, May 2020, doi: 10.3390/SYM12050754.

[11] O. Ibitoye, O. Shafiq, and A. Matrawy, “Analyzing adversarial attacks against deep learning for intrusion detection in IoT networks,” in *IEEE global communications conference (GLOBECOM)*, 2019, pp. 1–6.

[12] M. v. Kotpalliwar and R. Wajgi, “Classification of attacks using support vector machine (SVM) on KDDCUP’99 IDS database,” in *Proceedings - 2015 5th International Conference on Communication Systems and Network Technologies, CSNT 2015*, Sep. 2015, pp. 987–990. doi: 10.1109/CSNT.2015.185.

[13] W. Meng, W. Li, and L. F. Kwok, “Design of intelligent KNN-based alarm filter using knowledge-based alert verification in intrusion detection,” *Security and Communication Networks*, vol. 8, no. 18, pp. 3883–3895, Dec. 2015, doi: 10.1002/sec.1307.

[14] D. Moon, H. Im, I. Kim, and J. H. Park, “DTB-IDS: an intrusion detection system based on decision tree using behavior analysis for preventing APT attacks,” *Journal of Supercomputing*, vol. 73, no. 7, pp. 2881–2895, Jul. 2017, doi: 10.1007/s11227-015-1604-8.

[15] M. Nadeem, O. Marshall, S. Singh, X. Fang, and X. Yuan, “Semi-Supervised Deep Neural Network for Network Intrusion Detection,” 2016. [Online]. Available: https://digitalcommons.kennesaw.edu/ccerphttps://digitalcommons.kennesaw.edu/ccerp/2016/Practice/2

[16] K. Alrawashdeh and C. Purdy, “Toward an online anomaly intrusion detection system based on deep learning,” in *Proceedings - 2016 15th IEEE International Conference on Machine Learning and Applications, ICMLA 2016*, Jan. 2017, pp. 195–200. doi: 10.1109/ICMLA.2016.167.

[17] C. Yin, Y. Zhu, J. Fei, and X. He, “A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks,” *IEEE Access*, vol. 5, pp. 21954–21961, Oct. 2017, doi: 10.1109/ACCESS.2017.2762418.

[18] F. Deeba *et al.*, “A novel image dehazing framework for robust vision-based intelligent systems,” *International Journal of Intelligent Systems*, Dec. 2021, doi: 10.1002/INT.22627.

[19] Z. Zhang, M. S.-A. in neural information, and undefined 2018, “Generalized cross entropy loss for training deep neural networks with noisy labels,” *proceedings.neurips.cc*, Accessed: Jan. 20, 2023. [Online]. Available: https://proceedings.neurips.cc/paper/2018/hash/f2925f97bc13ad2852a7a551802feea0-Abstract.html