electronics

MDPI

Article

An Approach for the Application of a Dynamic Multi-Class Classifier for Network Intrusion Detection Systems

Xavier Larriva-Novo, Carmen Sánchez-Zas, Víctor A. Villagrá *, Mario Vega-Barbas and Diego Rivera

ETSI Telecomunicación, Universidad Politécnica de Madrid (UPM), Avda. Complutense 30, 28040 Madrid, Spain; xavier.larriva.novo@upm.es (X.L.-N.); carmen.szas@alumnos.upm.es (C.S.-Z.); mario.vega@upm.es (M.V.-B.); diego.rivera@upm.es (D.R.)

* Correspondence: victor.villagra@upm.es

Received: 3 September 2020; Accepted: 20 October 2020; Published: 23 October 2020

Abstract: Currently, the use of machine learning models for developing intrusion detection systems is a technology trend which improvement has been proven. These intelligent systems are trained with labeled datasets, including different types of attacks and the normal behavior of the network. Most of the studies use a unique machine learning model, identifying anomalies related to possible attacks. In other cases, machine learning algorithms are used to identify certain type of attacks. However, recent studies show that certain models are more accurate identifying certain classes of attacks than others. Thus, this study tries to identify which model fits better with each kind of attack in order to define a set of reasoner modules. In addition, this research work proposes to organize these modules to feed a selection system, that is, a dynamic classifier. Finally, the study shows that when using the proposed dynamic classifier model, the detection range increases, improving the detection by each individual model in terms of accuracy.

Keywords: intrusion detection system; dynamic classifier; ensemble machine learning; multiclass; cybersecurity

1. Introduction

Intrusion detection systems (IDS) are computer systems designed to monitor network traffic. These systems are capable to find atypical records and attack patterns based on the behavior of the networks. Thus, the aim of IDS is the early detection or prediction of possible real harm to the network, host, or cloud caused by a security issue. To perform this, IDS, as a software application, analyze possible anomalies detected at the network layer. This process is, traditionally, static and linked to the rules or algorithms used for detecting cyberattacks. Nevertheless, this static process is difficult to adapt to the detection of new types of attacks because it implies updating it with new rules in the cases of signature-based IDS [1], or the re-training of the detection model in the case of anomaly-based IDS [2]. Specifically, anomaly-based IDS are related directly to the application of machine learning (ML) techniques. These techniques, depending on the underlaying classification model, are capable of detecting anomalies by means of binary classifiers, or different types of attacks by means of multiclass classifiers. In general, the aim of ML-based IDS is to increase their ability of attack detection by reducing the quantity of possible false positives [2]. The reduction of possible false positives is a crucial issue in the design of IDSs, as the continuous development of automated malware forces the IDSs to be as accurate as possible, trying to be one step ahead of attackers. Nevertheless, the accuracy of this software still has important limitations, and therefore, there is a margin for their improvement.

In recent studies, it is possible to observe this use of ML techniques for detecting possible cyberattacks in a more efficient way [3,4]. Most of these studies are based on a binary classification,

but some investigations have improved the prediction by establishing a multiclass classification [5]. The main drawback of the systems presented in these studies is that they offer low levels of general precision [6,7] although they do offer a high level of efficiency in very specific attacks. Thus, it could be possible to identify and characterize certain types of models that allow a better detection of specific classes of attacks.

However, IDS performance is impaired and, in combination with the fact that the same algorithm is applied to a probably complete dataset, a problem to identify certain types of attacks has been posed. When a model is trained, it recognizes better patterns in some features, usually related to a type of intrusion. This is the main reason for the need of an adaptable system, to take advantage from the inclination of each algorithm to detect better some types of attacks over others.

In order to mitigate this disadvantage of current models, this research proposes a dynamic classification model that can obtain the main contributions in terms of rate detection of each individual ML model with the aim of generating reliable predictions of attacks from them. To perform this, the system introduces the separate training, validation, and metrics of different algorithms such as classical ML models and neural networks, and then join their predictions into a module, which aims to compare the different outcomes and extract the most suitable one.

This paper presents the related work in the state of the art in Section 2. Furthermore, in Sections 3–5, we explain the problem statement, the proposed methodology, and experimentation, defining the new architecture for the dynamic classifier introduced in this research. These sections also include several tests for different ML learning techniques, including data preprocessing and feature selection. Section 6 provides the obtained results for the individuals ML models proposed and the application of the dynamic classifier proposed using the dataset UNSW-NB15, concluding with a comparison between results obtained by the different tests done and related works. Finally, Section 6 includes the discussion of the main conclusions and future lines for this article.

2. Related Works

Nowadays, the research related to the selection of the most suitable IDS is addressed from different perspectives based on the description of the input problem and the analysis of the performance of underlaying algorithms. In general, the aim of these research works is to study the algorithm selection problem in order to determine the most proper one.

Some studies are based on basic ML techniques, selecting an algorithm, and training it with a complete dataset. Such as the case of [8], where the application of a multi-layer perceptron (MLP) network is applied to detect large scale datasets and predict malicious attacks. In this case, the selection of the attributes is based on the co-variance, standard deviation, or correlation. The authors of the paper obtained an accuracy of 0.9935 selecting the attributes with a near perfect correlation. In [2], Larriva-Novo et al. propose an analysis based on a categorization of a cybersecurity dataset, where they used a static algorithm based on MLP. The analysis was done in order to select the best hyperparameters related to the best performance in terms of accuracy. Furthermore, the different characteristics selected were based on: basic connection, content, traffic statistical and direction characteristics. The model proposed below presented an accuracy near to 99%, based on anomaly detection.

A typical approach for the design of IDS based on anomalies is the application of the support vector machine (SVM) algorithm, to predict if the input data represents an anomaly or not. The authors of [9] improved the results of their system using a non-linear scaling method for data preprocessing. The classifications conducted were binary and multi class, measured by accuracy (AC), detection rate (DR) and false positive rate (FPR). The accuracy obtained was 85.99% for binary classification and 75.77% for multi-class classification. In [10], they tried to go deeper in the classification based on SVM algorithms. The authors developed an efficient IDS, with an improved performance in terms of accuracy due to the binary gravitational search. The results obtained were 86.62% in terms of accuracy without the application of feature selection and 94.4% with its application, improving the results obtained in [9] for the binary classification.

Another approach based on deep learning techniques such as feed-forward neural networks is presented in [11]. This method consisted on selecting the optimal activation and features. The authors conducted three experiments: selecting the best activation function, deploying a feature selection and applying the obtained results to new data. The optimal results exposed an accuracy of 99.5%. Despite these results, basic algorithms for anomaly detection are not effective enough to perform the classification problem in IDS based on ML algorithms because they must be optimized, related to its hyperparameters and the quality of the data [12].

On the other hand, a large variety of algorithms have been used with the objective of finding a better performance in IDS multiclass attack detection. This is the case of the extreme gradient boosting (XGBoost) [13], which was used to deploy better results. The authors experimented with 39 numerical features, excluding those like IP Address or Protocol Type. The best results were achieved with an accuracy of 88% for the testing dataset [14]. Furthermore, the research conducted to include a data preprocessing phase, where the optimal features were reduced from 39 to 23. The same models were applied, obtaining again the best accuracy by the XGBoost model. However, in this case, the accuracy rate fell to a 76%. It should be noted that in this case the category "attack analysis" presented difficulties to be detected.

Recent works proved the effectiveness of different techniques to make predictions in IDS. However, just a few of them used time-series information and categorical information. This is the case in [15], where the authors used a long short-term memory (LSTM) network with feature embedding. The model makes a multi class classification based on chronologically order information. The research presented multiples combinations in order to deploy the most accurate model: feature embedding selection, transformation of categorical data, and projection of the features as vectors values in the space. The results in terms of accuracy was of 83% for a multiclass classification.

Another standpoint to confront the algorithm selection problem is based on the use of a Bayesian approach, because these models are able to reason under uncertainly. In [16], a static automatic selection is proposed, designing an expert system that presents knowledge representation, learning, and inference ability. The methodology presented in that work was based on the identification of representative characteristics and a list of suitable candidates to generate a random training dataset. They then use the dataset to analyze the performance of each algorithm. This method obtained an accuracy of 76.08% in its prediction, which was the best result in comparison with other proposed candidate algorithms. This approach was also used to select other hyperparameters among a wide space of choices as kernel functions [17].

