A Probabilistic-driven Ensemble Approach to Perform Event Classification in Intrusion Detection System

Roberto Saia, Salvatore Carta and Diego Reforgiato Recupero
Department of Mathematics and Computer Science
University of Cagliari, Via Ospedale 72 - 09124 Cagliari, Italy

Keywords: Intrusion Detection, Pattern Mining, Machine Learning, Ensemble Approach, Probabilistic Model.

Abstract: Nowadays, it is clear how the network services represent a widespread element, which is absolutely essential for each category of users, professional and non-professional. Such a scenario needs a constant research activity aimed to ensure the security of the involved services, so as to prevent any fraudulently exploitation of the related network resources. This is not a simple task, because day by day new threats arise, forcing the research community to face them by developing new specific countermeasures. The Intrusion Detection System (IDS) covers a central role in this scenario, as its main task is to detect the intrusion attempts through an evaluation model designed to classify each new network event as normal or intrusion. This paper introduces a Probabilistic-Driven Ensemble (PDE) approach that operates by using several classification algorithms, whose effectiveness has been improved on the basis of a probabilistic criterion. A series of experiments, performed by using real-world data, show how such an approach outperforms the state-of-the-art competitors, proving its better capability to detect intrusion events with regard to the canonical solutions.

1 INTRODUCTION

Cybersecurity is an increasingly crucial aspect in our times that is greatly dependent on network services, since they affect the everyday life, involving areas such as industry, education, finance, medicine, and so on. In such a scenario the *Intrusion Detection Systems* (*IDSs*) (Buczak and Guven, 2016) covered an important role, because their main task is to face the risks related to unauthorized use of such services by malicious people. Such systems have been designed in order to overcome the limits of the canonical approaches, such as, for instance, those based on authentication procedures, on data encryption, or on specific rules (e.g., firewalls).

They operate by exploiting a number of techniques aimed to detect anomalous activities related to a single machine or to an entire network. Examples of such techniques are the *Neural Networks*, the *Fuzzy Logic*, the *Genetic Algorithms*, the *Clustering Algorithms*, and the *Support Vector Machines*. In addition, hybrid techniques that combine several techniques can be used.

It should be observed how, regardless of the used approach, the intrusion detection represents a hard task, due to the high dynamism and heterogeneity of the involved scenarios, both in terms of network infrastructure and possible attacks. Considering that many attacks are quite similar to the behavior of the normal activities, it is clear that it is difficult for a classification model to distinguish one from another.

The proposed classification approach is based on several state-of-the-art techniques and the event classification is performed on the basis of the probabilities of possible outcomes of each classification. The aim is to improve the capability of an *IDS* to detect intrusions. The scientific contribution given by this paper is the definition of an ensemble approach of classification able to evaluate a new event by exploiting several state-of-the-art classification algorithms, which are driven by a probabilistic model.

2 BACKGROUND AND RELATED WORK

The concept of intrusion has been well discussed in (Bace and Mell, 2001), where the authors describe it as an activity aimed to compromise or bypass the security policies that regulate a network or a single machine. As defined in many authoritative studies (Pfleeger and Pfleeger, 2012), computer security attempts to ensure three main aspects: *confiden-*

Saia, R., Carta, S. and Recupero, D.
A Probabilistic-driven Ensemble Approach to Perform Event Classification in Intrusion Detection System.
DOI: 10.5220/0006893801410148
In *Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2018) - Volume 1: KDIR*, pages 141-148
ISBN: 978-989-758-330-8
Copyright © 2018 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

tiality, *integrity*, and *availability*, which are compromised (one or more) by an intrusion.

- *Confidentiality*: it grants that the resources, provided by a single machine or a network, are accessed only by the authorized users.
- *Integrity*: it grants that such resources can be modified only by the authorized users, according to the permissions with which they have been made available.
- *Availability*: it grants that such resources are available to authorized users in accordance with the established times, without any unwanted interruption or limitation.

An *Intrusion Detection System* (*IDS*) is aimed to analyze the network traffic or the activity of a single machine in order to discover non-authorized activities. Such activities can be originated by a malware (e.g., a virus) or can be related to an human attack (e.g., the attempt to gain an access to a resource) operated locally or remotely.

According to the literature (Ghorbani et al., 2010), there are several aspects that can be used to classify the *IDSs*, as reported in Table 1. Such aspects covered two different areas, *functional* and *non-functional*. *Functional* because they refer to intrinsic *IDS* characteristics, whereas *non-functional* because they refer to external characteristics that are not strictly related to the *IDS*.

Table 1: IDSs Classification.

