An Empirical Internet Protocol Network Intrusion Detection using Isolation Forest and One-Class Support Vector Machines

Gerard Shu Fuhnwi[1], Victoria Adedoyin[2], Janet O. Agbaje[3]
Gianforte School of Computing, Montana State University, Montana 59715, USA[1]
Department of Chemistry, Montana State University, Montana 59715, USA[2]
Department of Mathematical Sciences, Montana Technological University, Montana 59701, USA[3]

***Abstract*—With the increasing reliance on web-based applications and services, network intrusion detection has become a critical aspect of maintaining the security and integrity of computer networks. This study empirically investigates internet protocol network intrusion detection using two machine learning techniques: Isolation Forest (IF) and One-Class Support Vector Machines (OC-SVM), combined with ANOVA F-test feature selection. This paper presents an empirical study comparing the effectiveness of two machine learning algorithms, Isolation Forest (IF) and One-Class Support Vector Machines (OC-SVM), with ANOVA F-test feature selection in detecting network intrusions using web services. The study used the NSL-KDD dataset, encompassing hypertext transfer protocol (HTTP), simple mail transfer protocol (SMTP), and file transfer protocol (FTP) web services attacks and normal traffic patterns, to comprehensively evaluate the algorithms. The performance of the algorithms is evaluated based on several metrics, such as the F1-score, detection rate (recall), precision, false alarm rate (FAR), and Area Under the Receiver Operating Characteristic (AUCROC) curve. Additionally, the study investigates the impact of different hyper-parameters on the performance of both algorithms. Our empirical results demonstrate that while both IF and OC-SVM exhibit high efficacy in detecting network intrusion attacks using web services of type HTTP, SMTP, and FTP, the One-Class Support Vector Machines outperform the Isolation Forest in terms of F1-score (SMTP), detection rate(HTTP, SMTP, and FTP), AUCROC, and a consistent low false alarm rate (HTTP). We used the t-test to determine that OCSVM statistically outperforms IF on DR and FAR.**

***Keywords*—HTTP; SMTP; FTP; ANOVA F-test; AUCROC; OC-SVMs; FAR; DR; IF**

I. Introduction

Network Intrusion can be referred to as an unauthorized penetration of a computer in an establishment or an address in one's assigned domain [1]. The nature and types of network intrusion have evolved over the years and become more rampant in recent years [2].

An intrusion can be passive or active. In passive intrusion, the penetration is gained stealthily and without detection, while in active intrusion, changes to network resources are affected. Intrusion can either come from an insider or an outsider. By insider, we mean an employee, customer, or business partner. Outsider means someone not connected to the organization. Network intrusions can occur in different ways. Some announce their presence by defacing the website, while others are malicious, with the goal of siphoning off data until it's discovered. Some redirect users who are unaware of their website through cracking passwords or mimicking your website [1]. Sometimes, intruders absorb network resources intended for other uses or users, which can lead to a denial of service [3]. These unauthorized penetrations on the digital network are imperil on many occasions the security of networks and their data [4].

Network security breaches are rapidly increasing and result in a significant amount of loss to organizations, and often leads to a loss of confidence in them from their unaware customers that have fallen victims. The IBM report shows that the average cost of a data breach has risen 12 percent over the past five years to 3.92 million dollars per incident on average [5]. This is more than the cost of a breach caused by a system glitch or human error.

Many researchers have carried out research and projects on network intrusion detection [6], [7], [8]. Wang and Battiti identified intrusions in computer networks with principal component analysis [9]. Liao and Vemuri used a k-nearest neighbour classifier for intrusion detection [10]. Gaffney and Ulvila evaluated intrusion detectors using a decision theory approach [11]. But this area still longs for more work as a result of the rapid rise in network intrusion. Therefore, we need to design an efficient algorithm that can successfully defend against network intrusions in an ever-evolving threat landscape. To achieve proactive security control, organizations must put in place a good network security infrastructure and leverage the potential of machine learning, which has the capability of automatically and continuously detecting network intrusions. This will help block intruders and prevent them from achieving their goals. The remainder of this paper is organized as follows: In Section II, we briefly review some related work in anomaly detection based Network Intrusion Detection. Section III gives a description of the algorithms used in this paper. Section IV analyzes the empirical evaluation, where we review the data sets used, evaluation metrics description, results, and result discussion. Section V covers the conclusion.

