A Discretized Extended Feature Space (DEFS) Model to Improve the Anomaly Detection Performance in Network Intrusion Detection Systems

Roberto Saia, Salvatore Carta, Diego Reforgiato Recupero, Gianni Fenu and Maria Madalina Stanciu
Department of Mathematics and Computer Science, University of Cagliari, Via Ospedale 72, 09124 Cagliari, Italy

Keywords: Machine Learning, Anomaly Detection, Pattern Recognition.

Abstract: The unbreakable bond that exists today between devices and network connections makes the security of the latter a crucial element for our society. For this reason, in recent decades we have witnessed an exponential growth in research efforts aimed at identifying increasingly efficient techniques able to tackle this type of problem, such as the Intrusion Detection System (IDS). If on the one hand an IDS plays a key role, since it is designed to classify the network events as normal or intrusion, on the other hand it has to face several well-known problems that reduce its effectiveness. The most important of them is the high number of false positives related to its inability to detect event patterns not occurred in the past (i.e. zero-day attacks). This paper introduces a novel Discretized Extended Feature Space (DEFS) model that presents a twofold advantage: first, through a discretization process it reduces the event patterns by grouping those similar in terms of feature values, reducing the issues related to the classification of unknown events; second, it balances such a discretization by extending the event patterns with a series of meta-information able to well characterize them. The approach has been evaluated by using a real-world dataset (NSL-KDD) and by adopting both the in-sample/out-of-sample and time series cross-validation strategies in order to avoid that the evaluation is biased by over-fitting. The experimental results show how the proposed DEFS model is able to improve the classification performance in the most challenging scenarios (unbalanced samples), with regard to the canonical state-of-the-art solutions.

1 INTRODUCTION

Network security is undoubtedly one of the most crucial aspects of today's society, in which an ever increasing number of services is performed through local networks or through the Internet. Considering that many of these services involve sensitive information such as, for instance, banking or health-care, it is justified the growing interest of the scientific community in seeking efficient solutions able to ensure the security of these services. The *Intrusion Detection Systems* (*IDSs*) (Buczak and Guven, 2016; Yeo et al., 2017) represent the main instruments to face such a kind of problem, since they have been designed in order to classify the network events into two classes, legitimate or fraudulent.

An *IDS* enables us to overcome the limitations of other common security solutions (Kizza, 2017), such as those that exploit a series of defined rules (e.g., firewalls (Saboori et al., 2012)), authentication mechanisms, or data encryption (Deng et al., 2003). They can adopt different techniques and strategies in order to detect and classify the network events and, in addition, they can protect a single host or a whole network.

A *Back Propagation Neural Network* has been adopted in (Sen et al., 2015) in order to detect anomalous network activities, whereas a *Fuzzy Logic* technique has been used in (Orfila et al., 2003), with the aim to improve the *IDS* performance by adopting fuzzy thresholds. The *IDS* performance had been improved in (Scherer et al., 2011) by introducing *Clustering Algorithms* and *Support Vector Machines* supervised models, whereas the *Genetic Algorithms* were used in (Li, 2004) in order to take into account the spatial and temporal information related to the network events. Various technologies have been also combined to define *hybrid solutions*, such as those introduced in (Wang et al., 2010), where the performance of an *IDS* has been improved by using both the *Fuzzy Clustering* and the *Artificial Neural Networks* techniques. In brief, an *IDS* detects and analyzes all network events, alerting when a network activity can represent a potential intrusion, allowing security managers to face these attacks through manual or automatic countermeasures.

Saia, R., Carta, S., Recupero, D., Fenu, G. and Stanciu, M.
A Discretized Extended Feature Space (DEFS) Model to Improve the Anomaly Detection Performance in Network Intrusion Detection Systems.
DOI: 10.5220/0008113603220329
In *Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2019)*, pages 322-329
ISBN: 978-989-758-382-7

Although the literature proposes a large number of approaches based on different techniques and strategies, there are still some problems that transversely affect these approaches. The main cause of these problems is the high degree of dynamism that characterizes the domain taken into consideration, as it involves a heterogeneous and high number of events, whose correct classification represents a complex problem. This scenario often makes obsolete the information used to define an evaluation model, making it ineffective against attacks not previously known (e.g., *zero-days* ones) and, in addition, many attacks have a behavior very similar to that of a normal network activity, so it is very hard to discriminate them.

The intuition behind the proposed *DEFS* model relies onto the observation that the most important limitation of the state-of-the-art intrusion detection approaches is their inability to correctly classify the new network events, since the involved data domain is characterized by a huge number of heterogeneous activities. On the basis of this consideration, we introduced a new feature space where the features that characterized each event have been transformed by following discretization and extension criteria.

