Intrusion Detection in Computer Networks Using Hybrid Machine Learning Techniques

Deyban Perez[1], Miguel A. Astor[1], David Perez Abreu[1,3] y Eugenio Scalise[2]
Central University of Venezuela, Caracas, Venezuela
[1]Laboratory of Mobile and Wireless Networks - ICARO
[2]Center of Software Engineering and Systems - ISYS
University of Coimbra, Coimbra, Portugal
[3]Departament of Informatics Engineering - DEI
{deyban.perez,miguel.astor,david.perez,eugenio.scalise}@ciens.ucv.ve

***Abstract*— The emergence of new networking paradigms like Cloud Computing and the Internet of Things has introduced new security challenges that deserve new mechanisms to guarantee the integrity, availability and confidentiality of information and services to the users. One of the currently most studied strategies to satisfy these necessities is the use of hybrid Machine Learning techniques to automatize the process of intrusion detection in computer networks. This paper presents the design, implementation and performance analysis of multiple hybrid Machine Learning models for the task of intrusion detection in computer networks. Our results show that the combination of supervised and unsupervised learning algorithms complement each other to the task of creating a model capable of adapting to the detection of known and unknown attacks.**

***Keywords*—Computer networks security, intrusion detection, attack, Machine Learning, hybrid Machine Learning techniques.**

I. Introduction

The developments in the field of Information Technologies (IT) have allowed the deployment of an ample quantity of services that ease the daily activities of the citizens. Paradigms like Cloud Computing (CC) and the Internet of Things (IoT) [1], [2] have enhanced the services available to citizens, while introducing new requirements -such as ubiquity support, real-time response and security of shared data- which must be satisfied by the IT in ways that allow the acceptance of these new trends by society.

Data security stands out among the new requirements introduced by CC and IoT. Despite the many security strategies for the IT that have been widely studied in the last twenty years [3], new challenges related to data sharing and access have emerged from the new paradigms ensuring the need for revision and possible evolution of the ways to satisfy the new security demands of the services offered to users in the previously mentioned environments [4].

In the past, security systems relied on rules established by experts; however, with the constant evolution of attacks, the previously mentioned strategy showed high rates of false alarms due to it's rigidity and lack of adaptation to new kinds of attacks [5], making researchers focus their efforts in other strategies to improve the situation.

One of the currently most studied method for automation of the process of intrusion detection in computer networks is the use of Machine Learning (ML) techniques. This strategy is based on two fundamental approaches [3]: 1) the use of supervised learning, which in the context of intrusion detection in computer networks shows good performance for the detection of known attacks, but not for new attacks; 2) the use of unsupervised learning, which under the same context mentioned previously shows good performance for the detection of new attacks, but also shows a high rate of false alarms.

Considering that each day new vulnerabilities are found in the systems that support the IT, and that the attacks to this kind of systems are in a constant state of evolution; new mechanisms are necessary to face these challenges. During the last few years, research in this area has leaned towards the use of hybrid models that combine supervised and unsupervised learning with the intention of achieving high hit rates against known attacks, without neglecting the possibility of identifying new attacks [3], [5], [6].

This paper shows a study of the performance of hybrid ML techniques in the task of intrusion detection in computer networks. The algorithms used are specifically the Neural Network (NN) and Support Vector Machine (SVM) with radial kernel algorithms for supervised learning, and K-Means for unsupervised learning as a complement to the other algorithms. Additionally, the use of Principal Component Analysis (PCA) and Gradual Feature Reduction (GFR) as feature selection techniques is also studied.

Unlike other similar work, we measure the performance of the models developed no just by using the training data set, but also with the testing data set. Additionally, we show new combinations of the algorithms mentioned previously. Afterwards, we analyse the causes and implications of the results obtained by the different proposed models in the area of intrusion detection in computer systems.

The rest of this paper is organized as follows. Section II introduces the basic concepts needed for later discussions. Section III shows a compilation of works related to this research. Section IV presents the design and implementation

considerations of our solution. In Section V, the analysis and discussion of the obtained results is depicted. Section VI shows our conclusions and proposals for future work.

II. Background

To use as the foundations of analysis and discussion of the result obtained, the following section introduces the key terms and technologies used in this research.

A. Computer network security

Security in the context of computer networks refers to the protection conferred upon an automatic information system with the objective of preserving integrity, availability and confidentiality of the information and resources of the system. The activities that attempt to violate these three fundamentals of security are known as attacks [7].

Attacks can be classified in two categories [8]: 1) attacks, that represent those actions that attempt to damage the computer system directly or to degrade its performance; 2) intrusions, referring to those activities where outside entities try to infiltrate a network to steal information. In this paper both terms are used indistinctly to indicate illicit actions that must be detected in order to guarantee the basic principles of computer network security. Many security mechanisms have been proposed to resist attacks and intrusions that guarantee the preservation of the basic principles of security for the users.

Intrusion Detection Systems (IDS) are some of the most prevalent security mechanisms in use [3]. These systems are based on the premise that anomalous behavior is notably different and separable from normal traffic and as such it can be detected. The main activities of an IDS are: 1) monitoring the inbound and oubound network traffic; 2) detecting anomalies and 3) notifying about the anomalies detected. Depending on the purpose of the IDS, these can be classified as Network Intrusion Detection Systems (NIDS) or Host Intrusion Detection Systems (HIDS). The focus of this paper is on NIDS [7]–[9].

B. Machine Learning

Machine Learning is defined as the field of study on the capacity of computer programs to learn without being explicitly programmed to do so. Specifically, it is said that a program learns from an experience E with respect to some task T and some performance measure P if its performance on T, as measured by P, improves with the experience E [10], [11].

C. Supervised learning

Supervised learning is a kind of ML which presents a set of predicting variables X and a set of labels Y, where each entry in X must be labeled by an element of set Y. In this way, the ML models have a teacher in the form of set Y, which orients the model during the training process. This kind of ML is used frequently to solve classification and regression problems. Some of the most used supervised learning algorithms are NN and SVM [3], [12].

