Comparative Evaluation of Machine Learning Algorithms for Network Intrusion Detection and Attack Classification

Miguel Leon [§], Tijana Markovic [§], Sasikumar Punnekkat
School of Innovation, Design and Engineering
Mälardalen University, Västerås, Sweden
{miguel.leonortiz,tijana.markovic,sasikumar.punnekkat}@mdh.se

***Abstract*—With the increasing use of the internet and reliance on computer-based systems for our daily lives, any vulnerability in those systems is one of the most important issues for the community. For this reason, the need for intelligent models that detect malicious intrusions is important to keep our personal information safe. In this paper, we investigate several supervised (Artificial Neural Network, Support Vector Machine, Random Forest, Linear Discriminant Analysis, and K-Nearest Neighbors) and unsupervised (K-means, Mean-shift, and DBSCAN) machine learning algorithms, in the context of anomaly-based Intrusion Detection Systems. We are using four different IDS benchmark datasets (KDD99, NSL-KDD, UNSW-NB15, and CIC-IDS-2017) to evaluate the performance of the selected machine learning algorithms for both intrusion detection and attack classification. The results have shown that Random Forest is the most suitable algorithm regarding model accuracy and execution time.**

Keywords: Machine Learning, Supervised Learning, Unsupervised Learning, Intrusion Detection, Attack Classification

I. Introduction

The increasing use of the Internet, network applications, and cloud services, makes systems more vulnerable to external threats that can violate the main principles of security: confidentiality, integrity, and availability. One of the biggest challenges in the network security research area is identifying malicious activities on time and mitigating them promptly. The process of analyzing network traffic to identify signs of malicious activity is called intrusion detection [1] and a system that automates this process is called the Intrusion Detection System (IDS) [2]. There are two common methodologies that IDSs use to identify threats: signature-based and anomaly-based [3]. A signature-based IDS monitors network packets and searches for patterns that correspond to known network attack types. Anomaly-based IDS learns the general behavior of normal network traffic and raises an alarm when significant deviations are detected.

In recent years, Machine Learning (ML) has become a popular and effective method for developing new anomaly-based IDS [4], [5], [6], [7], [8]. The majority of experiments in the area of IDS were conducted using one or more benchmark datasets [9]. Some of the well-known IDS datasets are: KDD99, NSL-KDD, UNSW-NB 15, CIDDS-001, CICIDS2017, CSE-CIC-IDS2018 etc. All of those datasets consist of a combination of normal and malicious traffic. Data about network packets were preprocessed to create the features, and every entry was labeled as normal activity, or as some type of network attack. Various ML algorithms were applied to those datasets either to separate normal traffic for the malicious one (binary classification problems) or to detect specific attack types (multiclass classification problems). Buczak et al. [10] did a focused literature survey of ML methods used in IDs and recognized some of the most commonly used methods such as Random Forest (RF), Decision Trees, density-based clustering algorithms (e.g. DBSCAN), Support Vector Machine (SVM), Artificial Neural Networks (ANN), Naive Bayes (NB), association rules, etc. Many studies in this area analyze the accuracy of different ML algorithms on different benchmark datasets. Revathi et al. [11] presented an evaluation of supervised ML algorithms (RF, J48, SVM, Classification and regression trees and NB) for multiclass classification on one dataset (NSL-KDD) and derived the conclusion that RF has the highest accuracy compared to all other algorithms. Abedin et al. [12] worked on the same problem by applying NB, J48, NBTree, Multilayer Perceptron (MLP), and RF and their findings were that J48 and RF had the best performance. Tuan et al. [13] evaluated SVM, ANN, NB, Decision Tree, and unsupervised ML on the UNBS-NB 15 and KDD99 datasets. This paper considered only the Distributed Denial of Service (DDoS) attacks and unsupervised ML was the best at differentiating between DDoS and normal network traffic, but it was not specified which unsupervised ML algorithms were used. There are several papers that evaluate a single ML algorithm on one or more benchmark datasets, such as different types of neural networks [7], [14], [15], [16], [17], RF [18], SVM [19], K-means [20] etc.

Most of the above-mentioned papers were focused on evaluating ML algorithms on a single dataset or on evaluating single ML algorithm on one or more datasets. Also, most of the papers focused only on binary or multiclass classification, and the research goal of this paper is to address both.

The contributions of this work can be summarized as follows:

- five supervised and three unsupervised ML algorithms were evaluated and explained,

[§]Equal contribution

- four different datasets were used, including one of the most recent IDS datasets CICIDS2017, and
- evaluation was conducted for both binary classification (intrusion detection) and multiclass classification (attack classification).

In the following, we first present the methodology used in Section II. This section includes the details about the used datasets, the explanation of the applied preprocessing techniques, the overview of the evaluated machine learning algorithms, and the presentation of the experimental setup. In Section III we present experiments and results, followed by a discussion IV. Finally, a summary of our findings and the plans for future work are given in Section V.

II. Methodology

This section describes the methodology for evaluating the performance of different ML algorithms for attack detection and classification on four benchmark ID datasets, as well as the details of experimentation and implementation.

A. Datasets

The experiments presented in this paper were conducted on 4 preexisting datasets that will be presented in this section.