Alternatively, a method called HYDRA is presented in [18], which represents a new approach to ML models setup that combines automated algorithm configuration and portfolio-based algorithm selection techniques [19,20]. The base of this approach is to join the advantages of techniques, i.e., less domain knowledge required, mechanization from automated algorithm configuration, and variety of candidate algorithms. HYDRA accepts five inputs: a parametrized solver, a set of training problem instances, an algorithms configuration procedure, a performance metric to optimize, and portfolio-based algorithm selection. During the conducted test, it was proved that HYDRA outperformed in almost every case, the accuracy of the proposed algorithms.

Over time, completely automatic ML models were developed which tries to outperform the best accuracy of selected algorithms over a determined output. An example of this is Auto-WEKA [21], designed to choose the most appropriate algorithm and its optimal hyperparameters automatically, between a large number of possible combinations (39 WEKA's algorithms are available) using a Bayesian optimizer. Since its original release in 2013 some updates were included in Auto-WEKA 2.0 [22].

In [23], the authors discuss the need for a dynamic ML system. The objective they propose is to avoid the loss of efficiency, characteristic of a static model. This derives from the choice of the selected features, inferring that a single model is not sufficient. The authors discuss the need of a dynamic selection of a model. At first, Ensemble ML is proposed as a solution, but the drawback of

the need for continuous training makes it an inefficient option. The author of the research mentioned above, introduced a new methodology based on a cloud device operations architecture. The research proposed different containers to train a model, so each of them made a prediction to be sent to a model selector. The model selector was able to choose the most suitable one in terms of precision. Analyzing obtained results based on supervised options, an accuracy of 61% was obtained. This value increased substantially when the system was integrated with label categorization. Taking into account these results, unsupervised learning by clustering was only recommended for those attacks with lowest accuracy. The objective was to select the most accurate model according to real-time data received. The authors mentioned that it is imperative to automate this task, as there were at least 70 models to choose from. Furthermore, the dataset used for the researches introduced above was the benchmark dataset UNSW-NB15 [14]. Finally, [24] introduced a multiclassification approach which combines the outputs from related classifiers in the state of the art, such as the one proposed in [25], for mobile and encrypted traffic classification based on hard/soft combiner enhancing up to 9.5% of recall.

As a conclusion, this section has identified and detailed several related works focusing on the study of statics models for the application of IDSs based on ML algorithms. These studies conducted several analysis based on the performance of different ML models in terms of accuracy. The models and ML techniques applied obtained, in the best cases, an overall accuracy of 99% for a binary classification. Furthermore, some of these researches included the study and application of multiclass classification evaluation, experimenting a low rate of detection in terms of accuracy. In addition, other works such as [23], lead to the need of dynamic auto selection ML models with the objective to perform the rate detection by a dynamic selection of a model. All these works are based on several ML techniques such as correlation, feature selection, multiclass classification evaluation and dynamic model selection. Therefore, most of these techniques are covered and evaluated in this proposal.

3. Background

3.1. Cybersecurity Datasets

Nowadays, there exist different cybersecurity datasets that can be used for IDS based ML experimentation, i.e., UNB-ISCX-1012 [26], CTU-13 [27], MACCDC [28], UGR-16 [29], CICDS [30], KDD-99, NSL-KDD [31], or UNSW-NB15 [32]. Some of them have been widely used, like for instance the dataset KDD-99, which has been stablished as the main benchmark dataset for the different studies cases in the application of ML-based IDS. In [6], there is a representation of the most used datasets in the last decade for anomaly detection based on ML algorithms. This study points out that the NSL-KDD and the KDD-99 are the most used, with a 11.6% and 63.8% of use respectively.

In this study, however, the chosen dataset was UNSW-NB15 [14], because it has been considered as a benchmark dataset for the evaluation of IDS based on ML models thanks to the variety of the current cybersecurity attacks to date, now being widely used in cybersecurity [32].

3.1.1. Attack Categories in UNSW-NB15

The following are the nine different attack categories considered in the UNSW-NB15 dataset used in our study:

Fuzzers: Injection of an invalid random data into a program to cause crashes and exceptions.
Analysis: Involves network monitoring i.e., port scanning, spam, or penetration through HTML files.
Backdoor: Action of avoid security mechanisms of a system to access to stored data.
Denial of Service (DoS): Attempt to make a system temporarily unavailable to its users.
Exploits: Take the advantage of a security flaw to gain access of it.
Generic: Technique that works against all blockages, regardless of the encryption used.
Reconnaissance: Malicious actions that involves the collection of information.
Shellcode: Code used as a payload against an identified vulnerability in a system.

Worms: Replication of an attacker to spread an intrusion to other systems through the network.

3.1.2. UNSW-NB15 Description

This dataset contains information about multiple connections being labeled to differentiate those data that represent an attack and the normal behavior of the network, making suitable for supervised learning such as the case of this research.

The UNSW-NB15 includes 47 different features. In addition, it contains two fields that represent if the traffic flow is an attack or not, and the attack category, being one of those mentioned in Section 3.1.1. The features of the dataset are classified into flow features, basic features, time features, content features, additional features, and labeled features [14,32,33].

The dataset is divided in 10 categories including the nine presented before and the normal behavior among them, as presented in Table 1, where the number of records of each category and the percentage of records from the total are also shown.

Table 1. UNSW-NB15 distribution by types of attacks.

Type	% Records	No. Records
Normal	36.092	2,218,761
Fuzzer	9.409	24,246
Analysis	1.039	2677
Backdoors	0.904	2329
DoS	6.346	16,353
Exploits	17.279	44,525
Generic	22.847	215,481
Reconnaissance	5.428	13,987
Shellcode	0.586	1511
Worms	0.067	174

3.2. Machine Learning Applied to Intrution Detection Systems

Nowadays, ML has been incorporated in cybersecurity, specially to the design of IDS. It has been widely used, involving different levels of complexity, from the choice of an algorithm, to complex systems with the ability to autoconfigure themselves [23,34]. It is a field which focuses on the estimation of different mathematical functions with the objective of extract and represent behavioral generalizations of the data [3]. In addition, ML can be defined as a subset of the artificial intelligence with the ability to produce desired outputs referred to an input data without being programed.

ML algorithms can be classified into different categories: supervised learning, unsupervised learning, and semi-supervised learning. In this paper, we focus specifically in supervised learning methods, as the others are currently out of the scope of our research.

Supervised learning algorithms are based on what they are capable to learn about a set of features with an explicit label. Therefore, if the function is capable to adequate the input data to a desired output label, the algorithm is able to predict outputs given inputs in a similar scenario. After the training of the model, the algorithm can be executed until it reaches an acceptable level of performance [35].

3.3. Machine Learning Algorithms under Study

In the context of supervised learning, ML offers different types of algorithms because each one presents a different response to a unique input [23]. We have evaluated the results of the works presented in Section 2, analyzing the algorithms evaluated in those researches, in order to improve their results with our proposal. Additionally, other algorithms with similar procedures were taken into account, for those whose working methods fit the requirements of this research. After that evaluation, the models selected are the following:

1. K-nearest neighbors (KNN): It is an algorithm that calculates and orders the distance from new data to the existing one, classifying this input according to the frequency of the labels of the K-nearest ones. This distance is usually measured by the Euclidian norm presented in Equation (1). For the correct adjustment of this method [36], a correct value of the number of neighbors considered is essential.

$$d(x, y) = \sqrt{\sum_{i=1}^{k} (x_i - y_i)^2} \quad (1)$$

2. Decision trees (DT): In the DT algorithm each node represents a test over an attribute, and each branch is a possible outcome from it. It uses a decision support to show the prediction lead by features splits [37]. In this case, among the hyperparameters, we can list maximum depth of the diagram, minimum number of samples required to split an internal node or minimum number of samples required to be at a leaf node.
3. Random forest (RF): RF is composed by many DT [34], each one created from a subset of features to be considered. Each one votes and the algorithms compute all of them to choose the best prediction. The main advantage of this method is the prevention of the over-fitting [38].
4. Support vector machines (SVM): SVM allows find the optimal classification, maximizing the margin between classes, whose border is defined by support vectors. In case there is not a lineal separation, the kernel trick is applied to redefine inner products [39].
5. XGBoost: It is an algorithm that starts with weak classifiers over a set of data with the objective to enhance their results by means of a sequential processing with loss function to minimize the error with every iteration, obtaining a strong model at the end [40].
6. Multi-layer perceptron neural network (MLPNN): It is composed by a group of linear classifiers denominated perceptron. The perceptron itself contains a group of layers (i.e., input layer, output layer and hidden layer). MLPNN is a type of feed forward neural network (FFNN), where the layers have a non-linear activation function. MLPNN is trained by a back-propagation model which stablish the relation between the input features and the output features, with the objective to minimize the error [2]. The different hyperparameters (i.e., number of hidden neurons, layers, iterations, activation function, etc.) must be optimized in order to achieve better results.
7. Long short-term memory neural network (LSTMNN): LSTMNN are recurrent neural networks (RNN). These units are capable of connect previous information to the input data, learning those long-term dependencies. This model is configured to maintain the back-propagation error constant through the time between the different layers [2]. In this case, the hyperparameters to be optimized are batch size and number of hidden neurons.

