Feature selection by multi-objective optimization: application to network anomaly detection by hierarchical self-organizing maps

Emiro de la Hoz[a], Eduardo de la Hoz[a], Andrés Ortiz[b,*], Julio Ortega[c], Antonio Martínez-Álvarez[d]

[a] *Programa de Ingeniería de Sistemas*
Universidad de la Costa. Barranquilla, Colombia
[b] *Communications Engineering Department. University of Málaga. Málaga. Spain*
[c] *Computer Architecture and Technology Department. CITIC. University of Granada.*
[d] *Computer Technology Department, University of Alicante, Alicante, Spain*

Abstract

Feature selection is an important and active issue in clustering and classification problems. By choosing an adequate feature subset, a dataset dimensionality reduction is allowed, thus contributing to decreasing the classification computational complexity, and to improving the classifier performance by avoiding redundant or irrelevant features. Although feature selection can be formally defined as an optimisation problem with only one objective, that is, the classification accuracy obtained by using the selected feature subset, in recent years, some multi-objective approaches to this problem have been proposed. These either select features that not only improve the classification accuracy, but also the generalisation capability in case of supervised classifiers, or counterbalance the bias toward lower or higher numbers of features that present some methods used to validate the clustering/classification in case of unsupervised classifiers.

The main contribution of this paper is a multi-objective approach for feature selection and its application to an unsupervised clustering procedure based on Growing Hierarchical Self-Organizing Maps (GHSOM) that includes a new method for unit labelling and efficient determination of the winning unit. In the network anomaly detection problem here considered, this multi-objective approach makes it possible not only to differentiate between normal and anomalous traffic but also among different anomalies. The efficiency of our proposals has been evaluated by using the well-known DARPA/NSL-KDD datasets that contain extracted features and labeled attacks from around 2 million connections. The selected feature sets computed in our experiments provide detection rates up to 99.8% with normal traffic and up to 99.6% with anomalous traffic, as well as accuracy values up to 99.12%.

Keywords:
Feature selection, Multi-objective optimization, Unsupervised clustering, Growing self-organizing maps, Network anomaly detection, IDS

*Corresponding author
Email addresses: edelahoz@cuc.edu.co (Emiro de la Hoz), edelahoz6@cuc.edu.co (Eduardo de la Hoz), aortiz@ic.uma.es (Andrés Ortiz), jortega@ugr.es (Julio Ortega), amartinez@dtic.ua.es (Antonio Martínez-Álvarez)

1. Introduction

Frequently, existing classification problem features are not discriminative enough. Moreover, the use of correct features improves classification performance and reduces computational time. Thus, feature extraction and selection are two important classification problem issues that aim to obtain a subset of features in a lower dimensional space. They provide different advantages:

1. Representing the data in a lower dimensional space avoids the *curse of dimensionality* [1, 2]:
   - Diminishes the number of examples needed to train a classifier. The number of train samples grows exponentially with the data dimensionality.
   - Avoids overfitting and improves the classifiers' generalisation performance. In practice, there is an optimum number of features for maximum classification performance.
2. High informative features will represent the different class samples far away in the feature space, while similar samples will be represented close to each other.
3. The use of fewer features improves computational efficiency.
4. Data visualisation is easier and more intuitive in lower dimensional spaces (for example, 2D or 3D).

However, feature extraction and feature selection are different processes, although both can be used to obtain a discriminative subset. Thus, we will describe both separately.

Feature extraction can be stated as follows: Let $x \in \mathbb{R}^n$ be the existing feature set. The goal is to make existing features more descriptive through a mapping function $g(x) : \mathbb{R}^n \longrightarrow \mathbb{R}^m$ in such a way that $\widehat{x} = g(x)$ preserves the information and structure of data in $\mathbb{R}^n$. Thus, in general, $g(x)$ may implement a non-linear mapping. Frequently, linear transformations through a matrix $\mathbb{H}$, are usually applied to revise the initial feature set x $\widehat{x} = \mathbb{H}^T x$. In this case, $\widehat{x}$ is the representation of x in the subspace spanned by the basis vectors in $\mathbb{H}$ [3].
A representative feature extraction example through linear mapping is Principal Component Analysis (PCA). PCA generates an orthonormal basis vector indicating the maximum variance directions. Thus, projecting onto this basis maximises the sample scatter. The data samples projected onto the low dimensional space spanned by the central components are used in the classification task. Another popular feature extraction technique that uses a classification criterion instead of the representation error (as in PCA), is Linear Discriminant Analysis (LDA) [1]. In this case, the samples may not be accurately represented by the projected features (that is, reconstruction error is not minimised), but class discriminative information is enhanced. PCA and LDA have been used in classical problems, such as facial recognition, *eigenfaces* [4] and *fisherfaces* [5], respectively. Other techniques such as *Independent Component Analysis* (ICA), [6] aim to find a linear representation of non-Gaussian data in such a way that the components are as statistically independent as possible.

Unlike extraction, selection does not transform the existing features, but only searches for the most informative subset. Feature selection algorithms are classified into two categories: *filters*, and *wrappers*. *Filters* do not use a classifier and evaluate the features according to heuristics that accommodate different data characteristics. Thus, features are ranked according to their importance for separating classes using either statistical methods, information theory-based methods or searching techniques. Statistical methods include hypothesis testing, such as the Student's *t-test* [7, 1]. Other statistical methods, such as the Fisher Discriminant Ratio, can be used to quantify the discriminative power of individual features between two equiprobable classes [1]. Information theory-based methods can use different metrics, such as Entropy, Kullback-Leibler divergence [1] or the information gain measure [8] to rank the features. Moreover, [9, 10] use the Conditional Mutual Information (CMI) as the criterion for selecting feature subsets. Other *filter* algorithms use a correlation-based metric to evaluate feature usefulness. Specifically, the Correlation-Based Feature Selection (CFS) algorithm [11, 12] takes into account individual feature worth based on the hypothesis that *good feature subsets contain features highly correlated with the class, yet uncorrelated with each other* [11].

Nevertheless, most *filter* methods evaluate feature usefulness for predicting class labels by computing an average score on the different dataset classes. This may lead to removing features from the final selection that could be specially relevant for a certain class label. Thus, it is necessary to evaluate the discriminative power of each feature, selecting those that best describe each individual class. This is especially useful for imbalanced datasets or data with different statistical class distribution. In order to overcome this limitation, [13] proposed a framework focused on evaluating feature relevance and redundancy to a certain class label.

Unlike *filters*, *wrappers* use an objective function that returns the current feature selection goodness. This feedback is obtained from the classifier outcome (that is, classification accuracy or classification error) executed on the training set. However, these approaches are classifier-dependent, and require executing the training process in each iteration. Different searching strategies can be used depending on the way the features are selected or discarded in each iteration. However, their common goal is to keep the best feature combination (that is, features that optimise the objective function). In this way, there are two main searching strategies:

1. Suboptimal searching. These techniques aim to avoid trying all the feature combinations. Sequential Forward Selection (SFS) and Sequential Backward Selection (SBS) are well-known suboptimal searching methods [1].
2. Exhaustive searching. All possible feature combinations will be used to train the classifier and to assess performance. It is computationally expensive and may be unfeasible for high-dimensionality feature spaces or large datasets.

In *wrapper* feature selection approaches, evaluating a given set's utility presents different issues depending on the type, supervised or unsupervised, of the classification procedure used. If the procedure is supervised, it is relatively easy to define the utility cost function by using the classification error. Nevertheless, in unsupervised procedures, the utility should be determined from a clustering quality definition without having knowledge about the corresponding labels or even the

number of clusters. Frequently, the clustering quality measures use ratios between intra-cluster compactness measures and inter-cluster separation ones. Nevertheless, the distances between points tend to be similar values as the dimensions are higher, making these quality solutions biased toward lower dimension solutions [2]. This way, although, as is indicated in [2], formulating feature selection as a multi-objective optimisation problem could provide some advantages, results would depend on whether the procedure is either supervised or unsupervised. In the supervised classification procedures, the goal is usually maximising the classifier performance while the number of features is minimised as larger sets could produce overfitting and low generalisation problems. This way, a multi-objective optimisation approach that takes into account the classifier performance and the number of features allows for an adequate formulation of this goal. The situation in unsupervised classification problems is different. In this case, it is difficult to evaluate the clustering and, as has been previously indicated, the applied validation techniques usually present a dimensionality bias to either smaller or larger cardinality feature sets. Thus, a multi-objective approach could counterbalance the specific bias of the considered cluster validation method. Here, we propose using the NSGA-II algorithm [14] for multi-objective optimisation to build a wrapper approach that selects specific feature subsets for each class label. Some other works have been proposed to implement feature selection as a multi-objective optimisation, either for supervised or unsupervised classifiers. They are referenced and compared with the approach here proposed in Section 5.

In this paper, feature selection is considered in the context of network intrusion detection systems. With the growth of Internet, not only the number of interconnected computers, but also the relevance of network applications, have increased considerably. At the same time, the trend to online services has exposed a lot of sensitive information [15, 16]. This way, there are three main alternatives for protecting information. The first consists of avoiding sending information in clear (without any encryption). Such systems encrypt the information before sending for keeping its privacy. The second consists of using a separate physical or logical channel to transfer the information. This is the case of the Virtual Private Networks (VPN), which emulate a dedicated connection between two hosts. As a third alternative, the information on the VPNs can also be encrypted. Nevertheless, there is not any infallible encryption method and the encryption/decryption process can suppose a high overhead in high- speed networks that use the TCP/IP protocol stack.

However, the previous approaches do not react to attackers or intruders, but only suppose a passive protection to reduce exposure. The attacks can be classified into four categories [17, 18]:

1. *Denial of Service Attack (DoS).* In this case, the attacker tries to overload the victim machine in order to make it too busy to attend new legitimate requests. This attack can be performed by exhausting the memory, the network interface or other server resources.
2. *User to Root Attack (U2R).* In this case, the attacker logs into the system as a normal user, and then tries to change to a super-user by exploiting vulnerabilities.
3. *Remote to Local Attack (R2L).* In this case, the attacker uses system vulnerability to log into as a normal user. This attack may be a previous step

of the U2R attack.

4. *Probing Attack (PROBE).* In this case, the attacker tries to retrieve information about the computers attached to a network. Port scanning is an example of this probing attack.

In this way, new attack complexity necessitates using elaborated methods, such as pattern classification or artificial intelligence, to successfully detect an attack or just to differentiate bewteen normal and anomalous traffic. Thus, Intrusion Prevention or Intrusion Detection and Prevention (IPS or IDPS) are firewall-like active systems that protect the network by calculating some features from the network monitoring to be able to classify the traffic, detect abnormal behaviors and react according to some predefined rules [19]. This paper mainly deals with the Intrusion Detection system (IDS). There are two design approaches to IDS [20]. The first consists of looking for patterns corresponding with known intrusion signatures. The second searches for abnormal patterns by using more complex features that can reveal not only an intrusion, but also a potential intrusion. This can be figured by, for instance, discovering a misuse of the protocol flags or an abnormal number of certain events (such as the number of TCP connection attempts). Here, we approach network intrusion detection as a classification problem as in the second alternative. In works such as [20, 21, 14, 22, 23], unsupervised classification techniques with unlabelled data are applied to differentiate normal from anomalous traffic.

Unsupervised techniques for intrusion detection aim to group samples in clusters depending on the distance to other samples. On the other hand, supervised algorithms that work upon labelled data, such as k-nearest neighbour or Naïve Bayes classifiers, can be also used [20, 24, 25, 19, 26]. In this case, the algorithm classifies new points based on the distance to the training set, as in the k-nearest neighbour algorithm, where the majority class of the closest *k-neighbours* determines the class of the new instance. However, classical clustering techniques require a lot of iterations, making this technique not feasible for implementing real-time IDS.

Papers such as [27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 19], describe the use of artificial intelligence and other data-mining techniques to implement the intrusion detection and perform the attack classification. Paper [27] also applies perceptron-like neural networks for traffic classification, as well as fuzzy classifiers to improve decision making. In [28], SOMs [37] are used for data clustering and to classify the traffic anomalies according to several known attacks, taking advantage of SOM clustering properties to group the data instances. In [30], radial basis function (RBF) neural networks are used. Alternatively, several clustering techniques have been hybridised in order to improve the detection rate. Thus, in [31], RBFs with elliptical Gaussian basis functions are trained by using two different alternatives in order to obtain a high generative or discriminative behaviour. Generative models classify objects by building a probabilistic model for each category. Then, variational Bayesian inference is used to have a probabilistic class membership interpretation. On the other hand, discriminative classifiers estimate the probability that one object belongs to a specific category directly.