The use of NN is motivated by the emulation of the behavior of the brain of living beings. This algorithm simulates the structure of a brain with a graph, where each vertex corresponds to a neuron and the synaptic process is modeled by assigning weights to the edges, weights that are adjusted with an error minimization technique. This algorithm is commonly used for regression and classification [13], [14].

On the other hand, SVM is an algorithm that associates the input data inside a characteristic space of a higher dimension, and then generates an optimal separating hyperplane in said characteristic space. The separating hyperplane is chosen maximizing the distance between the support vectors, which represent the boundaries between the different classes that are closest to the separating hyperplane. The hyperplanes are generated using functions known as kernels that allow different kinds of separations. The most commonly used kernels are [3], [12]: 1) linear; 2) polynomial; 3) radial and 4) sigmoid. Kernels 1) and 2) are self-descriptive. Moreover, kernels 3) and 4) allow the creation of circular separating hyperplanes.

D. Unsupervised learning

Unsupervised learning is a ML technique in which there is no set of labels Y that allows the identification of all the entries in the set of predicting variables X, in contras with supervised learning as described in Section II-C. In this way, this approach is focused on the search of patterns that allow the description of the dataset. One of the most popular unsupervised learning algorithms is K-Means [3].

K-Means is a grouping algorithm based in centroids. This algorithm positions K centroids in an N-dimensional plane and, using distance measures, it repositions the centroids iteratively in such a way that the entries of the dataset are absorbed by the closest centroid, classifying the entries. This algorithm is scalable to large volume datasets, usually converges to local minimums and is able to detect spherical groups. On the other hand, the algorithm requires that the number of centroids must be specified during startup, is sensitive to noise (outliers) and, depending on the initialization of the centroids, it can produce widely differing results [15], [16].

E. Machine Learning approaches

The application of different ML techniques to solve a problem is valid. The most commonly used approaches can be classified in three categories: 1) simple, where only one ML algorithm, referring to either supervised or unsupervised learning, is used; 2) hybrid, that makes used of both a supervised learning and one unsupervised learning algorithm, with the interest of having both algorithms complement each other in order to improve the performance of the model with respect to a certain task; 3) levels, in which multiple ML algorithms are used, organized by levels, these being either supervised or unsupervised learning algorithms.

F. Feature selection techniques

Feature selection techniques allow to identify which are the features of a dataset that truly provide information to the ML model, reducing the dimensionality of the information with the objective of obtaining faster and more accurate models.

Two of the most used feature selection techniques are PCA and GFR.

PCA [12] rotates the dataset with the intention of capturing the greatest amount of variance by use of a covariance matrix. The first columns of this matrix represent the components that accumulate the most amount of variance and, due to this, are those that provide the greatest amount of information. The principal components correspond to linear combinations of the features of the dataset. This method is effective when it is used with datasets that possess lots of features that can be properly represented by the columns (components) of the covariance matrix.

On the other hand, GFR [17] takes the complete feature set and temporarily eliminates one of them while it evaluates the performance of the model with the rest. Then, the features that were shown to be less significant from a performance standpoint are removed definitely. This process is repeated n times where n represents the size of the complete feature set. When the last iteration is complete, the technique determines which is the most relevant feature while the rest of the features are sorted in descending order according to their relevance, being the less relevant feature that which was removed during the first iteration of the algorithm.

G. Confusion matrix

The confusion matrix is one of the performance metrics most used by researchers in the area of ML [3]. Using confusion matrices it is possible to visualize graphically using a table the performance of ML models for a determined task. This makes use of four fundamental criteria: 1) true positives (TP) and 2) true negatives (TN), which reference the correct classification of entries as belonging to either the positive or negative class respectively; 3) false negatives (FN) and 4) false positives (FP), that indicate the amount of entries incorrectly classified as belonging to the negative and positive class, respectively. In Table I an example of a confusion matrix for two classes can be observed, where the first column correspond to the real values of the entries and the first row to the predictions made by the model. Afterwards, the entries that conform the diagonal represent the hits made by the model, distributed among the TP and TN, while the classification errors are found outside the diagonal, distributed among the FN and FP.

TABLE I
EXAMPLE OF A TWO-CLASS CONFUSION MATRIX.

	Positive	**Negative**
Positive	TP	FN
Negative	FP	TN

From the confusion matrix shown in Table I, we can derive other performance metrics that are described below:

- **Accuracy** measures the percentage of entries correctly classified by the model.

$$Accuracy = \frac{TP + TN}{N} * 100 \tag{1}$$

- **Error rate** measures the percentage of incorrectly classified entries.

$$ErrorRate = (100 - HitRate) \tag{2}$$

- **Sensibility** measures the percentage of entries belonging to the positive class that were classified correctly.

$$Sensibility = \frac{TP}{TP + FN} * 100 \tag{3}$$

- **Specificity** measures the percentage of entries belonging to the negative class that were classified correctly.

$$Specificity = \frac{TN}{FP + TN} * 100 \tag{4}$$

- **Precision** measures the percentage of hits over the entries of the positive class that were classified as belonging to the positive class.

$$Precision = \frac{TP}{TP + FP} * 100 \tag{5}$$

- **ROC curve** is a graphical performance metric where predictions are organized in a descending form, using the certainty generated by the models. Then, an accumulated function of the accuracy vs the error rate is graphed, allowing the visualization, organization and selection of classifiers based on their performance [18].

III. Related work

Much effort has been spent in the area of intrusion detection in computer networks using ML techniques. These efforts can be classified in three ample categories: 1) feature selection, 2) pre-processing and 3) model training. Category 1) is the most critical because it concerns the transformation of raw network traffic data into a final dataset that can be understood by the different ML algorithms, a process that is costly both in effort and time and makes the researchers focus on categories 2) and 3); category 2) centers on the study of techniques for feature selection in datasets; category 3) focuses in the performance testing of different ML algorithms for intrusion detection in computer networks [6], [19], [20].

