Network Data Classification through Artificial Neural Networks and GenClust++ Algorithm

Ichrak LAFRAM, Siham EL IDRISSI, Aicha Marrhich, Naoual BERBICHE, Jamila EL ALAMI

Abstract - Information systems are becoming more and more complex and closely linked due to the exponential use of internet applications. These systems are encountering an enormous amount of traffic, this traffic can be a normal one destined for natural use or it may be a malicious one intended to violate the security of the system. Therefore, a defense method needs to be in place. One of the commonly used tools for network security is the Intrusion Detection System (IDS). An IDS, while ensuring real - time connectivity, tries to identify fraudulent activity using predetermined signatures or pre-established network behavior while monitoring incoming traffic. Intrusion detection systems based on signature or behavior cannot detect new attacks and fall when small deviations occur. Also, current anomaly detection approaches suffer often from high false alarms. As a solution to these problems, machine learning techniques are a new and promising tool for the identification of attacks. In this paper, the authors present a hybrid approach, combining artificial neural networks and a hybrid clustering algorithm based on k-means and genetic algorithm called GenClust++. The final framework leads to a fast, highly scalable and precise packets classification system. We tested our work on the newly published dataset CICIDS 2017. The overall process was fast, showing high accuracy classification results.

Index terms: Intrusion Detection, Machine Learning, Traffic Classification, Artificial Neural Networks, Clustering, GenClust++.

I. INTRODUCTION

The explosive growth of the Internet and the continued expansion of data systems is a major achievement in nowadays, but the world of massively connected computers has a damaging side caused by the malicious purposes of hackers intending to break the systems.

Since the internet ecosystem involves many menaces of attacks, different programs have been placed in action in order to block attacks on the Internet. One of these programs, is Intrusion Detection System (IDS) which aims to block external attacks helping the network withstand the damage.

IDSs goal is to provide a defense wall to confront computer networks attacks on the Internet. An IDS, aims to spot different forms of malign network communications and computer programs, while this task cannot be performed by conventional tools like firewalls.

Revised Manuscript Received on August 05, 2019.

Ichrak lafram, LASTIMI Laboratory, Superior School of Technologies of Sale, Mohammadia School of engineering, Mohamed V University city of Rabat, Morocco

Siham El idrissi, LASTIMI Laboratory, Superior School of Technologies of Sale, Mohammadia School of engineering, Mohamed V University city of Rabat, Morocco.

Aicha Marrhich, LASTIMI Laboratory, Superior School of Technologies of Sale, Mohammadia School of engineering, Mohamed V University city of Rabat, Morocco.

Naoual Berbiche, LASTIMI Laboratory, Superior School of Technologies of Sale, Mohammadia School of engineering, Mohamed V University city of Rabat, Morocco.

Jamila EL Alami, LASTIMI Laboratory, Superior School of Technologies of Sale, Mohammadia School of engineering, Mohamed V University city of Rabat, Morocco.

In general, IDSs can be subdivided into two categories:

Signature-based systems which may detect attacks based on a pre-established behavioral pattern, therefore a signature must be created for every attack in order to identify it.

The problem is that it easily falls against attacks with self-modifying behavioral characteristics.

On the other hand, anomaly-based detection in which the system finds intrusions based on deviations from normal patterns. Anomaly detection models compare incoming packets data to system profile learned from the training data. If the data collected show deviations from the normal behavior, the anomaly-based system classifies the data as abnormal/malicious. The main challenge of anomaly- based detection systems is the difficultly of defining rules to set the normal behavior.

Both approaches utilize patterns of already known attacks to identify unknown intrusions.

These static approaches are actually experiencing a myriad of issues and usually lead to poor detection systems. Unable to detect variations of known attacks therefore new and unknown attacks cannot be identified. Besides, they suffer from high false-alarm rates and require constant regular updates, therefore a loss of detection in real time [1].

Machine learning techniques have been proposed to start breaking further into the field of intrusion detection as a redress to these issues. These techniques are imprecision and uncertainty - tolerant which make them very useful where optimization and precision are most needed [2].

The use of these techniques aims to train algorithms to learn from existing data so that the class of each entry can be predicted. One big issue on training algorithms is the necessity of a training set, a labeled dataset in order to make the algorithm capable of generalization and predicting.

There are various ways by which we can implement an intrusion detection system. In recent years, artificial neural networks have been used in a wide range of applications, showing very good results, especially for network packets identification and attacks detection in networks. ANN has been successful because of their huge approximation capacity.

Unfortunately, there are limitations on using individual ANN which can lead to poor and inefficient system. Some of these limitations are due to the training process of ANN which completely relies on available training data. In case, the data suffer from an issue, such as non-representativity or bad distribution of features, the whole learning process will be biased and the artificial neural network will be incapable to accomplish a good generalization which will lead to the foremost downside of individual ANN based IDS: Poor detection accuracy. Another issue is the feeble detection stability caused by the imbalanced distribution of diverse attacks present in the dataset and particularly low frequent attacks [3].