4. Proposal

The dynamic classifier proposed in this research is designed to achieve the objective described throughout this document, a system capable of obtaining the best prediction results from various ML algorithms based on a multiclass classification. To develop the dynamic classifier, previously optimized models are required [41]. The proposed system is presented in Figure 1. As it can be seen in the figure, we propose an architecture composed of different modules: a series of static ML algorithms manually preconfigured by means of a study of hyperparameter selection and feature selection, and finally by a dynamic classifier.

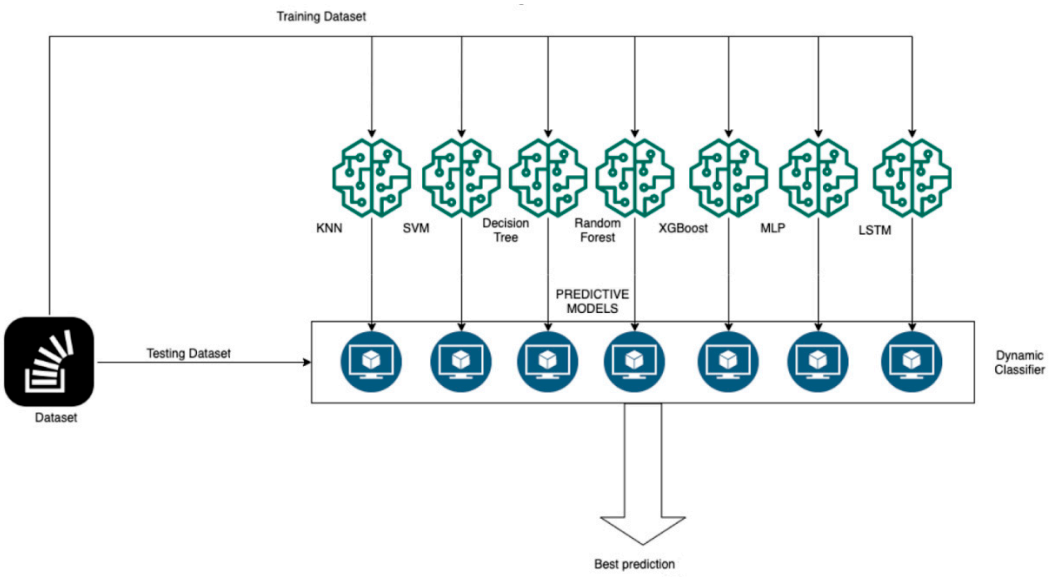


Figure 1. System proposed, composed by Multi ML (machine learning) algorithms, a real time streaming data and a dynamic classifier.

In our research work we include the development of that series of static ML models, capable of predicting possible attacks. Each model will count on its own configuration of hyperparameters and a specific feature selection. The related work study from Section 2 indicates that not all models are able to achieve good prediction accuracy for every type of attack. Instead, each ML model seems to be more suitable for certain attack categories. In the following sections we confirm this, by carrying out tests using different models to predict different attacks.

The dynamic classifier works by evaluating the incoming information by each one of these static ML models. The output of each one, consisting on a prediction on the possible attack (based on the attack categories defined by the UNSW-NB15 dataset, shown in Section 3.1.1 of this paper) is then used as input data for the actual classifier. The classifier is then able to determine the best prediction from those generated by the ML models, and select it, regardless of the attack category. This method improves the accuracy of attack prediction.

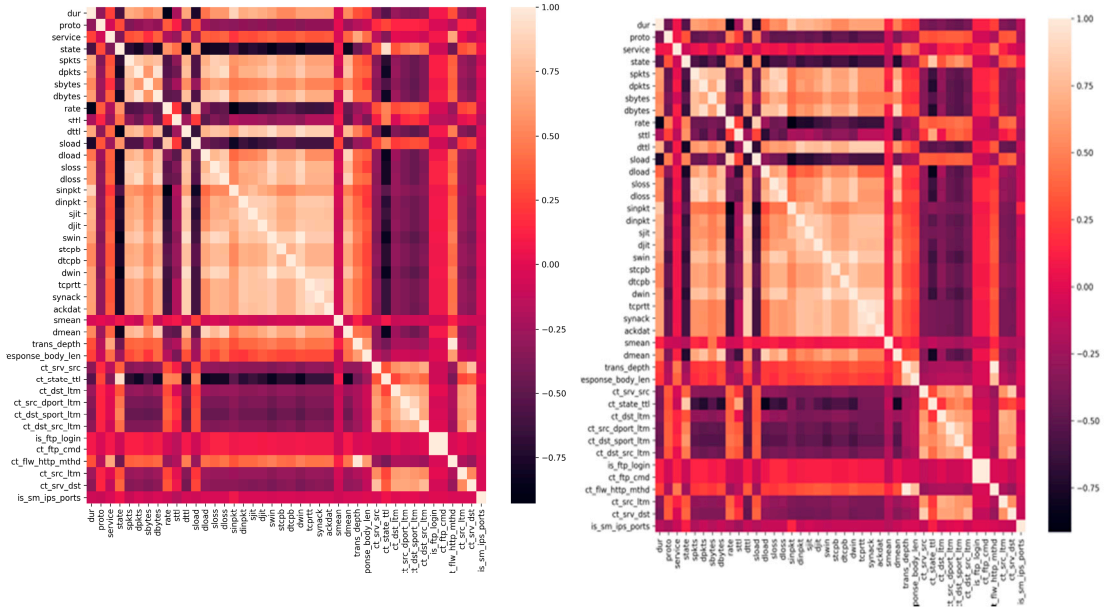
5. Methodology and Development

5.1. Dataset Preparation

For this research, the dataset selected was the UNSW-NB15 [14], as was mentioned before in Section 3.1. In [2], the importance of selecting the most important features between all the features available is established, as it has an enormous impact in the algorithm's performance. The less important features do not bring performance in terms of accuracy while consuming computer resources. For this purpose, the features were classified according to its type—i.e., numerical values were transformed into z-score [42] values—and categorical features were transformed into numerical values [2]. As was presented in Table 1, some attack categories expose a low distribution, what may not produce accordance results in terms of accuracy. To prevent this, some researches have used SMOTE [43] to balance the dataset UNSW-NB15, obtaining good results [44]. This algorithm was applied in this research with the objective to compare the results between a balanced dataset by SMOTE, and an unbalanced dataset as the original dataset.

Additionally, we have performed a feature selection based on the correlation Kendall coefficient [45], which has improved the results of the selection task. We improved the tests with both a balanced and unbalanced dataset. In Figure 2, the features correlation matrices for the UNSW-NB15 are presented.

For this research, the best results were achieved by removing the features with a correlation higher than 0.8 for both balanced and unbalanced dataset. Table 2 presents the features selected for both datasets.



(**a**) (**b**)

Figure 2. Features correlation matrix UNSW-NB15: (**a**) balanced dataset; (**b**) unbalanced dataset.

Table 2. UNSW-NB15 selected features after correlation Kendall coefficient feature selection.

Balanced Dataset	Unblanced Dataset
'sloss', 'dinpkt', 'rate', 'dwin', 'synack', 'dmean', 'dload', 'sjit', 'ct state ttl', 'stcpb', 'ct w http mthd', 'djit', 'dloss', 'ct ftp cmd', 'ct dst sport ltm', 'ackdat', 'dtcpb', 'tcprtt', 'swin', 'dbytes', 'sinpkt', 'sload', 'dpkts', 'dttl'.	'ackdat', 'synack', 'dpkts', 'ct srv dst', 'dloss', 'dwin', 'sload', 'sinpkt', 'ct w http mthd', 'ct dst sport ltm', 'sloss', 'stcpb', 'tcprtt', 'dinpkt', 'ct ftp cmd', 'dtcpb', 'sjit', 'dbytes', 'ct state ttl', 'rate', 'djit', 'swin', 'dmean'

After the selection process, the dataset was divided into training (75%) and testing (25%) data. The training data was used for the generation of models in each algorithm selected. The testing data was used for the validation of each algorithm and for testing the dynamic classifier proposed.