Evolutionary computation has been also applied to intrusion detection [32, 33, 35, 38]. In [32], evolutionary computation optimises the RBF used in the classification task and [33, 39] presents hybrid systems (for instance, [33] combines

artificial immune systems and SOM) to detect abnormal traffic patterns and to categorise them with SOM clustering. In addition, evolutionary techniques for mobile and ad-hoc network intrusion detection have been used in [35]. In general, SOMs are widely used for data-mining purposes to discover similarities in high-dimensional data, generating a number of prototypes corresponding to the number of SOM units, which generalises the data manifold. Moreover, a unique SOM feature is that the units are arranged in a 2D or 3D space where the ones corresponding to the most similar prototypes are moved close together. On the other hand, the units are moved away as the corresponding prototypes are dissimilar among them [37]. In accordance with the aforementioned characteristics, SOMs can be also seen as a dimension reduction method. Nevertheless, using SOMs presents some difficulties, especially when the feature (input) space dimension is very high. Moreover, the dimensions and size of the SOM have to be set before the training process, and their optimal values have to be determined by trial and error. However, the number of neurons on the output space determines the quality of the map and, then, the classification process performance (that is, classes should present enough differences among them and must be sufficiently apart from the rest of classes in the output space). In this way, works such as [27, 28, 29, 40] show the appropriateness of using hierarchical structures to improve the classification process. Thus, hierarchical architectures aiming to improve the classification performance have been used in these referenced works. As commented before, a hierarchical perceptron-like neural network is presented in [27] and the use of hierarchical SOM structures is considered in [28, 29]. The first approach uses a reduced feature set for training the perceptron networks, while the last one uses a first level comprised of three SOMs (each SOM representing one feature class) and a second level that summarises the clusters found by the first one. Although the proposal described in [28] tries to overcome some of the difficulties found on the classic SOM static structure by splitting it into three maps, the sizes of these maps are still static. In order to avoid some of these limitations, the Growing Hierarchical SOM (GHSOM) has been proposed in [41]. It is a dynamic structure in which the map grows during the training process to minimise its quantisation error. Thus, the GHSOM model presents a hierarchical architecture composed of several layers, also with several SOMs per layer. The number of SOMs per layer and the size of each SOM are determined during the GHSOM training. There are several papers that have applied GHSOM to IDS design [42] and have proposed several enhancements to improve GHSOM performance [36, 19].

In this paper, GHSOM is used for both anomaly detection and attack classification. Our approach slightly oversizes the GHSOM along the training process, leaving some units unlabeled (units that never win for any training pattern). Nevertheless, these units can be used to spread the clusters by applying the probability-based mechanism described in Section 3, to label the unlabeled units in the previously trained GHSOM structure.

As was said at the beginning of this section and in Section 2, feature selection clearly determines the classification performance. As is indicated in [2], feature selection formulation as a multi-objective optimisation problem could provide some advantages. Nevertheless, results would depend on whether the classification procedure is either supervised or unsupervised. In the supervised classification procedures, the goal has usually been the maximisation of the classifier performance while the number of features is minimised as larger feature sets could produce

overfitting and low generalisation problems. This way, a multi-objective optimisation approach that takes into account the classifier performance and the number of features adequately allows an adequate formulation of this goal.
The situation in unsupervised classification problems is different. In this case, it is difficult to evaluate the clustering, and the applied validation techniques usually present a dimensionality bias to either smaller or larger cardinality feature sets. Thus, a multi-objective approach can counterbalance the specific bias of the considered cluster validation method.
This paper proposes a multi-objective approach for feature selection that, although it implements feature selection for unsupervised classification procedures based on GHSOM, it also takes advantage of the labels that, in the available data set, indicate the class to which each input pattern belongs. This way, after training, it is possible to evaluate the classification accuracy, and the different objectives correspond to the classification accuracies for each of the different considered attacks (classes).

After this introduction, the remainder of this paper is organised as follows. Section 2 describes the main contribution of this paper: a dimensionality reduction technique based on multi-objective optimisation. Moreover, Section 2 provides a brief introduction to GHSOM-based classification procedure with the enhancements described in Section 3.1, as it has been used in the proposed wrapper method. In Section 4, the results obtained with and without feature selection are shown. In addition, statistical significance tests have been performed to demonstrate implementation improvement. Section 5 depicts the related works in IDS; finally, Section 6 provides the conclusions of this work.

2. Feature selection by multi-objective optimisation

Since classifier performance depends on the set of features used, it is necessary to accomplish an adequate feature selection. As it has been said, feature subset selection problem consist on applying a learning algorithm for selecting some subset of existing features upon which to focus its attention, while ignoring the rest. Specifically, in wrapper approaches, feature subsets are selected to maximize the value of a specific criterion as the classification accuracy in supervised classifiers [43, 44] or indirect measures as clustering quality in unsupervised ones. Thus, wrapper can be grouped into the *classifier-specific feature selection* (CSFS) methods [44]. Anyway, feature selection by exhaustive searching, which eventually results in testing all the possible feature subsets, is not feasible in terms of processing time. Subsequently, the use of suboptimal techniques that avoid to evaluate all the feature subsets provide an effective way to find relevant features. In what follows, we propose a feature selection procedure based on multi-objective optimisation. Here, as the problem consists of finding the features that maximise classifier performance, and since the dataset is labeled, a similarity measurement can be defined in order to simultaneously measure the number of elements being correctly classified. This way, we use the Jaccard's coefficient [45], which is a measurement of the asymmetric information on variables. In other words, it is a similarity measurement between datasets.

Let C_g be the ground truth labels provided by the dataset to identify the corresponding class to each pattern, and let C_s be the classification result. Thus, we can determine the Jaccard's coefficient for a specific class between the ground

truth labels and the labels calculated by our classifier, *J(class)*, as indicated in Equation 1.

$$J(class) = \frac{p}{p + fn + fp} \tag{1}$$

Where:

- p is the number of elements successfully classified (true positives).
- fn is the number of elements labeled as $a \in C_g$ in the dataset, but non-labeled as $a \in C_s$ by the classifier (false negatives).
- fp the number of non-labeled as $a \in C_g$ in the dataset and labeled as $a \in C_s$ by the classifier (false positives).
- class indicates whether traffic without attacks is considered class=*normal* or the type of attack (class is *DOS, PROBE, U2R* or *R2L*).

Thus, the Jaccard's coefficient provides a classifier performance measurement for the corresponding class. This coefficient can be used to select the features that fit better for a specific class, since these selected features will maximise the corresponding Jaccard's coefficient. Optimising classifier performance means maximising, at the same time, a number of objective functions equal to the number of classes. Each of these objective functions corresponds to the Jaccard's similarity coefficient for the given class.

Thus, the five objectives used for feature selection in the network anomaly detection problem are: J(*normal*), J(*DOS*), J(*PROBE*), J(*R2L*) and J(*U2R*). Thus, p, fn and fp in formula 1 refers to positives, false negatives and false positives, respectively, for each class label. As our work is focused on feature selection, we used a similarity index as the objective function for each class label in order to maximise this similarity between the predicted class labels and the ground truth. This way, accuracy is not directly used in the feature selection method, but rather in the assessment. On the other hand, our goal was to compare our results to previous work that used the same dataset; used as baseline, they only provided accuracy results without statistical validation.

Feature selection based on multi-objectives is suitable whenever:

- The experiments performed to reduce the feature set by using linear and multivariate techniques (such as PCA) show that it is not possible to effectively reduce the feature space dimension using linear techniques, in which case non-linear or stochastic methods should be used.
- The features that maximise one attack type do not always maximise the detection rate for all classes. In other words, maximising the detection rate for all classes deals with mutually exclusive objectives.

As a result of this multi-objective optimisation process implemented here by the NSGA-II algorithm [46, 14] and described in Algorithm 1 and Figure 1, we obtain the Pareto front which summarises the non-dominated solutions found by the multi-objective optimisation process. Thus, the Pareto front contains enough information to select the features for each attack type. The key point is to feed the

classifier with the most discriminant features to detect the elements belonging to the interest class. Hence, using a reduced set of features leverages the classification performance while the computing time for training is reduced.

Algorithm 1 Pseudo-code of the multi-objective optimisation process for feature selection with NSGA-II [46, 14]

1: Generate an initial population
2: **Evaluate objective values:**
3: Classifier training
4: Dataset classification
5: Jaccard's coefficients calculation
6: **NSGA-II Loop**
7: Assign rank based on Pareto Dominance as in NSGA-II to select non-dominated individuals
8: Apply evolutionary operators to generate a new population
9: Evaluate objective values
10: Classifier training
11: Dataset classification
12: Jaccard's coefficients calculation
13: **End NSGA-II loop**

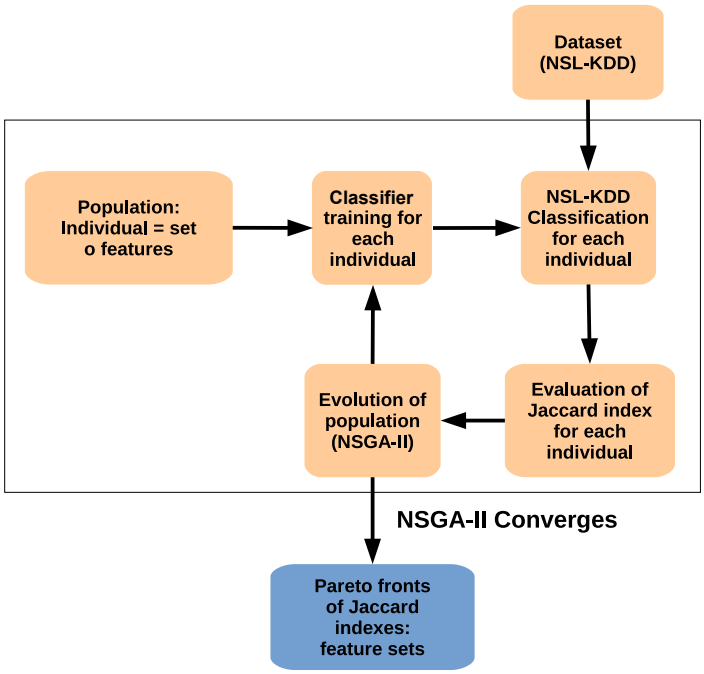


Figure 1: Scheme for Multi-objective optimisation with NSGA-II algorithm for feature selection. Solutions computed by NSGA-II correspond to feature selections

In Section 4, we compare the performance of this procedure with the results obtained by using the whole feature set and also with the use of PCA to reduce the dataset dimension. Parameter settings for NSGA-II execution are detailed in Section 4.

3. Classification with GHSOM

The classification procedure used in the wrapper procedure proposed in this work (step 10 in Algorithm 1), is based on the Growing Hierarchical SOM (GHSOM). Thus, this section introduces the main concepts of Self-Organizing Maps (SOMs) and GHSOM and describes some improvements in the GHSOM model consisting on a probabilistic labelling method. Although details on the SOM and GHSOM algorithms can be found elsewhere [37, 41], in this section, we briefly describe these neural models here for quick reference. We also motivate the need for GHSOM, and the way the winning map neuron, also called Best Matching Unit (BMU), can be obtained.