The previously presented categories are shown in Fig. 1, which shows the the workflow in the process of developing an ML model for the task of intrusion detection in computer networks. Specifically, the subcategories (2a), (2b), (3a) and (3d); whose efforts are reflected in Table II which shows the amount of papers published per category, from a revision of works by Atilla and Hamit [6], comprising the (2010 - 2015) period. Table III shows the amount of papers published in the same period ordered by ML algorithms used during the revision of works made by Atilla and Hamit. These tables allows us to identify the research trends in the area.

Because this paper is focused in categories 2) and 3), we only show a revision of related work belonging to those categories, particularly category 3) concerning the proposal, implementation and analysis of ML models.

Mukkamala et al. [9] shows a comparison of the NN and SVM algorithms, indicating that SVM with radial kernel

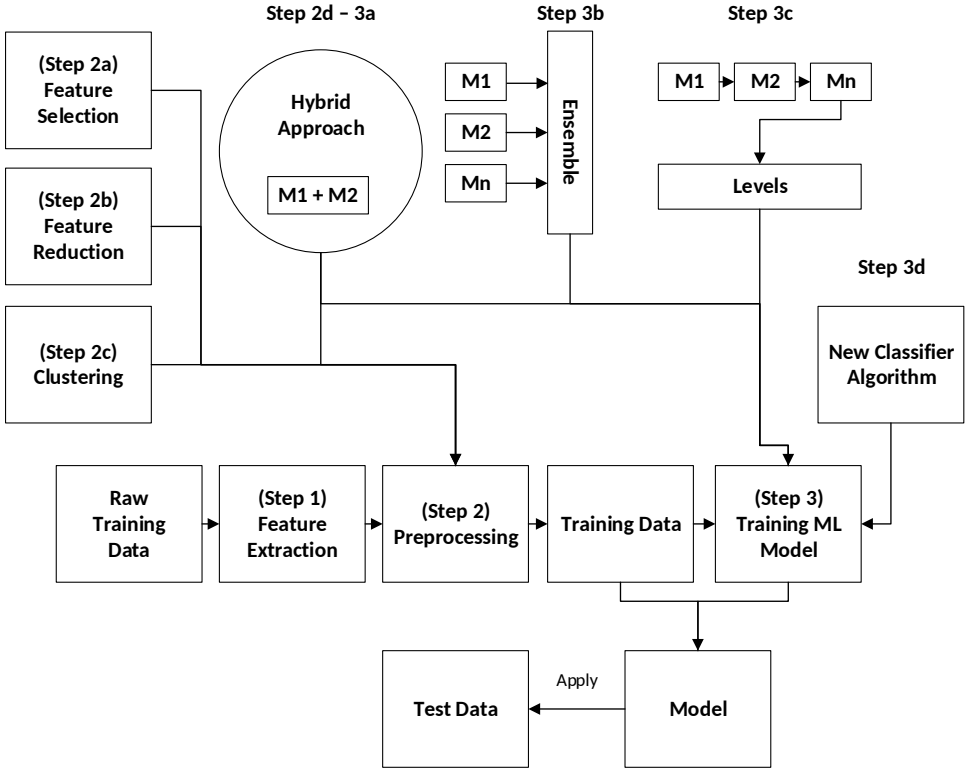


Fig. 1. General flow of an NIDS based in ML techniques. The three great categories in the process of ML model implementation in the area are shown, indicating the kind of activity conducted in each one (Adapted from [6]).

TABLE II
PUBLICATIONS IN THE AREA OF COMPUTER NETWORK SECURITY THAT USE ML TECHNIQUES [6].

Contribution	Quantity of Papers
Hybrid	50
New Classifier	45
Feature Reduction	38
Feature Selection	34
New Anomaly Detection Algorithm	33
New Optimization Algorithm	25
Levels	23
New Grouping Algorithm	19
Ensemble	14
Agent Based	12
Data Flow	7

TABLE III
MOST USED ALGORITHMS IN THE AREA OF COMPUTER NETWORK SECURITY USING ML TECHNIQUES [6].

Name	Quantity of Papers
Support Vector Machines	24
Decision Trees	19
Genetic Algorithms	16
Principal Components Analysis	13
Particle Concentration Analysis	9
K-Neighbors	9
K-Means	9
Naive Bayes Classifiers	9
Neural Networks (Multi-layer Perceptron)	8
Genetic Programming	6
Hard Sets	6
Bayesian Networks	5
Random Forests	5
Immune Artificial Systems	5
Fuzzy Logic	4
Neural Networks (Self-organizing Maps)	4

shows better performance than NN with two intermediate layers regarding hit ratio and processing time for the training of the model and prediction of entries.

Li et. al. [17] combine GFR and SVM with radial kernel and ant colony optimization. In the same way, Thaseen and Kumar [21] combine PCA and SVM with radial kernel. Both works show that the different feature selection techniques removed noise from the datasets, making the models more precise and faster during the process of training and prediction.

Kim et. al. [22] shows a hybrid model that combines Decision Trees (DT) with 1-class SVM, indicating that DT as a supervised learning algorithm has the strength of detecting known attacks and as such should be used first in order to guarantee their detection. Then, on a second level, 1-class SVM is used as an unsupervised learning technique in order to detect new kinds of attacks and in this way maximize the amount of detected attacks.

Tahir et. al. [23] proposes a hybrid model that combines SVM with radial kernel and K-Means. This model classifies the entries using K-Means first in order to form groups and afterwards applies SVM with radial kernel to classify with the groups created by K-Means, showing a high hit ratio in the

predictions made.