In this paper, we will introduce a new detection approach where the ANN classification decision is built upon a high-quality clustering method based on the combination of MK-means and a hybrid algorithm called GenClust++ [4].

The aim of applying a clustering algorithm before applying ANN models directly is, on one hand, to overcome the feebleness of neural networks in interpreting correlations between inputs and targets in high dimensionality problems [5], on the other hand, we didn't want to use dimensionality reduction techniques before clustering to not lose potential useful information. In our system, no labeled data is necessary for training.

The model presented here has been evaluated using the "CICIDS 2017" the recently published dataset from the Canadian Institute for Cyber security [6].

With regard to the current state of the art, our approach has several advantages. Primary and foremost, it works completely unsupervised, meaning that any real time system can use it directly and start working immediately, without any adjustment or training step. Moreover, it uses a high-quality clustering technique to overcome common clustering problems such as specification of number of clusters, sensitivity to initialization or structure-hiding by irrelevant features [7].

This paper is arranged in the next way: After introducing our work, we will discuss some works linked to the matching background. Then, we will present the methodology we actually followed in our research study in a third section. The fourth section will be dedicated to present our model and to present data and metrics used for evaluation. After that, we present the experimental results validating this approach. Finally, we draw a conclusion and discuss future work.

II. RELATED WORKS

In our approach, we combine unsupervised and supervised techniques for packets traffic identification. The majority of existing work using either supervised or unsupervised techniques deal with intrusion detection on the basis of the KDDCup99 dataset.

Numerous research studies based on unsupervised detection methods use clustering and detection of outliers.

In [8], Eskin et al. show good results with three partitioning algorithms: a fixed-width clustering, an optimized version of KNN, and a one-class support vector machine.

Xiangmei Li [9] proposed to optimize IDS based on neural networks using multi-sub-classifiers system. It deals with the four types of attacks present in the dataset, and for every type of attacks he assigned a specific classifier. The four types of attacks are DOS, Probe, Remote 2 Local and User 2 Root.

Even though the accuracy of the proposed system is optimized and better results have been shown, it has not shown how quickly the classification process is, since each sub-classifier should inspect the entire incoming traffic causing redundant and time - consuming process.

PCA is used for a large set of purposes, including subspace understandings, identification systems, abnormality detection and classification. PCA is actually processed in batch mode and all data must be accessible for the calculation of the PCA. Thus, batch PCA is impractical for stream data applications such real time intrusion detection, as it must be retrained whenever new data comes. To remedy this situation, several iterative algorithms were used, like the least square recursive, stochastic gradient, etc. [10]

Other studies proposed a hybrid approaches for intrusion detection by combining supervised ANN with unsupervised ANN, or by using ANN in combination with other data mining techniques.

Chen, et al. [11] proposed a hybrid intrusion detection approach based on a flexible neural tree combined to an evolutionary algorithm and particle swarm optimization. The results shown, has proved the efficiency of the method.

Jirapummin et al. [12] employed a hybrid approach to visualize intrusions using Kohenen's SOM and also to classify intrusions using a resilient propagation neural network.

Admittedly, different methods for constructing a hybrid IDS have been proposed, but how to design and implement this system greatly influences performance when detecting intrusions. Our motivation for using the hybrid ANN is to overcome the limitations of individual ANN discussed below in the first part.

III. METHODOLOGY

In this section, we present different stages of our framework of an IDS based on Genetic K-means Clustering and Artificial Neural Networks.

The first phase: vectors are constructed, normalized and then clustered using the GenClust++ algorithm.

The second phase: individual artificial neural networks classify every incoming vector based on its cluster.

1) Vector Construction:

The prediction accuracy may be altered by the presence of irrelevant or redundant attributes. We will perform two types of feature selection in order to improve the classification accuracy and the total computation time. The first one, is done before the clustering phase, in which we will use the Fast Correlation-Based Filter, described here [13] in order to eliminate irrelevant features, the ones taking only one single value for example, carrying no information about input vectors.

Once different clusters are built and since we have vectors of the same cluster, we used the recursive feature elimination algorithm where different subsets of features are evaluated and compared to each other in order to determine the best subset based on the model accuracy. We need to select features because redundant discriminators increase over-fitting or if highly correlated with another discriminator worsen the accuracy.

Some applications use IP layer encryption, our approach is non – payload based, so it is robust to encryption.

2) Unsupervised Clustering

Unsupervised machine learning is a branch of machine learning where we deal with unlabeled data, in other words, where data has not been labeled in advance to be identified (whether or not incoming network traffic is an attack in our case).

Clustering methods are either supervised or unsupervised. Generally, they are unsupervised which is the case in this study, because supervised clustering requires labeled datasets in order to train the algorithm. On the other hand, unsupervised

clustering places records of a dataset into groups based on their similarities with no need for class labels.

Unsupervised machine learning is effective and very useful for data security problems, because it is really difficult to set accurately labeled data in this field and unsupervised learning makes it possible to identify new (or abnormal) observations in data all without necessarily having seen an example of this behavior before [14].