In addition, some algorithms present a better performance using less features [45]. To achieve better results over the correlated feature selection, K-Best [46] function was applied. The final feature selection is presented in the following section.

5.2. Machine Learning Models Applied

All the considered algorithms were trained both for the balanced and Unbalanced dataset presented in Table 2, to analyze the performance in terms of rate detection differences of each method.

1. KNN: This model was trained using the six features obtained with K-Best. From the balanced dataset 'sbytes', 'dbytes', 'sload', 'smean', 'dmean', 'ct srv dst' were extracted, meanwhile for the unbalanced case, the selected features were 'dur', 'proto', 'service', 'sbytes', 'dttl', 'smean'. The best result was obtained for 12 neighbors (balanced dataset) and 27 neighbors (unbalanced dataset).
2. SVM: Due to the large increase of data from the balanced dataset, the training of this model for this dataset could not be completed, due to memory restrictions in the experiment environment. The model was trained for the unbalanced dataset using features from K-Best analysis ('dur',

'proto', 'service', 'sbytes', 'dttl', 'smean'). The best results were obtained with the hyperparameters: kernel = 'rbf', gamma = 'scale', C = 50 and max_iter = 50,000.

3. DT: The best results were achieved by using the entire dataset. The best hyperparameters obtained were max depth = 28 and random_state = 4 for both balanced and unbalanced datasets.
4. RF: The best results were achieved by using the entire dataset. The best hyperparameters obtained were criterion = 'entropy', max_depth = 50 and *n*_estimators = 50 for both balanced and unbalanced datasets.
5. XGBoost: The best results were achieved by using the entire dataset. The best hyperparameters obtained were *n*_estimators = 500.
6. MLPNN and LSTMNN: The best results were carried by using all the features of the dataset. The model configuration applied was with the model introduced in [2].

In the case of the KNN, SVM, DT, RF, and XGBoost algorithms the cross validation function with GridSearch [47] was applied in order to find the optimal hyperparameters.

5.3. Dynamic Classifier

The dynamic classifier, as presented in Figure 3, was designed to aggregate the predictions from the individual ML models, and make an automatic selection of the optimal prediction obtained from each one for a single sample, while the models are executed in parallel to make predictions over the testing dataset.

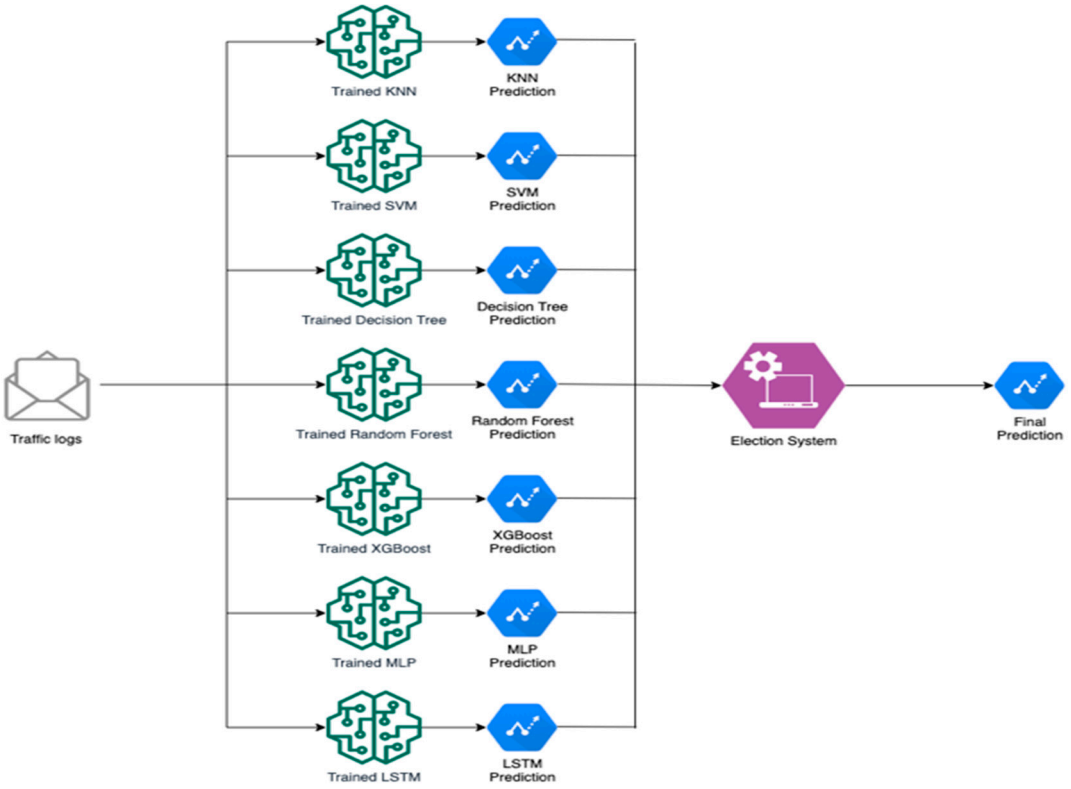


Figure 3. Dynamic classifier system.

Diverse tests were done in order to design the dynamic classifier system i.e., probability analysis, voter model [48], and ensemble ML model [12,23]. After several tests, the dynamic classifier was designed by an ensemble ML model based on XGBoost. The ensemble model was more accurate than the voter model by 3% of difference in comparison with the balanced dataset. In relation with the probability analysis the results were not relevant, so this method was excluded. Also, the use of a

ML algorithm permits increasing the detection rate by optimizing its hyperparameters. Taking into account these results, the ensemble model based on XGBoost was selected for the dynamic classifier.

The goal of the dynamic classifier proposed is to combine opinions of the base experts (individual ML models) in an ensemble, which is shown in the Algorithm 1. Each expert assigns a dynamic coefficient based on its own prediction, over the decision taken from the input data. This weight coefficient (0–1 scale) depends on the general accuracy by each individual model. For this purpose, an alternative dataset was created with the accuracy of each model presented in Section 5.2. The fitting was done selecting as input features, the predictions from the individual models mentioned below, and as outcome the desired classification attack. The proposed system classifies the predictions from the individual models, and the output is based on the relations that the ensemble model has found exposing the most relevant one from the individual algorithms.

The goal of the dynamic classifier is to bring together of several individual models to improve the accuracy of prediction with respect to an individual static model.

Algorithm 1 Ensemble proposed model

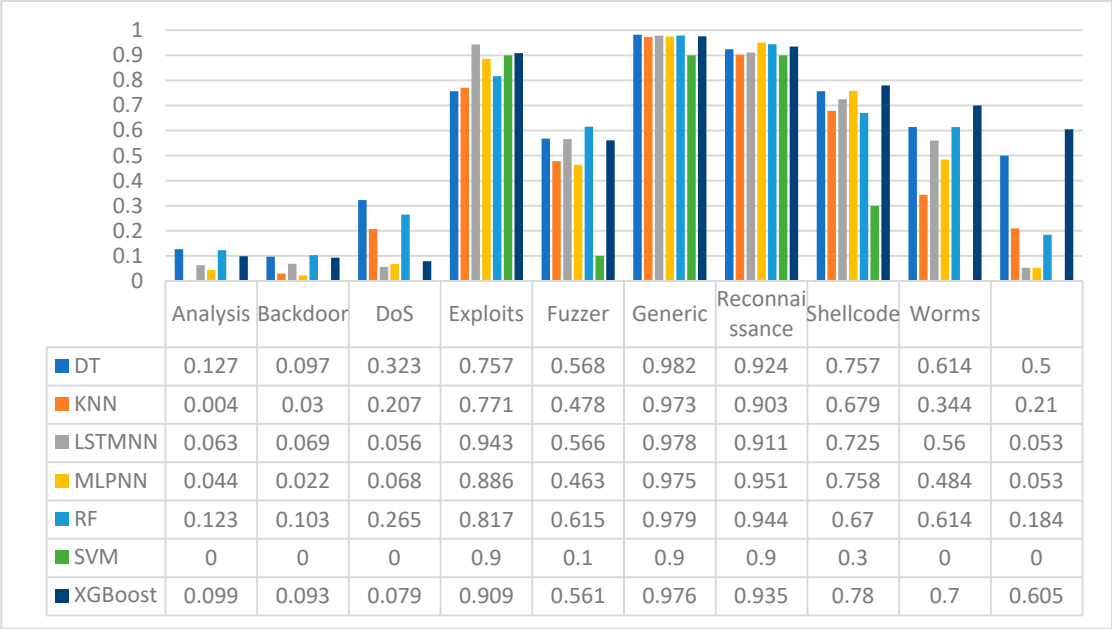
```
Ensembleproposedmodel
open full dataset file
drop incomplete rows
application of feature selection
for each individual ML model
    fit each individual machine learning model
validate each individual machine learning model
save accuracy coefficient
fit dynamic_classifier (x_test =accuracy_coefficient, y_test=attack_categories)
prediction = prediction each individual ML model (accuracy-multiclass)
validate dynamic_classifier (x_validate = prediction)
write dynamic_classifier (True Positive Rate and Accuracy (multiclass-attack))
```