The SOM [37] is one of the most used artificial neural network models for unsupervised learning. The main purpose of SOMs is to group similar data instances close into two- or three-dimensional lattices (output maps), while computing a number of prototypes that generalise the input data manifold. SOMs consist of a number of neurons also called *units* that are arranged following a previously determined lattice. During the training phase, in each iteration t, the distance between an input vector and the weights associated to the units on the output map are calculated. Usually, the Euclidean distance is used, as in Equation 2

$$U(t) = \underset{i}{\operatorname{argmin}} \parallel x(t) - \omega_i(t) \parallel \tag{2}$$

that provides the index, $U(t)$, of the *winning unit*, also called Best Matching Unit (BMU), that is, the unit with the smallest Euclidean distance to the input pattern $x(t)$). Then, the weights of the units in the neighbourhood of the *winning unit* are also updated as in Equation 3

$$\omega_j(t+1) = \omega_j(t) + \alpha(t)\, h_{U(t)}(t)(x(t) - \omega_j(t)) \tag{3}$$

where $\alpha(t)$ is the learning factor, that decreases with time linearly or exponentially, and hU(t) is the topological neighbourhood function, usually a Gaussian function (4)

$$h_{U(t)}(t) = e^{-\frac{\|r_{U(t)} - r_i\|^2}{2\,\sigma(t)^2}} \tag{4}$$

in which the size of the topological neighbourhood, $\alpha(t)$, shrinks with time as shown in (5).

$$\sigma(t) = \sigma_0\, e^{-\frac{t}{\tau_1}} \tag{5}$$

In (4), r_i represents the position on the output space (2D or 3D) and $\parallel r_{U(t)} - r_i \parallel$ is the Euclidean distance between the winning unit $U(t)$ and the unit *i-th* within the (2D or 3D) output space. The SOM performance can be evaluated by using two measures. The first is the quantisation error, q, a measure of the map resolution, defined in (6)

$$q = \frac{1}{n_{C_i}} \sum_{x_j \in C_i} \parallel \omega_i - x_j \parallel \tag{6}$$

where C_i is the set of input vectors mapped into the unit i, x_j is the *j-th* input vector belonging to C_i, ω_i is the weight associated to the unit i, and n_{C_i} is the

number of vectors in C_i. The other measure is the topographic error, t, that evaluates the SOM-quality topology-preservation characteristics (7)

$$t = \frac{1}{N} \sum_{i=1}^{N} u(\overrightarrow{x_i}) \tag{7}$$

where N is the total number of input vectors and $u(\overrightarrow{x_i})$ is 1 if the first and second BMUs for the input vector $\overrightarrow{x_i}$ are adjacent units, and is 0 otherwise [37]. The lower q and t, the better the SOM is adapted to the input patterns.

Although an SOM is a very useful tool for discovering structures and similarities in high-dimensional data, it is not able to figure out its inherent hierarchical structure [41], and its performance depends on the map size being determined in advance. GHSOM [41] is a hierarchical and non-fixed structure developed to overcome these problems. The GHSOM structure consists of multiple layers composed of several independent SOMs whose number and size are determined during the training phase. The adaptive growing process is controlled by two parameters that determine the hierarchy depth and each map's breadth. Therefore, these two parameters are the only ones that have to be initially set. In GHSOM, the quantisation error of each unit is given by Equation (8)

$$q_i = \sum_{x_j \in C_i} \parallel \omega_i - x_j \parallel \tag{8}$$

where C_i, x_j, C_i, and ω_i are defined as in (6). Initially, all the input vectors belong to C_0 (all the inputs are used to compute the initial quantisation error, q_0).

If $q_i < \tau_2 \cdot q_0$, neuron i is expanded in a new map on the next level of the hierarchy. Each new map is trained as an independent SOM, and the BMU calculation is performed as shown in Equation (3).

Alternatively, given a unit i, its mean quantisation error, mq_i, can be used to control the growing process. The value of mq_i is calculated by using Equation (6). Since mq_i measures the dissimilarity of the data already mapped to a specific unit, the growth process can be controlled in order to diminish mq_i until a minimum is reached.

Once the training of a map m is finished, the mean quantisation error, MQ_m, is obtained as the mean of the mq_i values for all units i in the map; it can be used to check map growth. Thus, the map should grow whenever $MQ_m \geqslant \tau_1 \cdot q_u$ is verified (q_u is the quantisation error of unit u on the upper layer and τ_1 is a parameter to control the depth of the GHSOM). To grow the map, a row or a column of units is inserted between the unit e with the highest quantisation error and its most dissimilar neighbour unit $d = arg\,max_i(\parallel w_e - w_i \parallel)$, where units i belong to the neighbourhood of e. In order to calculate the BMU in the GHSOM, we have to follow the hierarchy to determine the winning unit and the map to which it belongs.

Thus, an iterative algorithm is applied as shown in Figure 2, where an example of BMU calculation on a three-level GHSOM hierarchy is provided. Let us suppose that we calculate the distances between an input pattern and the weight vectors of the level 0 map, and then compute the minimum of these distances. As a result, the winning neuron on map 1 is found. Since another map could be grown from this winning neuron, we have to check whether the winning neuron is a parent unit (that is, a unit from which a new map has grown). This can be accomplished by

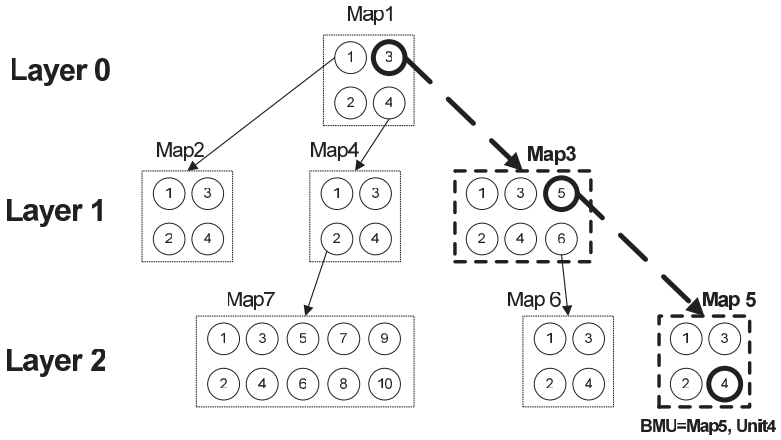


Figure 2: BMU calculation on the GHSOM hierarchy

using the parent vectors that contain the growing path (that is, the parent vectors and the index of this new map) resulting from the GHSOM training process. If a new map arose from the winning neuron, the BMU on this map would be calculated. This process is repeated until a BMU with no growing map is found. Thus, the GHSOM BMU is associated to the map in the corresponding hierarchy layer (that is, in Figure 2, Unit 4 of Map 5, in Layer 2).

3.1. GHSOM with probabilistic relabelling (GHSOM-pr)

In this section, we propose a procedure to label the GHSOM units that make possible a probabilistic clustering process behaviour. The values of parameters τ_1 and τ_2, previously introduced, have to be set carefully to achieve an efficient training process. For example, higher values for τ_1 and τ_2 make the GHSOM grow more than necessary, leaving some of the units unlabeled. Thus, the number of neurons on the output map surrounding the winning neuron for training data is increased. Whenever an input pattern similar to one of the training patterns is presented to the GHSOM, the winning neuron can be labelled or unlabeled. The label of a unit determines the data class to which it belongs. If the winning neuron (BMU) is unlabeled, a probability-based procedure is proposed in order to determine that neuron's label. In this procedure, the winning neuron is *relabeled* with the most repeated label in its neighbourhood by using a probability determined according to (9).

$$P_u = \frac{M_{\sigma(u,\epsilon)}(u)}{n} \tag{9}$$

In this expression, P_u is the probability for the winning unit u to be successfully relabelled, $M_{\sigma(u,\epsilon)}(u)$ is the number of units in the Gaussian neighbourhood, $\sigma(u,\epsilon)$, is the winning unit u that is labelled with the more frequent label in $\sigma(u,\epsilon)$, and n is the number of neurons belonging to $\sigma(u,\epsilon)$. In this equation, the parameter ϵ determines the width of the winning neuron neighbourhood. Although through this procedure we assign a label to an unlabeled unit, in most cases we have called it as relabelling, since the term "labelling" is usually applied along the training process. In Figure 3, an example of the relabelling process is shown. In this Figure 3, the BMU that has not been initially labelled is relabelled by using $\epsilon = 1$ to establish its neighbourhood. In this neighbourhood, we found four units labelled as L1, one unit labelled as L2 and one unit labelled as

L3. Then, P_u, the *a priori* probability for successful relabelling for this BMU is 4/6=0.66 (66%) ($M_{\sigma(u,\epsilon)}(u) = 4$, $n = 6$).

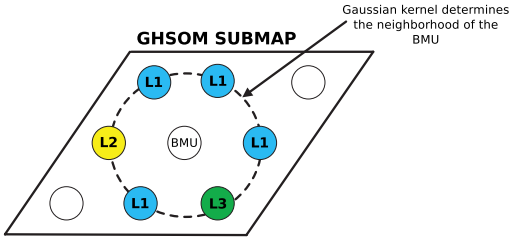


Figure 3: Example of the relabelling process (white units are unlabeled).

Moreover, the posterior relabelling probability for unit k can be computed by means of the Bayes' theorem:

$$p(\omega_k|x) = \frac{p(x|\omega_k)P(\omega_k)}{p(x)} \tag{10}$$

In Equation 10, $p(\omega_k|x)$ represents the posterior probability that a sample vector x belongs to class ω_k, while $p(x|\omega_k)$ is the conditional probability of x to the class k, $P(\omega_k)$ is the *a priori* probability and $p(x)$ is a normalisation constant, computed as

$$p(x) = \sum_{k=1}^{n} p(x|\omega_k)p(\omega_k) \tag{11}$$

This way, this posterior probability can be used to classify new samples. In this work, the posterior probabilities have been used to relabel the units that remain unlabelled during the SOM training process. This way, the dynamic relabelling of the map units presented in this section introduces a probabilistic clustering process behaviour that improves the GHSOM models, as shown in the results provided in Section 4.

In [47], GMM is used to model the clusters in SOM. Then, a probabilistic labelling method is applied, but it only takes into account a priori probabilities computed according to equation 9 (it uses the most frequent label in the neighbourhood). Moreover, [34] uses a similar labelling scheme for dead units [37] based on *a priori* probabilities. However, the method proposed in this work uses posterior probabilities through the Bayes formula to compute the most probable label, according to Equation 10. This constitutes an important improvement, as computing unit labels depends not only on the map labels, but also on the current sample, assigning the label of each BMU in an adaptive way.

4. Experimental results: application to Intrusion Detection

IDSs are firewall-like active systems that calculate some network monitoring features to be able to detect anomalous traffic. In this section, we present the experimental results obtained by applying the GHSOM classifier described in Section 3 and the multi-objective optimisation-based feature selection approach described in Section 2, to the NSL-KDD dataset [20]. This dataset comes from the KDD'99 dataset [48], that contains about 4GB of compressed data from captures

of *tcpdump* [49] in the DARPA'98 IDS evaluation program [17], corresponding to about seven weeks of network traffic. There are three extracted features included in the KDD'99 dataset:

- *Basic features*. These features summarise all the properties of a TCP/IP connection.
- *Traffic-based features*. These features are computed over a time interval (window) and contain information about the connections in which the destination port or the service remains the same after the corresponding time windows. In the KDD'99, the time window used to compute these features is two seconds.
- *Content-based features*. Since U2R or R2L attacks consist of repeatedly sending similar patterns on the packet payload, it is necessary to examine the packet contents to figure out these attacks. Thus, statistics regarding the packet contents are calculated and classified as content-based features. An example of content-based features statistics is the number of failed login attempts.

The KDD'99 dataset presents some inherent problems, such as the synthetic characteristic of the data [17, 23], and consequently, may not be representative of real attacks. Thus, the Network Security Lab - Knowledge Discovery and Data Mining (NSL-KDD) dataset was proposed. Moreover, in the NSL-KDD, redundant KDD'99 records were removed and the attacks labelled and sorted by their detection difficulty level. Taking into account all these characteristics, the NSL-KDD can be considered a good approximation of present known attacks. Moreover, NSL-KDD constitutes an adequate dataset to evaluate our procedure as the most recent reference works in IDS [50, 51, 26, 52, 40] also use the NSL-KDD.

The experiments have been performed in two ways: the first corresponds to classification experiments using the full feature set (41 features), while the second consists of applying dimensionality reduction techniques to the feature set in order to avoid using the full feature set. This dimension reduction is accomplished either by linear techniques (PCA) or by the multi-objective optimisation method described in Section 2 for comparison.

In order to prove that the system is not over-fit and thus, has a good generalisation performance, feature selection and training processes have been assessed by k-fold cross-validation (k=10). Since k=10, partitions containing 90% of samples were randomly selected to fit the model; the rest of the samples (10%) were used for testing. These subsets are different and do not share any samples. This process was repeated for the 10 folds, ensuring test data is never used in the feature selection or the classifier training. Hence, the results provided for selected feature subsets and classification accuracy are computed as the average of 10 evaluations throughout 10 folds. The main purpose of cross-validation is to estimate the generalisation error, ensuring that similar results will be obtained on new data (that is, low generalisation error). This method estimates the prediction error and avoids double-dipping. Moreover, it is worth noting that due to the high number of available dataset samples, both training and test processes are addressed using a high number of samples. This provides a lower generalisation error variance

estimate.