There exist a large set of clustering algorithms, and we will focus on (*k-means* and *GenClust++*) that have been used to achieve high quality clustering.

3) The Classification Framework

In order to achieve the network traffic classification task, we used a single artificial neural network for every cluster. The choice of ANN is due to their capability of dealing with uncertain and noisy data, which makes them the most used machine learning techniques in intrusion detection [15].

An ANN is an information processing system containing a large number of highly interconnected elements called neurons. The ANN learning process is an optimization process aiming to find the best weights for neurons connections. This is achieved by introducing a number of inputs (in our case, inputs are vectors coming from each cluster) used to generate an output that matches the desired output (in our case, the class of the vector whether an attack or not). The number of ANN classifiers of our architecture depends on the number of clusters generated by the GenClust++ algorithm used in first stage, which makes our architecture extensible and adjustable for any kind of networks.

IV. PROPOSED WORK

we built a two-level network traffic identification framework based on artificial neural networks through high quality clustering technique. It's a combination of GenClust++ Clusters and n-artificial neural network classifier.

The final framework is a three-stage framework:

Level one:

Stage I: The incoming traffic data is normalized and each input vector is constructed.

Stage II: Clusters identification.

Level Two:

Stage III: Every output vector of every cluster will pass through its specific artificial neural network trained to classify it.

Each ANN is trained only on one cluster in order to reduce the training time.

1) The diagram

In order to achieve high accuracy and minimize the false positives rate, the incoming vectors are rapidly identified and directed to its group and then it undergoes a fast feature selection process to be finally evaluated by the appropriate artificial neural network to determine whether it is a normal or an attack traffic.

The diagram below illustrates the proposed framework:

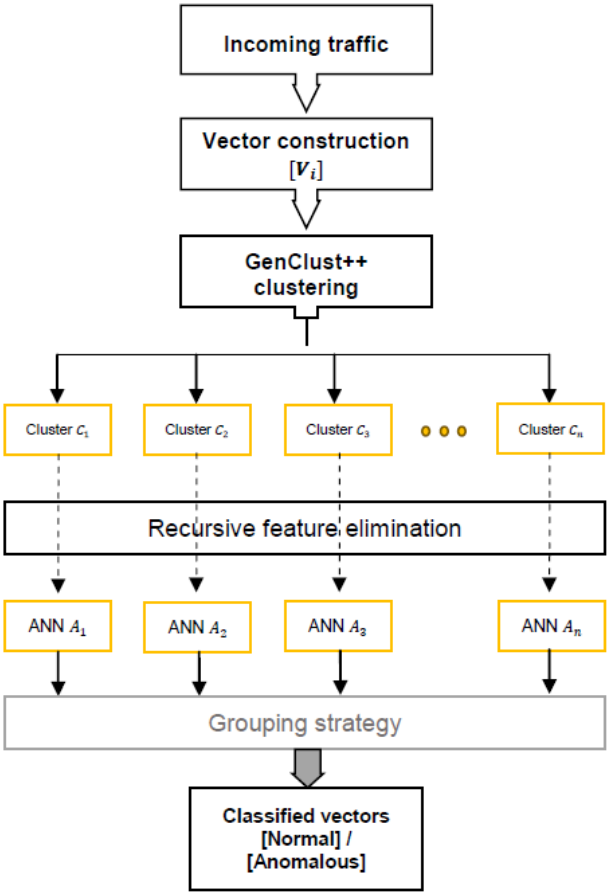

FIGURE 1: Classification Diagram

Incoming packet: Whole packet information passing by the network.

Vector V_i[id]: The incoming traffic is normalized processed and vector is constructed.

GenClust++ clustering: The algorithm specifies the cluster to which the incoming packet belongs.

RFE: A fast feature elimination process to avoid over-fitting.

ANN id: Individual artificial neural network trained only on a specific cluster.

Clusters are constructed based on the incoming data, the number of clusters is not fixed. It may increase or even decrease over time depending on the learning process which allows our model to be adaptable and evolutive. This is mainly for the unknown traffic which may not be classified correctly in order to refine the overall accuracy of our model.

The Algorithm

ALGORITHM: PACKETS CLASSIFICATION
INPUT: incoming packet Pi
OUTPUT: Class Pi: (normal/attack)
BEGIN
/*/**
FOR i **FROM** 1 **TO** n **DO**
/* incoming traffic packets Pi
FOREACH Pi **DO**
/* rapid packets identification*/
Create input vector
V_i
Normalize vector attributes

/* GenClust++ Clustering
GenClust++ clustering: V_i is clustered
Set corresponding identifier to V_i
V_i Becomes V_i [+id = cluster identifier]
/* Input feature selection */
FOREACH V_i (id) **FROM** Cluster_id ∈[1, N] **DO**
Select best features subset for V_i [id]
V_i [id] becomes U_i [id]
Send U_i [id] to ANN [id]
END
/* Artificial neural network classification */
Classify U_i [id]
U_i [id] classified: Normal/Anomalous
END
Return Class (Pi)
END
/*/**
END

A) TECHNICAL OVERVIEW

GenClust++ is an unsupervised clustering technique that deal with datasets without class labels, aiming to find hidden structure and groups within data. This process of clustering is omnipresent in a wide-ranging type of applications.
Zahidul et al. [16], states that even if K-MEANS algorithm offers a very efficient gradient descent approach to the total squared error of representation, it is profoundly affected by noise and demands setting up the parameter k.
As a solution to this, they built an effective genetic algorithm that combines the capacity of genetic operators to consolidate different solutions of the search space with the exploitation of the hill-climber. Fast hill-climbing cycles of K-MEANS results on an algorithm that is faster than its predecessor and achieves clustering results of higher quality.