II. Related Work

Liu and Ting [23] focused on using an Isolation Forest to detect anomalies that have many applications in the areas of fraud detection, network intrusion, medical and public health, industrial damage detection, and so on. The goal here is to build a tree-based structure that isolates anomalies rather than profiles anomalies like in the previous methods such as classification-based methods [12], and clustering-based methods [13]. Their proposed method, called Isolation Forest, builds a collection of individual tree structures that recursively partition a given data set, where anomalies are instances with a short path length on the trees. The anomaly score is used to determine instances that are anomalies, and has values between 1 and 0, with a score close to 1 being an anomaly and vice versa. The authors compared their results with other methods for anomaly detection techniques [14] like ORCA, LOF, and RF on real-world data sets with high dimensions and large data sizes using the metric AUC (Area Under the Curve) and run times. [15] proposed a hybrid of SVM and decision trees in classifying attacks of different forms of intrusion in knowledge discovery and data mining 1999 (KDDCUP99) data.

In [16], Sarumi et al. compared SVM and Apriori using Network Security Laboratory Knowledge Discovery and Data Mining (NSL-KDD) data and the University of South Wales NB 2015 (UNSW NB-15) dataset. From their results, they concluded that SVM outperformed Apriori in terms of accuracy, while Apriori showed a better performance in terms of speed.

In [17], Farnaaz and Jabbar proposed a detection intrusion system using random forest. Experimental results were conducted on the NSL-KDD dataset. Empirical results show that the proposed model achieved a low false alarm rate and a high recall. Similarly, [18], [19], [20], and [21] applied machine learning techniques for network intrusion detection systems.

All the above mentioned papers discuss intrusion detection methods without any statistics to compare their results, attacks using web services, and no user guidance for using the proposed algorithms. To overcome this, one can look at the statistical significance of the various evaluation metrics based on the different machine learning algorithms proposed by them and also change the various parameters in the machine learning algorithms to observe their performance.

This paper compares the performance of One-class SVM and Isolation Forest machine learning algorithms in network intrusion using a two-sample t-test and parameter alternation to provide some guidance on these algorithms' usage to new researchers in this field. Our approach can also guide evaluating and analyzing these techniques in solving intrusion detection problems. Also, this method can overcome one of the main challenges of intrusion detection techniques, accurate representative labels for normal and abnormal instances, which is a significant concern. To overcome this challenge in most intrusion detection problems, our approach can be used as a pre-labeling technique and then supervised anomaly detection techniques to solve intrusion detection problems. Overall, our empirical results demonstrate the potential of Isolation Forest and One-Class SVM and provide valuable insights for future research in this field.

III. Methods

This section presents the intrusion detection approach used in this paper. These approaches include the ANOVA F-test, the Isolation Forest, the One-Class Support Vector Machines, and the two-sample t-test.

A. ANOVA F-test

The ANOVA F-test, or Analysis of Variance F-test, is a statistical technique used to compare the means of two or more groups to determine whether significant differences exist. It is commonly employed in feature selection or variable ranking tasks, where the goal is to identify the most relevant features or variables for a particular analysis or model.

Applying the ANOVA F-test to a dataset can rank features based on their F-statistic or p-value. Features with high F-statistic values or low p-values are considered more relevant, as they exhibit significant differences between the groups or classes. These relevant features can then be selected for further analysis or modeling, while less informative features can be discarded to reduce dimensionality and improve computational efficiency. In the case of web network intrusion detection, the ANOVA F-test can be used to identify the most discriminative features that differentiate between normal network traffic and malicious intrusion attempts. By selecting the most significant features, it is possible to improve the performance and efficiency of intrusion detection systems by focusing on the most relevant information and reducing noise or irrelevant variables.