The performed data discretization process leads towards a new feature space able to reduce the high number of event patterns given by the original continuous and discrete feature values, grouping the similar patterns and facing the problems related to the classification of unknown event patterns. After this operation, we extend the number of event features by introducing a series of meta-information with the aim to well characterize each event. Differently from the canonical approaches of data preprocessing adopted in the machine learning context (e.g., *featurization*, *random projection*, etc.), the proposed data model combines the discretization process with the introduction of several meta-information in order to reduce the issues related to the loss of information caused by them.

It should be observed that most works in the literature validate their intrusion detection approaches on the same data they used to tune the parameters of the employed algorithms. This is not effective as the procedure is biased by over-fitting. The problem is due to the need of a complete separation between the data used to tune the internal algorithms, training data and test data. For this reason, we preferred to evaluate our model by following a more effective evaluation strategy largely used in other fields (e.g. financial), performing the parameter tuning on a set, then performing the training step on a different set of data and, finally, running the evaluation on the test set, a third different set of data. Basically we evaluate the performance of our approach (training and test) on data never seen before (named *out-of-sample*), performing the parameter tuning by using different data (named *in-sample*). Within the *in-sample* and the *out-sample* we performed two canonical cross-validation. The former was needed to define the internal parameters (discretization range and classification algorithm to use) of our approach. We created our training model and performing the testing on the latter.

We preferred to evaluate our novel data model in relation to individual machine learning algorithms, rather than comparing it with more sophisticated configurations (e.g., ensemble approaches). The reason is that if we verify that such a data model applied on a single classification algorithm produces better performance, it means that it is also able to improve the performance in the context of more complex approaches that exploit it. Our scientific contributions are:

- formalization of a *Discretized Extended Feature Space* (*DEFS*) model aimed to face the data heterogeneity by discretizing and extending the original event feature space;
- exploitation of the *DEFS* model in the context of a machine learning classifier, selecting through an exhaustive evaluation of several state-of-the-art approaches;
- definition of an algorithm based on the *DEFS* model, which is able to classify each new network event as *normal* or *intrusion*.

2 BACKGROUND AND RELATED WORK

An *Intrusion Detection System* (*IDS*) represents a network security technology aimed to detect attacks against a target application or computer. It works by analyzing each network event in order to detect unauthorized access, misuse, and potential violations (from now on, simply referred to as *attacks*) (Ghorbani et al., 2010). Such attacks can derive from automated actions such as, for instance, a malware (i.e. virus), or from a sequence of manual actions executed by a human attacker.

An *IDS* is aimed to capture, analyze, and classify all the events related to a machine/network. It is usually performed according to a binary criterion, where such events are divided into two classes, *normal* and *intrusion*. Literature works have used many metrics and real-world datasets in order to evaluate the performance of such systems (Munaiah et al., 2016), from those based on the *confusion* *matrix*[1], such as the *Accuracy*, the *Sensitivity*, the *Specificity*, and the *F-measure*, to more specific metrics able to evaluate the prediction model effectiveness, such as the *Matthews Correlation Coefficient* (*MCC*). The MCC presents some advantages with respect to other metrics based on the *confusion matrix*, since it takes into consideration the balance ratios of all the four classes of information in such a matrix (i.e., *true positives*, *true negatives*, *false positives*, and *false negatives*) (Chicco, 2017).

The main problem that reduces the effectiveness of an *IDS* is the domain where it operates, since it involves data that are characterized by a high level of heterogeneity and dynamism, making it difficult to define an effective evaluation model. In addition, the behavior of an attack can be almost identical to that of a legitimate activity, making it very difficult to identify.

3 FORMAL NOTATION

The formal notation adopted in this paper is shown in Table 1.

Table 1: Formal Notation.

Notation	Description	Note
$E = \{e_1, e_2, \ldots, e_X\}$	*Set of classified events*	
$E^+ = \{e_1^+, e_2^+, \ldots, e_Y^+\}$	*Subset of normal events*	$E^+ \subseteq E$
$E^- = \{e_1^-, e_2^-, \ldots, e_W^-\}$	*Subset of intrusion events*	$E^- \subseteq E$
$\hat{E} = \{\hat{e}_1, \hat{e}_2, \ldots, \hat{e}_Z\}$	*Set of unclassified events*	
$F = \{f_1, f_2, \ldots, f_N\}$	*Set of event features*	
$C = \{normal, intrusion\}$	*Set of event classifications*	

4 APPROACH DEFINITION

This section describes the implementation of our *DEFS* model, from its formal definition to its exploitation in the context of a classification algorithm.