KDD99 [21] is a dataset from a competition in intrusion detection (KDD cup). This dataset was created in a simulated environment, and it contains normal traffic and 22 types of network attacks grouped in 4 attack families: Denial of Service (DoS), Probe, User-to-Root (U2R), and Remote-to-Local (R2L). KDD99 dataset has 41 features. The full training set contains 4,898,431 entries. The distribution of attacks in the KDD99 dataset is 80%. In this paper, we used the 10% of the dataset, included in "kddcup.txt".

NSL-KDD dataset [22] is a modified version of the original KDD99 dataset created to solve some of the problems, such as the number of duplicate entries [23]. This dataset contains 37 attack types and has the same number of attack families and the same number of features as KDD99. NSL-KDD contains 125,973 entries in the training set and 22,544 entries in the test set, and the distribution of attacks is 48%. In this paper, we used the file: "KDDTrain+.txt".

UNSW-NB15 dataset [24] [25] was generated in the Cyber Range Lab of the Australian Centre for Cyber Security. This dataset is a hybrid of real normal traffic and 9 families of simulated attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. UNSW-NB15 dataset has 42 features and 2,540,044 entries. The distribution of attacks in the UNSW-NB15 dataset is around 13%. A partition from this data set was configured as a training set and a testing set. The training set has 175,341 records (attacks around 68%), and the testing set has 82,332 (attacks around 55%). In this paper, we used "UNSW_NB15_training-set.csv".

CIC-IDS-2017 dataset [26] was generated by the Canadian Institute for Cybersecurity. This dataset was created in a simulated environment, and it contains normal traffic and 15 types of network attacks grouped into 7 attack families: Brute force, PortScan, Botnet, DoS, DDoS, Web Attack, and Infiltration. CIC-IDS-2017 dataset has 78 features and 2,830,743 entries. The distribution of attacks is 19.7%. Two features were removed (flow of bytes and flow of packets per second), and we used the following files:

- "Friday-WorkingHours-Afternoon-DDos.pcap_ISCX.csv"
- "Friday-WorkingHours-Afternoon-PortScan.pcap_ISCX.csv"
- "Friday-WorkingHours-Morning.pcap_ISCX.csv"
- "Thursday-WorkingHours-Morning-WebAttacks.pcap_ISCX.csv"
- "Tuesday-WorkingHours.pcap_ISCX.csv"
- "Wednesday-workingHours.pcap_ISCX.csv"

All of the above-mentioned datasets were used for both intrusion detection and attack classification problems. Tables I and II present a summary for the whole datasets and for the subsets that were used in this paper, respectively. Additionally, we have removed instances that belong to classes with less than 800 cases for all the subsets we have used, hence the difference in instances and classes between tables I and II.

B. Preprocessing

These datasets had 3 different types of variables: binary, real, and categorical. These variables were preprocessed in the following manner:

- Binary: The values remained unchanged.
- Real: The values were normalized on the [0, 1] range.
- Categorical: The values were one-hot encoded. If a variable has 5 different possible alternatives, that variable was transformed into a binary vector with 4 zeros and 1 one indicating which alternative it is.

C. Machine Learning algorithms

Machine Learning (ML) is the part of Artificial Intelligence where algorithms learn patterns from datasets without explicit instructions [27]. It can be divided into the following areas:

- Supervised Learning (SL): Algorithms within the SL category use input-output pairs to learn a function that maps from inputs to outputs.
- Unsupervised Learning (UL): Algorithms within the UL category learn patterns within the input data without any output information given in the training phase.
- Reinforcement Learning (RL): Algorithms within the RL category learn by trial and error. A specific "reward" or "punishment" is given depending on their actions and consequences.

For IDS, only the SL and UL methods were used. The rest of the section will be used to give an insight into different algorithms that were used in our experiments. Algorithms 1) to 5) belong to the SL category, while 6) to 8) belong to the UL category.

TABLE I: Information and the distribution of the instances for KDD99, NSL-KDD, UNSW-NB15, and CIC-IDS-2017 dataset, considering the whole datasets. The number of classes in parenthesis is the subdivision of the attack families shown.

DATASET	KDD99	NSL-KDD	UNSW-NB15	CIC-IDS-2017
No. of Instances	4,898,431	148,517	2,540,044	2,830,743
Attacks (%)	80%	48%	13%	19.7%
No. of Classes	4 (22)	4 (37)	9	7
No. of Features	41	41	42	78
DoS	98.92%	74.70%	5.09%	45.31%
Probe	1.05%	19.70%	-	-
R2L	0.03%	5.40%	-	-
U2R	0.001%	0.20%	-	-
Fuzzers	-	-	7.55%	-
Analysis	-	-	0.83%	-
Backdoors	-	-	0.72%	-
Exploits	-	-	13.86%	-
Generic	-	-	67.07%	-
Reconnaissance	-	-	4.35%	-
Shell-code	-	-	0.47%	-
Worms	-	-	0.05%	-
Brute force	-	-	-	2.48%
Botnet	-	-	-	0.35%
DDoS	-	-	-	22.96%
Web attack	-	-	-	0.39%
Infiltration	-	-	-	0.01%
PortScan	-	-	-	28.50%

TABLE II: Information and distribution of the subsets of instances for the KDD99, NSL-KDD, UNSW-NB15, and CIC-IDS-2017 datasets, used in this paper. The number of features in parentheses is the actual number of features used by the ML algorithms after preprocessing.