B. Isolation Forest (IF)

IF has been applied in different scenarios. Isolation Forest is an unsupervised learning algorithm for anomaly detection that works on the principle of isolating anomalies, instead of the most common technique of profiling normal points [22] and [23]. It is different from other distance and density based algorithms (see Fig. 1). The underlying assumption for this algorithm is that fewer instances of anomalies result in a smaller number of partitions (shorter path length) and the instances with distinguishable attribute values are more likely to be separated in early partitioning [24]. This implies that data points that have a shorter path length are likely to be anomalies. The necessary input parameters for building Isolation Forest algorithm are the subsampling size,the number of trees, and the height of the tree [24]. The subsampling size was suggested to be smaller for the machine learning algorithm to function faster and yield a better detection result [25]. We can use log to base 2 (number of data points) to get the depth of trees needed, but the path length converge before $t = 100$. [25].

```
Algorithm 1: iForest($X$, $t$, $\psi$)
Inputs: $X$ – input data, $t$ – number of trees, $\psi$ – subsampling size
Output: a set of $t$ iTrees
  1: Initialize Forest
  2: set height limit $l$ = ceiling($\log_2 \psi$)
  3: for $i = 1$ to $t$ do
  4:     $X' \longleftarrow sample(X, \psi)$
  5:     $Forest \longleftarrow Forest \cup iTrees(X', 0, l)$
  6: end for
  7: return Forest
```

Fig. 1. Algorithm 1.

C. One-Class Support Vector Machines

One-Class Support Vector Machines (OC-SVMs) [26] are a natural extension of SVMs. One-Class SVM is an unsupervised learning technique capable of differentiating test samples from a particular class from other classes. The One-Class SVM works on the basics of minimizing the hypersphere of one class in the training set and then considers every other class not within the hypersphere as anomalies or outliers. In order to identify suspicious observations, an OC-SVM estimates a distribution that encompasses most of the observations and then labels as "suspicious" those that lie far from it with respect to a suitable metric. This model uses different kernel functions or hyperspheres: linear, radial basis, sigmoid, and polynomial.

D. Two Sample t-test

The two-sample t-test, also known as the independent samples t-test or unpaired t-test, is a statistical hypothesis test used to compare the means of two independent groups to determine if there is a significant difference between them. The test assumes that the data is normally distributed and that the variances of the two groups are equal (although there are modifications available if this assumption does not hold). In order to compare the performance of IF and OCSVM with the ANOVA F-test, we used a two-sample t-test to test whether there is a significant difference between the mean performances of DR, FAR, F_1 score, AUCROC, and precision. The null hypothesis (H_0) for the two-sample t-test states that there is no significant difference between the mean performances of the two models, while the alternative hypothesis (H_1) states that there is a significant difference, unlikely to have occurred by chance, between the mean performances of the two models.

IV. Empirical Evaluation

A. Data Description

The NSL−KDD dataset is an improved version of the KDD99 dataset, in which a large amount of data redundancy has been removed [27]. This dataset has the same attributes as the KDD99 having 41 features that are labeled as either normal or attacks using different web services (http, smtp, ftp, etc.). The NSL−KDD dataset repository has two files: KDDTrain.txt and KDDTest.txt. Table I shows the attack categories using different services and the number of data points per category in the NSL−KDD train and test datasets. The NSL−KDD dataset has 125973 data points in the training dataset and 22544 in the testing dataset.