Aspect	Possibilities	Area
Detection Approach	*Anomaly, Misuse, or Specification-based*	*Functional*
System Modality	*Host, Network, Network-Node, or Hybrid based*	*Functional*
System Response	*Passive or Active*	*Functional*
System Activity	*Continuous or Periodic*	*Non-functional*

On the basis of the adopted detection approach, the *IDSs* can be classified into three categories (Ghorbani et al., 2010):

1. *Anomaly Detection* approach (Bronte et al., 2016; Holm, 2014), also known as *Signature-based Detection* approach, operates by comparing the new events to a set of signatures related to a series of known intrusion events;
2. *Misuse Detection* approach (Garcia-Teodoro et al., 2009; Viegas et al., 2017), also known as *Anomaly-based Detection* approach, works on the basis of a model able to characterize the normal activities, which is used to detect intrusion attempts in the new events;
3. *Specification-based Detection* approach (Uppuluri and Sekar, 2001; Sekar et al., 2002) operates by defining the allowed behavior of the systems in the *IDS* area, considering as intrusion the events that does not respect it. This approach has been designed to combine the strengths of the *Anomaly Detection* and *Misuse Detection* approaches.

On the basis of its operative modality an *IDS* can operate as follows:

- *Host-based* modality (Bukac et al., 2012): it operates by exploiting a number of agents installed on a single machine that belongs to the network to be monitored. In this case, all events on the machine (e.g., active processes, log files, configuration alteration, etc.) are analyzed by the agents and the result is compared to a series of known attack patterns stored in a database. When an attack is detected, the system can send an alert or can activate appropriate countermeasures. Such a modality allows a system to use many agents in order to gain a higher level of protection, with the possibility to perform a more detailed configuration of them. A system that adopts this modality is named as *Host-based Intrusion Detection System* (*HIDS*) and its main disadvantages are the excessive number of false positives and false negatives, and the latency between when the event occurs and when it is processed.
- The *Network-based* modality (Das and Sarkar, 2014): it operates by monitoring and analyzing the network traffic in order to detect any attempt of intrusion. In this case, a twofold approach is followed: first a new event is compared with a database of known patterns (signature matching) and, when this operation fails, it is analyzed in order to detect anomalies (network analysis). It allows a system to detect also unknown attacks and the result can be a simple notification or the activation of automatic countermeasures, such as, for instance, the block of the IP address of the machine that is performing the attack. A system that adopts this modality is named *Network Intrusion Detection System* (*NIDS*) and the related disadvantages are the difficulty to well operate in networks characterized by a high level of traffic, since they have to analyze all the involved packets, and its inability to analyze encrypted data. Such a system can not operate proactively, because an attack can be detected only after it begins.
- The *Network-Node-based* modality (Potluri and Diedrich, 2016): it operates by analyzed the network traffic of a single network node. It is different from the *HIDS* and *NIDS* modalities, since in this case the traffic taken into account does not belong to a single machine or the entire network, but it is related to a precise node of the network, usually chosen for its strategical position. It can be considered a mix of the aforementioned *HIDS*

and *NIDS* modalities, and a system that adopts it is defined *Network Node Intrusion Detection System* (*NNIDS*).
- The *Hybrid-based* modality (Amrita and Ravulakollu, 2018): it combines yhe aforementioned modalities to improve the system performance. In such a context, the systems are usually defined as *Hybrid Intrusion Detection Systems* or as *Distributed Intrusion Detection Systems*.

An intrusion event detected by an *IDS* can lead toward a *passive* or *active* response (Bijone, 2016), as detailed in the following.

- *Passive* response: an *IDS* only sends alerts to security managers, without triggering any automatic countermeasure.
- *Active* response: an *IDS* operates by performing specific countermeasures, e.g., by closing a network connection or a process.

The *system activity* is another criterion adopted to classify the *IDSs*, as reported in the following.

- *Continuous Activity*: systems included in this group are those that collect and process the events in real-time. It means that such systems constantly monitor the area in which they operate-
- *Periodic Activity*: *IDSs* that not process the collected events in real-time but periodically are included in this group.

Ensemble Methods: Literature reports that the combined approaches are usually adopted in order to improve the classification performance. It means that an ensemble strategy commonly gets better performance than that based on a single approach (Dietterich, 2000; Zainal et al., 2008). However, it should be noted that it is not the norm, since a combination of approaches can also worsen the classification performance (Gomes et al., 2017).

The ensemble process is performed by using a *dependent framework* (i.e., the construction of each approach depends on the output of the previous one) or by using an *independent framework* (i.e., the construction of each approach is independent of the other ones) (Sagi and Rokach, 2018). In this paper we take into account only this last type of framework.

In the context of the intrusion detection the ensemble solutions are aimed to improve the detection of the intrusion attempts, reducing at the same time the number of misclassification. Such solutions are implemented by following two steps: in the first step a set of approaches is defined on the basis of their result complementarity (i.e., different correct and wrong classifications); in the second step the single results of the approaches are combined into one by using a strategy (e.g., *full agreement*, *majority voting*, or *weighted voting*).