1) GenClust++:

Genetic algorithms have been used along with K-MEANS to avoid the requirement of the user input on k and improve the cluster quality by exploring good quality optima [17]. Unlike many existing techniques [18], GenClust++ does not require any user input on the number of clusters k or radius of a cluster r.
To seed the initial population, GENCLUST++ applies K-MEANS multiple times with different k values ranging from k = 2 to k = 10, based on the supposition that generally the number of clusters k in a dataset varies between these values.

2) k-means

The *k-means* is a well-used unsupervised clustering algorithm which process different sets of clusters, the cluster having the lower sum of squared error is the best one to consider. Then every entry is allocated to just one cluster.
Sum of squared error is defined by:

$$SSE = \sum_{i=1}^{k} \sum_{C_i} dist(c_i, x)^2 \ where \ \forall x \in C_i \qquad (1)$$

C_i : i^{th} cluster
c_i : centroid of C_i

$$SST = \sum dist(C_m, x)^2 \ where \ \forall x \ in \ Dataset \ D \qquad (2)$$

Despite its advantages and its performance dealing with big datasets, k-means algorithm suffers from a key weakness:
In realistic settings, it may be extremely difficult for a user to guess the number of clusters in advance [7].
Bad initial seeds can easily lead K-MEANS to poor clustering results.

As a solution to the problem of choosing the right value for k number of clusters, an approach for providing the value of *k* is to determine the value of the objective function and analyze its change based on different values of *k*. Unfortunately, when dealing with data of big size such as the case of network traffic data, this method is inefficient. To overcome this, we will use a variant of k-means called x-means.
With *k-means* algorithm, we have to specify an array of the potential values of k. Then we set "two" as the min value and the value of the square root of half the data size as the max range [19].

$$k = \sqrt{\frac{n}{2}} \qquad (3)$$

Following, the algorithm improves its parameters starting with the min value of k and continues until finding its maximum.

$$BIC(M_j) = l_j(A) - \frac{p_j}{2} \times logS \qquad (4)$$

At the same time, it keeps improving its structure until the best centroids are found based on the calculation of the BIC values (Bayesian information criterion) (4).
Every cluster will then be divided into two slices in order to move the newly formed centroids in the opposite direction with distances proportionate to the size of the area.
The k-means are applied in each original clustered area through k equal to two and the BIC calculated for k equal 1 and 2.
Using Eq. (4):

$$\text{If: } BIC(k = 1) > BIC(k = 2),$$

The cluster then regains his original centroid.
Otherwise, the divided form is retained. Where:
l_j(A): log-likelihood of the dataset A for the j^{th} model in the point of max probability,
p_j : Number of parameters within M_j (a space of alternative models).
S: Size of the dataset *A.*

3) Genetic algorithm

In order to better understand the GenClust++ algorithm, we will briefly describe its multiple components. One of them is genetic algorithm which was used in combination with the k-means to generate better clusters.
Genetic algorithms make use of reproduction, natural selection and genes diversity as key concepts in the solution discovery process explained briefly as follows:
Generate initial population - The process of generating random sequences to initiate the genetic algorithm with regard to the possible space of allowed gene symbols.
Selection – Selecting the optimum symbols amongst all individuals.
Crossover – Selected individuals are combined with each other in order to produce new population individuals with the best possible genes inherited from their parents. The elitism guarantees, that the optimization function will not produce the worst results.
Mutation – Whenever no individuals had the solution genes, a mutation (random alteration on some genes) might guarantee generation of solution genes.
New generation – A combination of genes processed through crossover and mutation with individuals coming from the selection process.