TABLE I. The Attack Types (class) using Different Internet Protocols (http, smtp and ftp), the Number of Records in the NSL-KDD Training and Testing Dataset

Attacks using different Internet Protocol	**No. of records**		**Attack Types (class)**
	Training	**Testing**	
Normal	45,078	7,291	Normal traffic data
HTTP	2,289	1,180	Worm, Land, Smurf, Udpstorm, Teardrop, Pod, Mailbomb, Neptune, Process table, Apache2, Back
SMTP	284	316	Ipsweep, Nmap, Satan Portsweep, Mscan, Saint
FTP	648	48	WarezClient, Worm, SnmpGetAttack, WarezMaster, Imap, SnmpGuess, Named, MultiHop, Phf, SPy, Sendmail, Ftp_Write, Xsnoop, Xlock, Guess_Password

B. Data Pre-processing

The NSL-KDD dataset has 41 features, each representing an attack type described in Section 4.1 and an attack category (class feature). These features are both numeric (38 features) and categorical (3 features). The categorical features are protocol _type (3 types), service (70 types), and flag (11 types) that need to be converted to numeric features. We want to extract the most popular attacks caused by using different internet protocols (the service feature). These widespread attacks use web services (internet protocols) such as hypertext transfer protocol (HTTP), simple mail transfer protocol (SMTP), and file transfer protocol (FTP). After extracting the various internet protocols, the attack types (class) feature is labeled with a numeric type, starting with Normal, labeled as 0 and 1 for the different attack types.

Using ANOVA F-test feature elimination, the most relevant features with the highest F-statistic values in the dataset are identified, eliminating the least important features. These features are src bytes (number of data bytes transferred from source to destination in a single connection), dst bytes (number of data bytes transferred from destination to source in a single connection), and duration.

C. Confusion Matrix

The performance of machine learning techniques can be evaluated using different parameters. These parameters are calculated using True Positive (TP), False Negative (FN), False Positive (FP), and True Negative (TN) as shown in the confusion matrix [28] in Table II. The following parameters are used to evaluate our proposed approach.

1) Detection Rate (DR): It is the ratio between the total number of attacks detected by the NIDS and the total number of attacks present in the dataset [17] which can be calculated using the formula:

$$DR = \frac{TP}{TP + FN}$$

2) Precision: This measures the fraction of examples predicted as attacks that turned out to be attacks, which can be calculated using the formula:

$$Precision = \frac{TP}{TP + FP}$$

3) F_1 Score: It is the harmonic mean of the fraction of examples predicted as attacks that turned out to be attacks (precision). It can also be described as the ratio between the total number of attacks detected by the NIDS and the total number of attacks present in the dataset (the detection rate) which can be calculated using the formula:

$$F_1\ Score = \frac{2 * TP}{2 * TP + FN + FP}$$

4) False Alarm Rate (FAR): It is the fraction of non attacks that are misclassified as attacks, which can be calculated using the formula:

$$FAR = \frac{FP}{FP + TN}$$

TABLE II. CONFUSION MATRIX: A CONTINGENCY CONTAINING FOUR METRICS, TRUE POSITIVE (TP), TRUE NEGATIVE (TN), FALSE POSITIVE (FP), AND FALSE NEGATIVE (FN).

Attack		Predicted Class	
		Yes	No
Actual class	Yes	TP	FN
	No	FP	TN

5) Receiver Operating Characteristic (ROC) Curve: The Receiver Operating Characteristic (ROC) curve is a graphical representation used to evaluate the performance of binary classification models in machine learning. It is created by plotting the ratio between the total number of attacks detected by the NIDS to the total number of attacks present in the dataset (detection rate) against the fraction of non-attacks that are misclassified as attacks (False Alarm Rate) at various classification threshold levels. The area under the curve (AUC) of the ROC quantifies the overall performance of the classification model. AUC values range from 0 to 1, with a value of 0.5 representing a random classifier and a value of 1 indicating a perfect classifier. A higher AUC value suggests a better-performing classification model.

A good NIDS should have high detection rates, precision, AUCROC, F_1 score but low FAR.

Most machine learning algorithms are evaluated using predictive accuracy, but this is not appropriate for network intrusion detection because it mostly involves imbalanced data. In terms of imbalanced data, we mean that the proportion of data points in each class is not approximately equal. The evaluation metrics adopted in this paper for evaluation and comparison of our models are standard AUC (Area under curve). The area under the receiver operating curve gives an average measure of performance across all possible classification thresholds.