It should be noted how some approaches (e.g., *Gradient Boosting* and *AdaBoost*) work by following ensemble criteria, by exploiting an ensemble of weak prediction approaches (Guo et al., 2017).

Evaluation Metrics: Literature reports a number of metrics adopted in order to evaluate the *IDS* performance, as well as several datasets used in this process (Munaiah et al., 2016). Considering that the main objective of an *IDS* is the correct classification of the events in two classes (i.e., *normal* and *intrusion*), metrics aimed to evaluate the binary classifiers, such as those based on the *confusion matrix*[1] (i.e., *F-score*, *Accuracy*, *Sensitivity*, and *Specificity*), or those able to assess the effectiveness of the adopted classification model (i.e., *ROC* and *AUC*[2]) are usually used.

In addition, many works take also into account other aspects, such as those related to the *IDS* performance, by evaluating, for instance, the *computational load*, the *computational time*, or the *memory usage*. Together with the aforementioned functional aspects, further metrics can be taken into account to evaluate some financial aspects, such as, for instance, the costs related to the needed hardware and software or those related to the *IDS* administration (Sommer, 1999).

The systematic literature review performed in (Munaiah et al., 2016) shows that there is not an unique criterion to evaluate the performance of an *IDS*, whereas it is fairly common to use more than one metric to evaluate the proposed approaches.

Open Problems: We have previously highlighted how the intrusion detection task is not easy, mainly for the characteristics of the domain taken into account, which presents an high degree of dynamism and heterogeneity. In addition, we underlined how many *intrusion* attempts follow a behavior similar to that of the *normal* activities, therefore it is not simple for an *IDS* to correctly detect the *intrusions*. Some major issues related to the state-of-the-art solutions have been summarized in the following:

- one of the most important issues is related to the *Anomaly Detection* approach. It mainly depends on the difficulty to define, exhaustively, a signature for each possible intrusion event. It leads towards a huge number of false alarms;
- another issue arises in the context of the *Misuse Detection* approach. It is related to the inability for the system to recognize unknown intrusion

[1] A matrix 2x2 where are reported the number of *True Negatives* (TN), *False Negatives* (FN), *True Positives* (TP), and *False Positives* (FP).

[2] Respectively, *Receiver Operating Characteristic* and *Area Under the Receiver Operating Characteristic*.

patterns, and this prevents the detection of novel type of intrusion attempts;
- considering that the *Specification-based* approach combines the *Anomaly Detection* and *Misuse Detection* approaches, it inherits the aforementioned issues.

As far as the aforementioned problems are concerned, there is the issue related to the observation that many intrusion detection approaches are designed to operate with a specific class of attacks. This goes against the real-world scenarios, which are characterized by a various classes of attacks, also by considering the zero-day ones.

For this reason the ensemble approach proposed in this paper has been designed to operate in a heterogeneous scenario in terms of classes of attacks, and therefore, its performance has been evaluated both in terms of single class of attacks and in terms of all classes of attacks, by using a real-world dataset.

3 NOTATION AND PROBLEM DEFINITION

This section reports the formal notation adopted in this paper together with the formalization of the faced problem.

Formal Notation: Given a set of classified events $E = \{e_1, e_2, \ldots, e_N\}$, we denote as E_+ the subset of *legitimate* ones (then $E_+ \subseteq E$), and as E_- the subset of *anomalous* ones (then $E_- \subseteq E$).

Each event $e \in E$ is composed by a set of values $V = \{v_1, v_2, \ldots, v_M\}$ and each event can belong only to one class $c \in C$, where $C = \{normal, intrusion\}$.

We also denote a set of of unclassified events $\hat{E} = \{\hat{e}_1, \hat{e}_2, \ldots, \hat{e}_U\}$ and a set of classification algorithms $A = \{a_1, a_2, \ldots, a_Z\}$.

Problem Definition: First, we denote as Ξ the event classification process performed by our approach. Second, we define a function $\text{EVAL}(\hat{e}, \Xi)$ able to evaluate the correctness of the classification by the returned boolean value β (*0=misclassification*, *1=correct classification*). Finally, our problem can be formalized as maximization of the sum of the returned values, as shown in Equation 1.

$$\max_{0 \leq \beta \leq |\hat{E}|} \beta = \sum_{u=1}^{|\hat{E}|} \text{EVAL}(\hat{e}_u, \Xi) \tag{1}$$

4 IMPLEMENTATION

The probabilistic model used to drive the classification performed by our ensemble approach is based on the *Logistic Regression* algorithm. In more detail, a logistic model has been used in order to evaluate the probability of a binary response based on more independent predictors (i.e., the adopted state-of-the-art-approaches). More formally, we evaluate the probability that an unevaluated event $\hat{e} \in \hat{E}$ belongs to a predicted class $c \in C$ by mapping the predicted values to probabilities by using the sigmoid σ function[3].