Additionally, several tests with different values for τ_1 (to control the map breadth) and τ_2 (to control the depth) have been performed.

Data preprocessing

Despite data preprocessing playing an important role in classification performance, few works pay enough attention to it [53]. Data preprocessing comprises encoding non-continuous variables and normalisation. As described in the preceding section, KDD'99-based datasets consist of 41 features, which should be enough to characterise anomalous connections. They are classified into three groups: continuous, symbolic and binary features. As most classifiers only accept numeric values, the first issue is related to symbolic feature encoding. In several works, they are usually coded by simply substituting each different feature with an integer number [53, 54]. Although this can be acceptable in many situations, it is not the best encoding solution for classifiers based on the Euclidean distance [55]. This way, we adopt a different solution that maps each symbolic feature to a $\mathbb{R}^d$ subspace, where d is the number of possible discrete variable values. Although this solution increases data dimensionality (for instance, the *service* feature can take 65 different values), it is not critical for the classifiers used in this work. Furthermore, dimensionality reduction techniques are used to compress relevant information with fewer features. Thus, a different value on these features contributes $\sqrt{2}$ to the distance measure.

Data normalisation

Data normalisation ensures that no feature contributes more than another in the distance measure. There are different ways of normalising data [1]. In this work, continuous variables are normalised to zero mean and unity variance using the equation 12.

$$\hat{x} = \frac{x - \overline{x}}{\sigma} \tag{12}$$

where $\overline{x}$ and σ are the mean and the standard deviation of variable x, respectively. This is equivalent to expressing the variable x as the number of standard deviations away from its mean. Moreover, all the variables are scaled to $[0, 1]$. Symbolic (already encoded to binary vectors) and binary features are not normalised.

4.1. Experimental results

First, the NSL-KDD dataset with all the features is used for classification. The purpose of these first results is to demonstrate the improvements obtained by the GHSOM relabeling method described in Section 3.1.

In Figure 4, the detection rate for each attack present on the NSL-KDD dataset is shown (Anomalies in Figure 4 can be grouped as belonging to an attack type as indicated in Table 1). As this figure shows, our probability-based relabeling process performed with new data (black bar) increases the detection rate for most attacks. The average improvement is about 3.5% over all the attacks, and 5.5% over the attacks on which the relabeling creates improvements.

Detection rate in Figures reflects the percentage of anomalies correctly identified (that is, true positives). False positives (fp), true negatives (tn) and false

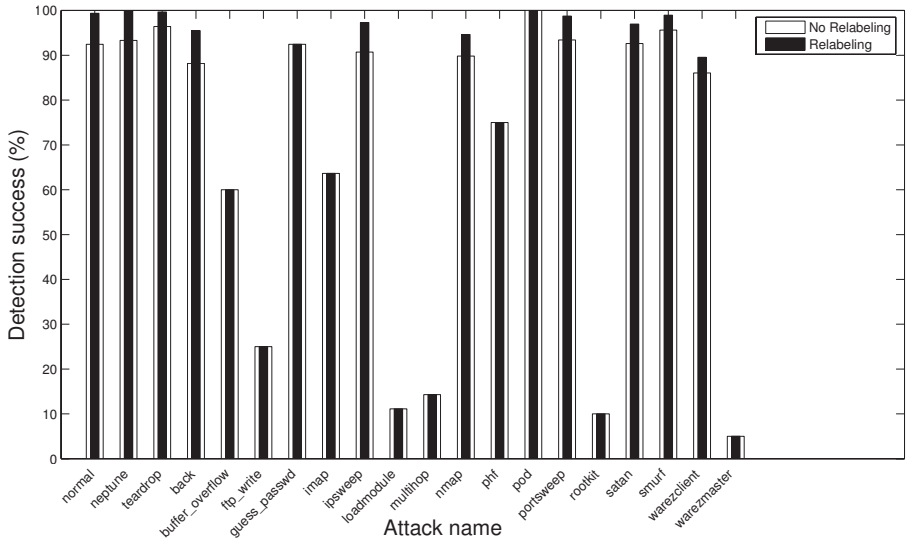


Figure 4: Detection rate with unit relabeling (black bar) and without unit relabeling (white bar). GHSOM is trained with the full feature set (1,384 neurons, five layers).

Attack type	*Attack name*
DOS	apache2, back, land, mailbomb, neptune, pod, processtable, smurf, teardrop, udpstorm
PROBE	ipsweep, mscan, nmap, portsweep, saint, satan
R2L	ftp_write, guess_passwd, httptunnel, imap, multihop, named, phf, sendmail, snmpgetattack, snmpguess, spy, warezclient, warezmaster, worm, xlock, xsnoop
U2R	buffer_overflow, loadmodule, perl, ps, rootkit, sqlattack, xterm

Table 1: Classification of network anomalies into different attack types

negatives (fn) are indicated by means of the ROC curves and accuracy values provided further in this section. In order to show the effectiveness of the relabeling process described in Section 3.1, the ROC curves are shown in Figure 5.

Figure 5a shows the ROC curve (false positive rate) and Figure 5b the mirrored ROC curve (true negative rate) obtained for classifying the test dataset (not used during training). Regarding the performance derived from these curves, we have computed the *Area Under ROC Curve* (AUC). Using AUC makes interpreting the results from the ROC curve easier. Thus, a perfect classifier will provide an AUC=1.0 and an AUC=0.5 in a random classifier. In these graphs, the cut-off point determines the best performance provided by the classifier.

Figure 6 presents the detection rate per attack type. As can be seen, the relabeling method increases classification performance in most cases, as well as the detection rate. It is worth mentioning that performance is worse for U2R than for other attack types, due to the smaller number of U2R training patterns in the

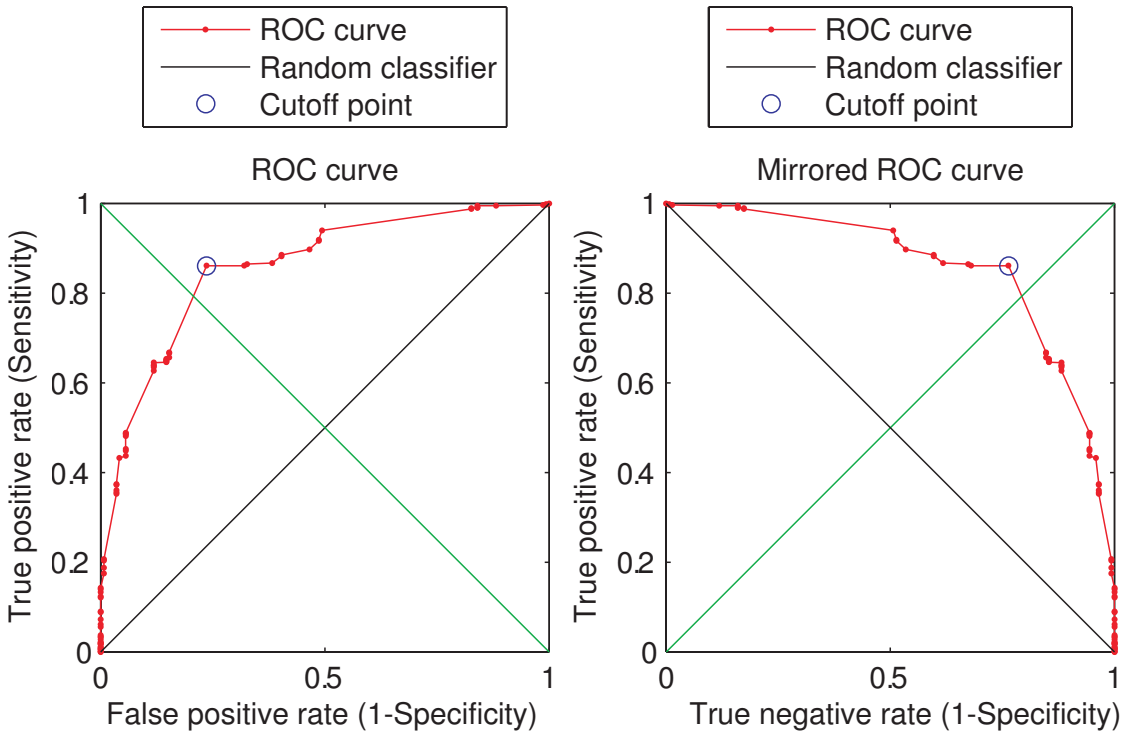


Figure 5: ROC curves for the relabeling process. GHSOM is trained with the full feature set (1384 neurons, 5 layers).

NSL-KDD dataset [52].

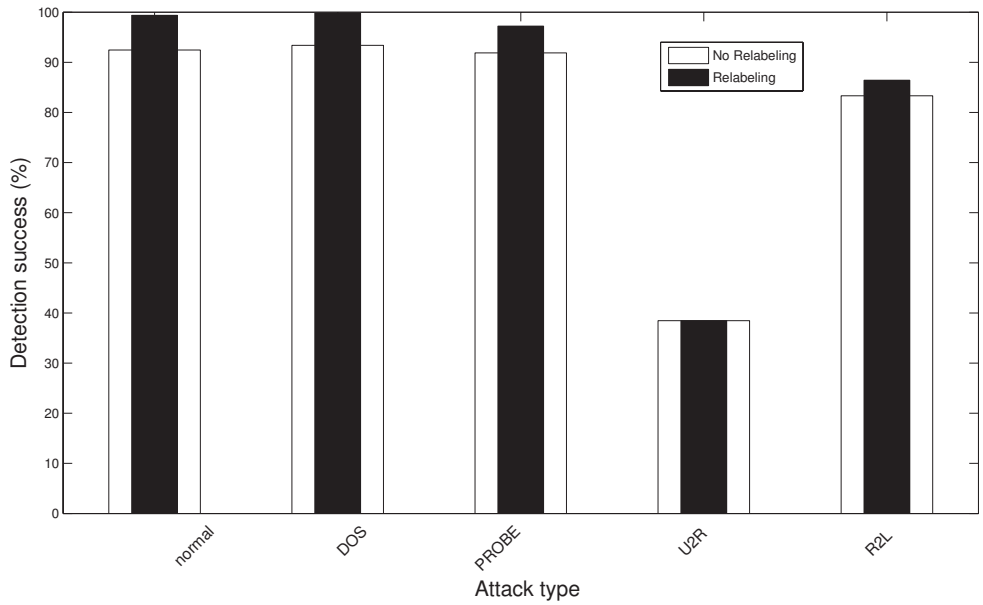


Figure 6: Detection rate for different types of attacks (GHSOM with 1384 neurons)

Results detailed at attack-type level in Figure 6 are summarised in Table 2, where detection and false positive rates are presented. As shown in this table, 99.8% of the patterns corresponding to normal traffic patterns and 99.5% of the patterns corresponding to the different attacks are correctly classified. High detection rates are possible, as shown in Figure 6, although most U2R attacks are

not classified correctly, since the KDD dataset is rather unbalanced. In fact, U2R and R2L samples represent only 0.001% and 0.02% of the training samples, respectively [40].

Connection	**Detection Rate (%)**	**False Positive rate (%)**
Normal	99.8 ± 0.10	1.10 ± 0.92
Attack	99.5 ± 0.33	4.33 ± 0.56

Table 2: Normal/Attack detection rates

Nevertheless, when the whole set of features is used, a large number of GHSOM neurons are required to reach the highest detection rate, as shown in Figure 6. Figure 7 shows the performance provided by the classification algorithm as a function of the number of GHSOM neurons. At the same time, Figure 7 shows the performance increase achieved by the relabeling process described in Section 3.1. The probability of structural dead units increases as the GHSOM grows. This effect is reflected in the classification accuracy as shown in Figure 7. However, units that remain unlabelled during the training process (that is, a dead unit), can be the BMU for new data instances. In this case, relabeling wakes up dead units depending on the label previously assigned to the neighbour units during training. Moreover, Figure 7 shows the effect of the relabeling method and its advantage when the number of units grows.