D. Experimental Results

All experiments were performed in Python with alternating parameters for Isolation Forest (Sklearn) and One-class support vector machines (Sklearn) using the Intel(R) Core(TM) i7-10510U CPU at 1.80 GHz and 2.30 GHz processor with 16 GB of RAM. Training and testing of the Isolation Forest took four seconds, while it took 40 seconds to train the One-Class support vector machine model on the three selected features from the NSL−KDD dataset. The experimental results for One-class support vector machines and Isolation Forest on different performance metrics are shown in Table III and Table IV respectively.

E. Discussion of Results

In Table III, the polynomial kernel outperformed the other kernels on HTTP and SMTP subsets with high DR, F_1 Score, AUCROC, Precision, and low FAR. On the other hand, in Table III, the sigmoid kernel performed much better than the different kernels on the FTP subset.

In Table IV, isolation forest with 100 estimators on the HTTP subset achieved the highest DR, F_1 Score, AUCROC, Precision, and low FAR. The SMTP and FTP subset performs best on the evaluation metrics with 50 estimators. Generally, both Isolation Forest and One-class support vector machines didn't perform well on the FTP subset, having very high FAR and low DR, F_1 Score, AUCROC, and Precision. It is evident in Table III and IV that the one-class support vector machines outperform Isolation Forest on all subsets, that is, HTTP, SMTP, and FTP having high DR, F_1 Score, and low FAR. Statistical analysis of overall performance on the one-class support vector machines and Isolation Forest results used a two-sample t-test with two-tailed probability to determine if each model's DR and FAR score on the test data yielded statistically significant differences ($p < 0.05$). In the HTTP, SMTP, and FTP, the one-class support machines had a significantly different DR,and FAR score ($p < 0.001$), which showed that our hypothesis was accepted.

Our approach improved detection capabilities on selected attack types when compared to other models and benchmarks in related work. The RFE process identified several key features highly relevant to network intrusion detection. These features align with our expectations and prior research [18], [20], and [25], confirming the importance of specific traffic characteristics in detecting malicious activities. The combination of IF, OC-SVM and ANOVA F-test not only improved the model's performance but also reduced the complexity of the model by eliminating redundant and irrelevant features.

The practical implications of our findings are significant for the field of network intrusion detection. The improved detection rates offered by our approach can help security practitioners identify and respond to cyber threats more effectively. Additionally, reducing false positives and negatives can minimize the operational overhead of manually investigating false alarms. Furthermore, our approach demonstrates potential scalability and adaptability for different network environments and evolving cyber threats.

V. CONCLUSION AND FUTURE WORK

The experiments performed on the NSL-KDD network intrusion data show that One-class support vector machines had the overall best performance in terms of DR and FAR scores over the Isolation Forest, with the best performance obtained by tuning the default parameters in both algorithms. Also, the number of estimators in Isolation Forest is comparable; using 100 and 50 estimators outperformed 200 estimators.

Therefore, One-class support is a good model for network intrusion detection by changing the default parameters in Sklearn. Also, polynomial or sigmoid kernel functions could be the best kernels to choose when using One-Class SVM on network intrusion data. Because of the usage of feature selection, the computational cost decreases (four seconds for Isolation Forest and forty seconds), and our experimental results indicate that our proposed approach increases the DR, F1 score, AUCROC, and precision and decreases FAR for three types of attacks. We equally compared one-class support vector machines and Isolation Forest selected attack types using a two-sample t-test and found that our proposed approach (with fewer features) is promising. For future work, we will experiment with deep learning approaches like GANs and autoencoders since they are capable of handling data of higher