4.1 Model Formalization

According to the formal notation introduced in Section 3, given the set $E = \{e_1, e_2, \ldots, e_X\}$ of classified events and the set $\hat{E} = \{\hat{e}_1, \hat{e}_2, \ldots, \hat{e}_Z\}$ of unclassified events, where each event is characterized by a series of $F = \{f_1, f_2, \ldots, f_N\}$ features, in the following we formalize the data *discretization* and *extension* processes we applied on the event feature space.

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

4.1.1 Data Discretization

The discretization process is applied on each feature in F and it is aimed to reduce the original (continuous and discrete) range of values by mapping them into a defined range of discrete values $\{0, 1, \ldots, \eta\} \in \mathbb{Z}$, where the value of η is defined through the tuning process previously described, which involves only the in-sample datasets in order to avoid the result from being influenced by over-fitting.

On the basis of the defined η value, we perform a data discretization process that we denote as $f \xrightarrow{\eta} d$. It converts each feature element $f \in F$ in a new discrete value taken from a range of integers $\{d_1, d_2, \ldots, d_\eta\}$. Such a process is aimed to reduce the high heterogeneity that characterizes the data domain by performing a kind of average between similar event patterns in terms of feature values. In short words, it transforms each feature value f in a discrete value d and this process is performed on both the datasets E and $\hat{E}$, before the data extension process described in Section 4.1.2, as formalized in Equation 1.

$$\begin{array}{l} \{f_1, f_2, \ldots, f_N\} \xrightarrow{\eta} \{d_1, d_2, \ldots, d_N\}, \ \forall \ e \in E \\ \{f_1, f_2, \ldots, f_N\} \xrightarrow{\eta} \{d_1, d_2, \ldots, d_N\}, \ \forall \ \hat{e} \in \hat{E} \end{array} \tag{1}$$

4.1.2 Data Extension

By using as data source the discretized datasets previously obtained, we further extend with the goal of integrating the new discretized feature space with several meta-information Ξ. In more detail, we integrate the event feature space with the values related to the *Minimum* (μ), *Maximum* (Λ), *Average* (α), and *Standard Deviation* (σ) information measured in such a space. It should be observed how these information are able to improve the characterization of each event, even when they refer to already discrete range of values or non-ordered values.

The objective of this operation is to combine the advantage related to the reduction of the number of event patterns, performed through the discretization process, with the advantages in terms of event characterization given by the meta-information Ξ formalized in Equation 2.

$$\Xi = \begin{cases} \mu = min(d_1, d_2, \ldots, d_N) \\ \Lambda = max(d_1, d_2, \ldots, d_N) \\ \alpha = \frac{1}{N} \sum_{n=1}^{N} (d_n) \\ \sigma = \sqrt{\frac{1}{N-1} \sum_{n=1}^{M} (d_n - \bar{d})^2} \end{cases} \tag{2}$$

The introduction of the Ξ mathematical indicators relies on the consideration that the literature presents several intrusion detection approaches enhanced by the adoption of metrics such as *minimum*, *maximum*, *average*, and *standard deviation* (Ahmed and Mohamed, 2018; Marino et al., 2018).

4.1.3 Transformed Feature Space

As a result of the *data discretization* and *data extension* processes, the original feature space F has been transformed in a new one, where the feature values $f \in F$ of the datasets E or $\hat{E}$ (i.e., respectively, related to the classified and unclassified events) are discretized in an integer range of η values, defined by following the tuning process previously described. In addition, this first transformation has been followed by a process aimed to integrate such a discretized feature space, with the additional meta-information Ξ reported in Equation 2. The result is the transformed feature space shown in Equation 3, which lies underneath our *DEFS* model (for exemplification reasons it is only applied to the dataset E, but it is identical for the dataset $\hat{E}$).

$$DEFS(E) = \begin{pmatrix} d_{1,1} & d_{1,2} & \dots & d_{1,N} & \mu_{1,N+1} & \Lambda_{1,N+2} & \alpha_{1,N+3} & \sigma_{1,N+4} \\ d_{2,1} & d_{2,2} & \dots & d_{2,N} & \mu_{2,N+1} & \Lambda_{2,N+2} & \alpha_{2,N+3} & \sigma_{2,N+4} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ d_{X,1} & d_{X,2} & \dots & d_{X,N} & \mu_{X,N+1} & \Lambda_{X,N+2} & \alpha_{X,N+3} & \sigma_{X,N+4} \end{pmatrix} \quad (3)$$

4.2 DEFS Classification Algorithm

The $DEFS$ model formalized in Section 4.1 is then adopted in the Algorithm 1 in order to classify all the new events in the set $\hat{E}$.