Such a function is able to map any real value into [0,1] and it is largely used in machine learning to map predictions to probabilities. It is shown in Equation 2, where $\sigma(a_z(p))$ denotes the probability estimate for the prediction p made by the algorithm a_z and expressed in the range $[0,1]$, and e is the base of natural log.

$$\sigma(a_z(p)) = \frac{1}{1+e^{-p}} \tag{2}$$

According to Equation 2, we define our probabilistic model by following the criterion indicated in Equation 3, where Z denotes the number of algorithms (i.e., $Z = |A|$) used to classify an event $\hat{e} \in \hat{E}$ by using our ensemble approach, and $c(\hat{e})$ denotes the classification given to the event $\hat{e}$.

$$\begin{aligned}
&\overline{\sigma} = \tfrac{1}{Z} \cdot \sum_{z=1}^{Z} \sigma(a_z(p)) \\
&w_n = \sum_{z=1}^{Z} 1 \;\; if \;\; \sigma(a_z(p)) > \overline{\sigma} \;\wedge\; a_z(p) = \textit{normal} \\
&w_i = \sum_{z=1}^{Z} 1 \;\; if \;\; \sigma(a_z(p)) < \overline{\sigma} \;\vee\; a_z(p) = \textit{intrusion} \\
&c(\hat{e}) = \begin{cases} \textit{normal}, & \text{if } w_n > w_i \\ \textit{intrusion}, & \text{otherwise} \end{cases}
\end{aligned} \tag{3}$$

Our model is aimed to exclude from the classification process those algorithms characterized by a low level of probability in their predictions, combining the effectiveness of the most performing ones through a weighted probabilistic criterion.

On the basis of the probabilistic model, previously formalized, the unevaluated events in $\hat{E}$ are classified by using the Algorithm 1.

It takes as input the set of adopted evaluation algorithms A, the set of past classified events E, and the event to evaluate $\hat{e}$. It returns the classification (i.e., as *normal* or *intrusion*) of the event $\hat{e}$.

5 EXPERIMENTS

This section describes the environment used to perform the experiments, the adopted real-world dataset, the metrics used for the performance evaluation, the experimental strategy, and the obtained results, which are discussed at the end.

[3] A mathematical function having a characteristic *sigmoid curve*.

Algorithm 1: *Transaction classification.*

Input: A=Set of algorithms, E=Past classified events, $\hat{e}$=Event to evaluate
Output: *result*=Event $\hat{e}$ classification

```
1: procedure CLASSIFICATION(A, E, ê)
2:     w1 ← 0
3:     w2 ← 0
4:     models = trainingModels(A,E)
5:     predictions = getPredictions(models, ê)
6:     μ ← getProbabilityAverage(predictions)
7:     for each p in predictions do
8:         if getProbability(p) > μ ∧ p == normal then
9:             w1 ← w1 + 1
10:        else
11:            w2 ← w2 + 1
12:        end if
13:    end for
14:    if w1 > w2 then
15:        result ← normal
16:    else
17:        result ← intrusion
18:    end if
19:    return result
20: end procedure
```

5.1 Environment

The approach proposed in this paper has been developed in *Python*, as well as the implementation of the state-of-the-art classification techniques used to define our ensemble approach, which are based on *scikit-learn* [4]. In order to make our experimental results reproducible, we have set the seed of the pseudo-random number generator used by the *scikit-learn* evaluation algorithms.

5.2 Dataset

The real-world dataset used to evaluate the proposed approach is *NSL-KDD*[5]. It represents an improved version of the old *KDD-CUP99*, which in addition to being dated it presents some problems (Wang et al., 2014) that do not make it suitable for the evaluation of the recent *IDSs*. The most significant issue related to the former version of the dataset is the data redundancy. It means that both the training and test sets include a big number of duplicate events and these influence the machine learning approaches, reducing their capability to detect the less frequent events correctly.

These issues have been addressed in the *NSL-KDD* as follows: (*i*) there are not redundant events in the train and test sets; (*ii*) the number of events that belong to each difficulty level group is inversely proportional to the percentage of events present in the former dataset; (*iii*) the events in the train and test sets are enough to perform the experiments without the need to use a subset of them (operation usually done in the former dataset in order to reduce the computational complexity). Such improvements mitigate the aforementioned issues and are able to reproduce a typical real-world scenario that allow us to evaluate the performance of the proposed approach and that of the competitors.