IV. Solution design and implementation

After revising the related work and identifying the research trends and gaps in the area, this paper proposes the design and implementation of seven hybrid ML models that use supervised learning in the first level and unsupervised learning in the second level, using the most commonly used algorithms as shown in Table III. The structure of the models is shown in Table IV, which presents the different configurations of the models indicating their id, combination of algorithms, if they present feature selection and the configuration of parameters for supervised learning. The nomenclature used Table IV is as follows:

- For the Feature Selection column, the format {Technique, #Features} is used, where Technique indicates the feature selection technique used and #Features indicates the number of predicting variables remaining after applying that technique. Model g) shows two feature selection techniques, because in the first level the features selected by GFR for NN were used, and in the second level the features selected by GFR for SVM were used.
- For the Parameters (NN/SVM) column, in the case of NN a single hidden layer was used and the following nomenclature is used $\{i,n,j\}$, where i, n y j represent the number of neurons in the input, hidden and output layers respectively. At the same time, in the case of SVM the following nomenclature is used $\{cost,gamma\}$ to identify the adjustable parameters of SVM with radial kernel.

TABLE IV
CONFIGURATION OF THE MODELS USED IN THIS PAPER.

Model ID	Algorithms	Feature Selection	Parameters (NN/SVM)
a)	NN + K-Means	{N/A,40}	{40,20,5}
b)	SVM + K-Means	{N/A,40}	{1,0.025}
c)	NN + K-Means	{PCA,24}	{24,20,5}
d)	SVM + K-Means	{PCA,24}	{1,0.042}
e)	NN + K-Means	{GFR(NN),19}	{19,20,5}
f)	SVM + K-Means	{GFR(SVM),19}	{1,0.052}
g)	NN + K-Means	{GFR(NN),19, GFR(SVM),19}	{19,30,5}

The parameter corresponding to K-Means is fixed and corresponds to the integer value two (2), due to the use of two centroids as explained in Section IV-B. Additionally, the radial kernel for SVM was chosen because Bhavsar and Waghmare [24] show that it presents the best accuracy for the task of intrusion detection when compared with the other kernels shown in Section II-C. Finally, model g) presents the features selected with GFR for NN in the first level, and those selected with GFR for SVM on the second level, because it was determined that the features selected by GFR for SVM provide more information for K-Means.

The different models shown previously were subjected to two different testing environments corresponding to: 1) analysis over the training dataset and 2) analysis over the testing dataset. Likewise, the performance of the different models in both scenarios were measured using the performance metrics show in Section II-G for their respective analysis and comparison. The methodology and design and implementation considerations used are presented below.

A. Methodology

This research was guided by the general ML work-flow shown in Fig. 2. This Fig. presents the ideal ML work-flow; however, it is possible to go back to a previous stage at any point and iterate over said work-flow. In this way, the strategy used was an iterative and empirical focus that allows to choose the models that adjust the best to the scenario.

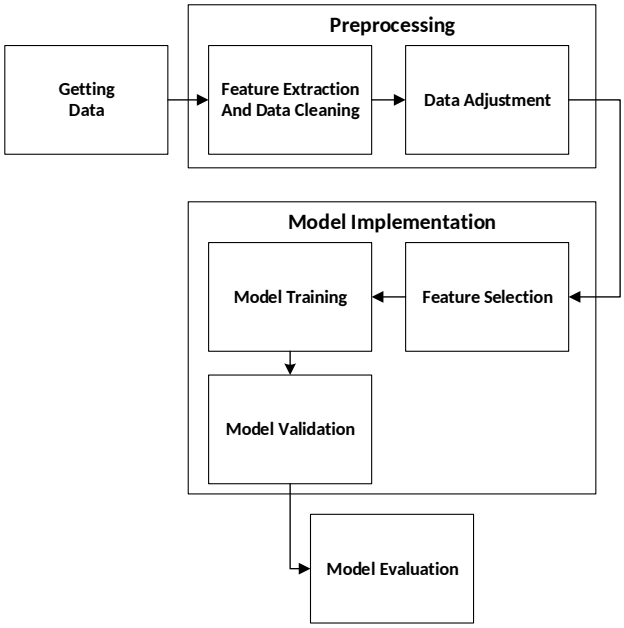


Fig. 2. General work-flow of the ML process.

B. Design considerations

The functionality of our solution is shown in Fig. 3 and Fig. 4. Both figures correspond to the work-flows for training and testing the models respectively.

The training work-flow used in this paper, shown in Fig. 3 is composed of four steps: 1) Dataset Selection, where we used the NSL-KDD[1] dataset in its training version; 2) Preprocessing, where the feature selection is performed using either PCA or GFR; 3) Training, where the best parameter configuration is defined for each algorithm by making used of cross-validation with ten sets; 4) Model Selection, where the optimal model for the defined environment is obtained.

On the other hand, the testing work-flow used in this paper shown in Fig. 4 consists of four steps: 1) Dataset Selection, where we used the testing version of the NSL-KDD dataset; 2) Preprocessing, where the feature selection is performed with either PCA or GFR; 3) Classification, where the anomalies

[1]http://www.unb.ca/cic/research/datasets/nsl.html

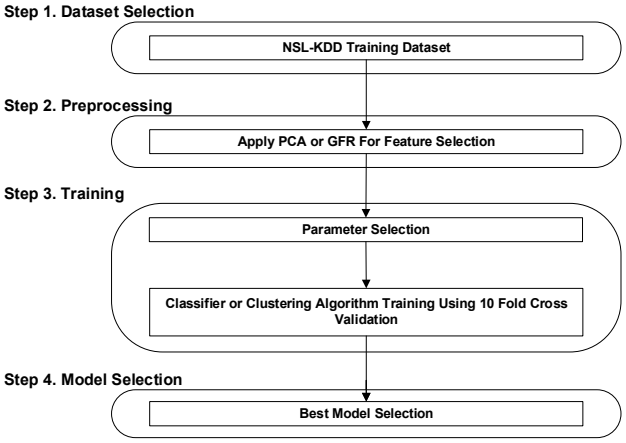


Fig. 3. Work-flow used for training the models.

detected by the first classification level corresponding to either NN or SVM are considered for their evaluation directly, while the entries classified as non-anomalous are passed to a second classification level corresponding to K-Means to separate the anomalies that remained undetected by the previous level; 4) Evaluation; where the predictions made during step 3) are evaluated using the performance metrics shown in Section II-G.