4) GenClust++

As it is a combination of a genetic algorithm and k-means, the GenClust++ main components are:
1: The dataset Normalization.
2: An alternative to K-MEANS called MK-MEANS.
3: Initial Population with Probabilistic Selection.
4: Crossover.
5: Elitism.
6: Mutation.
7: Probabilistic Cloning with MK-MEANS.
8: Chromosome selection
While genetic algorithm is initialized with random chromosomes, it is not the case for GenClust++, it is instead initialized with better chromosomes qualified as (good quality chromosomes) using the hill-climber property of the k-means algorithm. Therefore, GENCLUST++ makes use of numerous k-values and for each k-value it runs several instances of MK-MEANS [4] (or MK-MEANS++) variants of K-MEANS and (K-MEANS++) that can handle both numerical and categorical attributes; Exploring these different k values give good clustering results. Then the initial population of individuals (clustering solutions) are initialized with the best k-values.
After that, it applies elitism to the vanilla genetic algorithm in order to improve the average population fitness at regular intervals by cloning chromosomes (i.e. those with high fitness values) and substituting low performing chromosomes by the clones of high fitness chromosomes. Then, comes the mutation process during which cloned chromosomes are processed in a way that random new chromosomes are created for more divergence of the solutions space [4].
The diverse chromosomes are then processed in a short length by the hill-climber of MK-MEANS algorithm. This operation is performed for better results. This application of hill-climber here, is called polishing and it aims, not to obtain a final clustering decision but, to repair any slight fitness damage that might be caused by the mutation process in order to preserve a population of high achievers[4].
GenClust++ also performs a comparison between chromosomes of two following generations. Amongst ancestor chromosomes and descendants' chromosomes, then it picks the best chromosomes between them for the next resulting generation. This is to preserve the impact of the elitism in order to guarantee that strong progeny of cloning and polishing does improve the population fitness and that again, the random crossover and mutation operations do not introduce very low performers.

5) Artificial Neural Networks

A neuron is a neural network's basic processing unit. It is linked to information sources as input and returns output information.
The neuron receives a number of input data, each data point is processed and transmitted with its weight.

A weight is a coefficient w_i calculated for every data point x_i. The i^{th} neuron receives the information $(w_i \times x_i)$. This data is conceded to the activation function in order to get the final output.

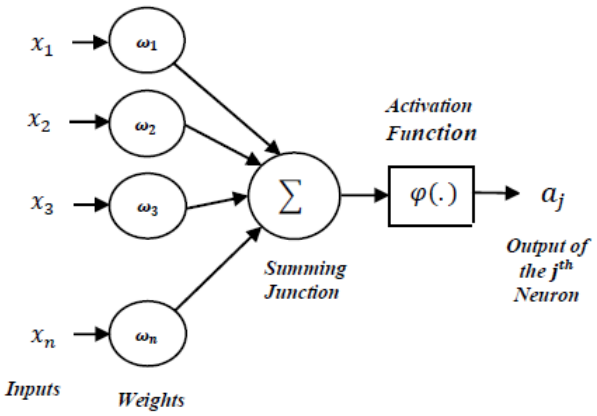

FIGURE 2: SINGLE ARTIFICIAL NEURON

We denote:

$\omega_{i,j}$ For $(1 \le i \le n)$ and $(1 \le j \le p)$, the weight connecting the information x_i and the neuron j
And
a_j the output of the j^{th} neuron, defined by the following equation:

$$\forall\, 1 \le j \le p : a_j = \varphi\left(\sum_{i=0}^{n} w_{i,j} \times x_i\right) \qquad (3)$$

The output a_j may become a stimulus for neurons in the next layer.
We used the sigmoid as the activation function given by:

$$\varphi(\gamma) = \frac{1}{(1+e^{-\gamma})} \qquad (4)$$

The non-linearity of the activation functions within hidden layers makes artificial neural networks one of the best universal approximator [20].
During the learning stage, training data is given as pairs of $(x_k, d_k), k = 1, \ldots, P$ where d_k is the desired outputs for inputs x_k of P training samples.
The Backpropagation learning algorithm is applied for minimization of error function [20] defined by:

$$E = \frac{1}{2}\sum_{k \in T_r}\sum_{j=1}^{m}(y_j(x_k, \omega) - d_{jk})^2 \qquad (5)$$

d_{jk} : j^{th} element of d_k
$y_j(x_k, \omega)$: j^{th} neural network output for input x_k
T_r : A set of training data

Validation & Evaluation metrics

1) Dataset

The lack of suitable free datasets to evaluate anomaly detection systems is the biggest challenge an evaluation can face.
To measure the efficiency of any detection approach, we must study it and experiment it with data that replicates the actual traffic of modern networks at an adequate level.
Datasets available in the field of network intrusion detection systems for machine learning are limited. DARPA dataset (KDD cup99, NSL-KDD) is one of the few yet at the same time widely

used datasets. Even though it is the most comprehensive datasets available, it may not be a great representative of currently existing realistic networks and yet still suffer from diverse problems discussed in [21].

Most works in the field of intrusion detection using machine learning techniques, focus on accuracy over productivity and are tested on the old DARPA dataset.

There exist other datasets for the same purpose, but some of these datasets actually lack volumes and variety of traffic and are composed of packets that do not reflect modern trends [22].

Based on the evaluation framework elaborated by [23], there are critical standards necessary to build a consistent benchmark dataset. In the following, here are these criteria:

- **Complete Traffic:** different machines and real-world attacks.
- **Labeled Dataset:** benign and attack labels
- **Complete Interaction:** separated networks
- **Complete Capture:** All traffic
- **Attack Diversity:** Most common attacks [24].

In our study, we wanted to test our approach on data close to the one generated in our network. To gauge the performance of any detection approach, we need to really experience it with data that simulates real traffic to an adequate level in modern networks.