An overview of the dataset characteristics is shown in Table 2, where we can observe its composition in terms of *normal* (i.e., set $|E_+|$) and *intrusion* (i.e., set $|E_-|$) events. It should be observed how the cardinality of classes (i.e., set $|C|$) is different in the training and test sets, since some classes of events are present in a dataset and not in the other one, and vice versa.

Table 2: Dataset Overview.

Dataset	Total events $\|E\|$	Normal $\|E_+\|$	Intrusion $\|E_-\|$	Values $\|V\|$	Classes $\|C\|$
Training	125,973	67,343	58,630	41	23
Test	22,543	9,710	12,833	41	38
Total	148,516	77,053	71,463		

An analysis of the distribution of event types is shown in Table 3, where we have grouped all types of events into the following categories:

- *Privilege Escalation Attack* (*PEA*): this category contains all the attacks aimed to obtain a privileged access by starting from an unprivileged status, as it happens, for instance, in a *buffer overflow* attack;
- *Denial of Service Attack* (*DSA*): this category includes all the attacks aimed to stop a service/system by using a very high number of legitimate requests, e.g., as it happens in a *syn flooding* attack;
- *Remote Scanning Attack* (*RSA*): this category classifies the attacks conducted to obtain information about one or more services/systems, by adopting non-invasive and invasive techniques, e.g., as it happens during a *port scanning* activity;
- *Remote Access Attack* (*RAA*): this category classifies all the attacks where the goal is to gain an access to a remote system, without using sophisticate techniques, e.g., by using a *brute-force* attack to guess the user credentials;
- *Normal Network Activity* (*NNA*): this last category contains the normal activity, then all the legitimate network traffic not related to intrusion attempts.

In more detail, in the context of the *NSL-KDD* dataset, the aforementioned four categories of intrusion events are related to the following major attacks:

- **PEA:** *Buffer_overflow*, *Loadmodule*, *Rootkit*, *Perl*, *Sqlattack*, *Xterm*, and *Ps*;
- **DSA:** *Back*, *Land*, *Neptune*, *Pod*, *Smurf*, *Teardrop*, *Mailbomb*, *Processtable*, *Udpstorm*, *Apache2*, and

[4] http://scikit-learn.org
[5] https://github.com/defcom17/NSL_KDD

Worm;
- **RSA:** *Satan*, *IPsweep*, *Nmap*, *Portsweep*, *Mscan*, and *Saint*;
- **RAA:** *Guess_password*, *Ftp_write*, *Imap*, *Phf*, *Multihop*, *Warezmaster*, *Xlock*, *Xsnoop*, *Snmpguess*, *Snmpgetattack*, *Httptunnel*, *Sendmail*, and *Named*.

Table 3: Events Distribution.

	Event	Training	Test	Type		Event	Training	Test	Type
01	*apache2*	0	737	*DSA*	21	*processtable*	0	685	*DSA*
02	*back*	956	359	*DSA*	22	*ps*	0	15	*PEA*
03	*buffer_overflow*	30	20	*PEA*	23	*rootkit*	10	13	*PEA*
04	*ftp_write*	8	3	*RAA*	24	*saint*	0	319	*RSA*
05	*guess_passwd*	52	1231	*RAA*	25	*satan*	3633	735	*RSA*
06	*httptunnel*	0	133	*RAA*	26	*sendmail*	0	14	*RAA*
07	*imap*	11	1	*RAA*	27	*smurf*	2646	665	*DSA*
08	*ipsweep*	3599	141	*RSA*	28	*snmpgetattack*	0	178	*RAA*
09	*land*	18	7	*DSA*	29	*snmpguess*	0	331	*RAA*
10	*loadmodule*	9	2	*PEA*	30	*sqlattack*	0	2	*PEA*
11	*mailbomb*	0	293	*DSA*	31	*spy*	2	0	*RAA*
12	*mscan*	0	996	*RSA*	32	*teardrop*	892	12	*DSA*
13	*multihop*	7	18	*RAA*	33	*udpstorm*	0	2	*DSA*
14	*named*	0	17	*RAA*	34	*warezclient*	890	0	*RAA*
15	*neptune*	41214	4657	*DSA*	35	*warezmaster*	20	944	*RAA*
16	*nmap*	1493	73	*RSA*	36	*worm*	0	2	*DSA*
17	*perl*	3	2	*PEA*	37	*xlock*	0	9	*RAA*
18	*phf*	4	2	*RAA*	38	*xsnoop*	0	4	*RAA*
19	*pod*	201	41	*DSA*	39	*xterm*	0	13	*PEA*
20	*portsweep*	2931	157	*RSA*	40	*normal*	67343	9710	*NNA*

Table 4 summarises the information previously presented in Table 3. It gives us a measure about the relevance of each class of attacks in the dataset.

Table 4: Events Overview.

