COMPUTERS & SECURITY 70 (2017) 255–277

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

A GA-LR wrapper approach for feature selection in network intrusion detection ☆

Chaouki Khammassi [a,b,*], *Saoussen Krichen* [a]

[a] *LARODEC Laboratory, Institut Supérieur de Gestion, Université de Tunis, 41 Rue de la liberté, Le Bardo 2000, Tunisia*

[b] *Institut Supérieur de Commerce et de Comptabilité, Université de Carthage, Bizerte 7000, Tunisia*

ARTICLE INFO

Article history:
Received 2 January 2017
Received in revised form 18 May 2017
Accepted 6 June 2017
Available online 15 June 2017

Keywords:
Intrusion detection systems
Anomaly detection
Feature selection
Wrapper approach
Genetic algorithm
Logistic regression
Classification
Decision tree
KDD99
UNSW-NB15

ABSTRACT

Intrusions constitute one of the main issues in computer network security. Through malicious actions, hackers can have unauthorised access that compromises the integrity, the confidentiality, and the availability of resources or services. Intrusion detection systems (IDSs) have been developed to monitor and filter network activities by identifying attacks and alerting network administrators. Different IDS approaches have emerged using data mining, machine learning, statistical analysis, and artificial intelligence techniques such as genetic algorithms, artificial neural networks, fuzzy logic, swarm intelligence, etc. Due to the high dimensionality of the exchanged data, applying those techniques will be extremely time consuming. Feature selection is needed to select the optimal subset of features that represents the entire dataset to increase the accuracy and the classification performance of the IDS. In this work, we apply a wrapper approach based on a genetic algorithm as a search strategy and logistic regression as a learning algorithm for network intrusion detection systems to select the best subset of features. The experiment will be conducted on the KDD99 dataset and the UNSW-NB15 dataset. Three different decision tree classifiers are used to measure the performance of the selected subsets of features. The obtained results are compared with other feature selection approaches to verify the efficiency of our proposed approach.

1. Introduction

A myriad of data have been created due to the widespread use of computers worldwide and the massive exchange of data supported by communications tools such emails and social networks. This has created a new paradigm named big data (Zhang et al., 2015). As we know, compromising the security of computerised system is becoming easier today due to the abundance of free hacking tools widespread over the Internet. Users do not need high skills to apply such malicious actions (Louvieris et al., 2013). Ensuring the integrity and the privacy of these data is becoming a real challenge. There are many software tools such as firewalls, antivirus, encryption, and authentication that provide the protection of data and networks from the incoming threats, yet they cannot be efficient for all existing threats (Chung and Wahid, 2012). The intrusion to a private network can wreak havoc within the whole system. Many enterprises lost their business and their credibility with their clients because of hacking actions such as stealing clients account passwords or getting sensitive information. To resolve this problem, many researches have been

☆ This work was done in the context of a master thesis in Computer Science Option Enterprise Systems Engineering.

* *Corresponding author.*
E-mail address: chaouki.khammassi@gmail.com (C. Khammassi).

http://dx.doi.org/10.1016/j.cose.2017.06.005
0167-4048/

conducted in this field. Intrusion detection systems (IDSs) have been developed to monitor and filter network activities by identifying attacks and alerting network administrators (Chung and Wahid, 2012). There are two main approaches for IDS: *misuse* and *anomaly* detection techniques. Neither is efficient for all kinds of threats, yet each has strengthens and limitations (Lin et al., 2015). The misuse detection approach is efficient for detecting known attacks but not for the unseen ones (Zhang et al., 2015). In contrast, the anomaly detection approach is efficient for detecting new attacks, but they are flawed by a high false positive rate (Kim et al., 2014). For that reason some IDSs use a hybrid approach that integrates both misuse and anomaly detection techniques (Depren et al., 2005).

In the literature, most of research in intrusion detection focused on the anomaly detection technique because it seems more promising (Gan et al., 2013; Karami and Guerrero-Zapata, 2015). Recently, many approaches emerged in the context of anomaly detection including data mining, machine learning, statistical analysis, and artificial intelligence techniques (Rastegari et al., 2015). The use of machine learning techniques in IDSs can be single, hybrid, or ensemble classifiers. The used classifier can be categorised in three operating modes: supervised, semi-supervised, and unsupervised. Generally, supervised mode outperforms the remaining modes (Garcia-Teodoro et al., 2009). In the anomaly detection, several machine learning algorithms can be used such as Fuzzy Logic, Nave Bayes, Artificial Neural Network, Support Vector Machine, Decision Trees, k-means, k-Nearest Neighbour, Genetic Algorithm, and Logistic Regression (Garcia-Teodoro et al., 2009; Ghorbani et al., 2009; James et al., 2013; Tsai et al., 2009).

Due to the high dimensionality of the exchanged data, applying those techniques would be extremely time consuming especially when we are looking for real time intrusion detection. For that reason feature selection, which is part of dimensionality reduction, is needed to select the optimal subset of feature that represents the entire dataset (Eesa et al., 2015). Thus, the size of data is reduced and the classification performance and accuracy is increased (Chandrashekar and Sahin, 2014). There are two main categories for feature selection: *filters* and *wrappers*. Filters are applied through statistical methods, information theory-based methods or searching techniques (De la Hoz et al., 2014) such as Principal Components Analysis, Information Gain, and Correlation-based Feature Selection (Boln-Canedo et al., 2016). Wrappers use machine learning algorithm to assess and classify features so as to identify the subset that mostly represent the dataset. They are based on two components: a feature search (such as branch and bound (Chandrashekar and Sahin, 2014), sequential search, genetic search (Guyon et al., 2008), etc.) and a learning algorithm which can be any classifier. In general, the filter method is less computational compared to the wrapper method, but the latter produces best results (De la Hoz et al., 2015).

In this work, we propose a feature selection approach for IDS to produce the optimal subset of features. The proposed approach is based on the wrapper approach integrating Genetic Algorithm (GA) as a feature search and Logistic Regression (LR) as a learning algorithm. GA is a popular and powerful algorithm inspired from the concept of natural selection to find the fittest solution by applying genetic operators (Ghorbani et al., 2009). It is widely used in different fields and can be used in intrusion detection by selecting the appropriate network features to improve the IDS accuracy (Garcia-Teodoro et al., 2009; Modi et al., 2013). LR is a supervised learning algorithm widely used in classification (James et al., 2013). The use of LR in intrusion detection is not frequent (Tsai et al., 2009) even if it can give good results in other domains such as banking, marketing, medicine, etc. This does not mean that LR cannot have good results in intrusion detection.

Despite the fact that the KDD99 dataset and its derived versions have been widely used in the IDS performance evaluation for many years (Kang and Kim, 2016), they are outdated and did not contain the contemporary attacks (Haider et al., 2017). The UNSW-NB15 is a recently created dataset by the cyber security research group at the Australian Centre for Cyber Security (ACCS) (Moustafa and Slay, 2015). This latter is considered as a new benchmark dataset that can be used for IDSs evaluation by the NIDS research community (Moustafa and Slay, 2016). Therefore, in this work we used the KDD99 and the UNSW-NB15 datasets for the evaluation of our proposed approach. Thus, we can compare the obtained results with the old and the new benchmark datasets.

The main objective of the GA-LR wrapper approach is to find the subset of features having the maximum accuracy of classification and containing the minimum number of features. The classification stage is performed with a family of three decision tree classifiers namely C4.5, Random Forest (RF), and Naive Bayes Tree (NBTree) for measuring the performance of the resulting subsets of features with the 10% KDD99 dataset and the training UNSW-NB15 dataset. As we know, the major drawback of the KDD99 dataset is the existence of a huge quantity of repeated records that affects any machine learning by biasing towards frequent records (Elhag et al., 2015; Sindhu et al., 2012). Furthermore, the KDD99 and the UNSW-NB15 datasets contain attribute values that cannot be handled with most of learning algorithms. Thus, a preprocessing stage is needed for removing all redundant records, resampling, and transforming attributes (Kang and Kim, 2016).

This paper is organised as follows. Section 2 presents an overview of the network intrusion detection problems such as network attacks, security mechanisms, and the existing approaches for IDSs. Section 3 presents an overview of the existing dimensionality reduction techniques and their application in the context of IDSs. Section 4 illustrates related work relevant to the context of feature selection in intrusion detection. Section 5 presents the three stages for the proposed approach for IDS: the preprocessing stage that changes the structure of the used datasets to be handled with the used classifiers, the feature selection stage based on the wrapper approach including GA and LR to select the best subset of features, and the classification stage performed using the three decision trees with the proposed subsets of features. Section 6 describes the conducted experiments and summarises the results of the proposed GA-LR wrapper and the used decision tree classifiers.

2. Network intrusion detection: an overview

Intrusions constitute one of the main issues in computer network security (Costa et al., 2015). Through malicious actions,

hackers can have unauthorised access that compromises the integrity, the confidentiality, and the availability of a resource. Of course, stealing, changing, or destroying data will automatically affect the reliability of the whole system (Rastegari et al., 2015). Any type of networks starting from LANs to cloud computing suffers from several types of attacks that can exploit systems vulnerabilities to reach a specific purpose, such as IP spoofing, Denial of Service (DoS), flooding, User to Root (U2R), Remote to Local (R2L), port scanning, probes, etc. (Ghorbani et al., 2009; Modi et al., 2013).

Network anomaly detection is a research field that started almost 40 years ago (Dangelo et al., 2015). Its main purpose is to create the perfect system that can detect and stop all intrusions and attacks to guarantee the reliability and the efficiency of the system (Rastegari et al., 2015), but this seems hard to reach with the continuous evolution of the hacking tools. The existing tools such as firewalls, antiviruses, and encryption are not efficient for all types of malwares and attacks. DoS attacks are too complex for firewalls to distinguish from normal traffic (Bulajoul et al., 2015). Intrusion detection systems (IDSs) have been developed to scan and monitor network traffic to find any malicious activities and then alert the network administrators (Rastegari et al., 2015). On the other hand, intrusion prevention system (IPS) has all the capabilities of an intrusion detection system and can block intrusions. Thus, an IPS is considered an extension of an IDS. However nowadays there is a fusion between those two systems and most IDSs include detection and prevention to give what we call an intrusion detection and prevention system (IDPS). To increase the security of the network, combining a firewall with an IDPS will be extremely useful to strengthen the network (Bulajoul et al., 2015).

IDSs operate through two main approaches: *misuse* and *anomaly* detection techniques. In the misuse detection approach (signature-based detection), the system compares every received packets from data stream with known signatures that have been already existing as patterns or rules. If there is a mismatch, the system raises an alert. The misuse detection approach is efficient for detecting known attacks but not for the unseen ones (Zhang et al., 2015). Furthermore, any mistake in the definition of the signatures will decrease the detection rate and inversely increase the false alarm rate. The anomaly detection approach is based on the concept that attackers behave differently from the normal user. For that the system creates a profile for normal traffic. If the received packet characteristic is too different from the normal traffic pattern, the system raises an alert. The anomaly detection approach is efficient for detecting new attacks, but they are flawed by a high false alarm rate (Kim et al., 2014). In order to avoid the disadvantages of these two approaches, some researchers proposed hybrid intrusion detection approach (Depren et al., 2005; Kim et al., 2014), while the others focused on the anomaly detection approach because it seems more promising than the misuse detection approach (Gan et al., 2013; Karami and Guerrero-Zapata, 2015).

In recent years, different approaches of IDSs based on anomaly detection have emerged using data mining, machine learning, statistical analysis, and artificial intelligence techniques such as genetic algorithms, artificial neural networks, fuzzy logic, swarm intelligence, etc. (Rastegari et al., 2015). The anomaly detection techniques can be classified in three operating modes: supervised, semi-supervised, and unsupervised. Supervised methods need a labelled training set to construct the model. In contrast, unsupervised methods do not require either a labelled training data or a prior knowledge of instances. They classify instances according to statistical models. The semi-supervised method is between supervised and unsupervised methods. It uses both unlabelled data and small labelled data which reduces the labelling cost while keeping the high performance of supervised methods. Basically, supervised methods outperform the remaining methods, but it is not the case all the time (Ghorbani et al., 2009).

As the number of attacks is increasing and the complexity is growing, manually building a model for intrusion detection is costly and time consuming. In supervised learning, the anomaly detection is performed through two stages: training stage and detection stage. In the training stage the normal behaviour of the system is determined and a model is established. Machine learning techniques have the ability to construct and maintain these models automatically with less human intervention by using a training dataset. A training dataset is composed of a collection of data instances. Each instance contains a set of features and associated labels often done manually. On the other hand, the detection stage consists of comparing the testing dataset with the available model. If there is a deviation that exceeds a fixed threshold, an alarm is raised (Garcia-Teodoro et al., 2009).