DATASETS		KDD99	NSL-KDD	UNSW-NB15	CIC-IDS-2017
No. of Instances		493,347	125,597	80,650	2,011,539
Attacks (%)		80.28%	46.38%	54.12%	27.69%
No. of Classes		9	10	6	11
No. of Features		41 (118)	41 (122)	42 (190)	76
DoS	**Smurf**	56.92%	2.11%	5.07%	-
	Teardrop	0.20%	0.71%		-
	Neptune	21.73%	32.81%		-
	Back	0.45%	0.76%		-
	Slowloris	-	-		0.29%
	Slowhttptest	-	-		0.27%
	Hulk	-	-		11.49%
	Golden Eye	-	-		0.51%
Probe	**Ipsweep**	0.25%	2.87%	-	-
	Portsweep	0.21%	2.33%	-	-
	Nmap	-	1.19%	-	-
	Satan	0.32%	2.89%	-	-
R2L	**Warezclient**	0.21%	0.71%	-	-
Fuzzers		-	-	7.52%	-
Exploits		-	-	13.80%	-
Generic		-	-	23.40%	-
Reconnaisance		-	-	4.33%	-
Brute Force	**FTP-Patator**	-	-	-	0.39%
	SSH-Patator	-	-	-	0.29%
DDoS		-	-	-	6.36%
Web Attack Brute Force		-	-	-	0.07%
Port Scan		-	-	-	7.90%
BotNet		-	-	-	0.10%

1) Artificial Neural Network (ANN): Inspired by the human brain and the connections that exist between millions of neurons, ANN simulates these connections between different artificial neurons [28]. The values in the neurons will be multiplied by a set of parameters, called weights, that will indicate the strength of the specific connections. The power of ANN relies on stacking many layers with neurons in order to create a model that better differentiates the boundaries of different non-linear separable classes.

2) Linear Discriminant Analysis (LDA): Similarly to ANN, LDA tries to find a hyperplane (w) that separates the classes [29]. This hyperplane is used to minimize the error between the predicted class and the actual class by using $y = w \times x$, where y is the class and x is the input data. LDA will achieve high accuracy for problems that have linearly separable data.

3) Support Vector Machine (SVM): In a similar way, the goal of SVM is to find the hyperplane that separates the different classes [27]. The difference lies in the position of the hyperplane. This will be placed with the maximum margin to the support vectors (closer solutions to the hyperplane). With this property, SVM is able to generalize better than LDA, for example. Additionally, SVM will transform the dimensionality of the data points using a kernel function. This dimensionality will depend on the number of training cases and will help to separate data that contain non-linearities.

4) Random Forest (RF): Differently from the other techniques, RF groups different versions of the same machine learning algorithm, called Decision Tree (DT) [30]. DT is a predictive model based on rules that are created by evaluating the importance of input parameters from the dataset, using methods such as entropy or information gain. RF is formed by a number of DTs. The difference among all DTs is that the models are created using different subsets of the dataset.

5) K-Nearest Neighbour (K-NN): K-NN is one of the simpler machine learning algorithms [27], [31]. A non-parametric model is built with all training examples. To classify a new example, a value d is calculated using some distance measure, such as the Euclidean distance. Distance (d) is calculated between the new example and all the cases that form the model. The majority class from the K nearest examples will be chosen as the predicted class.

6) K-means: K-means belongs to the UL category [20]. This means, that in theory, no specific desired output is given to the algorithm. K-means is a clustering algorithm, where K clusters are formed. To find the centroids of the clusters, the examples of the training set are assigned to the closest cluster. Then, the middle point of the assigned points is calculated as the new center. This process is repeated until the centroids do not change from one iteration to the next. For our experiments, after finding all the centers of the different clusters, the predominant class among the cases linked to the specific center is assigned to that cluster. This means that one class can have more than one cluster.

7) Mean-shift: Similarly to K-means, Mean-shift is a clustering algorithm. Mean-shift creates the centers of the clusters that cover the whole surface [32]. Then, the points assigned to the cluster, within a specified distance R, are selected to calculate the center of the cluster as the middle point of all of them. To finalize, clusters with the same points are removed, and the predominant class is assigned to the specific cluster.

8) Density-Based Spatial Clustering of Applications with Noise (DBSCAN): Differently from the previous unsupervised learning methods, DBSCAN is a density-based method [33]. Clusters are formed by neighboring points, starting from a minimal number of cases, m, within a radius R. Then, if the neighboring points to a cluster are within a distance R, they are added to the cluster. Once a point is added to one cluster, it will not be able to belong to a different cluster. The same as the other two UL methods, the predominant class is assigned as the class of the cluster.