Dataset	PEA	DSA	RSA	RAA	NNA
Training	52	45,927	11,656	994	67,343
Test	67	7,460	2,421	2,885	9,710
Total	119	53,387	14,077	3,879	77,053
%	0.08	35.95	9.48	2.61	51.88

5.3 Strategy

Algorithms Selections: The five competitor algorithms *Multilayer Perceptron* (*MLP*), *Decision Tree* (*DTC*), *Adaptive Boosting* (*ADA*), *Gradient Boosting* (*GBC*), and *Random Forests* (*RFC*), have been selected in order to have different classification techniques and they include the most performing ones. Considering that the differences between the results given by our *PDE* approach and those given by the single competitor approaches were almost the same even after a parameter tuning, all the experiments have been conducted by using the default *scikit-learn* values.

Data Preprocessing: Before the experiments, we converted all the categorical dataset features into a numerical form and, in addition, in order to be able to perform a binary classification task, we added a new feature (i.e., named *class*) that represents the two possible categories an event can be categorized (i.e., *1 = normal* and *2 = intrusion*).

Evaluation Criterion: Considering that the canonical *k-fold cross-validation* criterion used to reduce the impact of the data dependency during the experiments does not work in the case of *time series* data, since it does not take into account the event chronology, we adopted a different approach based on the *TimeSeriesSplit scikit-learn* functionalities. It is a *time series cross-validation* criterion able to divide a dataset into *n_splits* training and test sets, respecting the data chronology. In our case we used *TimeSeriesSplit* with *n_splits=10*.

Table 5: Events Detection Performance.

Approach	*Events*	*TNR*	*F-score*	*AUC*	*Average*
Multilayer Perceptron (MLP)	*PEA*	0.178	0.187	0.589	0.318
Decision Tree (DTC)	*PEA*	0.484	0.468	0.742	0.565
Adaptive Boosting (ADA)	*PEA*	0.496	0.565	0.748	0.603
Gradient Boosting (GBC)	*PEA*	0.369	0.410	0.684	0.488
Random Forests (RFC)	*PEA*	0.288	0.374	0.644	0.435
PDE	*PEA*	0.610	0.516	0.805	**0.644**
Multilayer Perceptron (MLP)	*DSA*	0.964	0.969	0.974	0.969
Decision Tree (DTC)	*DSA*	0.991	0.994	0.994	**0.993**
Adaptive Boosting (ADA)	*DSA*	0.990	0.993	0.994	0.992
Gradient Boosting (GBC)	*DSA*	0.991	0.994	0.994	**0.993**
Random Forests (RFC)	*DSA*	0.988	0.993	0.993	0.991
PDE	*DSA*	0.993	0.993	0.994	**0.993**
Multilayer Perceptron (MLP)	*RSA*	0.897	0.848	0.929	0.891
Decision Tree (DTC)	*RSA*	0.977	0.980	0.987	0.981
Adaptive Boosting (ADA)	*RSA*	0.974	0.977	0.985	0.979
Gradient Boosting (GBC)	*RSA*	0.971	0.976	0.984	0.977
Random Forests (RFC)	*RSA*	0.971	0.980	0.985	0.979
PDE	*RSA*	0.988	0.971	0.989	**0.983**
Multilayer Perceptron (MLP)	*RAA*	0.345	0.303	0.663	0.437
Decision Tree (DTC)	*RAA*	0.851	0.858	0.924	0.878
Adaptive Boosting (ADA)	*RAA*	0.803	0.846	0.901	0.850
Gradient Boosting (GBC)	*RAA*	0.836	0.858	0.917	0.870
Random Forests (RFC)	*RAA*	0.846	0.876	0.923	0.882
PDE	*RAA*	0.885	0.834	0.940	**0.886**
Multilayer Perceptron (MLP)	*NNA*	0.885	0.900	0.908	0.898
Decision Tree (DTC)	*NNA*	0.969	0.979	0.981	0.976
Adaptive Boosting (ADA)	*NNA*	0.947	0.965	0.967	0.960
Gradient Boosting (GBC)	*NNA*	0.966	0.978	0.979	0.974
Random Forests (RFC)	*NNA*	0.964	0.978	0.980	0.974
PDE	*NNA*	0.980	0.976	0.977	**0.978**

5.4 Metrics

According to the considerations made in Section 2, setting the focus on the functional aspect, we evaluated the performance of the proposed approach on the basis of three metrics. The first considered metric is the *Specificity* (*true negative rate*) as it is more crucial to perform a correct classification of the intrusion events rather than normal events, and, therefore, it is better to adopt a prudent policy preferring to get more *false negative* than *false positives*. The second considered metric is the *F-score* metric, since it offers an overview in terms of *Accuracy* and *Recall* performance. The last metric taken into account is *AUC*, since it is able to evaluate the capability of an evaluation model to distinguish between two different classes of events (i.e., *normal* and *intrusion*).