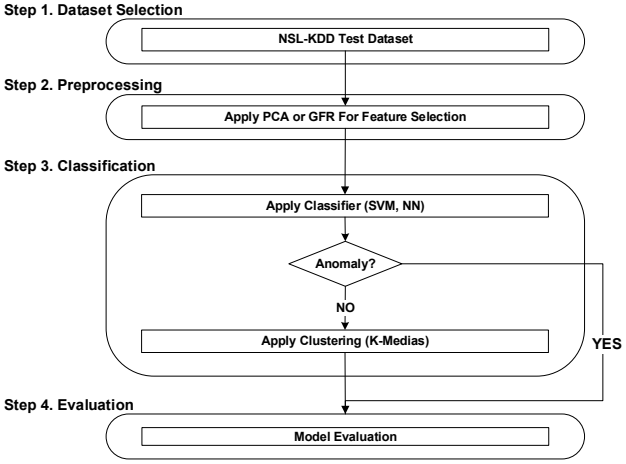


Fig. 4. Work-flow used for testing the model.

The classification made by the supervised learning algorithms have five classes as output, corresponding to the labels: DoS, Normal, Probing, R2L and U2R. These classes correspond to the labels present in the dataset used [3], [20]. Moreover, the second level corresponding to K-Means classifies on just two labels: Attack and Normal.

With the organization described previously, the first classification level provides detailed information about the kind of attack that is detected, allowing a hypothetical system administrator to narrow down the attack. On the other hand, the second classification level detects only the fact that an attack occurred but does not indicate its type. Because of this, K-Means receives as parameter two centroids that it has to position in order to separate normal from anomalous traffic.

C. Implementation considerations

In this section we expose all the generalization referring to the implementation of the solution presented in this paper.

- The NSL-KDD dataset was subjected to a pre-processing step, which was based in the work of Dhanabal and Shantharajah [25], and consisted of the following tasks:
  - Assigning names to columns;
  - Extracting the labels: DoS, Normal, Probing, R2L y U2R;
  - Removing the *Num_outbound_cmd* column, because it is a constant;
  - Transforming the *Protocol_type*, *Service* and *Flag* comlumns from a categorical to a numerical type;
  - Adding a column with the labels *Normal* and *Attack*, creating the last one from the grouping of the attack classes: DoS, Probing, R2L y U2R.
- During the training and testing of the models, the predicting variables were normalized with Equation 6 so that variables had a mean of zero and a standard deviation of 1, where X represents the current value of the entry, μ the mean, σ the standard deviation and X' the normalized value of the entry.

$$X' = \frac{X + \mu}{\sigma} \tag{6}$$

- To select the best models for evaluation with the testing dataset, we used the 10 set cross-validation technique since this is the most popular, trustworthy and easily implemented evaluation technique [12].
- We used the confusion matrix as a fundamental performance metric. From the same we derived the performance metrics described in Section II-G. Specifically, the ROC curve algorithm was implemented using the work of Fawcet [18] as a base.
- The training and testing times were measured only during the analysis over the testing dataset.
- We used the R programming language because of the great amount of packages for ML tasks with ample documentation available in CRAN[2]. Specifically, we used the *nnet*[3] package for NN because it has the most amount of documentation available. On the other hand, we used the *e1071*[4] package for SVM, because it is a binding for the R programming language of the *LibSVM* library, which is the most used SVM library [6], [26].
- For the reproducibility of our work by other researchers, we made use of fixed random seeds. The value of the seeds is explicitly defined in the source code of our solution[5]. By default these seeds have the value of 22;

[2]https://cran.r-project.org
[3]https://cran.r-project.org/web/packages/nnet/index.html
[4]https://cran.r-project.org/web/packages/e1071/index.html
[5]https://github.com/deybanperez/ML4NIDS

however, during the process of 10 set cross-validation, the seeds were set to the corresponding iteration number.

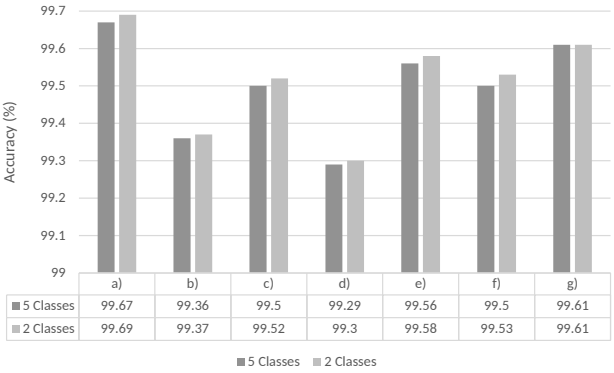
V. Analysis and discussion of results

In this section we show the results obtained with the different models described in Section IV on two scenarios: 1) analysis over the training dataset and 2) analysis over the testing dataset. Additionally, we present a final comparison of the best models obtained after the analysis of the previously mentioned scenarios.

A. Analysis over the training dataset

In this scenario, the models were tested using the 10 set cross-validation technique. With this test we measured the performance of the different models against known entry types. Specifically, in Fig. 5 we show a comparative graph of the performance of the different models in the first classification level from a accuracy point of view, using five target classes corresponding to: DoS, Normal, Probing, R2L and U2R; and using two target classes corresponding to: Attack and Normal. Moreover, Fig. 6 shows different performance metrics of the complete hybrid model.