Algorithm 1: New events classification by DEFS model.

```
Input: a=Classifier, E=Evaluated events, Ê=Unevaluated events
Output: Set of classified Ê events
procedure GETCLASSIFICATION(a, E, ê)
    D(E) ← dataDiscretization(E)
    D(Ê) ← dataDiscretization(Ê)
    D(E) ← addMetafeatures(D(E))
    D(Ê) ← addMetafeatures(D(Ê))
    model ← trainModels(a, D(E))
    for each d(ê) ∈ D(Ê) do
        c ← getClassification(model, d(ê))
        output.add(c)
    end for
    return output
end procedure
```

5 EXPERIMENTS

This section provides information about the used development environment during the experiments, the dataset taken into account, the evaluation metrics, the adopted experimental strategy, concluding by presenting and discussing the obtained results.

5.1 Environment

The development environment is based on the *Python* language, whose *scikit-learn* [2] library has been used in order to implement the state-of-the-art algorithms. It should be noted that we have set to *1* the seed of the *scikit-learn* pseudo-random number generator (i.e., the *random_state* parameter) in order to allow the reproducibility of all the performed experiments.

5.2 Dataset

We have chosen *NSL-KDD*[3] as real-world dataset to use in order to evaluate the performance of the proposed approach, since it represents a fixed and improved version of the *KDD-CUP99* (Tavallaee et al., 2009) dataset, which has been widely used in literature.

Details about this dataset are reported in Table 2, where it should be observed that the number of attack classes in the training set is not equal to that in the test set, since a certain attack might not be present in both the datasets. Information in terms of total number of events, number of *normal* and *intrusion* events, and values that characterize each event, have been reported according to the formal notation provided in Section 3.

Table 2: Dataset Composition.

Dataset	Total events $\lvert E\rvert$	Normal events $\lvert E_+\rvert$	Intrusion events $\lvert E_-\rvert$	Event values $\lvert V\rvert$	Classes of attacks
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		

The different types of events are grouped into four categories: *Privilege Escalation Attack* (*PEA*), *Denial of Service Attack* (*DSA*), *Remote Scanning Attack* (*RSA*), and *Normal Network Activity* (*NNA*). The *PEA* category contains all the attacks characterized by a *Privilege Escalation* (*PE*) activity, which is aimed to get a privileged access to one or more resources (e.g., *Buffer_overflow*, *Loadmodule*, *Rootkit*, *Perl*, *Sqlattack*, *Xterm*, and *Ps* attacks). The *DSA* category contains all the attacks characterized by a *Denial of Service* (*DoS*) activity, which is aimed to make a certain service/resource unusable by using a high number of legitimate requests (e.g., *Back*, *Land*, *Neptune*, *Pod*, *Smurf*, *Teardrop*, *Mailbomb*, *Processtable*, *Udpstorm*, *Apache2*, and *Worm* attacks). The *RSA* category groups instead all the attacks aimed to collect sensitive information about services/systems, performing this operation by adopting all possible non-invasive and invasive techniques (e.g., *Satan*, *IPsweep*, *Nmap*,

[2] http://scikit-learn.org

[3] https://github.com/defcom17/NSL_KDD

Portsweep, *Mscan*, and *Saint* attacks). The *RAA* category contains all the attacks focused on getting access to remote systems/services, leveraging simple techniques (e.g., *Guess_password*, *Ftp_write*, *Imap*, *Phf*, *Multihop*, *Warezmaster*, *Xlock*, *Xsnoop*, *Snmpguess*, *Snmpgetattack*, *Httptunnel*, *Sendmail*, and *Named* attacks). The last *NNA* category simply contains all the events related to the normal network activity.

The numeric relevance of each class of attacks is reported in Table 3.

Table 3: Classes of Attacks Numeric Relevance.

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
Percentage	0.08	35.95	9.48	2.61	51.88

5.3 Experimental Strategy

We compared our approach to different and highly performing state-of-the-art algorithms usually adopted in this domain, i.e., *Gradient Boosting* (*GBC*), *Adaptive Boosting* (*ADA*), *Random Forests*, *Multilayer Perceptron* (*MLP*), and *Decision Tree* (*DTC*).

The *NSL-KDD* dataset has been preprocessed in order to convert all the categorical features into numerical. It should be noted that the last feature of each dataset row contains the event classification, according to the notation *1=normal* and *2=intrusion*. We have also converted such a notation in the standard binary form *0=normal* and *1=intrusion*.