Table 6: Ensemble Strategies Performance.

Strategy	*Events*	*TNR*	*F-score*	*AUC*	*Average*
Full Agreement	*PEA*	0.048	0.081	0.524	0.218
Majority Voting	*PEA*	1.000	0.003	0.500	0.501
Weighted voting	*PEA*	0.409	0.521	0.705	0.545
PDE	*PEA*	0.610	0.516	0.805	**0.644**
Full Agreement	*DSA*	0.972	0.980	0.982	0.979
Majority Voting	*DSA*	1.000	0.581	0.500	0.694
Weighted voting	*DSA*	0.990	0.994	0.994	**0.993**
PDE	*DSA*	0.993	0.993	0.994	**0.993**
Full Agreement	*RSA*	0.900	0.927	0.946	0.924
Majority Voting	*RSA*	1.000	0.269	0.500	0.590
Weighted voting	*RSA*	0.974	0.979	0.985	0.979
PDE	*RSA*	0.989	0.971	0.990	**0.983**
Full Agreement	*RAA*	0.312	0.409	0.656	0.454
Majority Voting	*RAA*	1.000	0.089	0.500	0.530
Weighted voting	*RAA*	0.837	0.871	0.918	0.878
PDE	*RAA*	0.885	0.837	0.940	**0.886**
Full Agreement	*NNA*	0.931	0.946	0.950	0.935
Majority Voting	*NNA*	1.000	0.650	0.500	0.717
Weighted voting	*NNA*	0.963	0.977	0.978	0.972
PDE	*NNA*	0.980	0.975	0.976	**0.978**

Table 7: F-score and AUC Differences.

Events	F-score	AUC
PEA	-0.005	$+0.100$
DSA	-0.001	$+0.000$
RSA	-0.007	$+0.004$
RAA	-0.039	$+0.020$
NNA	-0.000	-0.001
Average	-0.010	$+0.025$

5.5 Results

The performed experiments were aimed to evaluate the performance of several state-of-the-art approaches when they are used in order to distinguish the *intrusion* events from the *normal* ones.

We performed this operation in two phases, initially by taking into account all normal events together with those related to a single class of intrusions, then by using all normal events together with those related to all classes of intrusions.

The experimental results are reported in Table 5 and Table 6, where, respectively, our approach has been compared to each single approach, then it has been compared to the canonical ensemble strategies. In such tables, the best average performances are highlighted in bold.

Premising that all the experiments have been made by following the *time series cross-validation* criterion described in Section 5.3, the analysis of the obtained results leads towards the following considerations:

- the results shown in Table 5 indicate that our *PDE* approach outperforms each single state-of-the-art approaches in terms of average performance (i.e., *TNR*, *F-score*, and *AUC*), except in the case of the *DSA* events, where it reaches the same performance of the best competitors;
- the results shows in Table 6 indicate that our *PDE* approach also outperforms all canonical ensemble strategies in terms of average performance, except in the case of the *DSA* events, where it reaches the same performance of the best competitors;
- the results shown in Table 7 indicate that such an improvement in terms of *TNR* (i.e., *Specificity*) does not lead to significant reduction of the classifier performance in terms of *F-score* and *AUC*, since although we have a moderate worsening in terms of average *F-score* (i.e., -0.01), we get an improvement in terms of average *AUC* (i.e., $+0.02$);
- considering that the adopted real-world dataset contains a large number of events related to intrusion events, as reported in Table 2, the obtained improvement allows a system to detect a significant number of attacks that otherwise would not have been detected;
- the last consideration should be made about the *False Negative Rate* (*FNR*) (i.e., the *Miss Rate*) related to our approach, in order to verify that the increase of the *True Negative Rate* (i.e., the *Specificity*) is not correlated with an equal increase of this value (i.e., the *FNR*): the experimental results report an average *FNR* of 0.007, whereas our average *TNR* is 0.056;
- the *TPR* and *FNR* rates indicate that a very low percentage of the new detected intrusions leads towards false alarms, also by considering that, prudentially, in the context taken into account the false negative classifications are preferable to the false positive ones.

6 CONCLUSIONS

The ensemble approach proposed in this paper has been designed to maximize the capability to detect intrusion events, independently from the operative scenario, exploiting several evaluation models in order to differentiate the normal events from those related to all classes of attacks.

The results show how the proposed ensemble approach outperforms the single classifiers in terms of *Specificity*, without being significantly penalized in other aspects, because it gets a slight deterioration in terms of average *F-score* but a positive average *AUC* (wrt the competitor approaches), demonstrating the effectiveness of our evaluation model.

Some ongoing efforts we are already focusing on are related to the employment of deep learning techni-

ques and frameworks (e.g. TensorFlow, Keras) on ad-hoc hardware (e.g. NVidia GPUs) in order to further improve the proposed approach.