D. Experimental Settings

Two different problems were solved: Intrusion Detection (ID) and Attack Classification. All experiments were performed in an HP Zbook with an Intel Core i9-988H CPU @ 2.30GHz and Matlab 2019b was used to implement the algorithms. The experiments were performed once per algorithm, dataset and problem with a distribution 70%-10%-20% for training, validation, and testing datasets, respectively. The final parameters for all the machine learning algorithms are:

- *ANN*: Epochs: 3, Batch size: 256 and Learning rate: 0.01
  - *Hidden Layer*: Number of neurons: 50, Activation function: Sigmoid
  - *Output Layer*: Number of neurons: number of classes, Activation function: softmax
- *SVM*: The function fitcsvm was used with 'gaussian' as the kernel function.
- *LDA*: The function fitcdiscr was used with 'pseudoQuadratic' as the 'DiscimType'.
- *RF*: The function fitcensemble was used with the standard parameters.
- *K-NN*: K was set to 5 and the Euclidian distance was used as the measure to find the most similar K cases.
- *K-means*: K was set to 200. If no cases are assigned to a centroid, the centroid is removed.
- *Mean-Shift*: The dimensionality for Mean-shift was reduced to 2, using Principal Component Analysis [34] since the algorithm will be unmanageable otherwise. The starting number of centroids was 100 and the radius R was set to 0.1.
- *DBSCAN*: The minimum number of points to create a cluster (m) was set to 2 and the radius (R) was set to 1. When a cluster is formed, the average point is calculated and used as a centroid in the same way as in K-means.

The decision to select these parameters was based on the experiments performed in [35] and [36].

III. Results

In this section, the findings of our experiments are presented. The results of intrusion detection are shown in Section III-A, while the results of attack classification are presented in Section III-B.

A. Intrusion Detection

A total of 8 different machine learning methods (5 SL methods and 3 UL methods) have been tested on 4 different datasets (KDD99, NLS-KDD, UNSW-NB15 and CIC-IDS-2017). The results for Intrusion Detection are presented in Table III.

For KDD99 dataset, the results can be obviated due to the similarity in the performance that all the ML methods obtained with a difference of 1% between RF and K-means, the best and the worst algorithm respectively.

In the NSL-KDD dataset, RF, K-NN and SVM obtained similar performance with a difference of 0.7% between each other. In the UNSW-NB15 dataset, RF emerges as the best algorithm with 97%, a 3% improvement with respect to SVM and a 5% with respect to K-NN, the second and third algorithm, respectively.

Following the performances of RF, SVM and K-NN, the second best group of algorithms was the one formed by the 3 unsupervised learning algorithms (K-means, Mean-shift and DBSCAN). Those 3 algorithms obtain a similar performance among them. In NSL-KDD, the performance was close to that obtained by the best algorithm with a difference of 3-5%, depending on the algorithm, while for the UNSW-NB15 dataset, the difference increases to 18%. The worst performance was obtained by ANN and LDA. For example, LDA obtained 74% and 65% in UNSW-NB15 and NSL-KDD respectively.

Finally, the results for CIC-IDS-2017 show that RF and K-NN were the best algrotihms with almost the same performance (99.8%). Closely to them, SVM obtained 97.7%. Then, the rest of the algorithms appear with an accuracy ranging from 86.6% for the best of them (ANN) to 78.72% for the worst of them (DBSCAN). When performing ID, CIC-IDS-2017 was the only data set in which LDA was not the worst algorithm.

B. Attack Classification

The same 8 ML algorithms, as for intrusion detection, have been tested on the same datasets, but in this section, the results for Attack Classification are presented.

Similarly to ID, the results of all algorithms for the KDD99 dataset (Table IV) were similar to each other with a difference of 2% between the best and the worst algorithm. The only interesting remark is that RF, SVM and K-NN were the only algorithms that properly recognized all the 9 different classes (more than 90% of the examples per class).

Regarding the second dataset, NSL-KDD (Table V), RF, K-NN and SVM obtained the best results, this time with a difference of 0.5% between the best and the worst one. The rest of the algorithms obtained similar performance to each other, and DBSCAN and LDA were the best options among them.

A similar conclusion can be drawn for UNSW-NB15 dataset (Table VI), where the results show that RF was the best algorithm, obtaining 85% accuracy, closely followed by K-NN with 83.22% and SVM with 82.44%. A second group, formed by UL methods and ANN obtained results ranging from 72.44% (DBSCAN) to 68% (ANN). LDA achieved poor performance (41%) compared to their counterparts.

Finally, the results of the algorithms for the CIC-IDS-2017 dataset are presented in Table VII. K-NN was the best algorithm with 99.86%, closely followed by RF with 0.6% lower accuracy. Also, SVM and LDA obtained decent performance with an accuracy of around 96%. In this dataset, the UL methods obtained the worse performance among all algorithms with a performance lower than 82%, with K-means obtaining the best results among the 3.

From the results for the 3 last datasets, the same conclusion can be derived regarding the recognition of the different classes: RF, K-NN and SVM were the only algorithms that could recognize all classes. Only one exception was detected, for the CIC-IDS-2017 dataset Web Attack Brute Force attacks were not recognized by RF. All the other algorithms fail to classify many of the attacks properly.

IV. Discussion

In this section, three different aspects will be discussed when comparing the different algorithms. Section IV-A discusses the performance of the algorithms. The execution time is discussed in Section IV-B, while the relation between the accuracy for different classes and the percentage of cases per class for the attack classification problem is analyzed in Section IV-C.