6. System Analysis and Evaluation

6.1. Analysis of Statics Models

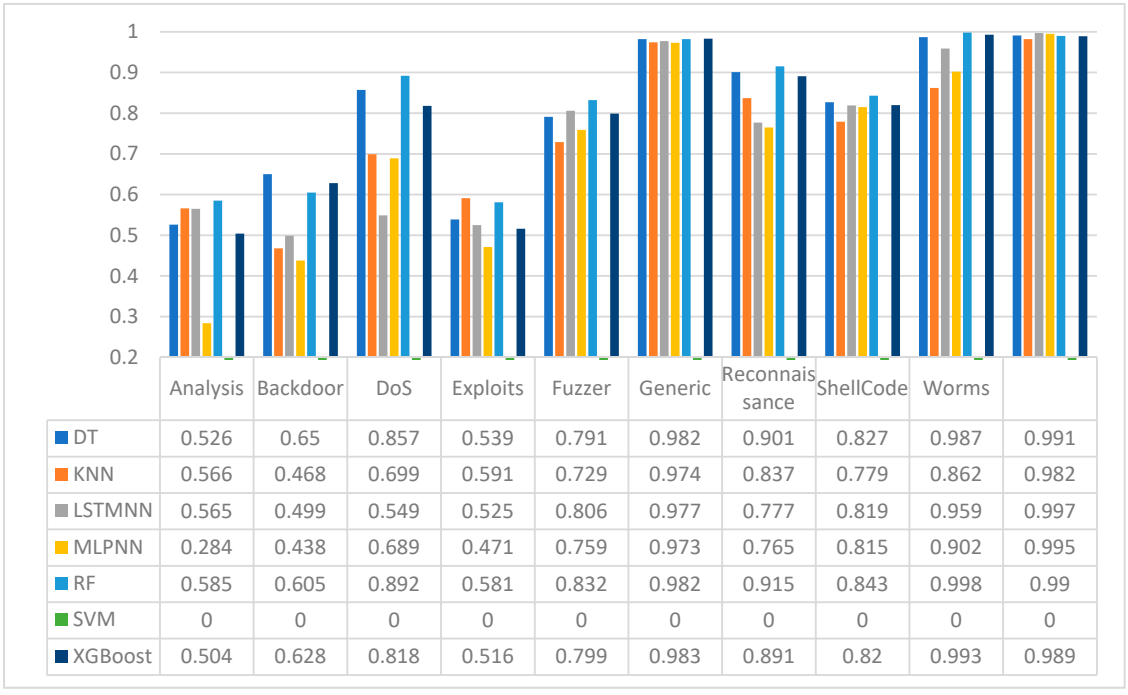
The evaluation metrics presented in this research are derived from the elements of the confusion matrix. The evaluation metric proposed for this research is the true positive rate (TPR) which considers the proportion of real positives correctly predicted, that is, the portion of attacks correctly identified over all possible attacks [12]. In Figure 4, we show the obtained results in terms of TPR of each individual model proposed in the present research for the unbalanced dataset. As we can see some models present a high detection over some types of attacks such as the case of exploits and generic. Instead, in the case of attacks such as analysis, backdoor DoS present a low range of detection, even less than 30% by each class.

In Figure 5, it is presented the obtained results in terms of TPR of each individual model proposed in the present research for the balanced dataset. It is shown how all the models raised the rate of detection over the attacks in comparison with the unbalanced dataset. Despite the increase of rates, in the case of attacks—such as analysis and backdoor exploits—there is still a low rate of detection, even less than 65% for each class. On the other hand, all the models show a decreased the detection rate over class exploits in comparison with the unbalanced dataset. In the case of worms, all the models except SVM (which could not be trained as was exposed in Section 3), raised the rate of detection over the 99%.



	Analysis	Backdoor	DoS	Exploits	Fuzzer	Generic	Reconnaissance	Shellcode	Worms	
DT	0.127	0.097	0.323	0.757	0.568	0.982	0.924	0.757	0.614	0.5
KNN	0.004	0.03	0.207	0.771	0.478	0.973	0.903	0.679	0.344	0.21
LSTMNN	0.063	0.069	0.056	0.943	0.566	0.978	0.911	0.725	0.56	0.053
MLPNN	0.044	0.022	0.068	0.886	0.463	0.975	0.951	0.758	0.484	0.053
RF	0.123	0.103	0.265	0.817	0.615	0.979	0.944	0.67	0.614	0.184
SVM	0	0	0	0.9	0.1	0.9	0.9	0.3	0	0
XGBoost	0.099	0.093	0.079	0.909	0.561	0.976	0.935	0.78	0.7	0.605

Figure 4. True positive rate metrics for static models applied to the unbalanced dataset.



	Analysis	Backdoor	DoS	Exploits	Fuzzer	Generic	Reconnaissance	ShellCode	Worms	
DT	0.526	0.65	0.857	0.539	0.791	0.982	0.901	0.827	0.987	0.991
KNN	0.566	0.468	0.699	0.591	0.729	0.974	0.837	0.779	0.862	0.982
LSTMNN	0.565	0.499	0.549	0.525	0.806	0.977	0.777	0.819	0.959	0.997
MLPNN	0.284	0.438	0.689	0.471	0.759	0.973	0.765	0.815	0.902	0.995
RF	0.585	0.605	0.892	0.581	0.832	0.982	0.915	0.843	0.998	0.99
SVM	0	0	0	0	0	0	0	0	0	0
XGBoost	0.504	0.628	0.818	0.516	0.799	0.983	0.891	0.82	0.993	0.989

Figure 5. True positive rate metrics for static models applied to the balanced dataset.

In Table 3, we summarize the best results in terms of TPR peer class of attack. The unbalanced dataset presents a low rate of detection in different attacks categories such as analysis, backdoor, and DoS. The improved detection is raised by the balance of the data. All the attack categories improved their detection rate once the data was balanced, except for the exploits and generic categories.

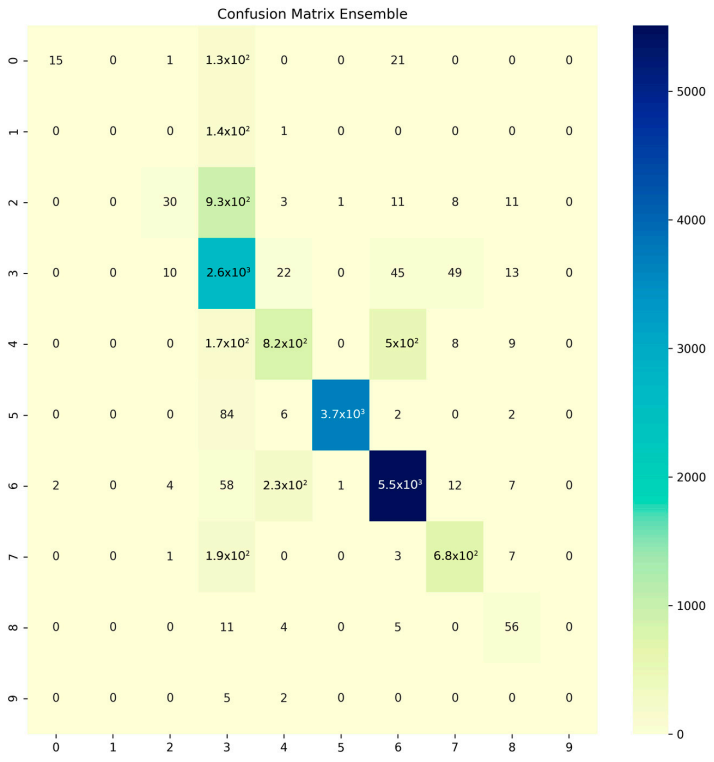
As can be seen in Table 3, different models are more suitable for each different type of attack.