TABLE III. One-Class SVM Performance Measure on NSL−KDD Test

Attacks using different Internet Protocol	Kernel	Gamma	DR	F_1 Score	AUCROC	Precision	FAR
HTTP	Linear	0.00005	0.9802	0.9303	0.6308	0.8852	0.0198
	Sigmoid	0.00005	0.9969	0.9386	0.8867	0.6383	0.0031
	Polynomial	0.00005	0.9969	0.9385	0.6378	0.8866	0.0031
SMTP	Linear	0.00005	0.8398	0.7228	0.4468	0.6345	0.1602
	Sigmoid	0.00005	0.9806	0.7953	0.5156	0.6689	0.0194
	Polynomial	0.00005	0.9838	0.7969	0.5172	0.6696	0.0162
FTP	Linear	0.00005	0.3333	0.5000	0.6667	1.0000	0.6667
	Sigmoid	0.00005	0.5208	0.6848	0.7604	1.0000	0.4791
	Polynomial	0.00005	0.3333	0.5000	0.6667	1.0000	0.6667

TABLE IV. Isolation Forest Performance Measure on NSL−KDD Test

Attacks using different Internet Protocol	Estimators	maximum samples	DR	F_1 Score	AUCROC	Precision	FAR
HTTP	50	256	0.9618	0.9563	0.8402	0.9508	0.0382
	100	256	0.9631	0.9570	0.8409	0.9509	0.0369
	200	256	0.9619	0.9563	0.8399	0.9507	0.0381
SMTP	50	256	0.9725	0.7846	0.6697	0.9725	0.0275
	100	256	0.9709	0.7838	0.6697	0.9709	0.0291
	200	256	0.9693	0.7835	0.6699	0.9693	0.0307
FTP	50	256	0.2917	0.3043	0.6225	0.3182	0.7083
	100	256	0.2083	0.2273	0.5809	0.2500	0.7916
	200	256	0.2708	0.2857	0.6121	0.3023	0.7292

dimensions and also evaluate one-class support vector machines and Isolation Forest using supervised learning methods like the random forest, Xgboost, and cost sensitive support vector machines.

References

[1] Michael West: Chapter 2 - Preventing System Intrusions: Network and System Security (Second Edition), pp.29–56. Syngress, Boston, 2014. doi:10.1016/B978-0-12-416689-9.00002-2.

[2] Thomas M. Chen and Patrick J. Walsh: Chapter 3 - Guarding Against Network Intrusions. Network and System Security (Second Edition). pp.57–82. Syngress, Boston, 2014. doi:10.1016/B978-0-12-416689-9.00003-4.

[3] Robert Moskowitz: Network Intrusion: Methods of Attack, 2014.

[4] Isabell Gaylord: Network Intrusion: How to Detect and Prevent It. Reducing Risk, United states Cybersecurity Magazine, 2020. https://www.uscybersecurity.net/network-intrusion/.

[5] David Bisson: How to Foil the 6 Stages of a Network Intrusion,Tripwire State of security news, 2019. https://www.tripwire.com/state-of-security/security-data-protection/security-hardening/6-stages-of-network-intrusion-and-how-to-defend-against-them/.

[6] Kumar, Amit and Maurya, Harish Chandra and Misra, Rahul: A research paper on hybrid intrusion detection system, International Journal of Engineering and Advanced Technology (IJEAT), vol.2, no.4, pp.294–297, Citeseer, 2013.

[7] Li Tian and Wang Jianwen: Research on Network Intrusion Detection System Based on Improved K-means Clustering Algorithm, International Forum on Computer Science-Technology and Applications, vol. 1, pp. 76–79, 2009. doi:10.1109/IFCSTA.2009.25.

[8] Dikshant Gupta and Suhani Singhal and Shamita Malik and Archana Singh:Network intrusion detection system using various data mining technique, International Conference on Research Advances in Integrated Navigation Systems (RAINS), pp.1–6, 2016. doi:10.1109/RAINS.2016.7764418.

[9] Wang, Wei and Battiti, Roberto:Identifying intrusions in computer networks based on principal component analysis, University of Trento, 2005.

[10] Liao, Yihua and Vemuri, V Rao:Use of k-nearest neighbor classifier for intrusion detection, Computers & security,vol.21,no.5, pp.439–448, Elsevier , 2002.