As introduced in Section 1, in order to evaluate the real effectiveness of the proposed approach with respect to the state-of-the-art solutions, we conducted the experiments by adopting an approach based on two different datasets, the first one (*in-sample*) used to select the internal algorithms parameters and the second one (*out-of-sample*) to evaluate the performance (training and testing). Such a strategy, largely used in many critical domains (Cleary and Hebb, 2016; Fenu and Surcis, 2009) in order to asses the real performance of a proposed classification approach/strategy, allows us to avoid that the prediction model is biased by over-fitting. Basically we perform the algorithms tuning on a different dataset (*in-sample*) than the one we adopted to build the model and perform the classification task (*out-of-sample*). In more detail, we used the 80% of each dataset (i.e., *PEA*, *DSA*, *RSA*, and *RAA*) as *in-sample* data, using the remaining 20% as *out-of-sample* data.

After the definition of the *in-sample* and *out-of-sample* datasets, in order to further limit the impact of the data dependency in the experiments, we used in each of them the *TimeSeriesSplit scikit-learn* function to perform the data validation process, since a canonical *k-fold cross-validation* strategy is not suitable when *time series* data are involved, as it does not respect the event chronology. The adopted *time series cross-validation* strategy is instead able to divide the data into *n_splits* training and test sets, tacking into account the chronology aspect (we used *n_splits=5*).

As anticipated earlier in Section 2, in order to get an effective evaluation of the performance reached by our approach, we evaluated our results on the basis of several metrics. The first metric is the *Specificity* (i.e., *true negative rate*), a metric based on the *confusion matrix* that provides a measure of the correctness of the performed classifications in terms of correct intrusion events detected respect to the normal ones. Another metric taken into account is the *Matthews Correlation Coefficient*, a metric largely used to evaluate the overall effectiveness of a classification model, in terms of its capability to correctly recognize the *normal* and *intrusion* events .

We also evaluate the *Dataset Class Imbalance* (DCI) as formalized in Equation 4, where 1 indicates a perfect data balance.

$$DCI(E^+,E^-) = \begin{cases} \frac{min(|E^+|,|E^-|)}{max(|E^+|,|E^-|)}, \; if\; (|E^+| \neq |E^-|) \wedge (max(E^+,E^-) > 0) \\ 0,\; if\; max(E^+,E^-) = 0 \\ 1,\; Otherwise \end{cases} \tag{4}$$

5.4 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 between *intrusion* and *normal* events. 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 first experimental step is the evaluation of the imbalance level that characterizes each involved dataset, since this information allows us to asses the real effectiveness of an evaluation model. Table 4 reports the *DCI* values related to all the datasets and it shows that three of them (i.e., *PEA*, *RSA*, and *RAA*) present a high level of imbalance, with regard to *DSA*, where exists a quite balance between the two classes of information (i.e., *normal* and *intrusion*).

Table 4: Datasets Class Imbalance Level.

Events	PEA	DSA	RSA	RAA
Normal	77,053	77,053	77,053	77,053
Intrusion	119	53,387	14,077	3,880
DCI	0.0015	0.6928	0.1826	0.0503

The experimental results related to the evaluation of the competitor algorithms are shown in Table 5, where the best obtained performance has been highlighted in bold. The evaluation has been performed by using the *in-sample* datasets with and without a preliminary data oversampling[4] process aimed to balance the two classes of samples (i.e., *normal* and *intrusion*). In order to perform this data oversampling we used a *Python* implementation of the *Synthetic Minority Over-sampling Technique* (SMOTE) (Chawla et al., 2002).

Premising that all the experiments that used the *in-sample* and *out-of-sample* datasets have been performed by following the *time series cross-validation* criterion described in Section 5.3, a first analysis of the obtained results indicates how some algorithms (e.g., *Random Forests*) perform better than the other ones in terms of *Specificity*, whereas other algorithms (e.g., *Adaptive Boosting*) perform better than other ones in terms of *MCC*.

Considering that an optimal algorithm of classification should balance the capability to detect the intrusion events (*Specificity*) with the capability to perform this operation without a significant increasing of the false negative (aspect reported by the *MCC* metric), in order to select the best algorithm we have taken into account the average value of these two metrics.

Table 5: Algorithms In-sample Performance.