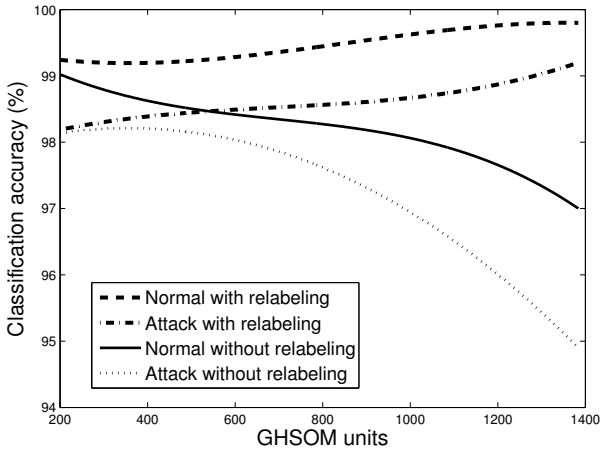


Figure 7: Detection rate with and without relabeling when the full feature set is used, as a function of the number of neurons on the GHSOM. (The GHSOM has been trained with the full feature set; its maximum size is 1,384 units and five layers)

Thus, when relabeling is applied, the detection rate depends less on the number of GHSOM neurons. Moreover, the performance of the system without relabeling decreases as the number of GHSOM neurons grows. Then, since the use of non-optimised feature sets allows wider and deeper GHSOM with more neurons, the relabeling process has a positive performance effect. This is precisely the situation in cases of unknown attacks, where the features cannot be selected according to their discriminative properties.

4.2. Experimental results with reduced feature sets

As is shown in Section 4.1, although GHSOM with relabeling allows high detection rates, using large GHSOM maps is necessary to provide detection rates higher than 98%. However, the goal of the algorithms proposed in this paper is to implement the IDS/IPS in a real-time module that could detect anomalous behaviours and decide whether or not to block a connection. Then, the whole process has to be fast enough; to speed up the detection process makes it necessary to reduce the complexity of both the feature selection and the classification processes.

Consequently, reducing the feature space dimension is convenient in order to optimise both the classification rate and the computation time while preserving labelling process performance. Training the classifier with a reduced set of features could improve performance, while producing smaller GHSOM if the features are suitably selected to discriminate among the input patterns. On the other hand, since the classifier is trained by using vectors with fewer dimensions, the training process, as well as the BMU calculation, will be faster. In fact, Figure 8 shows that GHSOM training time grows exponentially when the unit number increases.

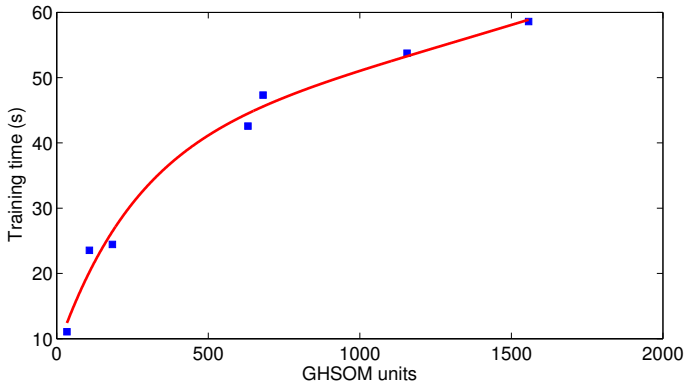


Figure 8: Training time depending on the number of GHSOM units

However, finding a reduced and discriminative enough set of features is not straightforward. In this section, we show the results obtained with feature sets selected for each attack type by using our multi-objective optimisation procedure described in Section 2. Each feature set is used to train a GHSOM and to classify the test patterns.

It is worth noting that Pareto fronts in Figure 9 have been computed using the training data, since labels are necessary to compute the Jaccard index used in fitness functions. As the Pareto fronts are five-dimensional in this case (that is, five objective functions are considered in the optimisation process). Figure 9 shows four projections of the obtained Pareto fronts into the planes corresponding to four pairs of objectives. From the Pareto front obtained by out multi-objective optimisation procedure, several different feature sets can be selected. In our experiments shown below, we have used the feature sets S1, S2, S3, S4 and S5, as is shown in Figure 9.

The obtained non-dominated feature sets selected are used for training the GHSOM. In the following, we show the classification results when the GHSOM is trained and tested with the optimised feature sets obtained from the Pareto front, as indicated in Figure 9.

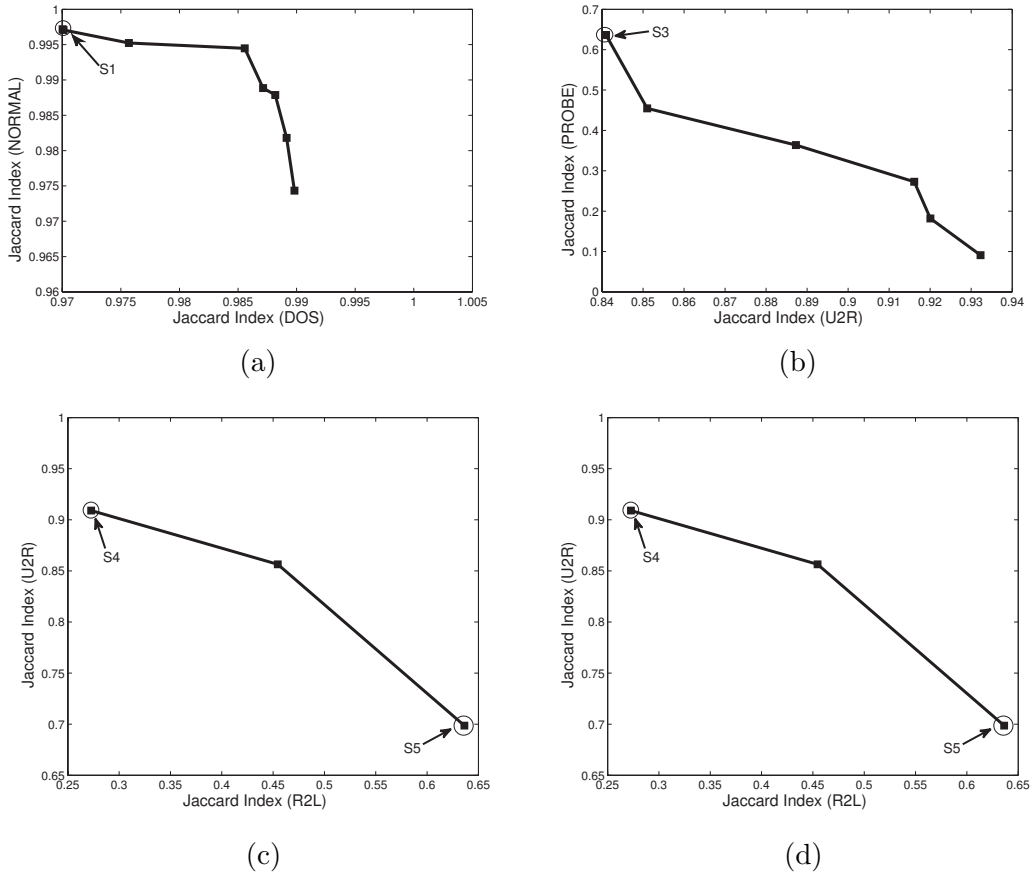


Figure 9: Pareto Fronts for (a) Normal/DOS, (b) PROBE/U2R, (c) U2R/R2L and (d) DOS/PROBE. GHSOM is trained with the non-dominated feature sets S1 (340 neurons, five layers), S2 (308 neurons, six layers), S3 (340 neurons, five layers), S4 (420 neurons, five layers) and S5 (225 neurons, five layers).

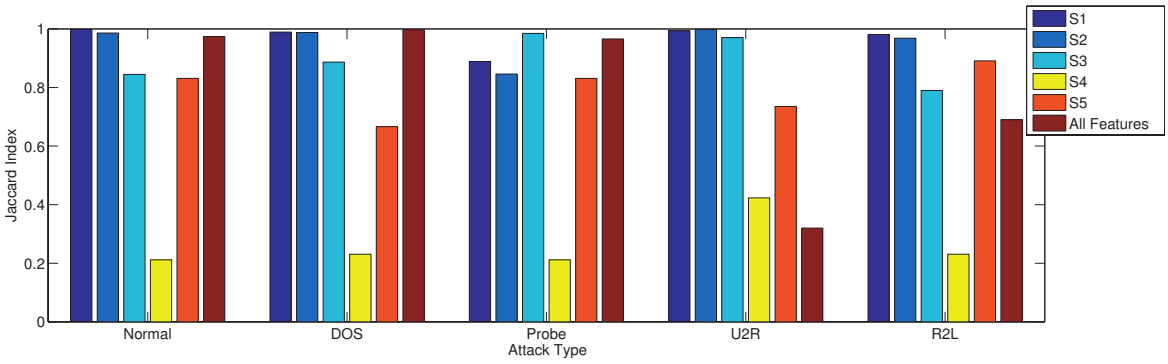


Figure 10: Detection Rate using non-dominated feature sets selected from multi-objective optimisation. GHSOM is trained with all feature (1384 neurons, five layers), the optimum feature sets for Normal attacks (340 neurons, five layers), DOS attacks (308 neurons, six layers), PROBE attacks (340 neurons, five layers), U2R attacks (420 neurons, five layers) and R2L attacks (225 neurons, five layers).

In Figure 10, detection rates obtained using non-dominated feature sets selected from multi-objective optimisation are shown. As detecting each attack type is addressed by different feature set, the GHSOM structure developed during training is different. As Figure 10 shows, non-dominated feature sets provide

higher detection rate values for different attacks than the alternative using all features. Additionally, in the case of the U2R attack, feature set S4, in which the Jaccard index for U2R is maximum, provides higher classification accuracy than the full feature set. It is important to indicate that while the non-dominated feature sets have been computed from the training samples, the classification results shown on Figure 10 are obtained with the test dataset. The use of all features provides high accuracy levels for most probable classes. This is the case of Normal, DOS or PROBE classes. However, selected features clearly provide a higher performance for less probable classes, such as U2R and R2L, as indicated in Figure 10. Moreover, the purpose of feature selection is not only to leverage the classification performance, but also to have a reduced set of discriminative features providing the high accuracy values. Thus, using fewer features reduces the computational burden associated with training and classifying new samples. In other words, as selected features provide at least the same accuracy as the full feature set, our method effectively discards non-informative or redundant features for each class label.

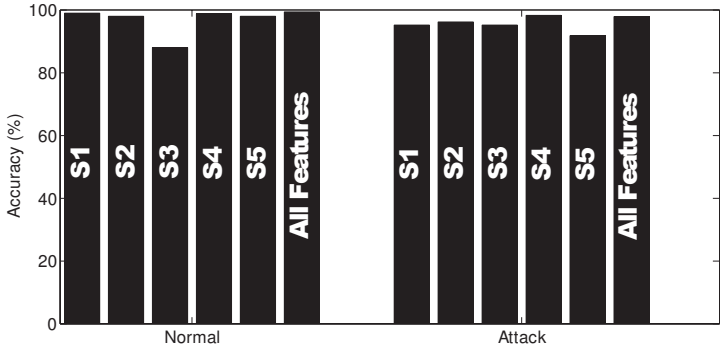


Figure 11: Detection Rate comparison for different number of PCs. GHSOM is trained with all features (1,384 neurons, five layers), S1 set (340 neurons, five layers), S2 set (308 neurons, six layers), S3 set (340 neurons, five layers), S4 set (420 neurons, five layers) and S4 set (225 neurons, five layers).

In Figure 11, the normal/attack detection rate is summarised when using feature sets from the Pareto front obtained by multi-objective optimisation procedure based on the NSGA-II algorithm. In this figure, it is clear that the S4 feature set provides better attack classification results than the full feature set. Additionally, it requires fewer GHSOM units; consequently, training and classification processes are less computationally expensive.

NSGA-II chromosomes consist of binary strings that determine the subset of features to be selected in each iteration. Thus, we use the single-point crossover and bitwise mutation operators. Initial NSGA-II population consists of 30 individuals. Moreover, crossover probability of $p_c = 0.9$ and mutation probability of $p_m = 1/l$ where l is the gene length, are used ($p_m = 1/41$). These values yield good results in different experiments performed in [46] on different datasets; they also provided a good solution spread, as well as algorithm convergence as shown in Section 4.3.