According to Tsai et al. (2009), machine learning techniques can be classified as follows: single classifier, hybrid classifier, and ensemble classifier. In the following, we present the major machine learning techniques and approaches that can be used in intrusion detection.

2.1. *Single classifier*

In the literature, the intrusion detection problems have been solved by applying a single machine learning technique such as Fussy Logic (FL), Nave Bayse (NB) networks, K-Nearest Neighbor (KNN), Artificial Neural Network (ANN), Support Vector Machine (SVM), Logistic Regression (LR), etc. (Ahmed et al., 2016; Garcia-Teodoro et al., 2009; Ghorbani et al., 2009; Modi et al., 2013; Tsai et al., 2009). For example, FL techniques can be used because the features to be considered can be seen as fuzzy variables. An activity is considered normal if it lies within a given spread. FL is effective against port scan attacks and probe attacks, but it presents some drawbacks. The main one is the high resource consumption (Garcia-Teodoro et al., 2009). NB is a classifier based on Bayes' theorem (Ghorbani et al., 2009) that can be used in intrusion detection. Although the effectiveness of NB is proved in certain situation, its results are equivalent to those given by threshold-based systems without having higher computational effort (Garcia-Teodoro et al., 2009).

There are different types of ANNs, depending on the number of hidden layers and their network architecture (Feedforward networks or Recurrent networks) that can be used in the field of intrusion detection such as Multilayer perceptron (MLP) and Self-Organising Map (SOM) (Ghorbani et al., 2009). To ameliorate the computational ability of ANNs and increase the intrusion detection accuracy, the number of hidden layers and neurons per layers should be increased (Modi et al., 2013). Decision Trees (DTs) have been used in intrusion detection by applying different algorithms. The best known algorithms for

implementing DTs are Iterative Dichotomiser 3 (ID3), C4.5 (successor of ID3 and a benchmark for supervised learning algorithm), and Classification and Regression Trees (CART). ID3, C4.5, and CART adopt a greedy and top-down approach to construct the tree (Han et al., 2011).

Logistic Regression (LR) is a supervised learning algorithm widely used in classification. In contrast to linear regression, LR deals with nominal attributes which is the case in intrusion detection. We have to predict the TCP/IP connection whether normal or attack. LR estimates the coefficients of the independent variables to make a linear model that generates a probability that the dependent variable belongs to a particular class (James et al., 2013). To estimate these coefficients, we used an iterative process using the *Maximum Likelihood* estimation and the *Newton–Raphson* method. The use of LR in intrusion detection is not frequent (Tsai et al., 2009) even it can give good results in other domains. But this does not mean that LR gives good results in intrusion detection.

2.2. *Hybrid classifier*

A hybrid classifier consists of the combination of more than one machine learning (either supervised or unsupervised) to improve the accuracy and avoid the disadvantages of single classifier. In fact, hybrid classifiers can include cascading classifiers where the output of one classifier is used as the input for another. A hybrid classifier can use a classifier for preprocessing or optimising the learning performance then the results are used by another classifier for the training stage or for prediction. Recently, the use of IDSs based on hybrid classifier is promising and the performance can be significantly improved (Tsai et al., 2009).

2.3. *Ensemble classifier*

An ensemble classifier consists of the combination of multiple weak machine learning algorithms (known as weak learners) to improve the classification performance. The combination of weak learners can be based on different strategies such as *majority vote*, *boosting*, or *bagging*. In intrusion detection, the use of ensemble classifiers is promising and some combination performs efficiently (Tsai et al., 2009).

3. Feature selection in network intrusion detection

In recent years, the rapid spread of the Internet, the number of devices, and the communication tools have created large amounts of data that have been stored without any clear potential use (Boln-Canedo et al., 2016). Nowadays, to process and extract knowledge from this big data, machine learning techniques and data mining tools are used. Nevertheless, applying those tools on a large volume of data with high dimensionality is time consuming and affects the accuracy of the extracted knowledge (Eesa et al., 2015). In fact, redundant and irrelevant features engender unsupportable memory requirements (Boln-Canedo et al., 2016) which reduce the algorithm and the predictor performance (Chandrashekar and Sahin, 2014). To overcome the curse of dimensionality and improve the predictor performance, a preprocessing phase is needed not only to remove noisy data but also to select, extract, or construct features (Boln-Canedo et al., 2016).

In intrusion detection, the datasets used are characterised by their large amounts and their high dimensionality. Thus, it is necessary to proceed with a dimensionality reduction step to improve the classification accuracy and reduce the computational time (Chandrashekar and Sahin, 2014). There are two main approaches to reducing dimensionality: feature transformation and feature selection. Feature transformation reduces the dimensionality of data by creating new features from the original features such as feature extraction and feature construction. In contrast, feature selection aims to select only relevant and informative features and remove irrelevant and redundant ones (Liu and Motoda, 2007). The interest in feature selection is increasing day by day due to the growing number of high dimensional datasets especially in machine learning fields such as classification, clustering, and regression (Boln-Canedo et al., 2016). There are three approaches for feature selection: *filter*, *wrapper*, and *embedded*. Each one of these approaches present strengthen and weaknesses (Liu and Motoda, 2007).

3.1. *Foundations of feature selection*

Feature selection is a process that widely uses machine learning and data mining applications. It aims to reduce the effect of the curse of dimensionality by removing redundant and irrelevant features to improve the predictor performance (Boln-Canedo et al., 2016). The prediction accuracy depends on the selection of the subset of features which efficiently represents the data. Therefore, there is a need to eliminate irrelevant and redundant features which contain lower discrimination information about the classes (Chandrashekar and Sahin, 2014). A feature is considered relevant if it integrates some information about the target or the class. Features can be classified into three categories: irrelevant, weakly relevant, and strongly relevant. Feature redundancy is usually determined by the correlation between features. Two features are redundant if they are totally correlated. If a feature is correlated with a set of features it will be difficult to judge on their redundancy (Boln-Canedo et al., 2016). Feature selection algorithms have two main components: feature search and feature evaluation.

3.1.1. *Feature search*

Feature search is a strategy that determines the optimal subset of features. The optimal solution can be found through an exhaustive search. However, the number of combinations to test is very high. If we have n features, then a search on $2^n - 1$ possible feature subsets it becomes a NP-hard[1] problem as the number of features grow (Chandrashekar and Sahin, 2014). Thus, it is time consuming and impractical. There are other search strategies that seem more feasible such as the branch and

[1] Non-deterministic polynomial-time hard.

bound algorithm,[2] sequential search methods[3] (e.g. sequential forward selection, sequential backward elimination, and bidirectional selection), and random search methods (e.g. genetic algorithms and random mutation hill climbing). If the dimensionality is very high, an individual search will be the affordable solution. In individual search methods, each feature is assessed based on some criteria. Features that satisfy a condition or that are top ranked will be selected (Liu and Motoda, 2007).

3.1.2. *Feature evaluation*

Through feature evaluation, it is possible to assess the relevance and redundance of the feature. It is necessary to fix the evaluation criteria to be applied on each feature or subset of features because different criteria lead to different feature selection. In classification, features relevant to the labelled classes will be kept. In contrast, the evaluation criteria in clustering to applying a feature selection seem difficult due to the unavailability of the class labels (Liu and Motoda, 2007).

3.2. *Feature selection methods*

Feature selection methods can be classified into two approaches: individual evaluation and subset evaluation (Yu and Liu, 2004). Individual feature evaluation assesses each feature individually according to its relevance which leads at the end to a feature ranking. The drawback of individual evaluation is that it is incapable of eliminating redundant features because they have the same rank. Differently, subset feature evaluation can overcome the inconvenience of individual evaluation, it uses certain search strategies to select and evaluate a subset of features according to certain evaluation measures and then compares it with the previous best one (Boln-Canedo et al., 2016). From this classification, three main approaches can be identified based on the relationship among the inductive learning method and the feature selection algorithm: *filters*, *wrappers*, and *embedded* methods (Guyon et al., 2008). Each one of these approaches contains several algorithms which create a large panoply of feature selection methods. Nevertheless, the best method does not exist yet. Different methods have been developed to perform feature selection with the constraints of minimising running time, memory allocation, and maintaining accuracy (Boln-Canedo et al., 2016).

3.2.1. *Filter methods*

Filter methods, which are more common in statistics, are feature selection algorithms totally independent from any predictors (Guyon et al., 2008). Applied directly on the training data, the filter approach is based on feature ranking techniques that use an evaluation criterion and a threshold to determine the feature relevance and decide whether to keep it or discard it. The feature relevance is determined by its capability to provide useful information about the different classes (Chandrashekar and Sahin, 2014). Filter algorithms are usually computationally less expensive than the other methods (Boln-Canedo et al., 2016). A common drawback for filter methods is that they are adequate only for independent features; otherwise, features will be redundant (Guyon et al., 2008).

3.2.2. *Wrapper methods*

Distinct from the filter methods, wrapper methods are feature selection based on three components: a search strategy, a predictor, and an evaluation function (Liu and Motoda, 2007). The search strategy determines the subset of features to be evaluated. The predictor (considered as a black box) can be any classification method and its performance is used as the objective function to evaluate the subset of features defined by the search strategy so as to find the optimum subset that gives the best accuracy of it (Guyon et al., 2008). The wrapper approach outperforms the filter approach but it is more time consuming and requires more computational resources (Boln-Canedo et al., 2016). As the evaluation of $2^n - 1$ subsets becomes a NP-hard problem, different search algorithms have been used to find the subset of features that maximises the accuracy of the predictor (Chandrashekar and Sahin, 2014) such as exhaustive search, sequential search, genetic search, etc. (Guyon et al., 2008).

3.2.3. *Embedded methods*

In contrast to wrapper methods, embedded methods incorporate an interaction between feature selection and learning process. Therefore, the solution is reached faster than wrappers because they make better use of the available data and avoid retraining the predictor for every selected feature subset (Guyon and Elisseeff, 2003). Embedded methods integrate a regularised risk function that is optimised taking into account the features designating parameters and the predictor parameters (Boln-Canedo et al., 2016).

4. Related work

Much related works in the literature focus on different detection approaches. Nevertheless, the most used approach is still the anomaly detection (Gan et al., 2013; Karami and Guerrero-Zapata, 2015). Of course, it is quite reasonable to focus on anomaly detection due to its efficiency in detecting new attacks. In the last recent years, the hybrid detection approach gain few steps (Depren et al., 2005; Kim et al., 2014) but it is still far from the anomaly approach and the misuse approach. Moreover, different approaches of IDSs based on anomaly detection have emerged (Rastegari et al., 2015) using numerous approaches and classifiers. Even so the use of hybrid classifiers and ensemble classifiers has increased (Tsai et al., 2009), single classifiers still exist and can output good results.

In the literature, different datasets are used to perform the IDS performance evaluation, but the one widely used is still the KDD99 dataset (Kang and Kim, 2016). The evaluation criteria adopted by most of papers to assess the performance of their approaches are accuracy, detection rate (DR), and false positive rate (FPR) (Lin et al., 2015; Wu and Banzhaf, 2010). Many related works to the intrusion detection applied a feature selection process to select the optimal subset of feature that represents the entire dataset instead of using the full feature

[2] For feature selection problems including more than 30 features, the branch and bound algorithm becomes impractical.

[3] Applying greedy techniques and the global optimality is not assured.

space (Eesa et al., 2015). Reducing the feature space can affect the size of the used dataset, the computational time, and the classification performance of several algorithms (Chandrashekar and Sahin, 2014). This can be done through different approaches. The wrapper approach for feature selection outperforms the other approaches even if it is more time consuming and requires more computational resources (Boln-Canedo et al., 2016). Interestingly, in the last years natural inspired approaches have been widely used in network intrusion detection.

Chung and Wahid (2012) proposed a new hybrid intrusion detection system using Intelligent Dynamic Swarm based Rough Set (IDS-RS) and Simplified Swarm Optimisation (SSO) which is a new version of Particle Swarm Optimisation (PSO) that incorporates a new Weighted Local Search (WLS) strategy. IDS-RS is performed to select the most relevant features to reduce the dimension of the dataset with a weighted sum fitness function. They obtained only six features from the 41 features containing in the KDD99 dataset. Finally, SSO classifier is used with the reduced dataset to classify the instances and achieve a classification accuracy with 93.3%.

De la Hoz et al. (2014) applied a multi-objective approach for feature selection based on the NSGA-II[4] as a feature search strategy and the Growing Hierarchical Self-Organising Maps (GHSOM) as a classifier. They used a fitness function based on the Jaccards coefficient which is a similarity measurement between datasets. The experiments are performed over the NSL-KDD datasets. They obtained 25 relevant features achieving 99.12% of classification accuracy.