To this end, we have opted for a fairly current IDS assessment dataset containing real life traffic data. The CICIDS2017 dataset from Canadian Institute for Cyber security in 2017.

In a physical testbed implementation, the CICIDS2017 data set was generated using real devices that interactively generate real traffic reflecting complex and realistic network traffic of nowadays infrastructures [6].

1.a) Traffic composition of the CICIDS2017 dataset

CICIDS2017 Dataset comprises fairly benign, most current, common attacks, which reflects real-life data. Besides, it contains labeled entries

of the traffic analysis based on the timestamp, source and destination IPs, source and destination ports, protocols and attacks.

Total number of instances	2830540
Number of features	83
Number of classes	15

Table 1: Characteristics of CICIDS2017 dataset

During five days period, the data collection began on Monday and ended on Friday. While there was only benign traffic on Monday, multiple attack scenarios were generated during other days of the period.

The table below shows the attacks which were performed in mornings and afternoons of the next four days.

Days	Labels
Monday	Benign
Tuesday	BForce, SFTP and SSH
Wednesday	DoS and Hearbleed Attacks slowloris, Slowhttptest, Hulk and GoldenEye
Thursday	Web and Infiltration Attacks Web BForce, XSS and Sql Inject. Infiltration Dropbox Download and Cool disk
Friday	DDoS LOIT, Botnet ARES, PortScans (sS,sT,sF,sX,sN,sP,sV,sU, sO,sA,sW,sR,sL and B)

Table 2: Class Labels of CICIDS2017 Dataset

Below is the traffic distribution of the captured data, it follows a normal distribution.

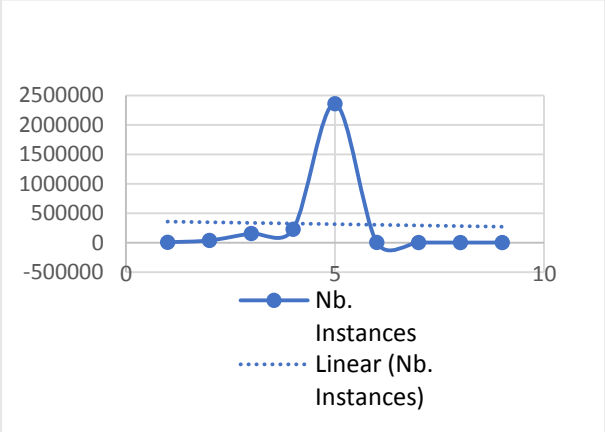

FIGURE 3: Traffic Distribution of the CICIDS2017

2) Metrics

In this stage of analysis, we used common metrics to evaluate information retrieval in order to be capable of comparing results with other studies, especially the ones based on the same dataset:

Precision (Pr):

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

Defined as the ratio of correctly classified attack entries TP (True positive), against all the classified entries TP+ FP (False positive)

Recall (Rc):

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

It is the ratio of correctly classified attack entries TP against all generated entries TP+FN (False negative)

F-Measure (F1):

$$F1 = \frac{2}{\frac{1}{Pr} + \frac{1}{Rc}}$$

A harmonic mixture of the precision and recall.

V. EXPERIMENTAL RESULTS

We tested our framework on both datasets (KDDCup/NSL-KDD and CICIDS17) in order to measure its performance and also to compare our results with studies based on these datasets.

1) Results comparison using KDD Cup99 dataset:

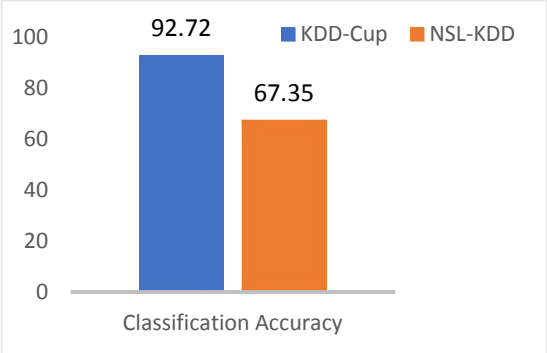

FIGURE 4: Performance of The model on KDDCup and NSL-KDD.

With regards to the (KDDCup / NSL-KDD) datasets, the results show that the performance of the classifier has declined clearly between the two datasets which confirm that the data used is the main important part of the learning process and affects the detection performance. This is to justify our choice of datasets.

The model performed well on the KDD-CUP99 dataset because of the high level of redundancy in it. It clearly declined when the same model was tested on the NSL-KDD which may be explained by an over fitting.

2) Results using CICIDS2017 dataset:

The detection model has to classify the incoming traffic whether normal or attack. We aimed to optimize the existing ANN based intrusion detection systems that classify the incoming traffic directly by analyzing the whole packets. This is done through a first clustering phase, then vectors of every cluster are classified separately.