A. Performance

A summary of the results can be found on Fig. 1 (Fig. 1a for ID and Fig. 1b for Attack Classification). As previously mentioned, RF had the highest accuracy. The reason for this is that it uses more than one model at the same time. This allows RF to create a better approximation of the perfect model. In CIC-IDS-2017, K-NN yielded the best results among all the algorithms. This happens due to the number of cases. CIC-IDS-2017 has over 2 million cases creating a huge model and allowing K-NN to have a better representation of the dataset.

With respect to the UL methods, it is important to mention that no specific information about the class was given when they form the clusters. Classes were used only at the end, to decide the class that the cluster will belong to. This means that the information that we take from the features is relevant when distinguishing between the different classes. Additionally, we can observe how the results were more or less consistent. Comparing results for the same dataset we can notice that if one UL algorithm is better than another UL algorithm for ID then the same is happening for attack classification. This means that the information from the dataset is relevant either for ID or attack classification.

The performance of LDA should also be discussed. LDA is a linear classifier which is not good when the classes were not linearly separable. In ID, LDA was not able to properly separate between two different classes, obtaining the worst results among all the classifiers. However, when the problem is attack classification, the performance of LDA improves due to

TABLE III: Accuracy of 8 ML algorithms for 4 different datasets for Intrusion Detection. Global stands for the Global Accuracy of the algorithm for a certain dataset. Boldface values represent the highest accuracy per dataset.

		KDD99			NSL-KDD			UNSW-NB15			CIC-IDS-2017		
Algorithm		**Global**	**Normal**	**Attack**	**Global**	**Normal**	**Attack**	**Global**	**Normal**	**Attack**	**Global**	**Normal**	**Attack**
SL	ANN	98.85	98.66	98.89	90.93	96.65	84.34	72.28	77.34	68.13	86.59	93.36	68.88
	SVM	**99.94**	99.89	99.95	99.14	99.55	98.67	94.41	95.20	93.77	97.68	99.19	93.72
	RF	**99.95**	99.92	99.95	**99.87**	99.89	99.85	**97.43**	97.76	97.16	**99.81**	99.90	99.57
	LDA	99.26	99.95	99.09	65.38	35.85	99.39	74.30	43.40	99.64	81.78	75.05	99.36
	KNN	**99.94**	99.89	99.95	99.54	99.63	99.43	92.97	95.55	90.86	**99.88**	99.91	99.81
UL	K-means	98.92	99.40	98.81	95.21	99.03	90.82	81.13	82.45	80.05	85.85	94.00	64.53
	Mean-Shift	98.93	99.21	98.86	94.03	95.91	91.85	79.07	91.08	69.23	83.45	82.33	86.36
	DBSCAN	99.02	99.67	98.86	96.96	97.54	96.28	79.64	85.75	74.62	78.72	98.26	27.57

TABLE IV: Accuracy of 8 ML algorithms for the KDD99 dataset for attack classification. Boldface values represent the highest global accuracy and the highest accuracy for each class.

Algorithm		**Global Acc.**	**No attack**	**Back**	**Ipsweep**	**Neptune**	**Portsweep**	**Satan**	**Smurf**	**Teardrop**	**Warezclient**
SL	ANN	98.13	98.74	0.00	0.00	**100.00**	0.00	0.00	**100.00**	0.00	0.00
	SVM	99.95	99.90	99.54	98.75	**100.00**	**99.49**	97.56	**100.00**	**100.00**	91.30
	RF	**99.99**	**99.99**	**100.00**	**99.17**	**99.99**	**99.49**	**97.87**	**100.00**	**100.00**	**100.00**
	LDA	98.68	99.42	9.98	97.08	98.27	98.46	93.29	99.86	9.85	32.85
	K-NN	99.95	99.89	98.61	98.75	**100.00**	98.97	**97.87**	**100.00**	**100.00**	96.62
UL	K-means	99.00	99.93	0.00	5.83	**100.00**	76.92	87.50	**99.99**	90.15	0.00
	Mean-Shift	98.93	99.11	0.00	97.92	99.95	30.77	81.10	**100.00**	20.20	62.80
	DBSCAN	98.99	99.97	3.48	69.17	**100.00**	91.28	79.27	**99.99**	0.00	0.00

TABLE V: Accuracy of 8 ML algorithms for the NSL-KDD dataset for attack classification. Boldface values represent the highest global accuracy and the highest accuracy for each class.

Algorithm		**Global Acc.**	**No attack**	**Back**	**Ipsweep**	**Neptune**	**nmap**	**Portsweep**	**Satan**	**Smurf**	**Teardrop**	**Warezclient**
SL	ANN	86.40	99.64	0.00	0.00	**99.99**	0.00	0.00	0.00	0.00	0.00	0.00
	SVM	99.18	99.47	78.13	**98.76**	**100.00**	94.01	**99.47**	95.85	**100.00**	98.80	82.14
	RF	**99.68**	**99.90**	**100.00**	98.06	**99.99**	94.01	99.30	**97.99**	**100.00**	**100.00**	91.07
	LDA	91.66	96.15	12.50	91.42	91.60	81.70	78.07	95.56	78.61	5.42	0.60
	K-NN	99.50	99.77	95.31	98.20	**100.00**	**96.21**	95.79	96.70	99.81	98.80	**93.45**
UL	K-means	90.08	98.73	0.00	86.45	99.47	16.09	72.81	0.00	0.00	0.00	0.00
	Mean-Shift	89.36	97.04	0.00	84.23	98.82	15.46	55.09	31.23	0.00	0.00	0.00
	DBSCAN	94.12	97.82	8.85	98.34	**100.00**	16.09	79.12	60.32	99.81	0.00	0.00

TABLE VI: Accuracy of 8 ML algorithms for the UNSW-NB15 dataset for attack classification. Boldface values represent the highest global accuracy and the highest accuracy for each class.