		Normal			Oversampling		
Algorithm	**Events**	**Specificity**	**MCC**	**Average**	**Specificity**	**MCC**	**Average**
Gradient Boosting	PEA	0.4019	0.4077	0.4048	0.5583	**0.6584**	0.6083
Adaptive Boosting	PEA	0.7629	0.5972	0.6801	0.5319	0.5259	0.5289
Random Forests	PEA	**0.9600**	**0.5984**	**0.7792**	**0.8467**	0.6227	**0.7347**
Multilayer Perceptron	PEA	0.3000	0.2079	0.2540	0.0455	0.1803	0.1129
Decision Tree	PEA	0.4794	0.5086	0.4940	0.4627	0.5257	0.4942
Gradient Boosting	DSA	0.9995	0.9990	0.9992	0.9993	0.9990	0.9991
Adaptive Boosting	DSA	0.9991	0.9986	0.9989	0.9989	0.9985	0.9987
Random Forests	DSA	**0.9997**	**0.9993**	**0.9995**	**0.9998**	**0.9992**	**0.9995**
Multilayer Perceptron	DSA	0.9818	0.9596	0.9707	0.9712	0.9589	0.9650
Decision Tree	DSA	0.9995	0.9990	0.9992	0.9993	0.9990	0.9991
Gradient Boosting	RSA	0.9964	0.9937	0.9950	0.9934	0.9926	0.9930
Adaptive Boosting	RSA	0.9925	0.9881	0.9903	0.9823	0.9848	0.9835
Random Forests	RSA	**0.9985**	**0.9956**	**0.9970**	**0.9981**	**0.9958**	**0.9970**
Multilayer Perceptron	RSA	0.8364	0.8601	0.8482	0.8016	0.8432	0.8224
Decision Tree	RSA	0.9917	0.9913	0.9915	0.9897	0.9898	0.9898
Gradient Boosting	RAA	0.9625	0.9427	0.9526	0.7892	0.8739	0.8316
Adaptive Boosting	RAA	0.9605	0.9337	0.9471	0.6733	0.8008	0.7370
Random Forests	RAA	**0.9895**	**0.9603**	**0.9749**	**0.9718**	**0.9511**	**0.9614**
Multilayer Perceptron	RAA	0.5749	0.2649	0.4199	0.2517	0.4465	0.3491
Decision Tree	RAA	0.9363	0.9387	0.9375	0.9074	0.9202	0.9138

We can observe that such a criterion does not allow us to detect the algorithm with the best overall performance, then we also calculated the average performance related to all the event scenarios, with and without oversampling preprocessing. In addition, we detailed this analysis by using two groups of events, those characterized by a low *DCI* value (low level of unbalance), and those characterized by an high *DCI* value (high level of unbalance), calculating their *DCI* average value, according to the information provided in Table 4. The results are shown in Table 6 and they indicate *Random Forests* as the best performing algorithm in both the aforementioned scenarios (i.e., high and low *DCI* value) and, in addition, such performances are reached without a preliminary data oversampling.

According to the *DEFS* model formalization provided in Section 4.1, this step is aimed to detect the optimal range to use for the discretization of the datasets feature values. Analogously to what was done for the choice of the best competitor algorithm, in order to avoid that such a process is biased by overfitting, we perform it in the context of the *in-sample* datasets. We obtain *430* as best value to use as discrete integer range of the *DEFS* model.

Table 6: Algorithms In-sample Mean Performance.

			Performance Average	
Algorithm	**Events**	$\overline{\textbf{DCI}}$	**Normal**	**Oversampling**
Gradient Boosting	PEA, DSA, RSA, RAA	0.2318	0.8379	0.8580
Adaptive Boosting	PEA, DSA, RSA, RAA	0.2318	0.9041	0.8121
Random Forests	PEA, DSA, RSA, RAA	0.2318	**0.9377**	**0.9232**
Multilayer Perceptron	PEA, DSA, RSA, RAA	0.2318	0.6225	0.5762
Decision Tree	PEA, DSA, RSA, RAA	0.2318	0.8556	0.8492
Gradient Boosting	PEA, RSA, RAA	0.0781	0.7842	0.8110
Adaptive Boosting	PEA, RSA, RAA	0.0781	0.8725	0.7498
Random Forests	PEA, RSA, RAA	0.0781	**0.9171**	**0.8977**
Multilayer Perceptron	PEA, RSA, RAA	0.0781	0.5064	0.4465
Decision Tree	PEA, RSA, RAA	0.0781	0.8077	0.7993

Out-of-sample Performance Comparison: On the basis of the results of the previous experiments, which indicate *Random Forests* (*RF*) without a preliminary data oversampling as the best performing algorithm in the *in-sample* datasets, the next series of experiments are aimed to evaluate its performance in the context of the *out-of-sample* datasets, measuring them before and after the introduction of our *DEFS* model. It means that the selection process we performed in order to choose our best competitor (i.e., *Random Forests*) has been made by using different data (*in-sample* datasets) respect to those used to evaluate its performance (*out-of-sample* datasets). This because we want that both *Random Forests* based on the standard feature space model and *Random Forests* based on our *DEFS* model are evaluated on data never seen before. The results are shown in Table 7, where the values highlighted and in bold indicate when our *DEFS* model overcomes the canonical one.