Eesa et al. (2015) introduced a new feature selection approach based on the cuttlefish optimisation algorithm and the decision tree for IDSs to remove irrelevant features and keep only features that represent the whole dataset. They used a wrapper approach having the Cuttlefish Algorithm (CFA) as a search strategy and the ID3 algorithm as a classifier to asses the features generated by the CFA. The CFA algorithm imitates the mechanism of a cuttlefish that allows it to change its color which is based on two main processes: reflection and visibility. These two processes represent the performed feature search strategy to reach the global optimal solution. The adopted fitness function is a weighted sum formula based on the DR and FPR. The experimental result was conducted on the KDD99 dataset. They found that the accuracy of classification and DR are increased when the number of features is equal to or less than 20 features. Furthermore they found that using only five features gives better results compared with the full dimension of the dataset.

Kang and Kim (2016) presented a wrapper approach for feature selection based on a Local Search Algorithm (LSA) and the k-means clustering algorithm. They used the accuracy of clustering resulted by the k-means algorithm as a cost function to measure the goodness of the feature subset generated by LSA. The paper focused on the detection of DoS attacks and the response time. For that reason, the k-means algorithm was performed in order to split the training dataset into two clusters. In order to avoid overfitting, Kang and Kim used MLP to evaluate the performance of the selected subset of features. The experiment was conducted over the NSL-KDD and the results proved that a feature subset composed of 25 features gives higher accuracy and DR than all 41 features, while having a lower FAR.

Kavitha et al. (2012) proposed an Emerging Neutrosophic Logic Classifier Rule based Intrusion Detection (ENLCRID) to overcome the problem of uncertainty. The main purpose of this approach is to classify instances into three categories: normal, abnormal, and indeterministic. The Neutrosophic Logic (NL) classifier is a combination of the fuzzy logic, intuitionistic logic, paraconsistent logic, and the three-valued logics that generates rules used to classify instances. The problem is that not all of these rules are efficient. For that reason, Kavitha et al. used an Improvised Genetic Algorithm (IGA) to evaluate the prediction power of the generated rules and keep only best rules for accurate classification. The fitness function used in IGA is based on the sensitivity, the specificity, and the rule length. The experiment was conducted over the KDD99 dataset after reducing its dimension to only seven features by using the Best First Search (BFS) algorithm. The results are promising with DR of 99.02% and FAR of 3.19%.

Lin et al. (2012) proposed an intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection. The proposed approach is based on three components: support vector machine, decision tree, and Simulated Annealing (SA). SA has been widely used in the context of optimisation problems because of its ability to converge to the optimal solution. Therefore, SA is used in this approach to adjust the parameter settings in SVM and DT to get the best values. In the proposed algorithm, feature selection process which is based on SVM and SA selected only relevant features and remove irrelevant ones to increase the search speed and the accuracy of classification. Next DT and SA are performed to generate the decision rules for the selected features. These two steps are repeated to reach the best testing accuracy for the selected features. The obtained results proved that the obtained rules can detect efficiently new attacks by using 23 features of the 41 features of the KDD99 dataset.

Louvieris et al. (2013) introduced a novel effects-based feature identification approach for network intrusion detection. The proposed approach involves three stages based on k-means clustering, NB, KruskalWallis (KW) statistical test and C4.5 decision tree classification. The first stage consists of using k-means to cluster the dataset instances into normal and abnormal in order identify clustered features relevant to a particular attack. The second stage consists of applying a feature selection process based on a wrapper based NB ranking and KW statistical test. The NB classifier was applied for ranking relevant features for each attack whereas the KW was applied for ranking significant features. The third stage consists of applying the C4.5 for the evaluation of the already selected significant features to illustrate their accuracy for intrusion detection. For the experiment, Louvieris et al. created four simulated attack datasets from a small office environment containing 20 features that represent the main part of the TCP/IP header. They obtained promising results with only 13 relevant features.

Sindhu et al. (2012) presented a decision tree based light weight intrusion detection using a wrapper approach. The proposed work integrates different techniques such as neural

[4] Non-dominated Sorting Genetic Algorithm-II.

Table 1 – Comparisons of related work.

Work	FS components	No.	Classifier	Dataset	Evaluation
Chung and Wahid (2012)	IDS-RS	6	SSO	KDD99	Accuracy
De la Hoz et al. (2014)	NSGA2, GHSOM	25	GHSOM	NSL-KDD	DR, FPR
Eesa et al. (2015)	CFA, DT	5	None	KDD99	Accuracy, DR, FPR
Kang and Kim (2016)	LSA, K-means	25	MLP	NSL-KDD	Accuracy, DR, FPR
Kavitha et al. (2012)	BFS	7	NL	KDD99	DR, FPR
Lin et al. (2012)	SVM, SA	23	DT, SA	KDD99	Accuracy
Louvieris et al. (2013)	K-means, NB, KW	13	DT	simulated	Accuracy
Sindhu et al. (2012)	GA, Neurotree	16	Neurotree	KDD99	DR, FPR
Mok et al. (2010)	SFS, FELR	5	RELR	KDD99	Accuracy

network (NN), genetic algorithm, and decision tree. They used a wrapper approach for feature selection based on GA and Neurotree to identify the optimal subset of features. The used fitness function in GA is based on the sensitivity, specificity and the number of features. A Neurotree is in fact a combination of NN and DT used for classification. Several decision trees are obtained after *n* iteration of GA-Neurotree wrapper. Therefore, the best decision tree obtained is used to generate rules and also the best subset of features. They obtained a subset composed of 16 features. In the experiment, Sindhu et al. compared their proposed approach with various feature selection approach and various decision tree classifiers. They obtained promising results with a DR of 98.38%.

Mok et al. (2010) proposed a Random Effects Logistic Regression (RELR) model to predict anomaly detection. They used a random effects model in order to contain the network environment characteristics and the uncertainty not explained by them. Furthermore, they applied a wrapper feature selection step based on a Stepwise Feature Selection (SFS) and a Fixed Effects Logistic Regression (FELR). They obtained a subset composed of five relevant features. The experimental results illustrated 98.74% accuracy of classification. Table 1 summarises the previously mentioned works relevant to the context of feature selection applied in intrusion detection. More precisely, we compare the used feature selection components, number of features selected, the classifier, the used dataset for the experiment, and the used metrics for evaluating the IDS approach.

5. The proposed feature selection approach for network intrusion detection

The existing approaches have presented good performance in resolving the intrusion detection problem. Nevertheless, the perfect system that provides the total security by detecting all attacks with no false alarm has not been created yet. Researchers are facing several constraints such as the continuous evolution of hacking tools, the large number of existing and emerging techniques and methods in data mining and machine learning, the high dimensionality of datasets, and so on. Several existing IDSs are based on combining multiple techniques to improve the system performance (Aburomman and Reaz, 2016; Kavitha et al., 2012), whereas others are based on new techniques (Zhang et al., 2015), techniques inspired from existing ones (Lin et al., 2015), or inspired from nature (Powers and He, 2008). Furthermore, several IDSs integrate a dimensionality reduction phase to increase the prediction accuracy and to reduce the computational time and save resources (Bahl and Sharma, 2016; Kang and Kim, 2016).

In this section, a feature selection approach in the context of intrusion detection is proposed. Although wrappers need more computational resources, they outperform filters and give better results (Alba et al., 2007; Boln-Canedo et al., 2016). For that reason, we have adopted the wrapper approach to perform the feature selection. As the number of features is important only heuristics can deal with this problem (Kang and Kim, 2016). We have decided to use the Genetic Algorithm (GA) as a feature search strategy with Logistic Regression (LR) as a classifier to evaluate the selected feature subsets. The anomaly detection approach is adopted in this proposal due to its efficiency in detecting unknown attacks (Kim et al., 2014). A family of three decision tree classifiers are performed in order to measure the performance of the selected subsets of features. The used DTs in the final classification are C4.5, RF, and NBTree. The evaluation criteria used to assess the prediction capability of our approach are the detection rate, the false positive rate, and the accuracy.

5.1. *The proposed IDS framework*

In this section, an overview of the different components of the proposed IDS with feature selection is presented. The implementation of the proposed IDS solution integrates three stages: preprocessing stage, feature selection stage, and the classification stage. The system architecture of the proposed approach is illustrated in Fig. 1.

5.1.1. *The preprocessing stage*

The dataset used in the context of intrusion detection contains different forms of features such as continuous, discrete, and symbolic with varying resolution and ranges (Aburomman and Reaz, 2016; Moustafa and Slay, 2015). Most of the existing classification algorithms are incompatible with such a structure. Therefore, it is necessary to proceed with a preprocessing stage to transform those featu–re formats in a way to be handled with the classification algorithms.

Preprocessing consists of two main steps. First, all nominal features are mapped to integer values ranging from 0 to $S-1$ where S is the number of symbols. Boolean features do not need any modification. Only the class label is mapped in a different way to be handled with LR in feature selection stage, 0 for

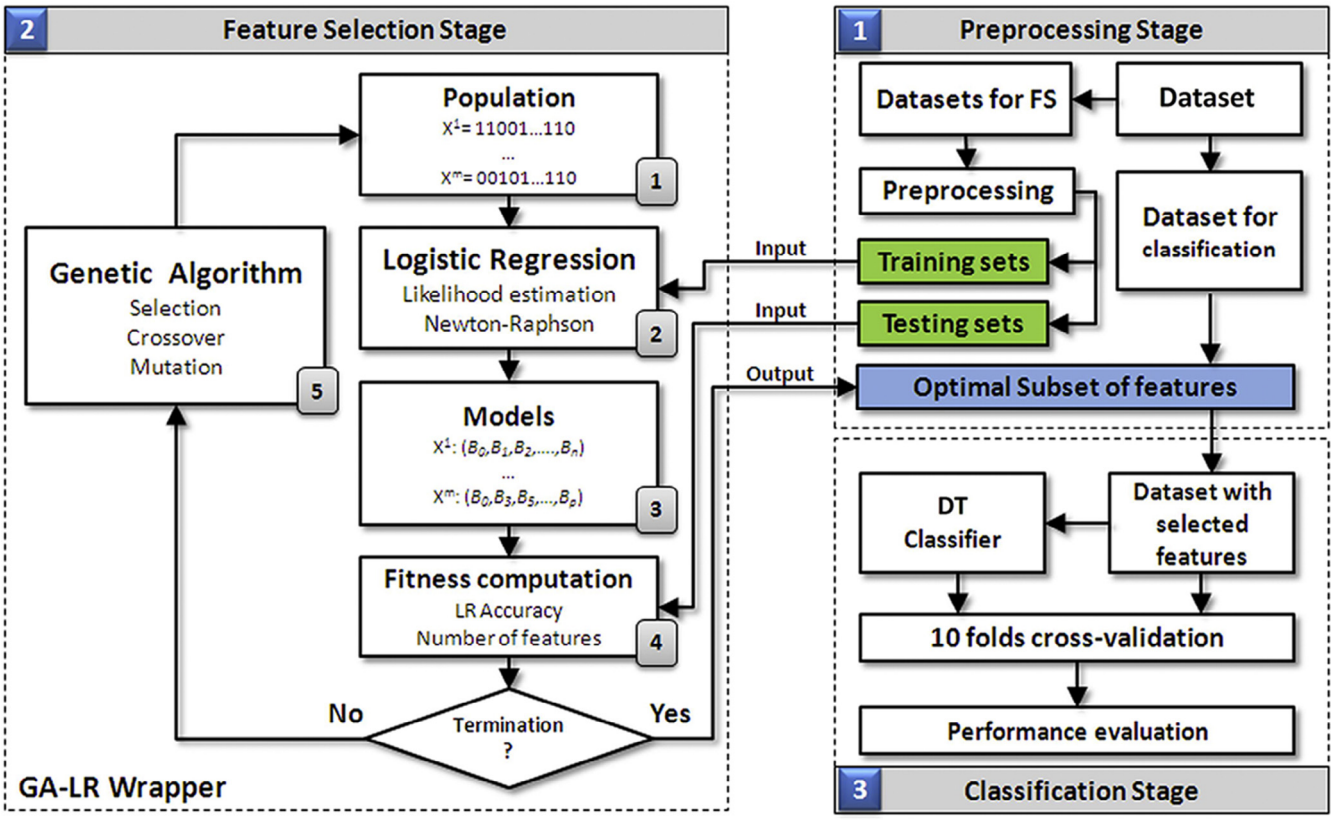

Fig. 1 – System architecture for the proposed IDS.