Algorithm		**Global Acc.**	**No attack**	**DoS**	**Exploits**	**Fuzzers**	**Generic**	**Reconnaissance**
SL	ANN	68.02	**99.85**	0.00	0.00	0.00	**96.16**	0.00
	SVM	82.44	96.69	20.83	68.98	38.44	**96.16**	51.64
	RF	**84.99**	98.07	36.03	76.93	28.77	**96.13**	**68.99**
	LDA	40.86	31.29	36.64	**81.98**	**54.02**	39.77	0.14
	K-NN	83.22	96.20	**39.83**	65.85	42.95	**96.43**	53.28
UL	K-means	70.87	95.34	0.00	35.54	0.00	**96.16**	0.00
	Mean-Shift	71.18	89.34	1.35	56.95	0.00	**96.16**	0.55
	DBSCAN	72.44	98.20	13.85	32.05	0.08	**96.19**	0.55

the fact that more than one classifier is being used to separate the classes.

B. Execution time

In Table VIII, the time expended by the algorithms during the training and testing phase is shown. It is obvious that the improvement in performance for the K-NN algorithm for CIC-IDS-2017 comes with a cost. We can observe how K-NN takes a huge amount of time compared to SVM or RF.

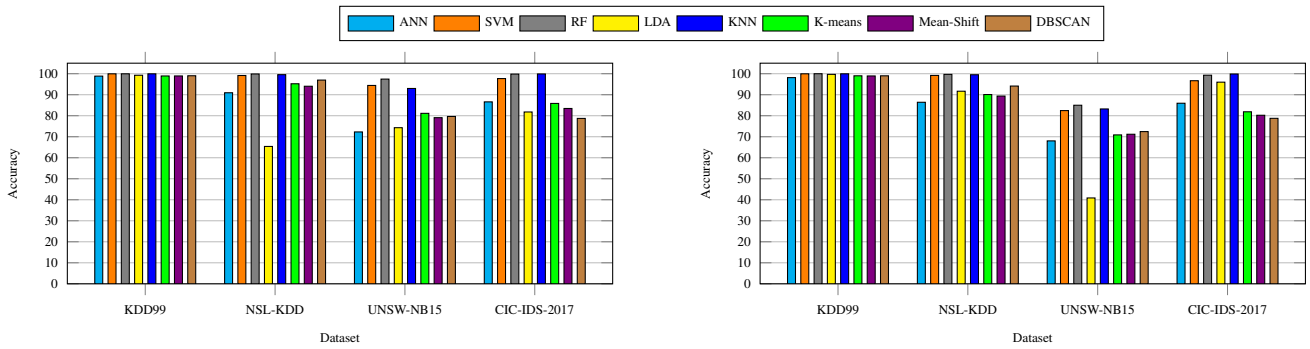
Additionally, we can see how SVM was faster than ANN, except for CIC-IDS-2017. The reason for this is the dimensionality change that SVM does, since it is directly dependent on the number of training cases. The dimensionality is equal to the number of cases.

It is also important to mention that LDA, K-means and Mean-shift take a really low amount of time in comparison to other algorithms, without forgetting that the dimensionality of Mean-shift was 2, otherwise the algorithm will be impossible to handle. The problem with these 3 algorithms is the low accuracy, especially in the case of LDA which was the worse algorithm among all.

Finally, DBSCAN was the slowest algorithm due to the fact that it has to calculate many more Euclidean distances than K-means for instance.

TABLE VII: Accuracy of 8 ML algorithms for the CIC-IDS-2017 dataset for attack classification. The bold-face values represent the highest global accuracy and the highest accuracy for each class.

Algorithm		Global Acc.	No Att.	Bot	DDoS	GoldenEye	Hulk	Slowhttptest	Slowloris	FTP-Patator	PortScan	SSH-Patator	WA Brute F.
SL	ANN	85.94	99.33	0.00	48.43	0.00	61.13	0.00	0.00	0.00	50.16	0.00	0.00
	SVM	96.66	99.35	0.00	94.23	87.52	86.64	83.01	51.36	49.42	99.23	0.42	0.00
	RF	99.28	99.75	0.00	**99.73**	86.03	99.13	73.45	79.10	99.55	99.43	**99.33**	0.00
	LDA	95.97	94.70	**99.19**	**99.73**	99.01	99.00	96.90	98.07	98.96	99.56	97.99	98.12
	K-NN	**99.86**	99.92	70.19	**99.73**	**99.30**	**99.85**	**98.67**	**98.95**	**99.81**	**99.97**	**99.33**	**98.43**
UL	K-means	81.87	99.35	0.00	46.25	0.00	61.09	0.00	0.00	0.00	0.00	0.00	0.00
	Mean-Shift	80.22	97.26	0.00	46.75	0.00	59.30	12.30	0.00	0.00	0.00	0.00	0.00
	DBSCAN	78.46	**99.99**	0.00	0.00	8.95	52.58	0.00	0.00	0.00	0.00	0.00	0.00



(a) Intrusion Detection

(b) Attack Classification

Fig. 1: Accuracy of 8 ML algorithms for 4 different datasets for (a) Intrusion Detection and (b) Attack classification.