[11] Ulvila, Jacob W and Gaffney Jr, John E: Evaluation of intrusion detection systems, Journal of Research of the National Institute of Standards and Technology, vol.108, no. 6, pp.453, 2003.

[12] Abe, Naoki and Zadrozny, Bianca and Langford, John: Outlier detection by active learning. Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, pp.504–509, 2006.

[13] He, Zengyou and Xu, Xiaofei and Deng, Shengchun:Discovering cluster-based local outliers,Pattern Recognition Letters,vol.24,no.9?10, pp.1641–1650, Elsevier, 2003.

[14] Fuhnwi, Gerard Shu and Agbaje, Janet O and Oshinubi, Kayode and Peter, Olumuyiwa James: An Empirical Study on Anomaly Detection Using Density-based and Representative-based Clustering Algorithms, Journal of the Nigerian Society of Physical Sciences, pp.1364–1364, 2023.

[15] Mulay, Snehal A and Devale, PR and Garje, GV: Intrusion detection system using support vector machine and decision tree, International Journal of Computer Applications, vol.3, no.3, pp.40–43, Citeseer, 2010.

[16] Sarumi, Oluwafemi A and Adetunmbi, Adebayo O and Adetoye, Fadekemi A: Discovering computer networks intrusion using data analytics and machine intelligence, Scientific African, vol.9, pp.e00500, Elsevier, 2020.

[17] Farnaaz, Nabila and Jabbar, MA: Random forest modeling for network intrusion detection system, Procedia Computer Science, Elsevier, pp.213–217, (2016).

[18] WS, Jenif D Souza and Parvathavarthini, B: Machine learning based intrusion detection framework using recursive feature elimination method, 2020 International Conference on System, Computation, Automation and Networking (ICSCAN), pp.1–4, IEEE, 2020.

[19] Ingre, Bhupendra and Yadav, Anamika: Performance analysis of NSL-KDD dataset using ANN, 2015 international conference on signal processing and communication engineering systems, pp.92–96, IEEE, 2015.

[20] Aljawarneh, Shadi and Aldwairi, Monther and Yassein, Muneer Bani: Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model, Journal of Computational Science, pp.152–160, Elsevier, 2018.

[21] Hamed, Tarfa and Dara, Rozita and Kremer, Stefan C: Network intrusion detection system based on recursive feature addition and bigram technique, Computers & Security, pp.137–155, Elsevier, 2018.

[22] Wikipedia contributors: Isolation forest — Wikipedia, The Free Encyclopedia, 2020, https://en.wikipedia.org/w/index.php?title=Isolation_forest&oldid=985700362.

[23] Liu, Fei Tony and Ting, Kai Ming and Zhou, Zhi-Hua:Isolation-based anomaly detection, Eighth IEEE International Conference on Data Mining, pp.413–422, IEEE, 2008.

[24] Arunraj, Nari S and Hable, Robert and Fernandes, Michael and Leidl, Karl and Heigl, Michael: Comparison of supervised, semi-supervised and unsupervised learning methods in network intrusion detection system (NIDS) application, Anwendungen und Konzepte der Wirtschaftsinformatik, no.6, 2017.

[25] Liu, Fei Tony and Ting, Kai Ming and Zhou, Zhi-Hua:Isolation-based anomaly, ACM Transactions on Knowledge Discovery from Data (TKDD). vol.6, no.1, pp.1–39, Acm New York, NY, USA, 2012.

[26] Larry M. Manevitz and Malik Yousef: One-Class SVMs for Document Classification, Journal of Machine Learning Research 2, pp.139–154, 2001.

[27] Ring, Markus and Wunderlich, Sarah and Scheuring, Deniz and Landes, Dieter and Hotho, Andreas, "A survey of network-based intrusion detection data sets," Computers & Security, vol.86, pp. 147–167, Elsevier, 2019.

[28] Beauxis-Aussalet, Emma and Hardman, Lynda, "IEEE Conference on Visual Analytics Science and Technology (VAST)-Poster Proceedings," pp. 1–2, 2014.