Figure 12 shows the ROC curves for normal/attack-type classification using relabeling with the corresponding non-dominated feature set. From these curves,

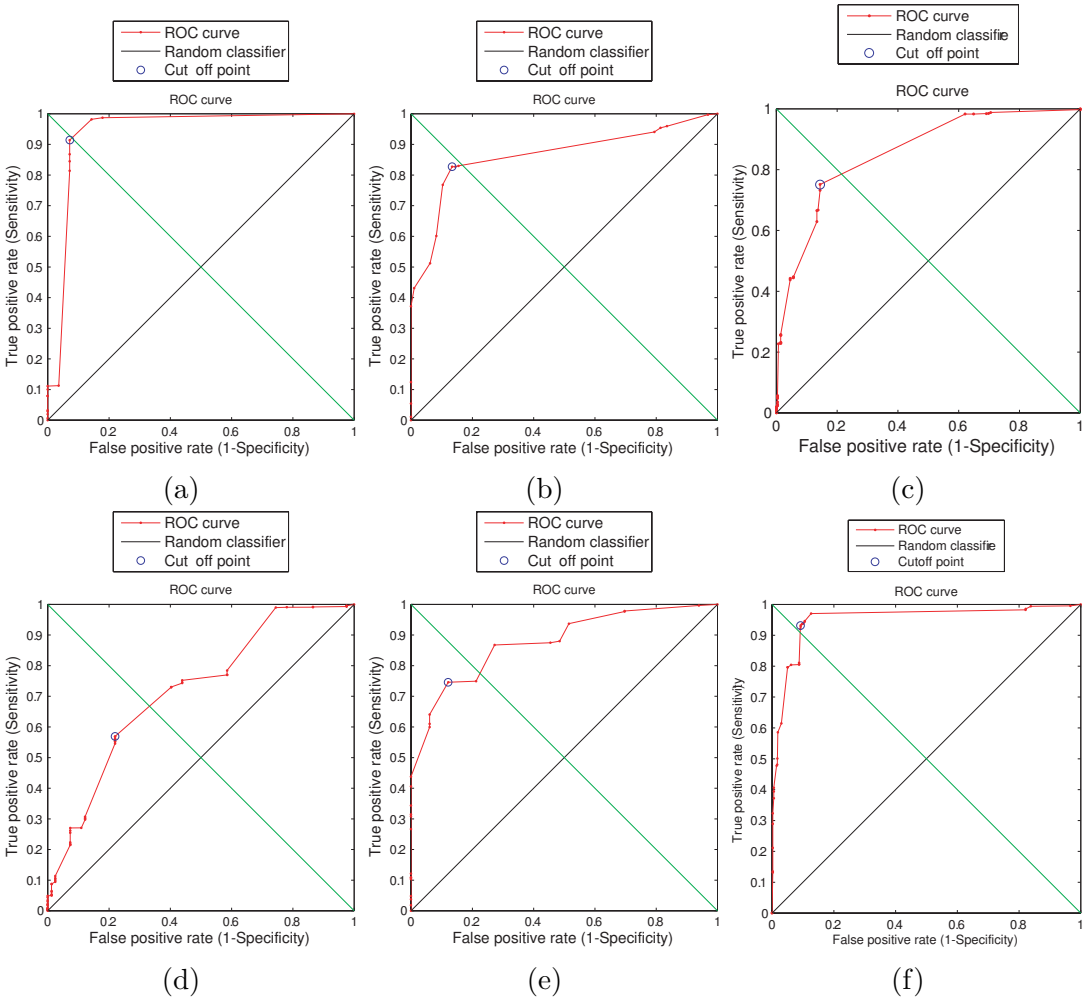


Figure 12: ROC curves for the relabeling process with the non-dominated feature set obtained for (a) S1 (340 neurons, five layers), (b) S2 (308 neurons, six layers), (c) S3 (340 neurons, five layers), (d) S4 (225 neurons, five layers), (e) S5 (420 neurons, five layers) and (f) PCA with 30 PCs (1654 neurons, five layers)

it is clear that non-dominated feature sets improve the results obtained with the full feature set in terms of sensitivity and specificity.

Feature set	Number offeatures	Accuracy (%)	False Positive (%)
All Features	41	99.6 ± 0.33	4.32 ± 0.80
S1	22	98.12 ± 1.10	3.10 ± 0.76
S2	**29**	**99.12 ± 0.61**	**2.24 ± 0.41**
S3	25	98.27 ± 1.19	2.13 ± 0.59
S4	25	98.10 ± 1.71	3.15 ± 0.50
S5	29	97.18 ± 1.43	4.16 ± 0.52
PCA	20	82.1 ± 5.13	6.50 ± 1.00

Table 3: Classification accuracy for different feature sets.

Table 3 gives the classification process accuracies, and the different feature sets determined by our multi-objective optimisation procedure. The last row of Table 3 provides the accuracy obtained with the best set of features determined

by Principal Component Analysis (PCA). An accuracy of 99.12% is achieved for the five different attacks here considered when using the non-dominated feature set with maximum Jaccard Index for S4.

In previous sections, we analysed the performance of the GHSOM when detecting different attack types. Nevertheless, detecting normal and attack traffic with maximum accuracy is essential for practical implementation, meaning high accuracy detection is not useful if other attack types are not similarly detected.

IDS implementation	Features used for classification	Detection Accuracy (%)	False Positive(%)
Without Feature Selection			
Naïve Bayes [23]	41	76.56	Not provided
Random Forest [23]	41	80.67	Not provided
Decision Tress (J48) [23]	41	81.05	Not provided
AdaBoost [50]	41	90.31	3.38
GHSOM-pr	41	99.59 ± 2.25	4.32 ± 1.93
GHSOM [56]	41	96.02	4.92
A-GHSOM [19]	41	96.63	1.80
Kayacik et al. [40]	41	90.40	1.38
Filter methods			
Naïve Bayes+N2B [50]	41	96.50	3.00
PCA	30	82.1 ± 5.13	6.50 ± 1.00
FDR + Kernel PCA [57]	23	90.0	8.00
Wrapper methods			
Decision tree-based [58]	16	98.38 ± 1.62	Not provided
GHSOM + Multiobjective Feature Selection (S4)	**25**	**99.12 ± 0.61**	**2.24 ± 0.41**

Table 4: Detection accuracy comparison for different classification methods. Standard deviation values are shown whenever available

Methods shown in Table 4 have been classified into 1) methods not using feature selection, 2) filter methods and 3) wrapper methods. In Table 4, the comparison with other existing methods using the NSL-KDD dataset is shown. As shown in this table, our approach reaches a high rate of detected attacks in the range of the best-performing existing approaches. Nevertheless, it has to be pointed out that with the results of Table 4 we only give an idea of the performance of our procedure compared with the results provided by other authors. Although, in the cases shown in Table 4, our proposal outperforms other previous procedures, we do not claim that our method is better than the rest ones in all the cases and classification problems. Such an exhaustive comparison is not possible because the only available performance results refer to detection rate, the standard deviation

is not even available in all cases, and we have not the specific implementations of the different procedures to accomplish the required set of executions.

4.3. NSGA-II performance evaluation

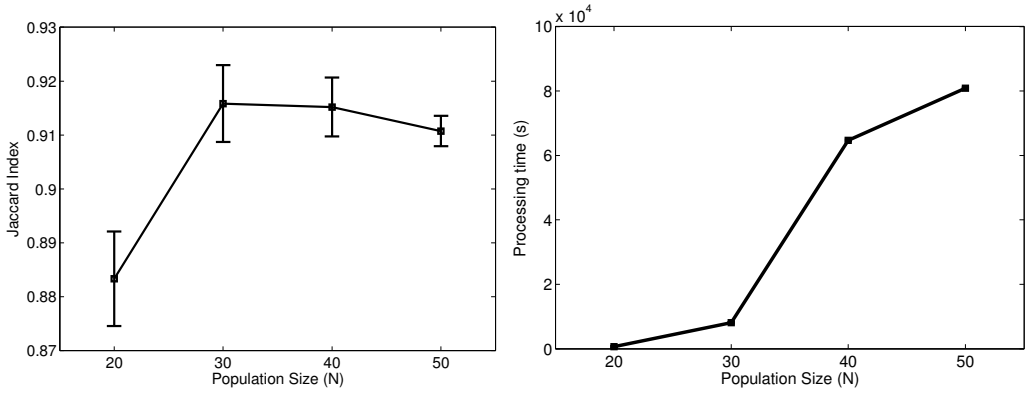
In this section, we provide NSGA-II convergence results to show the number of generations needed to provide good enough solutions. Although determination of convergence in multi-objective algorithms is not straightforward [59], in this work, we used the well-known hyper-volume metric [60]. This way, the higher the hyper-volume, the greater the number of non-dominated solutions, indicating better performance. In evolutionary algorithms, population size plays an important role as it is directly related to its diversity. Low diversity populations may need a higher number of generations to evolve to acceptable solutions. On the contrary, high-diversity populations increases the complexity and eventually the processing time. The experiments performed here have shown that populations of 30 individuals provide the best results as it is depicted in Figure 13a as well as a good trade-off between preserving time and classification accuracy as shown in Figure 13b. Moreover, Table 5 shows the mean Jaccard index for all the classes as the multi-objective approach aims to maximize it for all classes simultaneously as well as the hyper-volume computed for the corresponding non-dominated fronts. As it is shown in Table 5 and Figure 13a, populations of 30, 40 and 50 individuals (N) provide the same performance taking into account the standard deviation values. However, computational complexity is considerably higher for N=40 as shown in Figure 13b. Similarly, while N=20 provides lower hyper-volume values, the ones provided by N=30, N=40 and N=50 are not significantly different, indicating similar convergence levels.

Population size	Hyper-volume	Jaccard Index
20	0.95±0.35	0.88±0.01
30	**1.60±0.15**	**0.92 ±0.01**
40	1.57±0.12	0.910±0.005
50	1.50±0.12	0.910±0.005

Table 5: Hyper-volume metric for NSGA-II execution using different population sizes and 50 generations

4.4. Computational Efectiveness

Feature selection method aim to compute feature sets discriminative enough for all the classes in an acceptable amount of time. Thus, processing time is still an issue as exponential complexities limit the usefulness of some approaches for problems above a given dimension, but whenever this dimension is higher than the one corresponding to real problems, it is better to have a procedure that is able to provide a good solution of the problem in acceptable amount of time than a procedure that obtains a not so good solution, although very fast. Subsequently, it is important to characterize the space of feature selection procedures where we have a certain kind of multi-objective problem where the cost function are the computing time and the solution quality. The issue is to determine the non-dominant solutions corresponding to high quality feature selections although it could require more computing time than obtaining not so good feature selections. To complete



(a) (b)

Figure 13: Average Jaccard index (a) and processing time for $J = 0.88$ (b) for different population sizes with the corresponding standard deviation values. (Note that standard deviation values are not apparent in (b) as they are small in comparison with the mean values)

the characterization of our wrapper procedure, we have obtained details about their computing time requirements, including a comparison with other filter and wrapper methods. Thus, Table 6 shows the CPU time required for selecting the best feature set provided by each technique and using GHSOM-pr as classifier. Moreover, the training set has been used to determine the number of features providing the best classification results in terms of the Jaccard index.
It is worth noting that we evaluated the processing time required to compute acceptable solutions in terms of the mean Jaccard index computed for all classes, as it accounts for multi-class classification performance. Figure 13 shows that a population of 30 individuals provide a good trade-off between performance and complexity, as increasing the number of individuals (i.e. N=40) imply a considerably higher processing time. In this figure, standard deviations according to values in Table 6 are shown.