Normal class and 1 for the others. Second, numerical features having a large integer value ranges are scaled by applying a logarithmic scaling with base 10 to reduce their value ranges. Then all values of the used n features, except the Boolean features, are linearly scaled into the range of [0, 1]. Linear scaling is a min–max normalisation that consists of finding the minimum and maximum value of i^{th} feature and then transform each feature value v is linearly scaled to v' in the range [0, 1] by applying the following formula (Kang and Kim, 2016):

$$v' = \frac{v - min_i}{max_i - min_i}, \quad 1 \le i \le n \tag{1}$$

Applying preprocessing on the whole dataset is time consuming due to its large size. Therefore, it is recommended to apply sampling before the preprocessing step. The resulted dataset is divided into different datasets to be used in the feature selection stage for training and testing to select the optimal subset of features. In classification stage, we used a dataset applying a redundancy removal step to have only unique instances. The resulted dataset is then reduced using the optimal subset of features given by the feature selection stage to measure the performance of the proposed approach for IDS (Kang and Kim, 2016).

5.1.2. *The feature selection stage*

The feature selection stage consists of applying a wrapper approach to select the most relevant subset of features with the minimum size. As previously mentioned, the wrapper approach is based on three components: a search strategy, a predictor, and an evaluation function (Liu and Motoda, 2007). In this proposal, we use the GA as a search strategy and the LR as a classifier. The evaluation of the feature subsets is a fitness function based on the accuracy of the LR and the number of features in the subset.

Fig. 1 illustrates the different steps in the proposed GA-LR wrapper approach. Starting from a random initial population with each individual representing a possible feature subset, each feature subset is proceeded with the LR classifier to generate a model. The resulted models corresponding to the subsets of features are evaluated to compute their chance to be selected for reproduction. The subsets are then proceeded with the GA to generate the new population of subsets applying GA operations. This process is repeated until a given number of generations is attained or all subsets of features become identical. In the next section, a detailed description of the different components of the GA-LR wrapper approach is presented.

5.1.3. *The classification stage*

The classification stage consists of using three different decision tree classifiers namely C4.5, RF, and NBTree to evaluate the performance of the selected subsets of features given by the feature selection stage. In supervised learning, the classification stage is divided into two phases: a training phase and a testing phase (Depren et al., 2005). During the training phase the classifier is trained using the training dataset with the selected features. The resulting model constructed during the

training phase is used later in the testing phase to evaluate the classification performance (Eesa et al., 2015).

The C4.5 classifier was one of the 10 algorithms identified by the IEEE International Conference on Data Mining in December 2006. It was considered as one of the most influential data mining algorithms in the research community (Wu et al., 2008). It builds decision trees from training datasets based on the maximisation of the information gain (difference in entropy) at each attribute split. During the testing phase, the decision tree resulting from training the C4.5 is used to classify testing dataset instances. Starting from the root node of the decision tree, a test is applied on the same attribute of the testing dataset represented in the node. The branch is taken if the condition is satisfied to lead to the root node child. This process is repeated until a leaf node is reached which represents the class of the tested instance (Buczak and Guven, 2016). The RF classifier is one of the most powerful data mining techniques composed of a collection of tree structured classifiers. It has been used to prediction and probability estimation. It has the capability to deal efficiently with large datasets having many features (Zhang et al., 2008). The NBTree classifier is in fact a combination between DT and NB classifiers (Liang and Yan, 2006). DT is used as a general structure for the data segmentation whereas NB is used at leaves instead of splitting the attributes. NBTree is appropriate in machine learning particularly when having many relevant features. It produces smaller number of nodes than the C4.5 (Kohavi, 1996).

5.2. The GA-LR wrapper approach for solving feature selection

The problem of feature selection can be resolved with different methods as mentioned in Section 3. In this work, a wrapper approach is adopted due to its superiority compared to other approaches (Alba et al., 2007; Boln-Canedo et al., 2016). The wrapper approach includes a research strategy and a predictor. The predictor performance is used to evaluate the selected subsets. As previously mentioned, in this proposal the GA is used as a research strategy and the LR as a predictor. The general framework of the proposed GA-LR wrapper approach is summarised in Fig. 1. Below, we present the GA and the LR used in the proposed approach for feature selection.

5.2.1. Genetic algorithm as search strategy

The GA is a global metaheuristic search strategy based on a direct analogy to Darwinian natural selection and genetics in biological systems (Huang and Wang, 2006). GA is an evolutionary algorithm composed of four components: a population of individuals (or chromosomes) with each one representing a possible solution, a fitness function used to evaluate the fitness of an individual, a selection function which selects the fittest individuals to produce the next generation, and a genetic operator such as crossover and mutation to explore new search spaces. In the following, we describe the GA components used in the proposed wrapper approach.

5.3. Data encoding

In the GA, each chromosome presents a possible solution. The nature of the problem determines the chromosome coding (Fessi et al., 2014). In the context of feature selection, an X binary vector is used to codify the n features existing in the dataset. Therefore, we have $X = (x_1, x_2, \ldots, x_n)$ where $x_i \in \{0, 1\}$ and $1 \le i \le n$. If the i^{th} feature is used in the current subset then $x_i = 1$, otherwise $x_i = 0$.

5.4. Initial population

An initial population with N chromosomes is generated randomly regardless of any constraints. A large population provides more genetic diversity but it is weakened by slower convergence. In contrast, a small population converges rapidly but explores only a reduced part of the search space and may converge to a local extreme. Thus the choice of population size may affect the GA performance.

5.5. Fitness function

The fitness function of a given chromosome determines its chance of being chosen to create the next generation (Huang and Wang, 2006). In this work, the fitness function is based on two criteria: the classification accuracy of LR and the number of selected features. Therefore, the chromosome having the highest accuracy, and the smallest number of features produces the a highest fitness value. A combination of the two objectives is presented in the following formula:

$$fitness(X) = \alpha \times Accuracy(X) + (1-\alpha) \times \left(\sum_{i=1}^{n} x_i\right)^{-1} \quad (2)$$

where X is the binary vector representing the current features subset, $\alpha \in [0, 1]$ is a predefined weight adjusted according to user's requirements to indicate the importance of LR accuracy with respect to the subset length. $Accuracy(X)$ is the accuracy of LR based on the subset of features X for computing accuracy of a classifier, and x_i is a bit indicating whether the i^{th} feature is used or not. Thus the $\sum_{i=1}^{n} x_i$ gives the number of features within a given subset.

5.6. Selection function

After generating the new population of chromosomes by applying genetic operators, the fitness of each chromosome is computed. The fitter chromosomes have more chance for reproduction. The selection can be performed by using the roulette wheel, the tournament selection methods (Huang and Wang, 2006), or rank selection (Fessi et al., 2014). In this work, the roulette wheel is applied as a selection function.

5.7. Crossover operator

A crossover is performed among two randomly selected chromosomes with a probability p_c in [0.5, 0.9]. In this work, a random selection between a single point and two point crossovers determine the parts to be swapped between the two parent chromosomes to give the offspring. This operation is repeated N/2 times to create the new generation of chromosomes. Fig. 2 illustrates the genetic operators of a crossover.

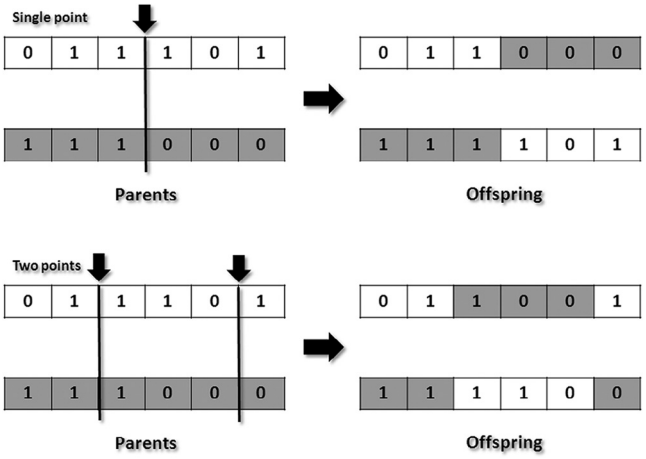

Fig. 2 – The genetic operators of a crossover.

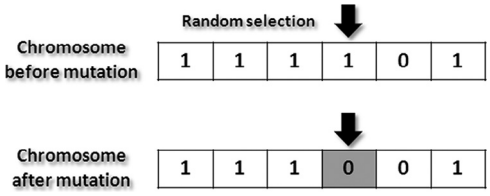

Fig. 3 – The genetic operators of mutation.

5.8. *Mutation operator*

Mutation serves to alter the value of one or more bits in the chromosome with a probability p_m in [0.01, 0.03]. It aims to improve the fitness value of chromosomes by introducing new heterogeneity in the population (Fessi et al., 2014). In this work, mutation consists of selecting randomly q bits of a chromosome ($q \leq 10\%$ of chromosome size) to be inverted from 1 to 0 or vice versa. Fig. 3 illustrates the genetic operators of mutation.

5.9. *Termination rules*

The algorithm is performed until one of the two conditions is satisfied. First condition is when G fixed number of generations is attained. Second condition is when all chromosomes are identical. Thus, the fittest chromosome within all generations is considered as the best subset of features.

5.9.1. *The proposed algorithm*

The proposed algorithm for feature selection is illustrated below. It consists of selecting the best subset of features that mostly represent the data by using the training and the testing datasets.

Algorithm: GA_LR_Wrapper_FS ( )

Input: Training dataset, Testing dataset
Output: Optimal subset of features

P = Generate_Initial_Population (N)
$i = 1$
While (All_Chromosomes_are_Different (P) and $i \leq G$)
  P_{new} = empty
  Repeat ($N/2$ times)
    Selection (P, X_{p_1}, X_{p_2})
    X_O = Crossover (X_{p_1}, X_{p_2})
    Mutation (X_{O_1}, X_{O_2})
    Add_to_Population (P_{new}, X_{O_1}, X_{O_2})
  End Repeat
  Compute_Fitness (P_{new})
  Save_Best_Chromosome (P_{new})
  $P = P_{new}$
  $i = i + 1$
End While
End.

The first step is to randomly generate an initial population with N chromosomes, denoted by P. To make a new population P_{new}, based on the previous population P, genetic operators are applied. First, two chromosomes are randomly selected, denoted by X_{p_1} and X_{p_2}, from P using the roulette wheel method to perform crossover. Second, the resulting offspring, denote by X_{O_1} and X_{O_2}, are then mutated before adding them to the new population P_{new}. Third, the fitness of all chromosomes of P_{new} is computed and the fittest chromosome overall generations is stored in memory. These operations are repeated until all chromosomes within P are similar or the number of generations G is attained. As mentioned above, the fitness is based on LR accuracy and the number of features. In the next subsection, LR is presented with more details.

5.10. *Logistic regression for classification*

LR is considered as one of the widely used classifiers such as linear discriminant analysis and k-nearest neighbour (James et al., 2013). LR is used to describe the relationship between a dependant variable and one or more explanatory variables. In contrast with Linear Regression, LR can deal with categorical dependent variables (Czepiel, 2002). The used dataset in the context of intrusion detection contains a categorical class label which indicates whether the connection is normal or an attack (Aburomman and Reaz, 2016). In this case, we have a binary dependent variable that can be represented by 0 for *Normal* class and 1 for the attack classes which can be handled with binomial LR (Ghosh and Mitra, 2015).

The purpose in LR (or Logit analysis) is to predict the probability p that the dependent variable is taking the value of 1 rather than 0, based on a set of independent variables (continuous or categorical variables) (Katos, 2007). As LR can be seen as a special case of generalised linear models, the linear component is assimilated to some function of the probability of a given result of the dependant variable. This function is the logit transform. To estimate the LR parameters, the method of least squares used in linear regression is not capable of giving minimum variance unbiased estimators for parameters. Therefore, maximum likelihood estimation is used for estimating the LR parameters with the Newton–Raphson method (Czepiel, 2002). Next, a presentation of the different components of LR is provided, more precisely we present the logit model and the parameters estimation method.