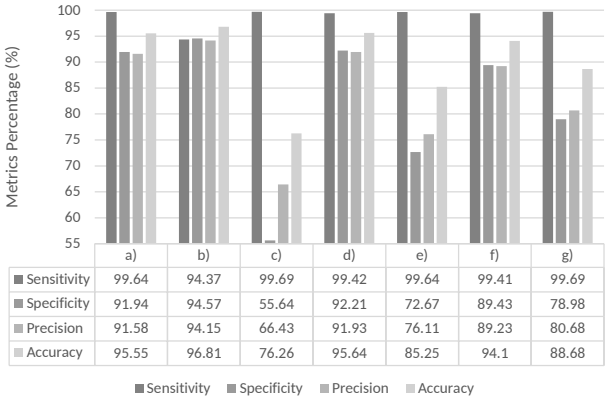
Fig. 5 shows four key aspects: 1) the accuracy of the first level is very high, result that shows that the supervised learning is effective in the classification of known entries; 2) the variation in the accuracy between two and five classes is low, indicating that the model classified the different types of attack correctly, since a high variance in the accuracy would indicate that the model committed many mistakes in the classification between attacks; 3) the effectiveness of the different feature selection techniques, since all models show a accuracy above 99%; 4) models b), d) and f) that use SVN, show a lower accuracy than the models that use NN, contradicting the results of Mukkamala et al. [9] who indicate that SVM shows a better performance than NN when using radial kernels and an NN architecture with two hidden layers using the training dataset and 10 set cross-validation.



	a)	b)	c)	d)	e)	f)	g)
5 Classes	99.67	99.36	99.5	99.29	99.56	99.5	99.61
2 Classes	99.69	99.37	99.52	99.3	99.58	99.53	99.61

Fig. 5. Comparison of the accuracy with the training dataset using five classes and two classes in the hybrid models on the first level, corresponding to supervised learning.

Fig. 6 shows the performance of the hybrid model under the following criteria: specificity, sensibility, precision and accuracy. The results show that the models b), d) and f), that use the SVM + K-Means algorithms, show better performance than the rest of the models that use NN + K-Means. This result contradicts what was previously shown in Fig. 5 and this behavior is justified in that K-Means as a complement to NN generates a high quantity of false positives that is reflected in the specificity of the models. The generation of false positives allows a revision of unusual entries and presents the opportunity of applying feedback to the models so that they can become more precise in future versions. On the other hand, K-Means as a complement of SVM provided practically no contribution, because both algorithms are based on the same approach of classification of circular groups and basically the actions performed by SVM are remade by K-Means, showing a high accuracy as shown by Li et. al. [23] using SVM with radial kernel as a complement of the K-Means algorithm and obtaining a high accuracy in the classification of entries over the training dataset using 10 set cross-validation.



	a)	b)	c)	d)	e)	f)	g)
Sensitivity	99.64	94.37	99.69	99.42	99.64	99.41	99.69
Specificity	91.94	94.57	55.64	92.21	72.67	89.43	78.98
Precision	91.58	94.15	66.43	91.93	76.11	89.23	80.68
Accuracy	95.55	96.81	76.26	95.64	85.25	94.1	88.68

Fig. 6. Performance of the hybrid models over the training dataset.

From the results obtained we conclude that the feature selection was positive for the models in the first classification level concerning supervised learning; however, in the global view, the models using the complete feature set show better performance than those that use a reduce feature set, specifically those using NN because these generate the greatest amount of false positives than those using SVM for the previously mentioned reasons.

B. Analysis over the testing dataset

In this scenario the models were trained using the best configuration obtained by each model over the training dataset, training made with the complete training dataset and whose performance was evaluated with the complete testing dataset. With this scenario we determined the performance of the different models against new kinds of entries. Just as in Section V-A, we show two figures that resume the results obtained on this scenario. Fig. 7 shows the comparison of the accuracy of the supervised learning level using five and two target classes. Moreover, Fig. 8 shows the general performance of the hybrid models.

Fig. 7 reflects three key aspects: 1) PCA feature selection is not effective because the covariance matrices of the training and testing dataset are very different, degrading the performance of the models notably when these receive new kinds of entries, result that was not shown by Thaseen and Kumar [21], who show the good performance of PCA only over the training dataset using 10 set cross-validation without using the testing dataset; 2) GFR feature selection works in a very positive way, because the models that were subjected to it use less than half of the feature set and possess a accuracy comparable to models a) and b) that were trained using the complete feature set, this being an extension of the results shown by Li et al. [17], who presented the effectiveness of GFR over the training dataset. On the other hand, the processing time of GFR was of eightenn days; 3) the supervised learning algorithms show good performance on the classification of traffic despite the fact that the training dataset contains new kinds of entries that are not present in the training dataset.

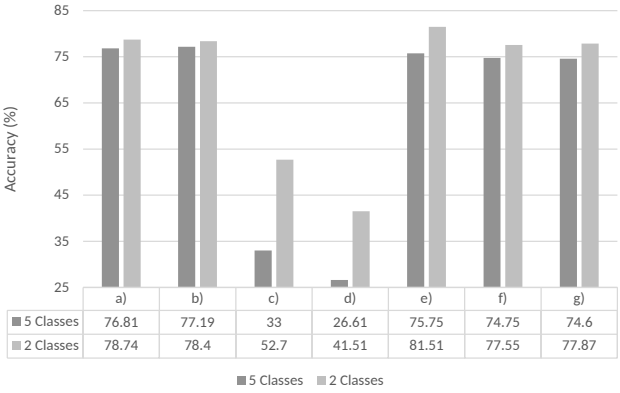


Fig. 7. Comparison of the accuracy over the training dataset using five and two classes in the first level of the hybrid models, corresponding to supervised learning.

Fig. 8 shows the global performance of the hybrid models, highlighting the fact that models a), f) and g) are the most balanced models for this scenario, since this models detect the greatest amount of attacks, classify the normal traffic in a better way, show better certainty when detecting attacks and a greater accuracy. On the other hand, the fact that the performance of model e) which uses the features selected by GFR for NN in both levels is lower than the performance of model g) which uses the features selected by GFR for SVM in the second level with K-Means shows that the features chosen for SVM provide more information to K-Means.