TABLE VIII: Average time of Machine Learning algorithms for training and testing. All values are represented in seconds.

Algorithm	KDD99	NSL-KDD	UNSW-NB15	CIC-IDS-2017
ANN	7,668	554	361	25,826
SVM	395	171	637	61,685
RF	489	138	146	12,551
LDA	12	4	4	37
KNN	71,915	4737	3103	835,852
K-means	324	110	59	1,172
Mean-Shift	40	24	6	358
DBSCAN	78,128	7550	4610	1,065,102

C. Relation between accuracy and number of training cases

A final reflection is made on the relation between the number of training classes and the accuracy obtained by the ML algorithms. This relation is presented in Fig. 2. We can observe that as the number of cases of a class increases, the accuracy obtained by the ML algorithms increases. This figure shows the importance of having all classes with a good representation so that the ML algorithms are able to represent that information within the model.

Some algorithms such as RF, SVM or K-NN were less affected, since they were able to represent every class as a different problem. However, some other algorithms, such as ANN or UL methods, fail to represent a big number of classes. ANN has a problem with backpropagation and the way in which the weights are updated. In the case of UL methods, when the clusters were formed and the final class was decided, having a low number of cases affects in the voting to select an underrepresented class.

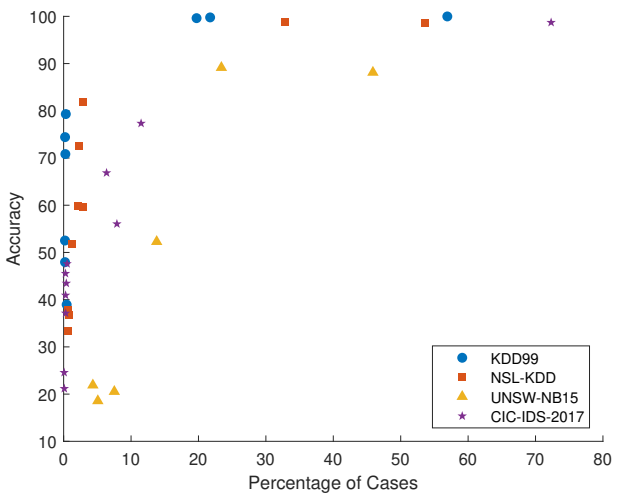


Fig. 2: Relation between the percentage of cases within a class with respect to the average accuracy obtained by the Machine Learning algorithms on that same class.

V. Conclusions and Future Work

In this paper, 8 different ML algorithms (5 SL methods and 3 UL methods) have been tested on 4 different datasets containing network attacks. Two different problems have been investigated: intrusion detection and attack classification.

From the experiments, it can be concluded that the best option regarding accuracy and time consumption was RF. K-NN yielded similar accuracy, but the biggest drawback was the time needed to classify new cases, especially when the size of

the dataset increases. Additionally, RF, K-NN and SVM were able to detect all attack types included in the used datasets.

As future directions, we can mention the usage of RF as a distributed learning algorithm when we have a distributed environment, or the usage of unsupervised learning methods for extracting new features or reducing the features. Also, we would like to investigate the possibility of using similarity among instances to apply structured ML models.

ACKNOWLEDGMENT

This work has been partially supported by the H2020 ECSEL EU Project Intelligent Secure Trustable Things (InSecTT). InSecTT (www.insectt.eu) has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876038. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Austria, Sweden, Spain, Italy, France, Portugal, Ireland, Finland, Slovenia, Poland, Netherlands, Turkey.

The document reflects only the author's view and the Commission is not responsible for any use that may be made of the information it contains.

REFERENCES

[1] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: A comprehensive review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[2] R. Bace and P. Mell, "Intrusion detection systems," *National Institute of Standards and Technology (NIST), Technical Report 800-31*, 2001.

[3] K. Scarfone, P. Mell *et al.*, "Guide to intrusion detection and prevention systems (idps)," *NIST special publication*, no. 800-94, 2007.

[4] M. M. Rathore, A. Ahmad, and A. Paul, "Real time intrusion detection system for ultra-high-speed big data environments," *The Journal of Supercomputing*, vol. 72, no. 9, pp. 3489–3510, 2016.

[5] Y. Fu, F. Lou, F. Meng, Z. Tian, H. Zhang, and F. Jiang, "An intelligent network attack detection method based on rnn," in *2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*. IEEE, 2018, pp. 483–489.

[6] A. K. Shrivas and A. K. Dewangan, "An ensemble model for classification of attacks with feature selection based on KDD99 and NSL-KDD data set," *International Journal of Computer Applications*, vol. 99, no. 15, pp. 8–13, 2014.

[7] S. Behera, A. Pradhan, and R. Dash, "Deep neural network architecture for anomaly based intrusion detection system," in *2018 5th International Conference on Signal Processing and Integrated Networks (SPIN)*. IEEE, 2018, pp. 270–274.