5.10.1. *Logit model*

Consider a population (Y, X) such that $Y \in \{0, 1\}$ a categorical dependent variable and $X = (X_1, \ldots, X_k)$ is a vector of k independent variables. The purpose is to build a model that predicts the probability that Y is taking a response y for any given value of X. This probability $p(X) = \Pr(Y = y|X)$ must range between 0 and 1 for all values of X. In LR, the *logistic function* (see Fig. 4) is used to model this probability as presented in Eq. (3) (James et al., 2013). Suppose we have $C(X) = \beta_0 + \beta_1 \times X_1 + \ldots + \beta_k \times X_k$, then we have

$$p(X) = \frac{e^{C(X)}}{1 + e^{C(X)}} = \frac{1}{1 + e^{-C(X)}} \tag{3}$$

After a bit manipulation of Eq. (3), we have

$$\frac{p(X)}{1 - p(X)} = e^{C(X)} \tag{4}$$

where the quantity $\frac{p(X)}{1-p(X)}$ is named the *odds* and range between 0 and ∞. By applying the logarithm to Eq. (4), we have the logit transform which is named *log-odds* or *logit* that is linear in X (Czepiel, 2002),

$$\log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 \times X_1 + \ldots + \beta_k \times X_k \tag{5}$$

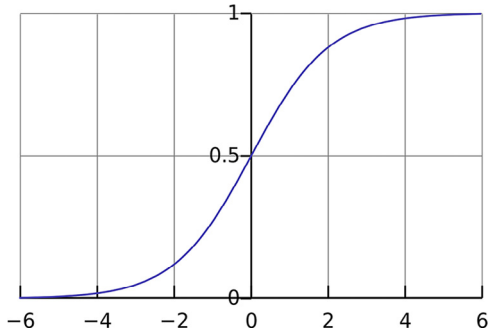

Fig. 4 – The logistic function.

The affectation rule can be in different ways (Czepiel, 2002):

$$Y = \begin{cases} 1 & \text{if } \frac{p(X)}{1-p(X)} > 1 \text{ or} \\ & \text{if } p(X) > 0.5 \text{ or} \\ & \text{if } C(X) > 0 \\ 0 & \text{otherwise} \end{cases}$$

5.10.2. *Maximum likelihood estimation*

The parameters $\beta = (\beta_0, \beta_1, \ldots, \beta_k)$ are unknown and must be estimated using the training data. The maximum likelihood method is used to estimate the coefficients of the logistic regression (James et al., 2013). The likelihood function of a sample Ω is presented by the following equation:

$$L = \prod_{\omega} \pi^{y} (1 - \pi)^{(1-y)} \tag{6}$$

where ω is an individual in Ω, π is the probability $P(Y(\omega) = 1|X(\omega))$ if $y = 1$, and $(1 - \pi)$ is the probability $P(Y(\omega) = 0|X(\omega))$ if $y = 0$. In fact, the likelihood corresponds to the probability of having the sample Ω from a given population. The maximum likelihood method consists of estimating the parameters $\hat{\beta} = (\hat{\beta}_0, \hat{\beta}_1, \ldots, \hat{\beta}_k)$ that maximise this likelihood function which is the probability of having the sample Ω. To facilitate handling, it is often preferred to work on the log-likelihood which range between $-\infty$ and 0 (Czepiel, 2002),

$$LL = \sum_{\omega} y \times \log(\pi) + (1 - y) \times \log(1 - \pi) \tag{7}$$

Often, the following quantity named *deviance* is used,

$$D = -2 \times LL \tag{8}$$

In contrast to log-likelihood, deviance is positive. The purpose here is to minimise this quantity. The log-likelihood is a convex function; there is therefore a single solution. Nevertheless, there is no analytical solution; thus it is necessary to use heuristics. Different optimisation methods are used in this context, the mostly used is the Newton–Raphson method (Czepiel, 2002).

5.10.3. *The Newton–Raphson method*

Maximum likelihood method is used to estimate the parameters of the LR. It uses the training data in a way to produce a distribution that gives the greatest probability of the observed data. However, solving a system of non-linear equations is not easy. Therefore, it is necessary to use an iterative process to numerically estimate the parameters of LR. The most widely used method for solving systems of non-linear equations is the Newton–Raphson method (Czepiel, 2002).

The Newton–Raphson method started with a random value for vector β then the algorithm applies an iterative process to approximate the final solution $\hat{\beta}$. To go from step (i) to step (i + 1), the algorithm uses the following formula (Czepiel, 2002):

$$\beta^{i+1} = \beta^{i} - \left(\frac{\partial LL}{\partial \beta . \partial \beta'}\right)^{-1} \times \frac{\partial LL}{\partial \beta} \tag{9}$$

where β^i is the current vector of parameters, β^{i+1} is the resulted vector, $\left(\frac{\partial LL}{\partial \beta . \partial \beta'}\right)^{-1}$ is the matrix of second partial derivatives, and $\frac{\partial LL}{\partial \beta}$ is the first partial derivatives of the log-likelihood. By using matrix notation, Eq. (9) becomes as follows:

$$\beta^{i+1} = \beta^i + \left[X^T W X\right]^{-1} \times X^T (y - \pi) \quad (10)$$

where X is the design matrix of independent variables, it contains $K + 1$ columns where K is the number of independent variables in the dataset and the first element of each row $X_{i0} = 1$ which represent the intercept. X^T is the matrix transpose of X. W is a square matrix of order equal to the dataset size, with elements $\pi(1 - \pi)$ on the diagonal and zeros everywhere else.

The iterative process based on Eq. (10) is repeated until there is no improvement in β. At that point, the algorithm has converged and the quantity $[X^T W X]^{-1}$ contains the variance–covariance matrix of the estimated coefficients.

5.10.4. *The proposed algorithm*

The following algorithm is used to compute the accuracy of LR by using the proposed subset of features. The algorithm uses the training dataset to estimate the model then evaluate it using the testing dataset. The accuracy of LR is used later to compute the fitness function of the current subset in GA.

The first step is to initiate the β vector of parameters for the proposed X subset of features. The log-likelihood is computed based on the proposed training dataset and then the new parameters are estimated using the Newton–Raphson method. The process is repeated until the improvement of the estimated parameters is less or equal to a given threshold σ or a given number of iteration M is attained. The final step is to evaluate the resulted model using the proposed testing dataset by computing its accuracy.

Algorithm: LR_Accuracy ( )

Input: Training dataset D_{tr}, Testing dataset D_{ts}, Subset of features X
Output: Accuracy of the subset of features

$i = 1$
While ($Imp > \sigma$ and $i \leq M$)
  LL = Compute_Log_Likelihood (D_{tr}, β, X)
  $\hat{\beta}$ = Newton_Raphson (i, β, LL)
  Imp = Improvement ($\hat{\beta}$, β)
  $i = i + 1$
  $\beta = \hat{\beta}$
End While
Accuracy = Evaluate_Model (D_{ts}, $\hat{\beta}$, X)
End.

6. Experimental results

The proposed approach for IDS integrates three main stages as mentioned in Section 4 which are preprocessing, feature selection, and classification stages. The preprocessing stage consists of resampling and transforming the dataset to be handled with the classifiers used in feature selection and final classification stages. The feature selection stage, which is based on a wrapper approach combing GA as search strategy and LR as a classifier, selects the best subset that mostly represents the data with the minimum number of features. The final stage consists of evaluating the performance of the optimal subsets of features given by the GA-LR wrapper approach by using three different decision tree classifiers.

6.1. *Datasets description*

Since many years, the evaluation of the performance of any proposed IDS approach is a real challenge. It is practically impossible to use a real-word network infrastructure to perform the IDS evaluation. This is quite reasonable because most companies never accept that their business will be disrupted or their data being destroyed or even stolen. Therefore, researchers used simulated datasets including normal and attack traffic to perform IDS evaluation. The problem is that these datasets are stored in limited lapse of time and generated through a limited network infrastructure which raises the problem of their concordance with reality. Moreover, measuring the degree of realism of the generated IDS datasets is also a challenge (Haider et al., 2017). Different publicly available datasets have been generated in this context (such as KDD99, NSL-KDD, Kyoto, ADFA-LD, etc.), but all of them are suffering in different ways from concordance to the real-word network traffic. In order to resolve this problem, researchers are trying to find metrics that quantify the quality of realism to generate more realistic IDS datasets that include dynamic scalable normal and attack behaviours (Haider et al., 2017; Shiravi et al., 2012). This is extremely important for the credibility and reliability of any IDS solution. Creating IDS datasets that do not take consideration of the evolution of the applications, Internet activities and the widespread of attack tools lead to an incorrect IDS evaluation which is unacceptable and extremely costly.

In the context of this research, we have decided to use an old benchmark which is the knowledge discovery and data mining 1999 (KDD99) dataset and a new benchmark which is the UNSW-NB15 dataset. Thus, we can compare the obtained results with the old and the new benchmark datasets in terms of accuracy and complexity which can reflect there degree of realism. The KDD99 dataset and its derived version have been widely used in the IDS performance evaluation (Kang and Kim,

2016). The problem is that this dataset is outdated and does not contain any contemporary normal and attack behaviours which bias any IDS evaluation (Haider et al., 2015). This may explain the high accuracy rate achieved by many works which is meaningless and no longer accepted in the IDS community. Moreover, this dataset contains a huge quantity of repeated records (Sindhu et al., 2012) and shows an unbalanced distribution between the different classes (Elhag et al., 2015). These facts proved the existence of issues related to the design and methodologies adopted for the dataset creation and validation (McHugh, 2000). Therefore, we used it in this context to prove these facts. The dataset was generated by Lincoln Labs based on a simulation of a typical U.S. Air Force LAN. It contains contained approximately 5M instances of raw tcpdump data containing normal and abnormal activities generated during several weeks (Elhag et al., 2015). Each instance is represented by 41 nominal and continuous features and a class label that indicates whether the connection is normal or an attack (Aburomman and Reaz, 2016). There are 22 different types of attacks that fall into one of the four categories: DoS, probe, U2R, and R2L. The 41 features can be classified into three groups: basic features, content features, and traffic features. Basic features (such as *duration*, *protocol_type*, *service*, etc.) are attributes that can be extracted from a TCP/IP connection. Content features (*Num_failed_logins*, *logged_in*, *num_compromised*, etc.) allow to detect suspicious behaviour. Traffic features, which are divided into the same host features (such as *serror_rate*, *rerror_rate*, etc.) and the same service features (such as *srv_serror_rate*, *srv_rerror_rate*, etc.), are computed on the basis of the examination of the established connections in the past two seconds (Kang and Kim, 2016).

The UNSW-NB15 is a recently created dataset by the cyber security research group at the Australian Centre for Cyber Security (ACCS) (Moustafa and Slay, 2015). The IXIA Perfect Storm tool and a tcpdump tool were used to capture 100 GB of raw data representing simulated network traffic including modern normal and contemporary attack behaviours. The raw data were captured during two simulation periods of 16 hours and 15 hours. The total size of the dataset is about 2.5M records. Argus, Bro-IDS tools, and a 12 developed algorithms were performed to create the total number of 49 features. These features are grouped into five categories: flow features, basic features, content features, time features, and additional generated features. Two features are used as a label: *attack_cat* which indicates the category of the attack and the normal state, and *label* which takes 0 for normal and 1 for attack. There are nine attack categories included in the dataset: Fuzzers, Analysis, Backdoor, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worms (Moustafa and Slay, 2016).

To evaluate the performance of the IDSs in supervised learning, it is necessary to have two distinct datasets: a training dataset and a testing dataset (Aburomman and Reaz, 2016). The original size of the KDD99 and the UNSW-NB15 datasets were too large (Kang and Kim, 2016; Moustafa and Slay, 2016). Therefore, for not affecting the time in the training stage, usually a 10% of the training dataset is used (Elhag et al., 2015). Table 2 illustrates the different categories and classes of the training 10% KDD99 and Table 3 illustrates the different categories of the training set extracted from the UNSW-NB15.

Table 2 – Distribution of records in the training 10% KDD99.

Category	Size	Distribution (%)
Normal	97,278	19.69
DoS	391,458	79.24
Probe	4,107	0.83
R2L	1,126	0.23
U2R	52	0.01
Total	494,021	100

Table 3 – Distribution of records in the training UNSW-NB15 datasets.

Category	Size	Distribution (%)
Normal	56,000	31.94
Backdoor	1,746	1
Analysis	2,000	1.14
Fuzzers	18,184	10.37
Shellcode	1,133	0.65
Reconnaissance	10,491	5.98
Exploits	33,393	19.04
DoS	12,264	6.99
Worms	130	0.07
Generic	40,000	22.81
Total	175,341	100

6.2. *Experimental settings*

In this section we present the experimental tools used to implement the proposed approach, the sample used to perform the experiment, and the evaluation method adopted to measure the performance of the proposed approach.

6.2.1. *Experimental tools*

In the literature, different tools are used to implement and evaluate the proposed IDS. Matlab (Aburomman and Reaz, 2016; Fessi et al., 2014; Huang and Wang, 2006; Kavitha et al., 2012), C++ (Gan et al., 2013), Visual C++ (Lin et al., 2008), C# (Eesa et al., 2015), Java (Sindhu et al., 2012), Weka (Chung and Wahid, 2012; Kim et al., 2014; Kou et al., 2009; Rastegari et al., 2015; Singh et al., 2015) are the most used tools in this context. In this proposal, Weka[5] data Mining and machine learning software (version 3.6.13) is used to perform the classification stage with C4.5 (implemented in Weka under the name of J48 (Rastegari et al., 2015), RF, and NBTree. Weka is a free software written in Java and developed at the University of Waikato, New Zealand. It integrates most of the machine learning and data mining techniques used for knowledge discovery.