Table 7: Out-of-sample Performance Comparison

		RF Default Model			RF DEFS Model		
Events	**DCI**	**Specificity**	**MCC**	**Average**	**Specificity**	**MCC**	**Average**
PEA	0.0015	0.5618	0.4077	0.4848	**0.7833**	**0.4835**	**0.6334**
DSA	0.6928	0.9957	0.9585	0.9771	0.9919	**0.9629**	**0.9774**
RSA	0.1826	0.9458	0.9102	0.9280	**0.9626**	**0.9175**	**0.9401**
RAA	0.0503	0.9704	0.7961	0.8833	**0.9714**	0.7891	0.8802

Discussion: The analysis of the *in-sample* and *out-of-sample* results leads towards the following considerations:

[4]A technique that increases the minority samples by generating other ones, synthetically.

- the preliminary analysis of the datasets imbalance level underlines that there are some classes of intrusion events that represent a high level of data unbalance, with respect to the normal cases. This generates a reduction of the effectiveness of the canonical classification approaches based on an evaluation model trained by using the previous known events collected by an *IDS* (Brown and Mues, 2012; Japkowicz and Stephen, 2002). Considering that also a slight performance increment in terms of intrusion event detection (i.e., *Specificity*) represents an important improvement, when it is not directly related to an equal reduction of the evaluation model event discrimination capability (i.e., *MCC*), we attribute more importance to this (unbalanced) scenario with respect to the other (balanced);
- according to the literature, the experiments indicate *Random Forests* as one of the best performing algorithm (Resende and Drummond, 2018) in this domain, as shown by the single and average performance measured in the context of the *in-sample* datasets, reported, respectively, in Table 5 and Table 6. On the basis of this consideration, we used this algorithm in order to experiment our *DEFS* model, considering that an improvement in its excellent classification performance would represent a good result able to demonstrate the effectiveness of the proposed model;
- the experimental results related to the *out-of-sample* performance shown in Table 7 indicate how the adoption of our *DEFS* model improves the intrusion detection performance in all the datasets, especially in those characterized by a high level of data unbalance (i.e., *PEA*, *RSA*, and *RAA*, with a *DCI* value of, respectively, 0.0015, 0.1826, and 0.0503);
- we can observe how the performance improves with the increase of data unbalance level, indicating the effectiveness of our *DEFS* model in terms of characterization of the network events, allowing an intrusion detection system to discriminate the intrusion events even when these are in much smaller numbers than the normal ones (e.g., as it happens in the *PEA* dataset, where the intrusion events are *119* and normal events are *77,053*);
- in the few cases where our model does not outperform the canonical one (i.e., *DSA Specificity* and *RAA MCC*), its performance are very close to those of the state-of-the-art competitor, since in the former case we have a difference of -0.0038 in terms of *Specificity* although we get better average performance, whereas in the latter case we have a difference of -0.0070 in terms of *MCC* but a *Specificity* improvement of $+0.0010$;
- summarizing, the proposed model proved to be able to improve the event characterization, especially in those cases where there are not enough samples in a class of events to train a reliable evaluation model, allowing us to define an intrusion detection system that operates in all the possible scenarios, also by recurring to hybrid strategies that combine the canonical data models with the proposed one.

6 CONCLUSIONS AND FUTURE WORK

Nowadays, *Intrusion Detection Systems* play an important role, since the exponential increase in services offered through computer networks is jeopardized by an equally exponential growth of the number of attempts to exploit them fraudulently. These real-world scenarios involve various classes of attacks that frequently need targeted approaches that do not work well in a canonic heterogeneous scenario, where different classes of attacks are present.

The *Discretized Extended Feature Space* model proposed in this paper has been designed in order to reduce the issues related to the classification of unknown and heterogeneous events by adopting a data discretization process and, at the same time, to better characterize each event by introducing a series of meta-information.

The results show how such a novel model outperforms the best competitor in terms of capability to detect the intrusion in a context of datasets with a high level of unbalance, in terms of *Specificity* (in all the cases) and *MCC* (in two datasets out of three), demonstrating its potential effectiveness in a real-world scenario. It should be observed that our approach can be parallelized by using *GPUs* or big data frameworks such as *Apache Spark* [5] in order to reduce the computational time.