Table 3. Best true positive rate detection per class of attack for balanced and unbalanced datasets applying static models.

Class	Balanced Dataset TPR	Model	Unbalanced Dataset TPR	Model
Analysis	0.585	RF	0.127	DT
Backdoor	0.65	DT	0.103	RF
DoS	0.892	RF	0.323	DT
Exploits	0.591	KNN	0.943	LSTMNN
Fuzzer	0.832	RF	0.615	RF
Generic	0.983	KNN	0.982	DT
Reconnaissance	0.843	RF	0.78	XGBoost
Shellcode	0.998	RF	0.7	XGBoost
Worms	0.997	LSTMNN	0.605	XGBoost

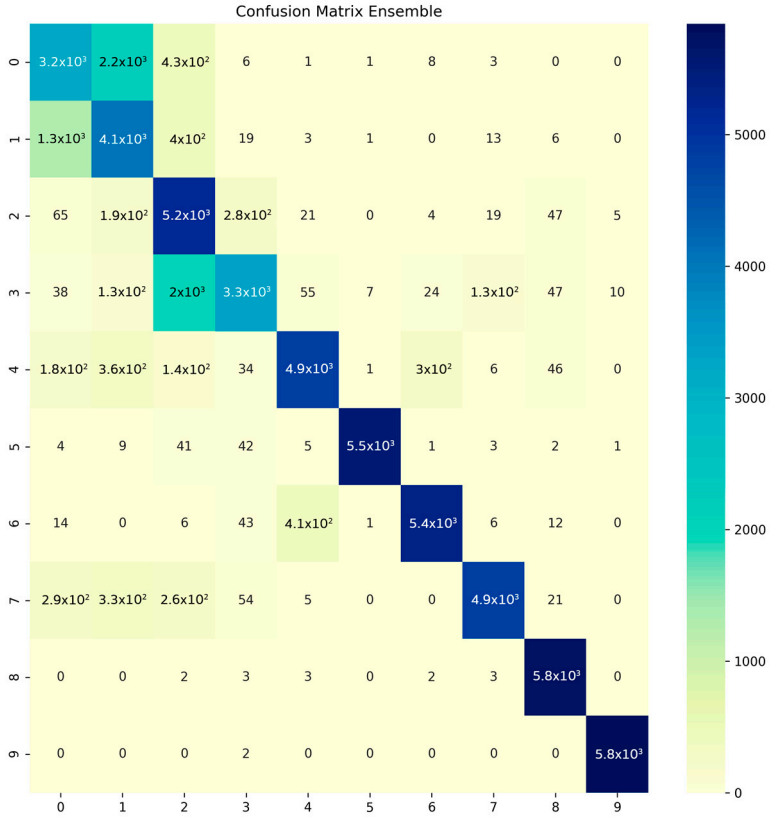
6.2. Dynamic Classifier Results

The dynamic classifier model proposed was based on an ensemble ML model, which consists in obtaining the major benefits per each individual static model, using those introduced in this research work. In Figure 6, we show the confusion matrices for the balanced and unbalanced dataset. The most important differences between both datasets appear on worms, DoS, analysis, and backdoor being practically undetectable using the original dataset. The detection was improved using the SMOTE algorithm, where the lowest rates of detection belongs for analysis and exploits. According to this results, balanced model can improve the accuracy of the ensemble model, since it is able to distinguish better the traffic that is not normal.



(**a**)

Figure 6. *Cont.*



(**b**)

Figure 6. Confusion matrix: (**a**) balanced dataset; (**b**) unbalanced dataset. The class categories in the figure corresponds to: analysis "0", backdoor "1", DoS "2", exploits "3", fuzzers "4", generic "5", normal traffic "6", reconnaissance "7", shellcode "8", worms "9".

We applied different metrics in order to understand better the results given by the proposed models. Table 4 show the results given by the metric TPR for both datasets balanced an unbalanced. In reference to balanced dataset some attacks present a high detection rate over 90% (worms, shellcode, generic, and DoS). In any case, the dynamic classifier presents an acceptable TPR for some attack categories (reconnaissance, fuzzer, and backdoor). On the other hand, classes such as analysis and exploits present the lowest detection rates.

Table 4. True positive rates by class category obtained by the dynamic classifier.

Class	Balanced Dataset TPR	Unbalanced Dataset TPR
Analysis	0.551	0.089
Backdoor	0.701	0.09
DoS	0.921	0.030
Exploits	0.577	0.95
Fuzzer	0.819	0.601
Generic	0.981	0.995
Reconnaissance	0.835	0.744
Shellcode	0.998	0.737
Worms	0.999	0.65

Table 4 also shows the results given by the classification of the dynamic classifier. The results obtained by some attacks are near to 0% (analysis, backdoor, and DoS). These results are related to the

percentage of data by each class, as can be seen Section 3.1.2. On the contrary, the results for the classes exploits and generic and worms were improved in comparison with the results shown in Table 3.

Table 5 shows the results obtained by each individual model proposed and the dynamic classifier. The dynamic classifier improves the results obtained in terms of accuracy up to 5% for the balanced dataset and 3% for the unbalanced dataset being similar for the results obtained the by the F1Score.

Table 5. Comparison between statics models and dynamic classifier in terms of accuracy for balanced and unbalanced dataset.

Model	Balanced Dataset		Unbalanced Dataset	
	Accuracy	F1-Score	Accuracy	F1-Score
KNN	0.739	0.7406	0.779	0.7673
SVM	-	-	0.739	0.685
DT	0.816	0.8066	0.801	0.8062
RF	0.823	0.824	0.828	0.820
XGBoost	0.795	0.796	0.824	0.803
MLPNN	0.709	0.715	0.811	0.784
LSTMNN	0.747	0.754	0.816	0.792
Dynamic Classifier	0.876	0.879	0.851	0.829

As stated in Section 2, there are different approaches for IDS based on ML techniques. Table 6 compares the proposed dynamic classifier for both applied datasets in this research, balanced and unbalanced, with related works. This table shows an improvement in multiclass detection for auto machine learning selection in terms of accuracy in comparison with other studies in the state of the art. The table mention different aspects such as the model, dataset, TPR by attack, and accuracy obtained.

Table 6. Comparison between proposed solution and related works.

Study	Algorithm	Dataset	Data Preprocessing	Feature Selection	Selection Model	Dataset Balancing	TPR	Accuracy
[9]	SVM multiclass	UNSW-NB15	Non-linear scaling method	X	X	X	X	75.77%
[13]	XGBoost multiclass	UNSW-NB15	X	X	X	X	X	86%
[15]	LSTM multiclass	UNSW-NB15	One-hot encoding	X	X	X	X	83%
[23]	Dynamic classifier multiclass	UNSW-NB15	Label encoding	K-best	Voter classifier	X	X	61%
[12]	Dynamic classifier multiclass	NSL-KDD	One-hot encoding	X	Ensemble voting	X	X	85.2
Proposed solution	Dynamic classifier multiclass	UNSW-NB15	Label encoding	Correlation Kendall Coefficient and K-best	Ensemble model	Smote	✓	87.6%

7. Discussion and Conclusions

Most of the diverse studies that proposes IDS based on ML algorithms are based on a static model with an improved performance in terms of accuracy [3,6] with a variation over its hyperparameters. The dataset proposed for this study is the UNSW-NB15 which is considered up to date a benchmark dataset [2]. This dataset has been highly used because its relevance to its recent cyberattacks.

This article exposes through several tests that multiple algorithms can detect significantly better some type of attack over different hyperparameters configurations, as it can be concluded from the data shown in Table 3. Thus, after experimentation with static ML algorithms described in Section 5.2, some categories of attacks presented a better rate of detection as well as RF for attacks—such as analysis, DoS, fuzzers, reconnaissance, and shellcode. However, when comparing these same attacks with the unbalanced dataset, other models allowed obtaining a better detection such as DT, RF, and XGboost.