Despite its ability to perform wrapper approaches for feature selection, Weka presents a weakness which is the process time. We tested a GA-LR wrapper over Weka with a sample of 494 instances and with a population composed of 6 chromosomes. To generate only the initial population and the first generation, the process takes almost more than one hour. For that reason, we selected the C++ programming language to implement the GA-LR wrapper for feature selection. We

[5] The Waikato Environment for Knowledge Analysis.

Table 4 – Distribution of records in training and testing datasets used in the feature selection stage after preprocessing.

Dataset	Label	Training set			Testing set		
		1000	1500	2000	1000	1500	2000
KDD99	Normal (0)	508 (50.80%)	719 (47.93%)	985 (49.25%)	495 (49.50%)	746 (49.73%)	934 (46.70%)
	Attack (1)	492 (49.20%)	781 (52.07%)	1015 (50.75%)	505 (50.50%)	754 (50.27%)	1066 (53.30%)
UNSW-NB15	Normal (0)	330 (33%)	552 (63.20%)	661 (33.50%)	402 (40.20%)	574 (38.3%)	838 (41.90%)
	Attack (1)	670 (67%)	948 (36.80%)	1339 (66.95%)	598 (59.80%)	926 (61.70%)	1162 (58.10%)

selected this programming language to reduce the computational time of the GA-LR wrapper algorithm that integrates heavy computational work. We used also R software which is a free software environment for statistical computing and graphics to verify the results given by the implemented LR module.

The preprocessing stage modules such as sampling, nominal to numeric, logarithmic scaling, and normalisation are also implemented with C++ language. The platform adopted to develop and test the proposed approach is a PC with the following features: Intel Core Duo 2.16 GHz CPU, 1 GB RAM, and a Windows XP operating system.

6.2.2. *IDS evaluation method*

There are different metrics used to evaluate the performance of an IDS. In the literature, most of research works in the field of intrusion detection focused on the accuracy, the detection rate (DR), and false alarm rate (FAR) (Lin et al., 2015). In this work, we have adopted the same metrics to evaluate the performance of our proposed approach. As aforementioned, the fitness function that evaluates each selected subset of features in the feature selection stage is based on the accuracy of LR classifier and the number of selected features. Furthermore, in the classification stage the performance of the three decision tree classifiers with the selected subsets of features is measured with the three metrics mentioned above and then compared with other work results.

6.3. *Preprocessing of the KDD99 and the UNSW-NB15 datasets*

As aforementioned, the KDD99 and the UNSW-NB15 datasets are used in this proposal. Due to the huge size of the original datasets (about 5M records for KDD99 and 2.5M for UNSW-NB15), we used 10% of the KDD99 dataset (kddcup.data_10_percent.gz) (KDD Cup, 1999) and the training dataset of the UNSW-NB15 (UNSW_NB15_training-set.csv) (UNSW-NB15 dataset, 2015). Some research works (Chung and Wahid, 2012; Dangelo et al., 2015; Eesa et al., 2015; Gan et al., 2013; Karami and Guerrero-Zapata, 2015) use different dataset sizes randomly selected from the training dataset. We randomly selected 10,000, 15,000, and 20,000 unique instances from the 10% KDD99 dataset and from UNSW-NB15 training dataset. All extracted datasets are preprocessed and then splitted into 10 folds. We selected from each group two datasets presenting the best accuracy of classification with LR, one used for the training and the other for the testing. Thus, we have six datasets extracted from the KDD99 and six datasets extracted from the UNSW-NB15 with different size: two datasets with 1000 records, two datasets with 1500 records, and two datasets with 2000 records. These datasets are used to perform the feature selection stage.

Table 4 illustrates the distribution of records between normal and attack in training and testing datasets. Tables 5 and 6 present some descriptive statistics of the 42 features for training and testing datasets extracted from the KDD99 dataset (see Figs. 5 and 6). Features having zero variability (standard deviation) are not used in the analysis and must be removed before the feature selection stage. As shown in Tables 5 and 6, we decided to remove 8 irrelevant features which are 7, 9, 11, 14, 15, 18, 20, and 21. Differently, there is no zero variability in the features of the training and testing datasets extracted from UNSW-NB15 dataset. Furthermore, there are only 44 features in the available training UNSW-NB15 dataset (UNSW_NB15_training-set.csv) (UNSW-NB15 dataset, 2015), which means 42 attributes and 2 class labels. Thus, we removed the attribute *attack_cat* that indicates the category of the attack and the normal state, and keep *label* and the 42 other attributes.

For the classification stage, we decided to use the 10% of KDD99 dataset and the UNSW-NB15 training dataset. The problem is that the KDD99 dataset contains a huge quantity of repeated records (Elhag et al., 2015; Sindhu et al., 2012) which is not the case with the UNSW-NB15 dataset (Moustafa and Slay, 2016). Redundant records affect any machine learning by biasing towards frequent records (Elhag et al., 2015; Sindhu et al., 2012). Therefore, we proceeded to the elimination of all redundant records, obtaining a total number of 145,586 instances (see Table 7). Finally, we extracted 50,000 unique records from each dataset with keeping the same distribution of records between categories to perform the classification stage with the three DTs. In contrast to the feature selection stage, we removed from the extracted UNSW-NB15 dataset the attribute *label* and keep the *attack_cat* attribute. Table 8 illustrates the distribution records between categories for the two datasets extracted from the KDD99 and UNSW-NB15.

6.4. *Implementation of GA-LR wrapper approach*

Four major packages including different modules are created using C++: a file management package including modules such as loading, saving, etc., a matrix management package including several modules (such as matrix multiplication, transpose, inverse, etc.) that are used in the remaining packages, an LR package including modules such as computing the likelihood, the Newton–Raphson method, the LR accuracy, etc. The LR program needs a training dataset, the selected features represented by a binary string, and a testing dataset to generate the model and measure its performance. The model is illustrated by the coefficient estimated by LR. The performance is measured by all metrics extracted from the confusion matrix.

Table 5 – Descriptive statistics of the different sample KDD99 training datasets after the preprocessing stage.

Feature	Training set					
	1000		1500		2000	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
1	0.003886	0.007562	0.001564	0.003059	0.003436	0.006670
2	0.532500	0.079475	0.529333	0.074052	0.537250	0.083758
3	0.543667	0.196413	0.544076	0.192277	0.552468	0.188307
4	0.633000	0.259472	0.605083	0.256794	0.601389	0.279561
5	0.182444	0.180119	0.171666	0.181026	0.175004	0.181395
6	0.197765	0.218757	0.181779	0.214257	0.188973	0.217319
7	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
8	0.007000	0.013902	0.004667	0.009277	0.005500	0.010928
9	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
10	0.004000	0.007904	0.004422	0.008715	0.002217	0.004387
11	0.000000	0.000000	0.000133	0.000266	0.000000	0.000000
12	0.393000	0.477102	0.365333	0.463730	0.371500	0.466975
13	0.000021	0.000042	0.000317	0.000631	0.000357	0.000710
14	0.000000	0.000000	0.000667	0.001332	0.000000	0.000000
15	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
16	0.000041	0.000082	0.000611	0.001220	0.000708	0.001412
17	0.000500	0.000998	0.000667	0.001332	0.000571	0.001139
18	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
19	0.000500	0.000999	0.000667	0.001332	0.001500	0.002991
20	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
21	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
22	0.007000	0.013902	0.005333	0.010610	0.003000	0.005982
23	0.187075	0.185328	0.193971	0.186441	0.186101	0.182150
24	0.026853	0.025495	0.024856	0.020450	0.025997	0.021578
25	0.383010	0.470694	0.375853	0.466985	0.380715	0.470216
26	0.380510	0.470572	0.375013	0.467157	0.380265	0.470128
27	0.112200	0.199023	0.149973	0.254315	0.134905	0.232641
28	0.113630	0.200658	0.150713	0.255129	0.135180	0.232640
29	0.563290	0.460333	0.538053	0.460653	0.552710	0.460061
30	0.041580	0.043584	0.047367	0.047186	0.043245	0.044289
31	0.071160	0.117893	0.072040	0.120823	0.074505	0.123653
32	0.785315	0.303950	0.778275	0.307043	0.770968	0.317973
33	0.434976	0.429742	0.408345	0.420463	0.422819	0.426789
34	0.469760	0.446281	0.442580	0.440781	0.460315	0.447528
35	0.065970	0.062407	0.066340	0.058491	0.067535	0.062366
36	0.072140	0.114469	0.080380	0.128497	0.086340	0.137558
37	0.013310	0.020130	0.017693	0.026668	0.016495	0.024494
38	0.378850	0.468287	0.374760	0.466840	0.381250	0.468660
39	0.378080	0.469672	0.374327	0.467337	0.379895	0.470120
40	0.117320	0.203950	0.153313	0.255962	0.134075	0.229049
41	0.113610	0.200090	0.152967	0.256808	0.132285	0.227288
42	0.508000	0.499872	0.479333	0.499146	0.492500	0.499887

Other statistical information is provided such as Wald test, χ^2 test, AIC,[6] BIC,[7] and pseudo R^2 (James et al., 2013).

A GA package that performs the GA operations such as generating populations, selection, crossover, mutation, etc. All these packages are integrated into a main program called *GA-LR-Wrapper*. The inputs of the main program are: a training dataset, a testing dataset, population size, crossover rate, mutation rate, maximum generation, and the coefficient α used in the fitness function. To reduce the program execution time, all evaluated chromosomes and their fitnesses are stored in memory whether they are generated again. If all chromosomes are the same or the number of generation is attained the program is terminated and a log file is generated. The log file contains all input parameters, all generations and their mean fitnesses, the started and the terminated time, and the best subset of features.

6.5. *Classification stage using Weka*

The classification stage is performed using Weka software with the new dataset obtained after the redundancy removal from 10% KDD99 and UNSW-NB15 training datasets. We converted the two datasets into ARFF file format to be handled with Weka. The conversion consists of adding header information including attribute names, types, possible values (for nominal attributes), and separating data fields with commas (Garner, 1995). Before starting classification, we proceeded with a

[6] Akaike information criterion.
[7] Bayesian information criterion.

Table 6 – Descriptive statistics of the different sample KDD99 testing datasets after the preprocessing stage.

Feature	Testing set					
	1000		1500		2000	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
1	0.004797	0.009315	0.002474	0.004816	0.002371	0.004627
2	0.533500	0.083970	0.530667	0.075719	0.533750	0.082992
3	0.548267	0.193636	0.535781	0.194063	0.557758	0.187614
4	0.629333	0.262670	0.626250	0.251403	0.595833	0.278989
5	0.183420	0.183196	0.185231	0.184793	0.171454	0.180453
6	0.196478	0.219876	0.194289	0.220194	0.183472	0.214764
7	0.000000	0.000000	0.000000	0.000000	0.000500	0.000999
8	0.004000	0.007968	0.007333	0.014540	0.004833	0.009613
9	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
10	0.001600	0.003165	0.005089	0.010001	0.002617	0.005168
11	0.000000	0.000000	0.000667	0.001332	0.000250	0.000500
12	0.382000	0.472152	0.392667	0.476959	0.357000	0.459102
13	0.000038	0.000075	0.000540	0.001069	0.000405	0.000803
14	0.000000	0.000000	0.000667	0.001332	0.000500	0.000999
15	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
16	0.000037	0.000074	0.000944	0.001884	0.000292	0.000583
17	0.001000	0.001994	0.001667	0.003324	0.000036	0.000071
18	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
19	0.000500	0.000999	0.002667	0.005308	0.002000	0.003984
20	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
21	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
22	0.001000	0.001998	0.006667	0.013244	0.002500	0.004987
23	0.190378	0.189929	0.189108	0.186419	0.204335	0.192871
24	0.028388	0.027229	0.026115	0.021985	0.029382	0.026234
25	0.373220	0.466458	0.365427	0.461462	0.387910	0.473228
26	0.373200	0.467593	0.366880	0.463444	0.387895	0.473811
27	0.123280	0.215380	0.129473	0.224580	0.137270	0.236212
28	0.123290	0.215337	0.128673	0.223099	0.138020	0.236564
29	0.561890	0.459752	0.560007	0.461033	0.534135	0.463373
30	0.042310	0.044359	0.045580	0.047510	0.046225	0.045865
31	0.072850	0.118471	0.083833	0.136430	0.059195	0.099745
32	0.790512	0.297557	0.767511	0.318477	0.789181	0.298373
33	0.440280	0.440056	0.431325	0.425064	0.410585	0.423884
34	0.468580	0.450595	0.466207	0.444736	0.448100	0.444616
35	0.071860	0.070538	0.066747	0.061925	0.067320	0.059903
36	0.090240	0.144834	0.084973	0.134461	0.085345	0.138163
37	0.015440	0.023228	0.017753	0.026378	0.016990	0.025952
38	0.374060	0.465751	0.366613	0.462008	0.387750	0.472819
39	0.373180	0.467368	0.366513	0.463712	0.387390	0.474160
40	0.125070	0.214791	0.130207	0.222810	0.139040	0.235154
41	0.124130	0.214963	0.127520	0.220201	0.139490	0.237537
42	0.495000	0.499950	0.497333	0.499986	0.467000	0.497822

dimensionality reduction to the two datasets using Weka to keep only the features given by the feature selection stage.