A possible future work would be the experimentation of our model in the context of a classification approach based on several and different algorithms configured through an ensemble strategy (Saia et al., 2018), in order to improve the overall performance, as well as its evaluation in other data domains, such as those related to the *Fraud Detection* (Carta et al., 2019) and *Credit Scoring* (Saia and Carta, 2016) fields.

[5]https://spark.apache.org

ACKNOWLEDGEMENTS

This research is partially funded by Italian Ministry of Education, University and Research - Program Smart Cities and Communities and Social Innovation project ILEARNTV (D.D. n.1937 del 05.06.2014, CUP F74G14000200008 F19G14000910008). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.

REFERENCES

Ahmed, M. A. and Mohamed, Y. A. (2018). Enhancing intrusion detection using statistical functions. In *2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*, pages 1–6. IEEE.

Brown, I. and Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Syst. Appl.*, 39(3):3446–3453.

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.

Carta, S., Fenu, G., Recupero, D. R., and Saia, R. (2019). Fraud detection for e-commerce transactions by employing a prudential multiple consensus model. *Journal of Information Security and Applications*, 46:13–22.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

Chicco, D. (2017). Ten quick tips for machine learning in computational biology. *BioData mining*, 10(1):35.

Cleary, S. and Hebb, G. (2016). An efficient and functional model for predicting bank distress: In and out of sample evidence. *Journal of Banking & Finance*, 64:101–111.

Deng, H., Zeng, Q.-A., and Agrawal, D. P. (2003). Svm-based intrusion detection system for wireless ad hoc networks. In *2003 IEEE 58th Vehicular Technology Conference. VTC 2003-Fall (IEEE Cat. No. 03CH37484)*, volume 3, pages 2147–2151. IEEE.

Fenu, G. and Surcis, S. (2009). A cloud computing based real time financial system. In *2009 Eighth International Conference on Networks*, pages 374–379. IEEE.

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.

Japkowicz, N. and Stephen, S. (2002). The class imbalance problem: A systematic study. *Intell. Data Anal.*, 6(5):429–449.

Kizza, J. M. (2017). *Guide to Computer Network Security, 4th Edition*. Computer Communications and Networks. Springer.

Li, W. (2004). Using genetic algorithm for network intrusion detection. *Proceedings of the United States Department of Energy Cyber Security Group*, 1:1–8.

Marino, D. L., Wickramasinghe, C. S., and Manic, M. (2018). An adversarial approach for explainable ai in intrusion detection systems. In *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society*, pages 3237–3243. IEEE.

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

Orfila, A., Carbó, J., and Ribagorda, A. (2003). Fuzzy logic on decision model for IDS. In *The 12th IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 2003, St. Louis, Missouri, USA, 25-28 May 2003*, pages 1237–1242. IEEE.

Resende, P. A. A. and Drummond, A. C. (2018). A survey of random forest based methods for intrusion detection systems. *ACM Computing Surveys (CSUR)*, 51(3):48.

Saboori, E., Parsazad, S., and Sanatkhani, Y. (2012). Automatic firewall rules generator for anomaly detection systems with apriori algorithm. *CoRR*, abs/1209.0852.

Saia, R. and Carta, S. (2016). A linear-dependence-based approach to design proactive credit scoring models. In *KDIR*, pages 111–120.

Saia, R., Carta, S., and Recupero, D. R. (2018). A probabilistic-driven ensemble approach to perform event classification in intrusion detection system. In *KDIR*, pages 139–146. SciTePress.

Scherer, P., Vicher, M., Drázdilová, P., Martinovic, J., Dvorskỳ, J., and Snášel, V. (2011). Using svm and clustering algorithms in ids systems. In *Proc. Int Conf. Dateso 2011, 2011*.

Sen, R., Chattopadhyay, M., and Sen, N. (2015). An efficient approach to develop an intrusion detection system based on multi layer backpropagation neural network algorithm: Ids using bpnn algorithm. In *Proceedings of the 2015 ACM SIGMIS Conference on Computers and People Research*, pages 105–108. ACM.

Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A. A. (2009). A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA 2009, Ottawa, Canada, July 8-10, 2009*, pages 1–6. IEEE.

Wang, G., Hao, J., Ma, J., and Huang, L. (2010). A new approach to intrusion detection using artificial neural networks and fuzzy clustering. *Expert Syst. Appl.*, 37(9):6225–6232.

Yeo, L. H., Che, X., and Lakkaraju, S. (2017). Modern intrusion detection systems. *CoRR*, abs/1708.07174.