Following this idea, this research proposes a dynamic auto-selection classifier over different ML models with the objective of obtaining the best capabilities of each individual model to detect cyberattacks. For this purpose, different models purposed in Section 5.3 were studied and tested, obtaining as the best option an ensemble ML based on the XGBoost algorithm [48].

The need of different techniques for data preprocessing and data balancing with the objective to enhance the rate detection was presented in Section 2. We selected the SMOTE to achieve data balancing in our dataset. Also, a feature selection was applied with a correlation index of 0.8 between different features which improved rate detection of the models proposed. Furthermore, two datasets—balanced and unbalanced—were created with the objective to compare the importance of the models proposed.

Several tests and configuration were carried out, with the objective of improving the algorithms' rate detection proposed in Section 3.3. The metric chosen to expose the results is the TPR, which allows evaluate the results in terms of correctly predicted individual classes of attacks. In Figure 4, TPR is taken into account for each individual model for the unbalanced dataset, where the lowest rate of detection was for the categories analysis, backdoor, fuzzers, and worms. In the case of the category attack worms; the algorithms XGBoost and DT could obtain a TPR over 0.5. Other attacks—such as rxploits, generic, and reconnaissance—presented overall an acceptable TPR for each individual algorithm. Comparing the results between Tables 3 and 4, some attacks—such as exploits, fuzzers, generic, and worms—enhanced the detection rate. Instead other categories remaining decreased insignificantly the detection rate. This could be determined by the low numbers of samples for the unbalanced dataset for each class.

On the other hand, for the balanced dataset, static models showed better results in terms of TPR. Figure 5 exposes the enhancing rate detection rate detection up to 50% in comparison with Figure 4. However, the category exploit was the only one decreasing the TPR, up to 40%. The balancing algorithm was successful because it was able to improve the results obtained by the static models. This comparison can be appreciated in Table 3. Furthermore, the application of the dynamic classifier proposed in this research could improve the overall results in terms of TPR. The following classes increased the detection rate: backdoor 5.1%, DoS 2.9%, exploits 8.5%, generic 0.7%, and worms 0.2%. The shellcode class upheld its TPR. Classes such as analysis, fuzzers, and reconnaissance decreased the TPR with a mean of 1.83% which is acceptable, maintaining the highest TPR for the static models proposed.

Finally, Table 5 shows a comparison of the accuracy reached by each individual model and by the dynamic classifier proposed. The dynamic classifier was able to improve the results in 5.3% and 2.3% compared with the best static model for the balanced and unbalanced dataset. The key idea of the model proposed in this research is to auto select, by a ML model, the best rate detection, gathering the individual advantages of each model. We used the method based on an ensemble model based on XGBoost to improve the detection rate. Compared with other related works such as the mentioned in Table 6, it can be seen that our research work achieves a noticeable performance increase in terms of detection rate for IDS based on multiples classes of attacks.

Although our model increases the detection rate, it takes longer time in execution to detect an attack. This happens because the data has to be processed by each individual algorithm and the dynamic classifier model. In a practical scenario, this could delay the time of detection of a possible attack.

For unbalanced classification scenarios, our proposed model also could enhance detection rate as was mentioned above. In practical applications in the area of IDS based on ML algorithms, one of the principal approaches is to improve the quality of the training data, optimizing the features and apply preprocessed methods in order to improve the data quality for the training of the ML algorithms. As for future research lines, some future directions can be considered. The main one is the application of our proposal into a real scenario with real collected data in real-time taking into account the features applied during this research and the proposed models. This can be done with technologies such as Apache Spark Structured Streaming [49] which is designed for real time processing data.

Author Contributions: Conceptualization, V.A.V. and X.L.-N.; Methodology, X.L.-N., C.S.-Z., and V.A.V.; Software, X.L.-N. and C.S.-Z.; Validation, X.L.-N., M.V.-B. and D.R.; Formal analysis, X.L.-N., V.A.V. and M.V.-B.; Investigation, X.L.-N. and C.S.-Z.; Resources, X.L.-N.; Data curation, X.L.-N. and C.S.-Z.; Writing—original draft preparation, X.L.-N.; Writing—review and editing, X.L.-N., V.A.V., M.V.-B. and D.R.; Supervision, V.A.V., M.V.-B. and D.R.; Project administration, V.A.V.; Funding acquisition, V.A.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Masdari, M.; Khezri, H. A survey and taxonomy of the fuzzy signature-based Intrusion Detection Systems. *Appl. Soft Comput.* **2020**, *92*, 106301. [CrossRef]
2. Larriva-Novo, X.A.; Vega-Barbas, M.; Villagra, V.A.; Sanz Rodrigo, M. Evaluation of Cybersecurity Data Set Characteristics for Their Applicability to Neural Networks Algorithms Detecting Cybersecurity Anomalies. *IEEE Access* **2020**, *8*, 9005–9014. [CrossRef]
3. Mishra, P.; Varadharajan, V.; Tupakula, U.; Pilli, E.S. A Detailed Investigation and Analysis of Using Machine Learning Techniques for Intrusion Detection. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 686–728. [CrossRef]
4. Aboueata, N.; Alrasbi, S.; Erbad, A.; Kassler, A.; Bhamare, D. Supervised Machine Learning Techniques for Efficient Network Intrusion Detection. In Proceedings of the 2019 28th International Conference on Computer Communication and Networks (ICCCN), Valencia, Spain, 29 July–1 August 2019; pp. 1–8.
5. Elmasry, W.; Akbulut, A.; Zaim, A.H. Empirical study on multiclass classification-based network intrusion detection. *Comput. Intell.* **2019**, *35*, 919–954. [CrossRef]
6. Hindy, H.; Brosset, D.; Bayne, E.; Seeam, A.; Tachtatzis, C.; Atkinson, R.; Bellekens, X. A Taxonomy and Survey of Intrusion Detection System Design Techniques, Network Threats and Datasets. *arXiv* **2018**, arXiv:1806.03517.
7. Berman, D.S.; Buczak, A.L.; Chavis, J.S.; Corbett, C.L. A Survey of Deep Learning Methods for Cyber Security. *Information* **2019**, *10*, 122. [CrossRef]
8. Teoh, T.T.; Chiew, G.; Franco, E.J.; Ng, P.C.; Benjamin, M.P.; Goh, Y.J. Anomaly detection in cyber security attacks on networks using MLP deep learning. In Proceedings of the 2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE), Kuala Lumpur, Malaysia, 11–12 July 2018; pp. 1–5.
9. Jing, D.; Chen, H.-B. SVM Based Network Intrusion Detection for the UNSW-NB15 Dataset. In Proceedings of the 2019 IEEE 13th International Conference on ASIC (ASICON), Chongqing, China, 29 October–1 November 2019; pp. 1–4.
10. Gauthama Raman, M.R.; Somu, N.; Jagarapu, S.; Manghnani, T.; Selvam, T.; Krithivasan, K.; Shankar Sriram, V.S. An efficient intrusion detection technique based on support vector machine and improved binary gravitational search algorithm. *Artif. Intell. Rev.* **2020**, *53*, 3255–3286. [CrossRef]
11. Zhiqiang, L.; Mohi-Ud-Din, G.; Bing, L.; Jianchao, L.; Ye, Z.; Zhijun, L. Modeling Network Intrusion Detection System Using Feed-Forward Neural Network Using UNSW-NB15 Dataset. In Proceedings of the 2019 IEEE 7th International Conference on Smart Energy Grid Engineering (SEGE), Oshawa, ON, Canada, 12–14 August 2019; pp. 299–303.
12. Gao, X.; Shan, C.; Hu, C.; Niu, Z.; Liu, Z. An Adaptive Ensemble Machine Learning Model for Intrusion Detection. *IEEE Access* **2019**, *7*, 82512–82521. [CrossRef]
13. Husain, A.; Salem, A.; Jim, C.; Dimitoglou, G. Development of an Efficient Network Intrusion Detection Model Using Extreme Gradient Boosting (XGBoost) on the UNSW-NB15 Dataset. In Proceedings of the 2019 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), Ajman, UAE, 10–12 December 2019; pp. 1–7.
14. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the 2015 military communications and information systems conference (MilCIS), IEEE, Canberra, Australia, 11–12 November 2015; pp. 1–6.
15. Gwon, H.; Lee, C.; Keum, R.; Choi, H. Network Intrusion Detection based on LSTM and Feature Embedding. *arXiv* **2019**, arXiv:1911.11552.