The cross-validation technique (Dangelo et al., 2015; Depren et al., 2005; Huang and Wang, 2006; Lin et al., 2015; Sindhu et al., 2012; Singh et al., 2015) is used for performance evaluation to guarantee the reliability of the results. It consists of randomly dividing the dataset into k disjoint parts. One part is used for validating the model and the remaining others are used for training the classifier (Dangelo et al., 2015). This process is repeated k times with different choices of the validation subset (Lin et al., 2015). In the experiment, 10-fold cross validation (Huang and Wang, 2006; Lin et al., 2015; Sindhu et al., 2012) has been performed because of low bias and good error estimate (Singh et al., 2015).

6.6. *Results and comparison*

We executed the GA-LR wrapper program 200 times with different parameter settings. We used the extracted datasets from the KDD99 and the UNSW-NB15 datasets to perform the feature selection stage with different values of population size $p = \{10, 20, 30\}$, different values of crossover probability $p_c = \{0.6, 0.75, 0.9\}$, and different values of mutation probability $p_m = \{0.01, 0.02, 0.03\}$. We set the maximum generation equal to 1000 for GA. Generally, for the predefined weight α used in the fitness function referring to the importance of the accuracy with respect to the number of features can be set from 0.7 to 1 (Chung and Wahid, 2012; Eesa et al., 2015; Huang and Wang, 2006). We defined $\alpha = 0.8$ for all experiments.

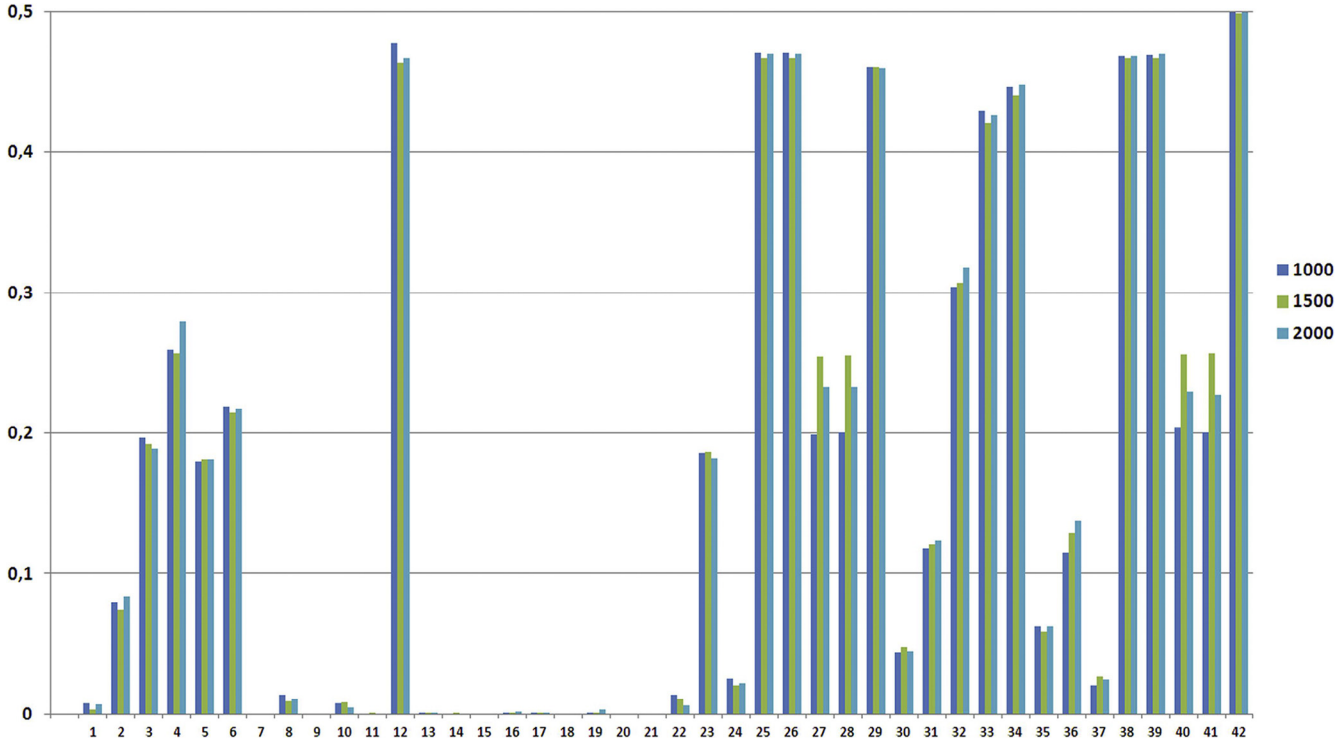

Fig. 5 – Standard deviations of the 42 features in KDD99 training datasets.

We compare the subset of features, the number of features, the accuracy of the subset, the fitness value, and CPU time. According to the obtained results, we deduced that the population size p, crossover probability p_c, mutation probability p_m affect the execution time but not the quality of the resulted subset.

We selected from the obtained results the subsets of features having the best classification accuracy. The selected

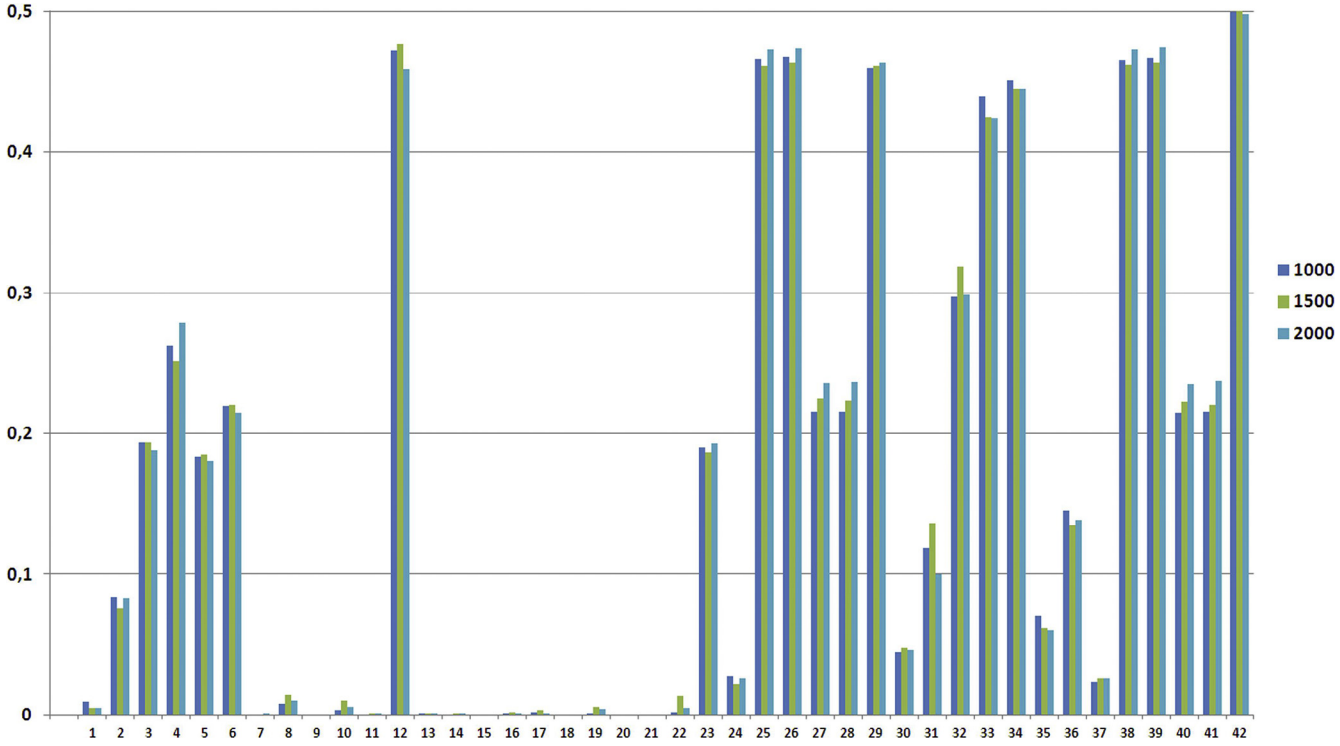

Fig. 6 – Standard deviations of the 42 features in KDD99 testing datasets.

Table 7 – Distribution of records between categories in the KDD99 dataset after redundancy removal.

Categories	Original size	New dataset	New distribution (%)	Reduction (%)
Normal	97,278	87,832	60.33	9.71
DoS	391,458	54,572	37.48	86.06
Probe	4,107	2,131	1.46	48.11
R2L	1,126	999	0.69	11.28
U2R	52	52	0.04	0.00
Total	494,021	145,586	100	70.53

Table 8 – Distribution of records between categories in datasets used in the classification stage.

Dataset	Category	Size	Distribution (%)
KDD99	Normal	30,235	60.47
	DoS	18,620	37.24
	Probe	768	1.54
	R2L	354	0.71
	U2R	23	0.05
	Total	50,000	100
UNSW-NB15	Normal	15,959	31.92
	Backdoor	491	0.98
	Analysis	564	1.13
	Fuzzers	5,180	10.36
	Shellcode	316	0.63
	Reconnaissance	3,065	6.13
	Exploits	9,541	19.08
	DoS	3,501	7.00
	Worms	39	0.08
	Generic	11,344	22.69
	Total	50,000	100

subsets are compared according to AIC and McFadden R^2. Generally, the subset that has the lowest AIC value and the highest R^2 is the best (James et al., 2013).Table 9 illustrates the best subsets obtained after the feature selection stage on the KDD99 datasets. Table 10 illustrates the best subsets obtained after the feature selection stage on the UNSW-NB15 datasets. The subsets presenting the best significant model are used to perform the classification stage with the C4.5, RF, and NBTree classifiers to select the one that provides the best classification accuracy.

The classification stage is performed using Weka. The dataset (extracted from KDD99 or from UNSW-NB15) is loaded and then reduced using the selected subsets before starting classification. Table 11 illustrates the best subsets and their accuracy for the three DTs. We can deduce from these results that the best subset that mostly represent the data is X^1 for the KDD99. X^1 subset gives the best accuracy of classification with 99.90% by using RF classifier. The X^1 accuracy is almost the same as the full features space but with only 18 features.

For the UNSW-NB15, Z^7 subset provides the best accuracy of classification which is about 81.42% by using C4.5 classifier. Also, Z^7 accuracy is almost the same as the full features space but with only 20 features. Table 12 presents a description of the features of the two final selected subsets. The confusion matrix of all KDD99 categories with the subset X^1 by using RF classifier is illustrated in Table 13. The best DR is obtained for *DoS* category with 99.98% while the worst DR is obtained for *U2R* with 52.17%. This can be explained by the limited number of instances in the training dataset for this category of attack. But it looks promising compared to the obtained

Table 9 – The best subsets of features for KDD99 datasets given by the GA-LR wrapper.