The results below show the accuracy of the model on every cluster:

	X-means	**GenClust++**
Cluster 1	95.11	99.97
Cluster 2	91.31	98.62
Cluster 3	98.18	98.94
Cluster 4	93.82	99.96

Table 3: Classification accuracy results before and after cluster optimization

The best results given by the GenClust++ algorithm is for four clusters. We wanted to see if we give the same number of clusters k=4 for k-means algorithm to compare the results and to analyze the quality of the clusters. The results show that even for the same number of clusters there are differences in entries which explain the quality of every clustering method.

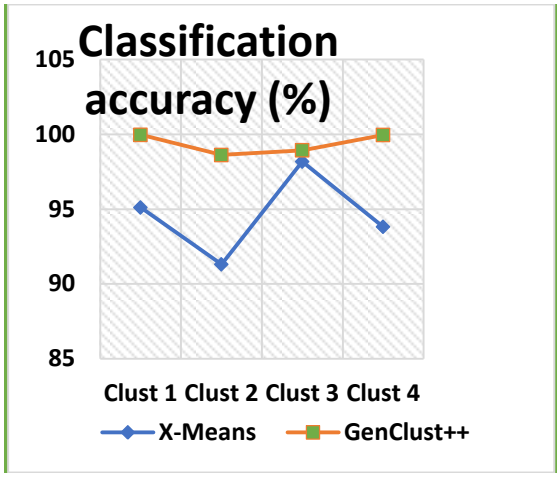

FIGURE 5: MODEL ACCURACY COMPARISON ON BOTH K-MEANS AND GENCLUST++ CLUSTERS.

We also, tested different classification model on the formed clusters and here are the results comparison between our model and models used by [40]:

Algorithm	***Pr***	***Rc***	***F1***
MLP	0.77	0.83	0.76
Adaboost	0.77	0.84	0.77
Naive-Bayes	0.88	0.04	0.04
KNN	0.96	0.96	0.96
Random Forest	0.98	0.97	0.97
GenClust++ANN	0.99	0.98	0.98

TABLE 4: CLASSIFICATION RESULTS COMPARISON

The table above gives an overview of the performance evaluation in terms of average of our evaluation metrics for five commonly used machine learning algorithms, Multilayer perceptron (MLP), Adaboost, Naive-Bayes, K-Nearest Neighbors (KNN), Random Forests and our model.

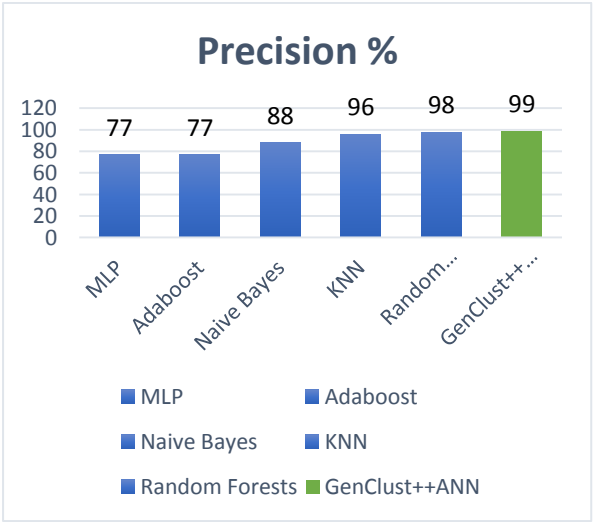

FIGURE 6: DIFFERENT ALGORITHMS APPLIED TO CICIDS2017

Another important metric to take into consideration while designing a model for network traffic classification is the execution time. The detection time has decreased by applying our model.

Algorithm	*Execution time (Second)*
MLP	75.73
Adaboost	26.24
Naive-Bayes	4.77
KNN	18.23
Random Forest	4.39
GenClust++ANN	0.27

Table 5: Comparison of the model time execution

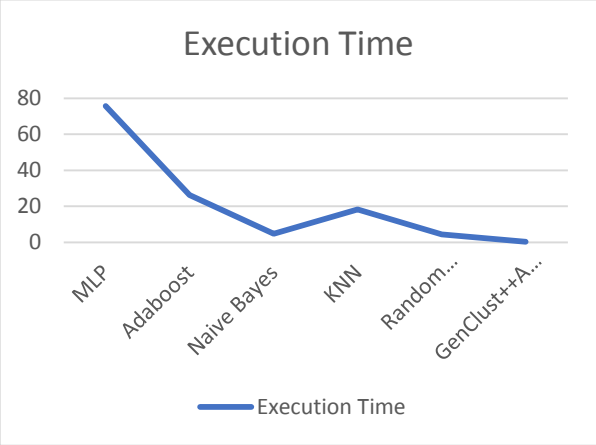

FIGURE 7: ALGORITHMS EXECUTION TIME ON CICIDS2017

Better results are achieved by our model, this is because when artificial neural network processes on one cluster, it makes it faster and accurate. The training time is reduced because the ANN doesn't need to make computational operations on the whole traffic.

The results show that each sub classifier is more accurate, shorter in training time than the classifier of all features and the entire intrusion detection system is higher in the detection rate and less in the false negative rate than the primal intrusion detection system.

VI. CONCLUSION

In this study, we presented an approach of intrusion detection optimization based on the combination of both unsupervised and supervised machine learning techniques in order to cover the whole process of traffic identification.