[8] S. Wang, J. F. Balarezo, S. Kandeepan, A. Al-Hourani, K. G. Chavez, and B. Rubinstein, "Machine learning in network anomaly detection: A survey," *IEEE Access*, vol. 9, pp. 152 379–152 396, 2021.

[9] M. Ghurab, G. Gaphari, F. Alshami, R. Alshamy, and S. Othman, "A Detailed Analysis of Benchmark Datasets for Network Intrusion Detection System," *Asian Journal of Research in Computer Science*, vol. 7, no. 4, pp. 14–33, 2021.

[10] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications surveys & tutorials*, vol. 18, no. 2, pp. 1153–1176, 2015.

[11] S. Revathi and A. Malathi, "A detailed analysis on NSL-KDD dataset using various machine learning techniques for intrusion detection," *International Journal of Engineering Research & Technology (IJERT)*, vol. 2, no. 12, pp. 1848–1853, 2013.

[12] M. Abedin, K. N. E. A. Siddiquee, M. Bhuyan, R. Karim, M. S. Hossain, K. Andersson *et al.*, "Performance analysis of anomaly based network intrusion detection systems," in *43nd IEEE Conference on Local Computer Networks Workshops (LCN Workshops), Chicago, October 1-4, 2018*. IEEE Computer Society, 2018, pp. 1–7.

[13] T. A. Tuan, H. V. Long, L. H. Son, R. Kumar, I. Priyadarshini, and N. T. K. Son, "Performance evaluation of Botnet DDoS attack detection using machine learning," *Evolutionary Intelligence*, vol. 13, no. 2, pp. 283–294, 2020.

[14] 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. 21 954–21 961, 2017.

[15] B. Ingre and A. Yadav, "Performance analysis of NSL-KDD dataset using ANN," in *2015 international conference on signal processing and communication engineering systems*. IEEE, 2015, pp. 92–96.

[16] J. Kim, J. Kim, H. Kim, M. Shim, and E. Choi, "CNN-based network intrusion detection against denial-of-service attacks," *Electronics*, vol. 9, no. 6, p. 916, 2020.

[17] B. Roy and H. Cheung, "A deep learning approach for intrusion detection in internet of things using bi-directional long short-term memory recurrent neural network," in *2018 28th International Telecommunication Networks and Applications Conference (ITNAC)*. IEEE, 2018, pp. 1–6.

[18] N. Farnaaz and M. Jabbar, "Random forest modeling for network intrusion detection system," *Procedia Computer Science*, vol. 89, pp. 213–217, 2016.

[19] M. S. Pervez and D. M. Farid, "Feature selection and intrusion classification in NSL-KDD cup 99 dataset employing SVMs," in *The 8th International Conference on Software, Knowledge, Information Management and Applications (SKIMA 2014)*. IEEE, 2014, pp. 1–6.

[20] V. Kumar, H. Chauhan, and D. Panwar, "K-means clustering approach to analyze NSL-KDD intrusion detection dataset," *International Journal of Soft Computing and Engineering (IJSCE) ISSN*, pp. 2231–2307, 2013.

[21] S. Hettich and S. D. Bay, "The UCI KDD Archive," [http://kdd.ics.uci.edu], 1999, irvine, CA: University of California, Department of Information and Computer Science.

[22] "NSL-KDD," [https://www.unb.ca/cic/datasets/nsl.html], 2009.

[23] 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*. IEEE, 2009, pp. 1–6.

[24] 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)*. IEEE, 2015, pp. 1–6.

[25] ——, "The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Information Security Journal: A Global Perspective*, vol. 25, no. 1-3, pp. 18–31, 2016.

[26] 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.

[27] S. Russell and P. Norvig, "Artificial Intelligence: A Modern Approach," in *Malaysia: Pearson Education Limited.*, 2016.

[28] A. Jain, J. Mao, and K. Mohiuddin, "Artificial neural networks: A tutorial," *Computer*, vol. 3, pp. 31–44, 1996.

[29] S. Balakrishnama and A. Ganapathiraju, "Linear Discriminant Analisys - a brief tutorial," *Institute for signal and information processing*, vol. 18, pp. 1–8, 1998.

[30] P. A. A. Resende and A. C. Drummond, "A survey of random forest based methods for intrusion detection systems," *ACM Computing Surveys*, vol. 51, no. 3, 2018.

[31] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE transactions on information theory*, vol. 13, no. 1, pp. 21–27, 1967.

[32] Y. Cheng, "Mean Shift, Mode Seeking, and Clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[33] K. Khan, S. Fong, S. U. Rehman, K. Aziz, and I. Science, "DBSCAN : Past , Present and Future," in *The fifth international conference on the applications of digital information and web technologies (ICADIWT 2014)*. IEEE, 2014, pp. 232–238.

[34] H. Lee and S. Choi, "PCA+HMM+SVM for EEG Pattern Classification," in *Seventh International Symposium on Signal Processing and Its Applications*, 2003, pp. 541–554.

[35] G. Sarossy, "Anomaly detection in Network data with unsupervised learning methods," [urn:nbn:se:mdh:diva-55096], 2021.

[36] J. Hautsalo, "Using Supervised Learning and Data Fusion to Detect Network Attacks," [urn:nbn:se:mdh:diva-54957], 2021.