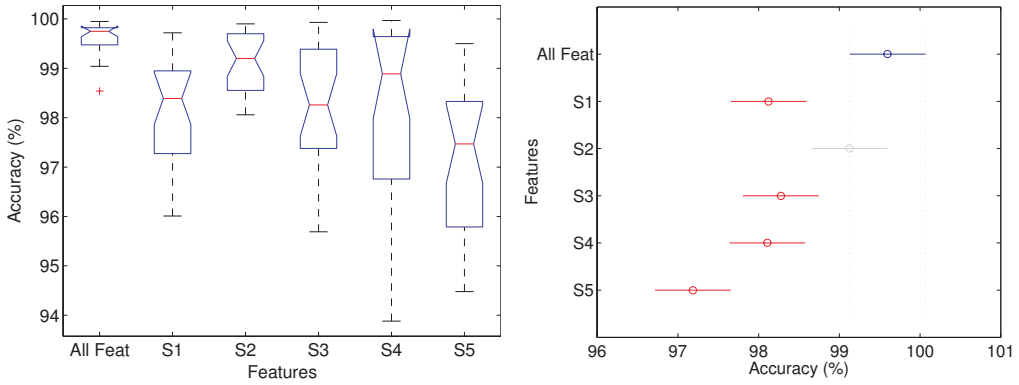
Feature Selection method	Processing time (s)	Jaccard Index
Filter methods		
PCA	796±20	0.83±0.04
FDR	597±3	0.74±0.03
ReliefF	32112±32	0.92±0.04
Wrapper methods		
BackwardFS	17194±121	0.82±0.07
Multiobjective Feature Selection	**8084±180**	**0.92±0.01**

Table 6: Computational complexity and accuracy for different feature selection methods. GHSOM-pr classifier is used in all cases for comparison. Average Jaccard index along all the classes is shown

4.5. Statistical Significance

Due to the variability imposed by both the GHSOM pseudo-random initialisation process and the evolutionary algorithm used for multi-objective optimisation, classification outcomes may vary among different runs. In this way, in Tables 3 and 4 respectively, the mean results corresponding to feature reduction by PCA and the multi-objective optimisation method are provided with the standard deviation computed over 50 training/classification algorithm runs. Moreover, statistical significance tests are necessary in order to guarantee that mean values obtained by different experiments are different. This has been addressed through hypothesis testing under the assumption that results from the different experiments are drawn from a distribution with the same average (null hypothesis). While ANOVA [61] is used to find significant differences among group means, multiple comparison tests try to identify the specific groups whose means are significantly different.

ANOVA has been performed on results obtained using multi-objective optimisation to reduce the feature space dimension; these results are presented in Figure 14. The *p-value* provided by the ANOVA analysis ($p < 10^{-7}$) indicates that all the means are significantly different. However, a multi-comparison test reveals that results obtained using non-dominated feature sets S1, S2, S3, S4 and S5 are not statistically different for normal/attack detection, providing the same accuracy.



(a) (b)

Figure 14: ANOVA test for experimental results using multi-objective optimisation (a) and multiple comparison tests to identify the source of mean differences (b).

5. Related works

This section comments on the work done vis-à-vis the paper's main contributions. The feature selection for network-based IDS is not straightforward and a number of papers have been published dealing with this topic. There are works [52, 62, 63] that use multivariate techniques such as PCA [52] or LDA [62], thus supposing that the other components are not important or are just noise. Although good results are obtained in [52, 62, 63], other works show that using the full set of features outperforms using feature selection with PCA/LDA [24]. This way, feature reduction has to be applied to each input vector since the most

discriminative component depends on the specific attack. Thus, in [15], an SOM classifier for attack detection is presented without reducing the feature space. However, the authors use a selection of 28 features from the 41 available ones in the KDD-NSL dataset. In this work, the influence of each feature is figured through each component's U-Matrix [37] on the input (feature) space.

In [40], the feature selection, along with its interaction with the partition of training data, and the number of layers of an SOM hierarchy are analysed. They concluded that a two-layer SOM hierarchy based on the 41 features of the KDD set is the better choice. A detection rate of 90.4% and a false positive rate of 1.38% are reported.

Other proposals, such as [47] use Gaussian Mixture Models (GMM) to model SOM unit activation. This provides an activation level for each SOM unit and allows tuning the map response by modifying the prior activation probabilities. However, the labels are assigned by simple majority voting and are static. Moreover, the map can not be oversized as dead units have to be necessarily avoided (that is, the activation likelihood of a dead unit is zero, and its posterior activation level is also zero). On the other hand, the GMM method in [47] is used to model the SOM and aims to differentiate between *normal* and *abnormal* samples. In GHSOM sub-maps (where this method could be applied independently), samples are small enough to make the GMM method unfeasible. On the other hand, probabilistic relabelling in [34] only takes into account a priori probabilities to label any unlabelled unit (that is, uses the most frequent label in the neighbourhood). However, the method proposed in this paper uses posterior probabilities through the Bayes' formula to compute the most probable label, according to equation 10. This constitutes an important improvement, as computing unit labels depends not only on the map labels, but also on the current sample, assigning the label of each BMU in an adaptive way. Moreover, the relabelling method in this work computes the BMU label for each new sample, and not only for BMUs that are never activated during the training stage.

As commented in the introduction, computing an average score on different dataset classes may lead to removing features from the final selection that could be specially relevant for a certain class label. Thus, it is necessary to consider methods that evaluate each feature's discriminative power, selecting those that best describe each individual class. Consequently, the method proposed in [57] uses an ensemble of specialised SVC classifiers, each trained using specific features computed for each anomaly type. Nevertheless, the experiments performed in [57] reveal that neither linear nor non-linear projection techniques are able to find discriminative enough features. Thus, some works have been proposed in the formulation of feature selection as a multi-objective optimisation problem, either as supervised or unsupervised classifiers. A very good review of the alternatives and previous references on this topic is the paper by J. Handl and J. Knowles [2]. With respect to supervised classifiers, in [64], multi-objective feature selection procedures that take into account the number of features and the performance of the classifier are provided. In [65], a variation of the NPGA algorithm [66] for multi-objective optimisation is applied, whilst [67] uses NSGA [68] and [64] uses NSGA-II. In this paper, we have also used NSGA-II, as in [64]. In our case, a similarity measurement between the predicted labels and the ground truth for each anomaly type is treated as a different objective to be maximised jointly. Al-

though the number of features is not taken into account as an objective function, our results show that subsets corresponding to non-dominated solutions contain reduced, but discriminative feature subsets for each class label. Thus, the approach proposed in [64] does not take into account each class label independently in the selection process, and it uses k-NN as the wrapper procedure classifier instead of GHSOM, as in our case. In [13], a feature selection framework is proposed that takes into account the relevance and redundancy of the selected features for the different class labels. Nevertheless, that paper does not approach the problem as a Pareto-front searching. From a multi-objective optimisation perspective, as in our proposal, the Jaccard's coefficients for the different labels (used as a classifier's performance measure for each label) are the objectives of the problem considered. In [69] NSGA-II, along with other proposed procedures, is applied to select the global non-dominated feature subsets for customer churn prediction in telecommunications. In this multi-objective feature selection procedure, there are three cost functions: the proportion of the total number of correct predictions, the proportion of churn cases that were correctly identified, and the proportion of non-churn cases that were correctly identified. This kind of information is considered in our approach, but through a Jaccard's coefficient for each label. In our case, the objectives are precisely the Jaccard's coefficients and we have as many objectives as labels. Moreover, the classification in [69] is done through decision tree C4.5 [70], so it corresponds to a different approach to that described herein, which is based on Self-Organizing Maps.

In the case of unsupervised classification, we have the papers [2, 71, 72]. In [71], given a feature selection, the k-means algorithm is used to build a clustering and is evaluated via four objectives (number of features, number of clusters, compactness of the clusters, and separation between clusters). The paper [72] also uses k-means for clustering and the number of features and the Davies-Boulding Index (DBI) [73]. Along with a critical review of papers [71] and [72], and an experimental study of different alternatives for unsupervised feature selection with multi-objective optimisation, the paper [2] provides a strategy to select (without external knowledge) the most adequate solution from the obtained Pareto front approximation.

In this paper, we propose the use of a dynamic structure, GHSOM-pr, based on the previously proposed GHSOM, and a new approach for dimensionality reduction through feature selection based on multi-objective optimisation. Some works on GHSOM for IDS design have been previously proposed [42, 36, 19]. After the results provided by [42], some enhancements in the GHSOM procedure have been proposed in [36, 19]. In [36], a new metric including numerical and symbolic data is introduced along with a procedure to allow an automatic map growth control that avoids the use of parameter τ_1. The paper shows that the improved GHSOM is better than an SOM used as the base map configuration and provides detection rates below 96.9%. The same strategy is considered in [19], where four GHSOM enhancements are proposed. These enhancements are related to training, input normalisation, adaptation of the quantisation error, and a mechanism for confidence filtering and forwarding. The resulting hierarchical organising map, called A-GHSOM, provides an overall accuracy of 99.63% and a false positive rate of 1.8%.

With respect to those works, we have proposed using a dynamic structure

that includes a new probabilistic relabelling in the previously proposed clustering procedure GHSOM, along with a multi-objective approach for feature selection. These tools have been applied to the IDS design; the results obtained outperform those provided by all the considered approaches except those of false positive rates shown in [19]. In this case, there is only a 0.44 % of difference in false positive rate.

6. Conclusions

In this paper, we present an intrusion detection approach that takes advantage of the discriminating properties of Self-Organizing Maps. More specifically, we have considered GHSOM where we have introduced a relabelling procedure that improves its intrusion detection and prevention performance. This clustering procedure allows a better labelling of the incoming data by taking into account the clusters found by the previously trained GHSOM. It also includes a multi-objective procedure based on NSGA-II algorithm for feature selection in order to reduce the complexity of the GHSOM and to improve the classification performance.

The experiments performed with the KDD-NSL dataset show the relabelling method efficiency through the corresponding ROC curves and the classification rate improvements. Thus, the results obtained with GHSOM and the proposed relabelling method reaches a detection rate of 99.4% for normal patterns, and 99.2% for attack connections. A reduction in the GHSOM size that is able to provide a given classification rate is important for the computational efficiency of the intrusion detection approach. Thus, if normal and anomalous behaviours are accurately detected with lightweight processing, it is possible to perform other actions, such as IP blocking, in real time. An approach to achieve this goal is to select the most suitable set of features instead of using the whole feature set. For the most adequate set of features, the Jaccard index for each type of attack, evaluated after training the GHSOM, is used as one of the objectives of a multi-objective procedure based on the NSGA-II. In our experiments, we have considered five different non-dominant solutions (sets of selected features) that belong to the obtained Pareto front. These five solutions have been chosen in such a way that they optimise the Jaccard index for the normal traffic situations and one of the attack types (NORMAL, DOS, PROBE, U2R, and R2L). The results obtained on the KDD-NSL show that the feature set that maximises the Jaccard index for the U2R attack, along with the relabelling procedure applied in the GHSOM training procedure, provides detection rates up to 99.8% for normal traffic, and up to 99.4% for anomalous traffic with five levels and 225 neurons. The detection accuracy achieved by the proposed procedure is 99.12%, thus improving the results obtained by the considered proposed procedures (Table 4).
In future work, we plan to analyse how the current GHSOM could be improved by hybridising it with other improved clustering techniques, such as the Gaussian Mixture Model [22] or using Support Vector Machines (SVM). On the other hand, alternatives for efficiently implementing intrusion prevention systems by using optimised kernel modules in computers with several processors and/or programmable network interface cards (for example, including network processors) will also be analysed. We consider that taking advantage of the parallelism present on the

current processing nodes will improve IPS performance, thus enabling efficient active intrusion prevention systems.

Acknowledments

This work has been funded by FEDER funds and the Ministerio de Ciencia e Innovación of the Spanish Government under Project No. TIN2012-32039. The authors would like to thank the reviewers for their suggestions and comments to improve the paper.

References

[1] S. Theodoridis, K. Koutroumbas, Pattern Recognition, Academic Press, 2009.

[2] J. Handl, J. Knowles, Feature subset selection in unsupervised learning via Multiobjective Optimization, International Journal of Computational Intelligence 2 (3) (2006) 217–238.

[3] T. Hastie, T. Tibshirani, J. Friedman, The elements of Ststistical Learning. Data mining, Inference and Prediction., second edition Edition, Springer, 2009.

[4] M. Turk, A. Pentland, Eigenfaces for recognition, J. Cognitive Neuroscience 3 (1) (1991) 71–86.

[5] P. Belhumeur, J. Hespanha, D. Kriegman, Eigenfaces vs. fisherfaces: Recognition using class specific linear projection, IEEE Trans. Pattern Anal. Mach. Intell. 19 (7) (1997) 711–720.

[6] A. Hyvärinen, E. Oja, Independent component analysis: Algorithms and applications, Neural Netw. 13 (4-5) (2000) 411–430.

[7] W. Navidi, Statistics for Engineers and Scientists, third edition Edition, McGraw-Hill, 2010.

[8] J. Quinlan, Induction of decision trees, Machine learning (1986) 81–106.

[9] F. Fleuret, Fast binary feature selection with conditional mutual information, J. Mach. Learn. Res. 5 (2004) 1531–1555.

[10] G. Wang, F. H. Lochovsky, Feature selection with conditional mutual information maximin in text categorization, in: Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management, CIKM '04, ACM, New York, NY, USA, 2004, pp. 342–349.