In addition to these high performances, the proposed framework is tested on a newer dataset and thus more representative of current computer networks. We count on that to make sure that the proposed framework is best suited for modern network architectures and we are designing a cloud based distributed infrastructure for real time traffic analysis for better results.

The framework of the elaborated model will be combined with our previous work [25] to constitute a final complete architecture for network traffic classification. Both studies showed great results and encourage us to elaborate an optimal model based on a mixture of advanced machine learning techniques that will be tested with a platform for real traffic capture [26].

Our focus in the coming studies will be on stream data from real time network traffic.

REFERENCES

1. A.Jelsiana Jennet, Dr. J Frank Vijay. International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 17 (2015) pp.12635-12641.
2. Piero P.Bonissone, "Soft computing: the convergence of emerging reasoning technologies," Soft Computing Journal, vol 1, no 1, pp. 6-18, Springer-Verlag 1997
3. Beghdad, R. (2008). Critical study of neural networks in detecting intrusions. Computers and Security, 27(5-6), 168–175
4. Rahman, M. A., & Islam, M. Z. (2014). A hybrid clustering technique combining a novel genetic algorithm with k-means. Knowledge-Based Systems, 71, 21 - 28. Retrieved from http://dx.doi.org/10.1016/j.knosys.2014.08.011
5. Jacob Kogan, "Introduction to Clustering Large and High-Dimensional Data", University of Maryland, Baltimore County
6. Sharafaldin, I., Gharib, A., Habibi Lashkari, A., and Ghorbani. Towards a reliable intrusion detection benchmark dataset. Software Networking, 2017.
7. A. K. Jain, "Data Clustering: 50 Years Beyond K-Means", in Pattern Recognition Letters, vol. 31 (8), pp. 651-666, 2010
8. E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo, "A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data," in Applications of Data Mining in Computer Security, Kluwer, 2002
9. 978-1-4244-5143-2 ©2010 IEEE DOI**:** 10.1109/ITAPP.2010.5566641 Internet Technology and Applications, 20-22 Aug. 2010
10. Bannour S. and M. R. Azimi-Sadjadi. Principal component extraction using recursive least squares learning, Neural Networks, IEEE Transactions on, 1995,6, 2. 457–469
11. Chen, Y. H., Abraham, A., & Yang, B. (2007). Hybrid flexible neural-tree-based intrusion detection systems. International Journal of Intelligent Systems, 22(4), 337–352
12. Jirapummin, C., Wattanapongsakorn, N., & Kanthamanon, P. (2002). Hybrid neural networks for intrusion detection system. Proceedings of ITC–CSCC, 928–931
13. Lei Yu and Huan Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In Proceedings of the Twentieth International Conference on Machine Learning (ICML 2003), 2003
14. Jacob Kogan, "Introduction to Clustering Large and High-Dimensional Data", University of Maryland, Baltimore County
15. Axelsson S.: Intrusion detection systems: A taxonomy and survey. Technical Report No 99-15, Dept. of Computer Engineering.
16. Zahidul Islam, et al. Combining K-Means and a Genetic Algorithm through a Novel Arrangement of Genetic Operators for High Quality Clustering. Journal of latex class files, vol. 13, no. 9, september 2014 2
17. Maulik & Bandyopadhyay, 2000; Bandyopadhyay & Maulik, 2002; Laszlo & Mukherjee, 2007; Liu, Wu, & Shen, 2011
18. Lloyd, S. P. (1982). Least squares quantization in PCM. IEEE Transactions on Information Theory, 28(2), 129 –136
19. Mardia KV, Kent JT, Bibby JM (1979) Multivariate analysis. Academic Press, London
20. Hornik, K., M. Stinchcombe, and H. White, ''Multilayer Feedforward Networks are Universal Approximators,'' Neural Networks, Vol. 2, 1989, pp. 359–366
21. J. McHugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory," ACM Transactions on Information and System Security, vol. 3, no. 4, pp. 262–294, 2000
22. Brown C, Cowperthwaite A, Hijazi A, Somayaji A. Analysis of the 1999 DARPA/Lincoln laboratory IDS evaluation data with netadhict. Computational intelligence for security and defense applications. Piscataway, NJ, USA: IEEE Press; 2009. p. 67e73
23. Gharib, A., Sharafaldin, I., Habibi Lashkari, A., and Ghorbani, A. A. (2016). An evaluation framework for intrusion detection dataset. In 2016 International Conference on Information Science and Security (ICISS), pages 1–6
24. https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-dec-2016.pdf
25. Ichrak Lafram, N.Berbiche, J.Elalami, A random forest estimator combined with n-Artificial neural network classifiers to optimize network intrusion detection. IJAER, ISSN 0973-4562 Volume 12, Number 16 (2017) pp. 5835-5843
26. S. El idrissi, N.Berbiche, F.Guerouate, Performance evaluation of web application security scanners for prevention and protection against vulnerabilities. IJAER Vol 12 n 17 (2017)