ACKNOWLEDGEMENTS

This research is partially funded by *Regione Sardegna*, *Next generation Open Mobile Apps Development* (NOMAD) project, *Pacchetti Integrati di Agevolazione* (PIA) - *Industria Artigianato e Servizi* (2013).

REFERENCES

Amrita and Ravulakollu, K. K. (2018). A hybrid intrusion detection system: Integrating hybrid feature selection approach with heterogeneous ensemble of intelligent classifiers. *I. J. Network Security*, 20(1):41–55.

Bace, R. and Mell, P. (2001). Nist special publication on intrusion detection systems. Technical report, BOOZ-ALLEN AND HAMILTON INC MCLEAN VA.

Bijone, M. (2016). A survey on secure network: Intrusion detection & prevention approaches. *American Journal of Information Systems*, 4(3):69–88.

Bronte, R., Shahriar, H., and Haddad, H. M. (2016). A signature-based intrusion detection system for web applications based on genetic algorithm. In *Proceedings of the 9th International Conference on Security of Information and Networks, Newark, NJ, USA, July 20-22, 2016*, pages 32–39. ACM.

Buczak, A. L. and Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys and Tutorials*, 18(2):1153–1176.

Bukac, V., Tucek, P., and Deutsch, M. (2012). Advances and challenges in standalone host-based intrusion detection systems. In *TrustBus*, volume 7449 of *Lecture Notes in Computer Science*, pages 105–117. Springer.

Das, N. and Sarkar, T. (2014). Survey on host and network based intrusion detection system. *International Journal of Advanced Networking and Applications*, 6(2):2266.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *Multiple Classifier Systems*, volume 1857 of *Lecture Notes in Computer Science*, pages 1–15. Springer.

Garcia-Teodoro, P., Díaz-Verdejo, J. E., Maciá-Fernández, G., and Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *Computers & Security*, 28(1-2):18–28.

Ghorbani, A. A., Lu, W., and Tavallaee, M. (2010). *Network Intrusion Detection and Prevention - Concepts and Techniques*, volume 47 of *Advances in Information Security*. Springer.

Gomes, H. M., Barddal, J. P., Enembreck, F., and Bifet, A. (2017). A survey on ensemble learning for data stream classification. *ACM Comput. Surv.*, 50(2):23:1–23:36.

Guo, H., Li, Y., Shang, J., Mingyun, G., Yuanyue, H., and Bing, G. (2017). Learning from class-imbalanced data: Review of methods and applications. *Expert Syst. Appl.*, 73:220–239.

Holm, H. (2014). Signature based intrusion detection for zero-day attacks: (not) A closed chapter? In *47th Hawaii International Conference on System Sciences, HICSS 2014, Waikoloa, HI, USA, January 6-9, 2014*, pages 4895–4904. IEEE Computer Society.

Munaiah, N., Meneely, A., Wilson, R., and Short, B. (2016). Are intrusion detection studies evaluated consistently? a systematic literature review.

Pfleeger, C. P. and Pfleeger, S. L. (2012). *Security in Computing, 4th Edition*. Prentice Hall.

Potluri, S. and Diedrich, C. (2016). High performance intrusion detection and prevention systems: A survey. In *ECCWS2016-Proceedings fo the 15th European Conference on Cyber Warfare and Security*, page 260. Academic Conferences and publishing limited.

Sagi, O. and Rokach, L. (2018). Ensemble learning: A survey. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery*, 8(4).

Sekar, R., Gupta, A., Frullo, J., Shanbhag, T., Tiwari, A., Yang, H., and Zhou, S. (2002). Specification-based anomaly detection: a new approach for detecting network intrusions. In *ACM Conference on Computer and Communications Security*, pages 265–274. ACM.

Sommer, P. (1999). Intrusion detection systems as evidence. *Computer Networks*, 31(23-24):2477–2487.

Uppuluri, P. and Sekar, R. (2001). Experiences with specification-based intrusion detection. In *Recent Advances in Intrusion Detection*, volume 2212 of *Lecture Notes in Computer Science*, pages 172–189. Springer.

Viegas, E., Santin, A. O., and Oliveira, L. S. (2017). Toward a reliable anomaly-based intrusion detection in real-world environments. *Computer Networks*, 127:200–216.

Wang, Y., Yang, K., Jing, X., and Jin, H. L. (2014). Problems of kdd cup 99 dataset existed and data preprocessing. In *Applied Mechanics and Materials*, volume 667, pages 218–225. Trans Tech Publ.

Zainal, A., Maarof, M. A., Shamsuddin, S. M. H., and Abraham, A. (2008). Ensemble of one-class classifiers for network intrusion detection system. In *IAS*, pages 180–185. IEEE Computer Society.