[11] M. A. Hall, Correlation-based feature selection for discrete and numeric class machine learning, in: Proceedings of the Seventeenth International Conference on Machine Learning, ICML '00, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2000, pp. 359–366.

[12] L. Yu, H. Liu, Efficient feature selection via analysis of relevance and redundancy, J. Mach. Learn. Res. 5 (2004) 1205–1224.

[13] Y. Zhang, S. Li, T. Wang, Z. Zhang, Divergence-based feature selection for separate classes, Neurocomput. 101 (2013) 32–42.

[14] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimisation: NSGA-II, in: Proceedings of the 6th International Conference on Parallel Problem Solving from Nature, PPSN VI, Springer-Verlag, London, UK, UK, 2000, pp. 849–858.

[15] H. Oh, I. Doh, K. Chae, Attack Classification Based on Data Mining Technique and Its Application for Reliable Medical Sensor Communication, International Journal of Computer Science & Applications 6 (3) (2009) 20–32.

[16] A. Patcha, J.-M. Park, An overview of anomaly detection techniques: Existing solutions and latest technological trends, Computer Networks 51 (12) (2007) 3448–3470.

[17] J. McHugh, Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory, ACM Transactions on Information and System Security 3 (4) (2000) 262–294.

[18] W. Khreich, E. Granger, A. Miri, R. Sabourin, Adaptive ROC-based ensembles of HMMs applied to anomaly detection, Pattern Recognition 45 (1) (2012) 208–230.

[19] D. Ippoliti, X. Zhou, A-GHSOM: An adaptive growing hierarchical self organizing map for network anomaly detection, Journal of Parallel and Distributed Computing 72 (12) (2012) 1576–1590.

[20] The KDD-NSL dataset. Last accessed on 04/28/2014.

[21] Uriarte, E.A., F. Martín, Topology Preservation in SOM, Journal of Mathematical and Computer Sciences 19 (22).

[22] M. Bahrololum, M. Khaleghi, Anomaly Intrusion Detection System Using Gaussian Mixture Model, in: Third International Conference on Convergence and Hybrid Information Technology, Vol. 1, 2008, pp. 1162–1167.

[23] M. Tavallaee, E. Bagheri, W. Lu, A. Ghorbani, A detailed Analysis of the KDDCup 1999 dataset, in: Proceedings of the IEEE International symposium on Computational Intelligence in Security and defense Applications (CISDA-2009), 2009, pp. 1–6.

[24] S. Mukkamala, A. H. Sung, Feature Ranking and Selection for Intrusion Detection Systems Using Support Vector Machines, in: Proceedings of the Second Digital Forensic Research Workshop, 2002, pp. 1–10.

[25] J. Z. Lei, A. A. Ghorbani, Improved competitive learning neural networks for network intrusion and fraud detection, Neurocomputing 75 (1) (2012) 135–145.

[26] T. Wang, S. Mabu, N. Lu, K. Hirasawa, A novel intrusion detection system based on the 2-dimensional space distribution of average matching degree, in: Proceedings of SICE Annual Conference, 2011, pp. 2829–2834.

[27] A. Hofmann, C. Schmitz, B. Sick, Intrusion detection in computer networks with neural and fuzzy classifiers, in: Proceedings of the 2003 Joint International Conference on Artificial Neural Networks and Neural Information Processing, ICANN/ICONIP'03, Springer-Verlag, Berlin, Heidelberg, 2003, pp. 316–324.

[28] P. Lichodzijewski, A. Nur Zincir-Heywood, M. Heywood, Host-based intrusion detection using self-organizing maps, in: Proceedings of the 2002 International Joint Conference on Neural Networks, Vol. 2, 2002, pp. 1714–1719.

[29] C. Zhang, J. Jiang, M. Kamel, Intrusion detection using hierarchical neural networks, Pattern Recognition Letters 26 (6) (2005) 779–791.

[30] D. Fisch, A. Hofmann, B. Sick, On the versatility of radial basis function neural networks: A case study in the field of intrusion detection, Information Sciences 180 (12) (2010) 2421–2439.

[31] D. Fisch, B. Sick, Training of radial basis function classifiers with resilient propagation and variational Bayesian inference, in: International Joint Conference on Neural Networks, 2009, pp. 838–847.

[32] M. Bauer, O. Buchtala, T. Horeis, R. Kern, B. Sick, R. Wagner, Technical data mining with evolutionary radial basis function classifiers, Applied Soft Computing 9 (2) (2009) 765–774.

[33] S. T. Powers, J. He, A hybrid artificial immune system and Self Organising Map for network intrusion detection, Information Sciences 178 (15) (2008) 3024–3042.

[34] A. Ortiz, J. Ortega, A. F. Díaz, A. Prieto, Network intrusion prevention by using hierarchical self-organizing maps and probability-based labeling, in: J. Cabestany, I. Rojas, G. Joya (Eds.), Advances in Computational Intelligence, Vol. 6691 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2011, pp. 232–239.

[35] S. Sen, J. A. Clark, Evolutionary computation techniques for intrusion detection in mobile ad hoc networks, Computer Networks 55 (15) (2011) 3441–3457.

[36] Y. Yang, D. Jiang, M. Xia, Using improved GHSOM for Intrusion Detection, Journal for Information Assurance and Security 5 (2009) 232–239.

[37] K. T., Self-Organizing Maps, third edition Edition, Springer, 2001.

[38] S. Sen, J. A. Clark, Evolutionary computation techniques for intrusion detection in mobile ad hoc networks, Computer Networks 55 (15) (2011) 3441–3457.

[39] M. Govindarajan, R. Chandrasekaran, Intrusion detection using neural based hybrid classification method, Computer Networks 55 (8) (2011) 1662–1671.

[40] H. Gunes Kayacik, A. Nur Zincir-Heywood, M. I. Heywood, A hierarchical SOM-based intrusion detection system, Engineering Applications of Artificial Intelligence 20 (4) (2007) 439–451.

[41] A. Rauber, D. Merkl, M. Dittenbach, The Growing Hierarchical Self-Organizing Map: Exploratory Analysis of High-Dimensional Data, IEEE Transactions on Neural Networks 13 (2002) 1331–1341.

[42] E. J. Palomo, E. Domínguez, R. M. Luque, J. Muñoz, Network security using growing hierarchical self-organizing maps, in: Proceedings of the 9th international conference on Adaptive and natural computing algorithms, ICANNGA'09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 130–139.

[43] R. Kohavi, G. John, Wrappers for feature subset selection, Artificial Intelligence 97.

[44] N. Abe, M. Kudo, J. Toyama, M. Shimbo, Classifier-independent feature selection on the basis of divergence criterion, Pattern Analysis and Applications 9.

[45] tcpdump packet analyzer. Last accessed on 04/28/2014, `http://tcpdump.org`.

[46] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.

[47] E. de la Hoz, E. de la Hoz, A. Ortiz, J. Ortega, Network anomaly detection with Bayesian self-organizing maps, in: Proceedings of the International Work-Conference on Artificial Neural Networks (IWANN), LNCS 7092, Springer-Verlag, 2013, pp. 532–537.

[48] KDD Cup 1999 dataset. Last accessed on 04/28/2014, `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`.

[49] R. Lippmann, D. Fried, I. Graf, J. Haines, K. Kendall, D. McClung, D. Weber, S. Webster, D. Wyschogrod, R. Cunningham, M. Zissman, Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation, in: DARPA Information Survivability Conference and Exposition, 2000. DISCEX '00. Proceedings, Vol. 2, 2000, pp. 12–26.

[50] M. Panda, A. Abraham, M. Patra, Discriminative multinomial Naïve Bayes for network intrusion detection, in: Sixth International Conference on Information Assurance and Security (IAS), 2010, pp. 5–10.

[51] J.-P. Nziga, Minimal dataset for Network Intrusion Detection Systems via dimensionality reduction, in: Digital Information Management (ICDIM), 2011 Sixth International Conference on, 2011, pp. 168–173.

[52] S. Lakhina, S. Joseph, B. Verma, Feature Reduction using Principal Component Analysis for Effective Anomaly–Based Intrusion Detection on NSL-KDD, International Journal of Engineering Science and Technology 2 (6) (2010) 1790–1799.

[53] M. Tavallaee, N. Stakhanova, A. Ghorbani, Toward credible evaluation of anomaly-based intrusion-detection methods, IEEE Transactions on Systems, Man, and Cybernetics, Part C 40 (5) (2010) 516–524.

[54] H. Kayacik, A. Zincir-Heywood, M. Heywood, A hierarchical SOM-based intrusion detection system, Journal Engineering Applications of Artificial Intelligence 20 (4) (2007) 439–451.

[55] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, S. Stolfo, A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data, in: Applications of Data Mining in Computer Security, Kluwer, 2002, pp. 77–101.

[56] Z. Yu, J. J. P. Tsai, T. Weigert, An adaptive automatically tuning intrusion detection system, ACM Transactions on Autonomous and Adaptive Systems 3 (3) (2008) 10:1–10:25.

[57] E. de la Hoz, A. Ortiz, E. de la Hoz, J. Ortega, Network Anomaly Classification by Support Vector Classifiers Ensemble and Non-Linear Projection Techniques, in: Proceedings of the International International Conference on Hybrid Artificial Intelligence Systems (HAIS), LNAI 8073, Springer-Verlag, 2013, pp. 103–111.

[58] S. S. Sivatha Sindhu, S. Geetha, A. Kannan, Decision tree based light weight intrusion detection using a wrapper approach, Expert Syst. Appl. 39 (1) (2012) 129–141.

[59] H. T, On the convergence of multiobjective evolutionary algorithms, European Journal of Operational Research 117.

[60] E. Zizler, L. Thiele, Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach, IEEE Transactions on Evolutionary Computation 3 (4) (1999) 257–271.

[61] W. Navidi, Statistics for Engineers and Scientists, third edition Edition, McGraw-Hill, 2010.

[62] R. Datti, B. verma, Feature Reduction for Intrusion Detection Using Linear Discriminant Analysis, International Journal on Computer Science and Engineering 2 (4) (2010) 1072–1078.

[63] G. R. Zargar, P. Kabiri, Selection of effective network parameters in attacks for intrusion detection, in: Proceedings of the 10th Industrial Conference on Advances in Data Mining: applications and theoretical aspects, ICDM'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 643–652. doi:10.1007/978-3-642-14400-4_50.

[64] T. Hamdani, J.-M. Won, A. A.M., F. Karray, Multi-objective feature selection with nsga ii, in: Proceedings of the 8th International Conference on Adaptive and Natural Computing Algorithms (ICCANGA), LNCS 4431, Springer, 2007, pp. 240–247.

[65] C. Emmanouilidis, A. Hunter, J. MacIntyre, A multiobjective evolutionary setting for feature selection and a commonality-based crossover operator, in: Proceedings of the 2000 Congress on Evolutionary Computation, IEEE Press, New York, 2000, pp. 209–316.

[66] J. Horn, N. Nafpliotis, D. E. Goldberg, A niched pareto genetic algorithm for multiobjective optimization, in: Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on, 1994, pp. 82–87 vol.1.

[67] L. Oliveira, R. Sabourin, F. Bortolozzi, C. Suen, A methodology for feature selection using multiobjective genetic algorithms for handwritten digit string recognition, International Journal of Pattern Recognition and Artificial Intelligence 17 (6) (2003) 903–929.

[68] N. Srinivas, K. Deb, Multiobjective optimization using nondominated sorting in genetic algorithms, Evol. Comput. 2 (3) (1994) 221–248.

[69] B. Huang, B. Buckley, T. M. Kechadi, Multi-objective feature selection by using nsga-ii for customer churn prediction in telecommunications, Expert Syst. Appl. 37 (5) (2010) 3638–3646.

[70] J. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo, CA, USA, 1993, 1993.

[71] Y. Kim, W. Street, F. Menczer, Evolutionary model selection in unsupervised learning, Intelligent Data Analysis 6 (6) (2002) 531–556.

[72] M. Morita, R. Sabourin, F. Bortolozzi, C. Suen, Unsupervised feature selection using multi-objective genetic algorithms for handwritten word recognition, in: Proceedings of the Seventh International Conference on Document Analysis and Recognition, IEEE Press, New York, 2003, pp. 666–670.

[73] J. Davies, D. Bouldin, A cluster separation measure, IEEE Transactions on Pattern Analysis and Machine Intelligence 1 (1979) 224–227.