From the results obtained in this scenario we conclude that PCA feature selection is not effective is the models receive new entry types, whilst GFR feature selection works better in the same scenario. Additionally, we identified that models a), f) and g) present a global performance that is very balanced and good. The following Section shows these models in greater detail.

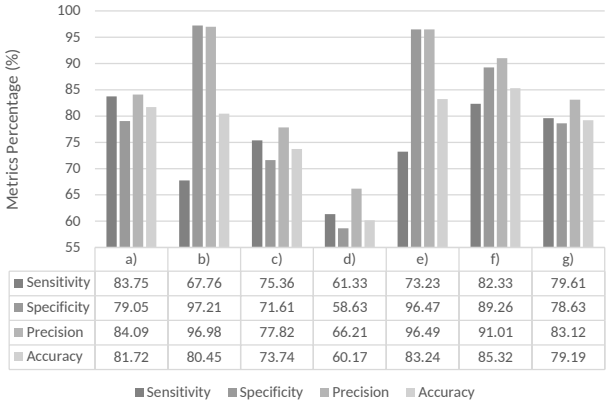


Fig. 8. Performance of the hybrid models over the testing dataset.

C. Comparison of the best models

In Section V-B we identified three models with good performance over the testing dataset: a), f) and g). These models also present good performance over the training dataset, with model g) being the model with the better performance due to the generation of false positives by K-Means in the second classification level. To determine which of these models is the most complete, we show the ROC curves of the first classification level, corresponding to supervised learning, over the training and testing datasets in Fig. 9. This Fig. shows that model g) is the one presenting a better ROC curve on both scenarios, indicating that its predictions on the first level are more accurate than those of the other models. That model g) shows better performance than model a), indicates that the GFR feature selection removed noise from the model, thus improving its performance.

The effectiveness of hybrid model g) is reflected in Table V, that shows how the strength of supervised learning lies in the detection of known attacks, whilst the strength of the unsupervised learning approach is the detection of unknown attacks, showing that NN and K-Means are a good complement in the task of intrusion detection in computer networks.

TABLE V
ATTACKS DETECTED BY HYBRID MODEL g).

Level	New Attacks Detected	Effectiveness Known Attacks	New Attacks Detected	Effectiveness New Attacks
NN	6282	69.16%	1059	28.24%
K-Medias	612	24.40%	1526	67.91%
Total	6894	75.90%	2585	68.93%

The training and prediction times for models a), f) and g) are shown in Fig. 10 and Fig11 respectively. In said Figures, it can be observed how the training and prediction times of SVM with radial kernel are much higher than those of NN using a single hidden layer. Additionally, it can also be observed how the GFR feature selection for NN not only increased the certainty of the predictions, but it also reduced the training and prediction times.

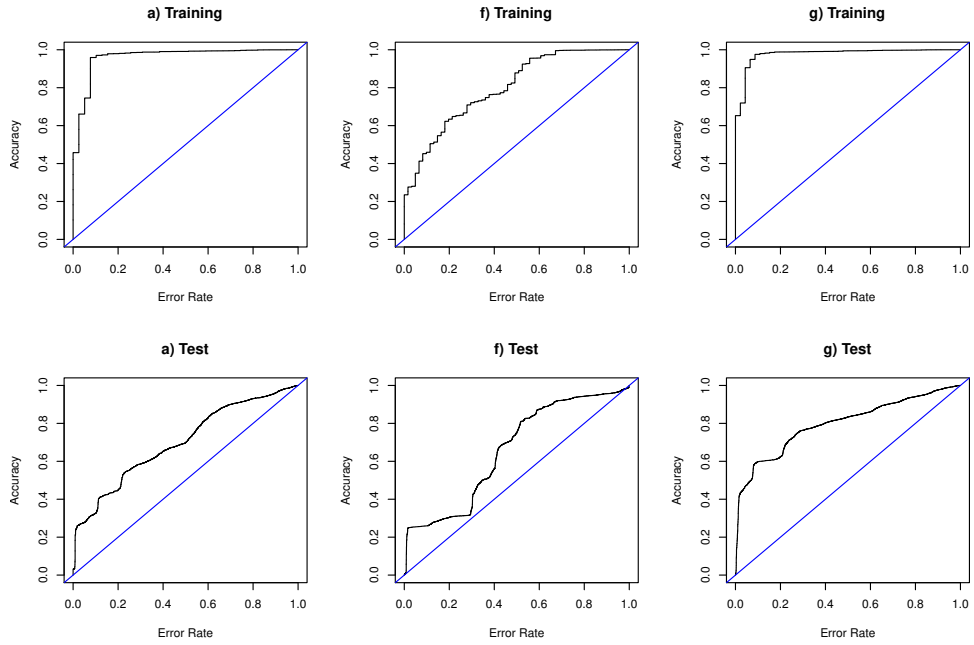


Fig. 9. ROC curves for models a), f) and g) over the training and testing scenarios.

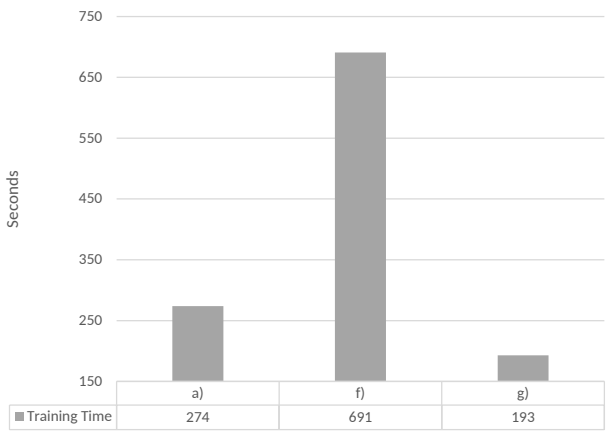


Fig. 10. Training times for models a), f) and g).