16. Guo, H. A bayesian approach for automatic algorithm selection. In Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI03), Workshop on AI and Autonomic Computing, Acapulco, Mexico, 9–15 August 2003; pp. 1–5.
17. Malkomes, G.; Schaff, C.; Garnett, R. Bayesian optimization for automated model selection. In *Advances in Neural Information Processing Systems 29*; Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2016; pp. 2900–2908.
18. Xu, L.; Hoos, H.H.; Leyton-Brown, K. Hydra: Automatically configuring algorithms for portfolio-based selection. In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, Atlanta, GA, USA, 11–15 July 2010; AAAI Press: Palo Alto, CA, USA, 2010; pp. 210–216.
19. Kerschke, P.; Hoos, H.H.; Neumann, F.; Trautmann, H. Automated Algorithm Selection: Survey and Perspectives. *Evol. Comput.* **2018**, *27*, 3–45. [CrossRef]
20. Bischl, B.; Kerschke, P.; Kotthoff, L.; Lindauer, M.; Malitsky, Y.; Fréchette, A.; Hoos, H.; Hutter, F.; Leyton-Brown, K.; Tierney, K.; et al. ASlib: A benchmark library for algorithm selection. *Artif. Intell.* **2016**, *237*, 41–58. [CrossRef]
21. Thornton, C.; Hutter, F.; Hoos, H.H.; Leyton-Brown, K. Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, IL, USA, 11–14 August 2013; Dhillon, I.S., Ed.; Association for Computing Machinery: New York, NY, USA, 2013; pp. 847–855.
22. Kotthoff, L.; Thornton, C.; Hoos, H.H.; Hutter, F.; Leyton-Brown, K. Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. *J. Mach. Learn. Res.* **2017**, *18*, 826–830.
23. Karn, R.R.; Kudva, P.; Elfadel, I.A.M. Dynamic Autoselection and Autotuning of Machine Learning Models for Cloud Network Analytics. *IEEE Trans. Parallel Distrib. Syst.* **2019**, *30*, 1052–1064. [CrossRef]
24. Aceto, G.; Ciuonzo, D.; Montieri, A.; Pescapé, A. Multi-classification approaches for classifying mobile app traffic. *J. Netw. Comput. Appl.* **2018**, *103*, 131–145. [CrossRef]
25. Dainotti, A.; Pescapé, A.; Sansone, C. Early Classification of Network Traffic through Multi-classification. In *Proceedings of the Traffic Monitoring and Analysis*; Domingo-Pascual, J., Shavitt, Y., Uhlig, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; pp. 122–135.
26. Kato, K.; Klyuev, V. Development of a network intrusion detection system using Apache Hadoop and Spark. In Proceedings of the 2017 IEEE Conference on Dependable and Secure Computing, Taipei, Taiwan, 7–10 August 2017; pp. 416–423.
27. Terzi, D.S.; Terzi, R.; Sagiroglu, S. Big data analytics for network anomaly detection from netflow data. In Proceedings of the 2017 International Conference on Computer Science and Engineering (UBMK), Jakarta Indonesia, 20–21 July 2017; pp. 592–597.
28. Krovich, D.; Cottrill, A.; Mancini, D.J. A Cloud Based Entitlement Granting Engine. In Proceedings of the National Cyber Summit, Huntsville, AL, USA, 4–6 June 2019; Choo, K.-K.R., Morris, T.H., Peterson, G.L., Eds.; Springer: Cham, Switzerland, 2019; pp. 220–231.
29. Maciá-Fernández, G.; Camacho, J.; Magán-Carrión, R.; García-Teodoro, P.; Theron, R. UGR '16: A new dataset for the evaluation of cyclostationarity-based network IDSs. *Comput. Secur.* **2018**, *73*, 411–424. [CrossRef]
30. Stiawan, D.; Bin Idris, M.Y.; Bamhdi, A.M.; Budiarto, R. CICIDS-2017 Dataset Feature Analysis with Information Gain for Anomaly Detection. *IEEE Access* **2020**, *8*, 132911–132921. [CrossRef]
31. Revathi, S.; Malathi, D.A. A Detailed Analysis on NSL-KDD Dataset Using Various Machine Learning Techniques for Intrusion Detection. *Int. J. Eng. Res. Technol.* **2013**, *2*, 1848–1853.
32. Kumar, V.; Das, A.K.; Sinha, D. Statistical Analysis of the UNSW-NB15 Dataset for Intrusion Detection. In *Computational Intelligence in Pattern Recognition*; Das, A.K., Nayak, J., Naik, B., Dutta, S., Pelusi, D., Eds.; Springer: Singapore, 2020; pp. 279–294.
33. Moustafa, N.; Slay, J. The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. *Inf. Secur. J. A Glob. Perspect.* **2016**, *25*, 18–31. [CrossRef]
34. Larriva-Novo, X.; Vega-Barbas, M.; Villagrá, V.A.; Rivera, D.; Álvarez-Campana, M.; Berrocal, J. Efficient Distributed Preprocessing Model for Machine Learning-Based Anomaly Detection over Large-Scale Cybersecurity Datasets. *Appl. Sci.* **2020**, *10*, 3430. [CrossRef]

35. Brownlee, J. Supervised and unsupervised machine learning algorithms. *Mach. Learn. Mastery* **2016**, *16*. Available online: https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/ (accessed on 11 August 2020).
36. sklearn.neighbors.KNeighborsClassifier—Scikit-Learn 0.23.2 Documentation. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html (accessed on 11 August 2020).
37. sklearn.tree.DecisionTreeClassifier—Scikit-Learn 0.23.2 Documentation. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html (accessed on 11 August 2020).
38. 3.2.4.3.1. sklearn.ensemble.RandomForestClassifier—Scikit-Learn 0.23.2 Documentation. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (accessed on 11 August 2020).
39. sklearn.svm.SVC—Scikit-Learn 0.23.2 Documentation. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html (accessed on 11 August 2020).
40. Yıldırım, S. Gradient Boosted Decision Trees-Explained. Available online: https://towardsdatascience.com/gradient-boosted-decision-trees-explained-9259bd8205af (accessed on 11 August 2020).
41. Cszas cszas. Intrusion Detection Models. Available online: https://github.com/cszas/IDSModels (accessed on 10 June 2020).
42. Kathiresan, V.; Sumathi, P. An efficient clustering algorithm based on Z-Score ranking method. In Proceedings of the 2012 International Conference on Computer Communication and Informatics, Coimbatore, India, 10–12 January 2012; pp. 1–4.
43. Wang, J.; Xu, M.; Wang, H.; Zhang, J. Classification of Imbalanced Data by Using the SMOTE Algorithm and Locally Linear Embedding. In Proceedings of the 2006 8th international Conference on Signal Processing, Beijing, China, 16–20 November 2006; Volume 3.
44. Zhang, H.; Huang, L.; Wu, C.Q.; Li, Z. An effective convolutional neural network based on SMOTE and Gaussian mixture model for intrusion detection in imbalanced dataset. *Comput. Netw.* **2020**, *177*, 107315. [CrossRef]
45. Gottwalt, F.; Chang, E.; Dillon, T. CorrCorr: A feature selection method for multivariate correlation network anomaly detection techniques. *Comput. Secur.* **2019**, *83*, 234–245. [CrossRef]
46. Huang, L.; Chiang, D. Better k-best Parsing. In Proceedings of the Ninth International Workshop on Parsing Technology, Vancouver, BC, Canada, 9–10 October 2005; Association for Computational Linguistics: Stroudsburg, PA, USA, 2005; pp. 53–64.
47. sklearn.model_selection.GridSearchCV—Scikit-Learn 0.23.2 Documentation. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (accessed on 12 August 2020).
48. Orlenko, A.; Moore, J.H.; Orzechowski, P.; Olson, R.S.; Cairns, J.; Caraballo, P.J.; Weinshilboum, R.M.; Wang, L.; Breitenstein, M.K. Considerations for automated machine learning in clinical metabolic profiling: Altered homocysteine plasma concentration associated with metformin exposure. In *Biocomputing–World Scientific 2018*; Altman, R.B., Dunker, A.K., Hunter, L., Ritchie, M.D., Murray, T.A., Klein, T.E., Eds.; World Scientific Publishing Co.: Singapore; pp. 460–471. ISBN 978-981-323-552-6.
49. Structured Streaming. In Proceedings of the 2018 International Conference on Management of Data, Houston, TX, USA, 10–15 June 2018; Available online: https://dl.acm.org/doi/abs/10.1145/3183713.3190664 (accessed on 18 October 2020).