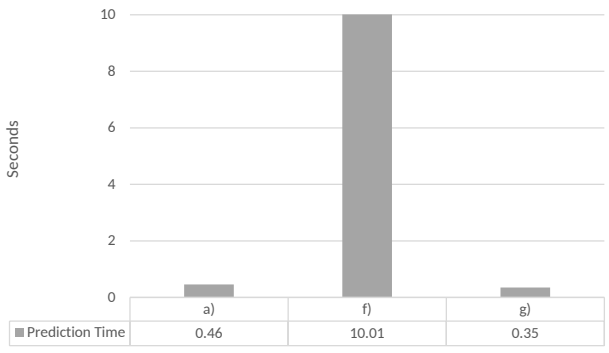


Fig. 11. Prediction times for models a), f) and g).

Lastly, the results indicate that model g) is a good prototype hybrid ML model for the task of intrusion detection in computer networks because it shows good performance in the training and testing scenarios. Additionally, it is the model that shows greater certainty in the predictions made in the first level and is also the model that trains and predicts faster.

VI. Conclusions

This paper shows the design, implementation and evaluation of different hybrid ML models with the intention of showing their applicability to the task of intrusion detection in computer networks. To accomplish this we used both versions of the NSL-KDD data set corresponding to training and testing, in order to evaluate the performance of the model against known and unknown entries.

The models use the latest research trends in the subject as shown in Section III, using the NN and SVM algorithms for supervised learning, the K-Means unsupervised learning algorithm and the PCA and GFR feature selection techniques, as shown in Section IV.

The results shown in Section V indicate that: 1) the PCA feature selection technique is not a good strategy if the models are subject to a considerable quantity of new attacks; 2) the GFR feature selection technique is useful for both scenarios of known and unknown attacks; however, it is also very computationally expensive regarding processing time; 3) NN shows better performance than SVM with radial kernels regarding hit ratio; 4) hybrid models are a good strategy because they allow the detection of known and unknown attacks; 5) Model g) is who better adjusts to the scenario of intrusion detection in computer networks since it shows the most balanced results, making it the most trustworthy for the task.

For future work we propose the integration of a feedback system to allow the models to increase their performance in future versions. Additionally, we suggest the implementation of different hybrid models, making use of other algorithm combinations in order to compare results with the models created for this research.

REFERENCES

[1] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," 2011.

[2] F. Wortmann and K. Flüchter, "Internet of Things," *Business & Information Systems Engineering*, vol. 57, no. 3, pp. 221–224, 2015.

[3] D. Bhattacharyya and J. Kalita, *Network Anomaly Detection: A machine Learning Perspective*. CRC Press, 2013.

[4] H. Liao, C. Lin, Y. Lin, and K. Tung, "Intrusion Detection System: A Comprehensive Review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[5] K. Veeramachaneni, "AI 2: Training a Big Data Machine to Defend," in *International Conference in Big Data security on Cloud*. IEEE, 2016.

[6] O. Atilla and E. Hamit, "A Review of KDD99 Dataset Usage in Intrusion Detection and Machine Learning Between 2010 and 2015," *PeerJ Preprints*, vol. 4, p. e1954v1, 2016.

[7] W. Stallings, *Cryptography and Network Security Principles and Practice*. Pearson, 2014.

[8] P. Ning and S. Jajodia, "Intrusion Detection Techniques," *The Internet Encyclopedia*, 2003.

[9] S. Mukkamala, G. Janoski, and A. Sung, "Intrusion Detection Using Neural Networks and Support Vector Machines," in *Proceedings of the 2002 International Joint Conference on Neural Networks, 2002. IJCNN'02.*, vol. 2. IEEE, 2002, pp. 1702–1707.

[10] A. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM Journal of research and development*, vol. 3, no. 3, pp. 210–229, 1959.

[11] T. Mitchell, *Machine Learning*. McGraw Hill, 1997.

[12] G. James, D. Witten, T. Hastie, and R. Tibshiriani, *An Introduction to Statistical Learning*. Springer, 2013.

[13] L. Fausset, *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*. Prentice-Hall, Inc., 1994.

[14] S. Samarisinghe, *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition*. CRC Press, 2006.

[15] A. Jain, M. Murty, and P. Flynn, "Data Clustering: A Review," *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.

[16] C. Aggarwal, *Data Mining: The Textbook*. Springer, 2015.

[17] Y. Li, J. Xia, S. Zhang, J. Yan, X. Ai, and K. Dai, "An Efficient Intrusion Detection System Based on Support Vector Machines and Gradually Feature Removal Method," *Expert Systems with Applications*, vol. 39, no. 1, pp. 424–430, 2012.

[18] T. Fawcett, "An Introduction to ROC Analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[19] W. Lee and S. Stolfo, "A Framework for Constructing Features and Models for Intrusion Detection Systems," *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, no. 4, pp. 227–261, 2000.

[20] M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set," in *Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications 2009*, 2009.

[21] S. Thaseen and C. Kumar, "Intrusion Detection Model Using fusion of PCA and optimized SVM," in *International Conference on Contemporary Computing and Informatics (IC3I)*. IEEE, 2014, pp. 879–884.

[22] G. Kim, S. Lee, and S. Kim, "A Novel Hybrid Intrusion Detection Method Integrating Anomaly Detection With Misuse Detection," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1690–1700, 2014.

[23] H. Tahir, W. Hasan, and A. Said, "Hybrid Machine Learning Technique for Intrusion Detection System." 5th International Conference on Computing and Informatics (ICOCI) 2015, 2015.

[24] Y. Bhavsar and K. Waghmare, "Intrusion Detection System Using Data Mining Technique: Support Vector Machine," *International Journal of Emerging Technology and Advanced Engineering*, vol. 3, no. 3, pp. 581–586, 2013.

[25] L. Dhanabal and S. Shantharajah, "A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, no. 6, pp. 446–452, 2015.

[26] C. Tsai, Y. Hsu, C. Lin, and W. Lin, "Intrusion Detection by Machine Learning: A Review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11 994–12 000, 2009.