X^i	Subset	No.	Accuracy	AIC	McFadden R^2
Datasets size = 1000					
X^1	1, 2, 4, 5, 6, 8, 10, 12, 13, 23, 27, 29, 34, 35, 37, 39, 40, 41	18	0.995	52.5921	0.9894
X^2	2, 4, 5, 6, 10, 13, 16, 17, 19, 23, 25, 29, 30, 36, 39	15	0.995	67.9851	0.9740
X^3	4, 6, 8, 13, 16, 17, 19, 22, 23, 24, 25, 27, 28, 29, 31, 32, 37, 38, 39, 41	20	0.995	72.3113	0.9781
Datasets size = 1500					
X^4	5, 6, 8, 10, 12, 13, 17, 19, 23, 26, 27, 28, 30, 34, 35, 39, 40	17	0.992	133.5	0.9530
X^5	1, 2, 4, 5, 8, 10, 13, 17, 19, 24, 26, 28, 29, 30, 33, 37	16	0.9933	82.3545	0.9767
Datasets size = 2000					
X^6	1, 4, 6, 8, 10, 12, 16, 19, 22, 23, 25, 26, 27, 29, 31, 35, 37, 40	18	0.9915	179.924	0.9488
X^7	1, 2, 4, 5, 8, 10, 13, 16, 17, 22, 23, 24, 26, 28, 29, 30, 34, 35, 37, 38, 39, 41	22	0.9915	266.735	0.9203
X^8	2, 3, 4, 6, 8, 10, 12, 17, 22, 23, 24, 26, 27, 33, 35, 37, 38, 39	18	0.994	179.073	0.9491

Table 10 – The best subsets of features for UNSW-NB15 datasets given by the GA-LR wrapper.

Z^i	Subset	No.	Accuracy	AIC	McFadden R^2
Datasets size = 1000					
Z^1	1, 3, 4, 5, 6, 9, 11, 12, 13, 16, 20, 21, 25, 30, 33, 38, 39, 41	18	0.925	433.894	0.6878
Z^2	1, 4, 9, 11, 14, 16, 18, 20, 22, 26, 27, 29, 30, 33, 37, 38, 39	17	0.924	513.049	0.6238
Z^3	3, 4, 5, 11, 12, 13, 20, 22, 25, 32, 33, 34, 36, 38, 39	15	0.924	472.603	0.6526
Datasets size = 1500					
Z^4	2, 4, 8, 10, 11, 12, 16, 17, 18, 19, 20, 21, 23, 24, 25, 29, 31, 33, 35	19	0.928	562.641	0.7351
Z^5	1, 4, 6, 7, 8, 10, 11, 13, 16, 17, 19, 20, 23, 24, 25, 26, 29, 31, 32, 33, 34, 35, 37, 42	24	0.928	550.972	0.7461
Z^6	4, 8, 10, 11, 12, 15, 16, 17, 18, 19, 20, 21, 22, 25, 27, 28, 31, 32, 33, 34, 35, 38, 39, 40, 42	25	0.9273	571.633	0.7367
Datasets size = 2000					
Z^7	2, 3, 4, 5, 6, 7, 8, 11, 15, 16, 19, 20, 24, 27, 28, 29, 30, 31, 35, 42	20	0.9235	783.107	0.7080
Z^8	1, 2, 3, 4, 6, 10, 11, 14, 16, 17, 19, 23, 24, 27, 31, 32, 35, 37, 40, 41, 42	21	0.922	805.066	0.7001

Table 11 – Accuracy of the three DTs with the selected subsets.

Dataset	Subset	C4.5	RF	NBTree
KDD99	X^1	0.99804	**0.99902**	0.99854
	X^5	0.99756	0.99856	0.99788
	X^8	0.99752	0.99818	0.99720
	All 41	0.99806	0.99914	0.99844
UNSW-NB15	Z^1	0.79572	0.80092	0.79086
	Z^5	0.80984	0.80724	0.80730
	Z^7	**0.81418**	0.81286	0.81090
	All 42	0.81490	0.81408	0.81252

The values in boldface represent the best accuracy of classification obtained with our proposed approach for each dataset.

results by Lin et al. (2015), Powers and He (2008) and Sindhu et al. (2012).

Table 14 illustrates the confusion matrix of C4.5 for all categories over the UNSW-NB15 dataset with the subset Z^7. The best DR is obtained for *Generic* with 97.94% whereas the worst is obtained for *DoS* with 4.11%. The reason is that the majority of *DoS* instances are predicted as *Exploits* which represent a new challenge to defeat in a future work.

Now, if we consider only the normal and attack classes, the problem of misclassification between attack categories will disappear and the classification accuracy is increased. Thus, we deduce that our proposed method gives promising results with an accuracy of classification of about 99.91% with X^1 subset for KDD99 dataset and an accuracy of 94.65% with Z^7 for UNSW-NB15 dataset.

Tables 15 and 16 illustrate the performance of our GALR-DT method compared to other methods using feature selection respectively with X^1 subset on KDD99 and Z^7 subset on UNSW-NB15 datasets. More precisely, we compare the feature selection method, the number of selected features, the classifier, accuracy, DR, and FAR with the used classifier. X^1 subset illustrates the highest DR with 99.81% and gives a lower FAR with 0.105%. Z^7 subset provides the lowest FAR with 6.39% and a good classification accuracy compared to the other mentioned approaches. Figs. 7 and 8 illustrate a comparison between our proposed GALR-DT method and other IDS approaches respectively on the KDD99 and the UNSW-NB15 datasets.

Table 12 – Feature description of X^1 and Z^7 subsets.

Subset	No.	Features
X^1	18	duration, protocol_type, flag, src_bytes, dst_bytes, wrong_fragment, hot, logged_in, lnum_compromised, count, rerror_rate, same_srv_rate, dst_host_same_srv_rate, dst_host_diff_srv_rate, dst_host_srv_diff_host_rate, dst_host_srv_serror_rate, dst_host_rerror_rate, dst_host_srv_rerror_rate
Z^7	20	proto, service, state, spkts, dpkts, sbytes, dbytes, dttl, dloss, sinpkt, djit, swin, tcprtt, smean, dmean, trans_depth, response_body_len, ct_srv_src, ct_dst_sport_ltm, is_sm_ips_ports

Table 13 – Confusion matrix of all categories over the KDD99 dataset with the subset X^1 using RF classifier.

Actual	Predicted					Recall (%)
	Normal	DoS	Probe	R2L	U2R	
Normal	30228	1	1	3	2	99.977
DoS	3	18617	0	0	0	99.984
Probe	10	2	756	0	0	98.438
R2L	13	1	0	338	2	95.480
U2R	11	0	0	0	12	52.174
Precision (%)	99.878	99.979	99.868	99.120	75	

Table 14 – Confusion matrix of all categories over the UNSW-NB15 dataset with the subset Z^7 using C4.5 classifier.

Actual	Predicted										Recall (%)
	Normal	Backdoor	Analysis	Fuzzers	Shellcode	Reconnaissance	Exploits	DoS	Worms	Generic	
Normal	14478	3	48	1197	16	18	174	23	0	2	90.720
Backdoor	6	34	0	6	3	3	430	8	1	0	6.925
Analysis	57	0	56	4	0	0	439	7	0	1	9.929
Fuzzers	918	8	4	3580	56	4	587	19	1	3	69.112
Shellcode	32	2	0	69	150	7	45	10	0	1	47.468
Reconnaissance	5	2	0	3	3	2334	704	12	2	0	76.150
Exploits	143	10	14	155	35	185	8808	164	11	16	92.317
DoS	28	6	4	50	17	16	3227	144	2	7	4.113
Worms	0	0	0	2	0	1	18	3	15	0	38.462
Generic	5	1	0	22	0	0	197	9	0	11110	97.937
Precision (%)	92.381	51.515	44.444	70.362	53.571	90.888	60.209	36.090	46.875	99.731	

Table 15 – Performance of our proposed method compared to other methods using feature selection for KDD99 dataset.

Work	FS components	No.	Classifier	Accuracy (%)	DR (%)	FAR (%)
Chung and Wahid (2012)	IDS-RS	6	SSO	93.3	–	–
Eesa et al. (2015)	CFA, DT	5	None	91.99	91	3.917
Kavitha et al. (2012)	BFS	7	NL	–	99.02	3.19
Lin et al. (2012)	SVM, SA	23	DT, SA	99.96	99.95	0.021
Sindhu et al. (2012)	GA, Neurotree	16	Neurotree	–	98.38	1.62
Mok et al. (2010)	SFS, FELR	5	RELR	98.74	99.39	1.42
GALR-DT	GA-LR	18	DT	**99.90**	**99.81**	**0.105**

The values in boldface represent the results obtained by our proposed approach for the KDD99 dataset.

7. Conclusion

In this paper, we have proposed a feature selection approach for IDS to produce the optimal subset of features that can be used to classify the instances of KDD99 and UNSW-NB15 datasets. The proposed approach is based on three stages: a preprocessing stage, a feature selection stage, and a classification stage. The preprocessing stage consists of reducing the size of the datasets through resampling, changing the attribute values in a way to be handled with LR classifier, and removing redundant records if they exist. The feature selection stage is based on GA-LR wrapper that is consists of an interaction between a feature search which is GA and a learning algorithm which is LR. The selection of the best subset is based on maximising the accuracy of classification and minimising the number of features. The best resulted subsets of features from the GA-LR wrapper are used to perform the classification stage. The classification stage is performed by using three decision tree classifiers namely C4.5, RF and NBTree to evaluate the generated subsets of features and compare them with other existing approaches.

Table 16 – Performance of our proposed method compared to other methods using feature selection for UNSW-NB15 dataset.

Work	FS components	No.	Classifier	Accuracy (%)	FAR (%)
Moustafa and Slay (2016)	None	42	DT	85.56	15.78
			LR	83.15	18.48
			NB	82.07	18.56
			ANN	81.34	21.13
			EM	78.47	23.79
GALR-DT	GA-LR	20	DT	**81.42**	**6.39**

The values in boldface represent the results obtained by our proposed approach for the UNSW-NB15 dataset.

The experimental results are promising with an accuracy of classification equal to 99.90%, 99.81% DR and 0.105% FAR with a subset of only 18 features for the KDD99 dataset. Furthermore, the selected subset provides a good DR for *DoS* category with 99.98%. The obtained results for the UNSW-NB15 provide

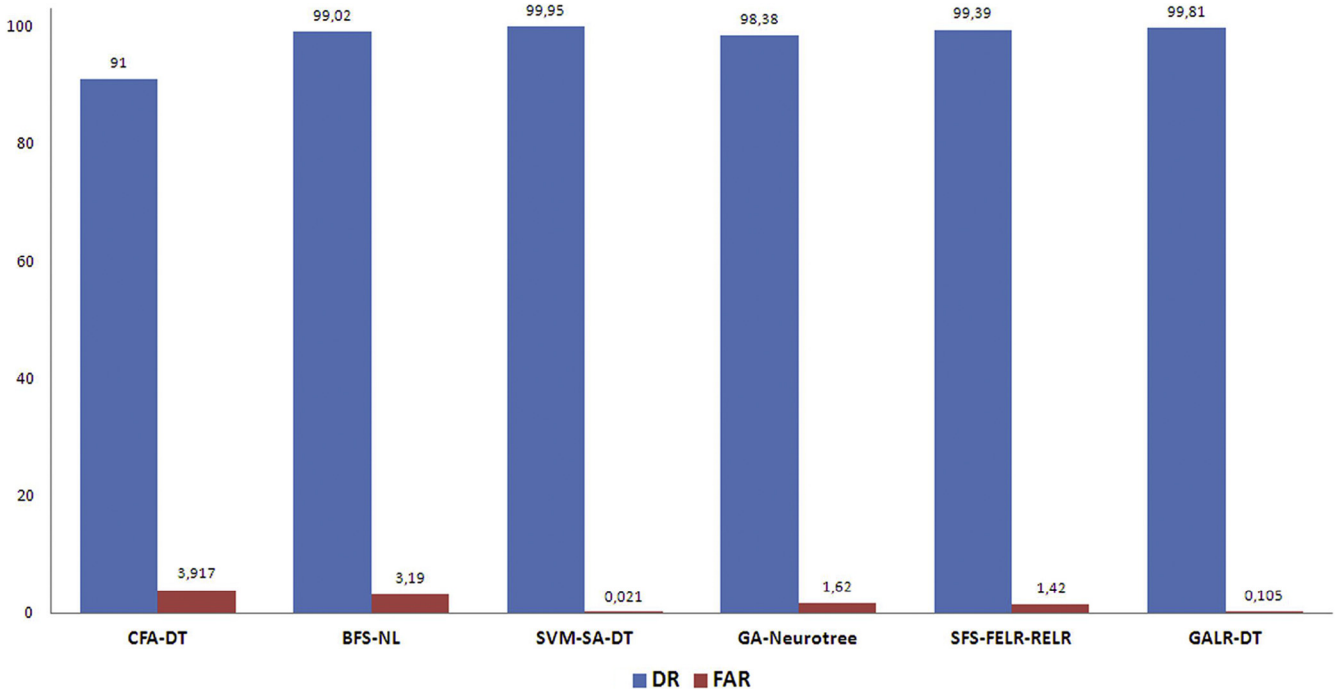

Fig. 7 – DR and FAR of our proposed method compared to other IDSs KDD99 dataset.

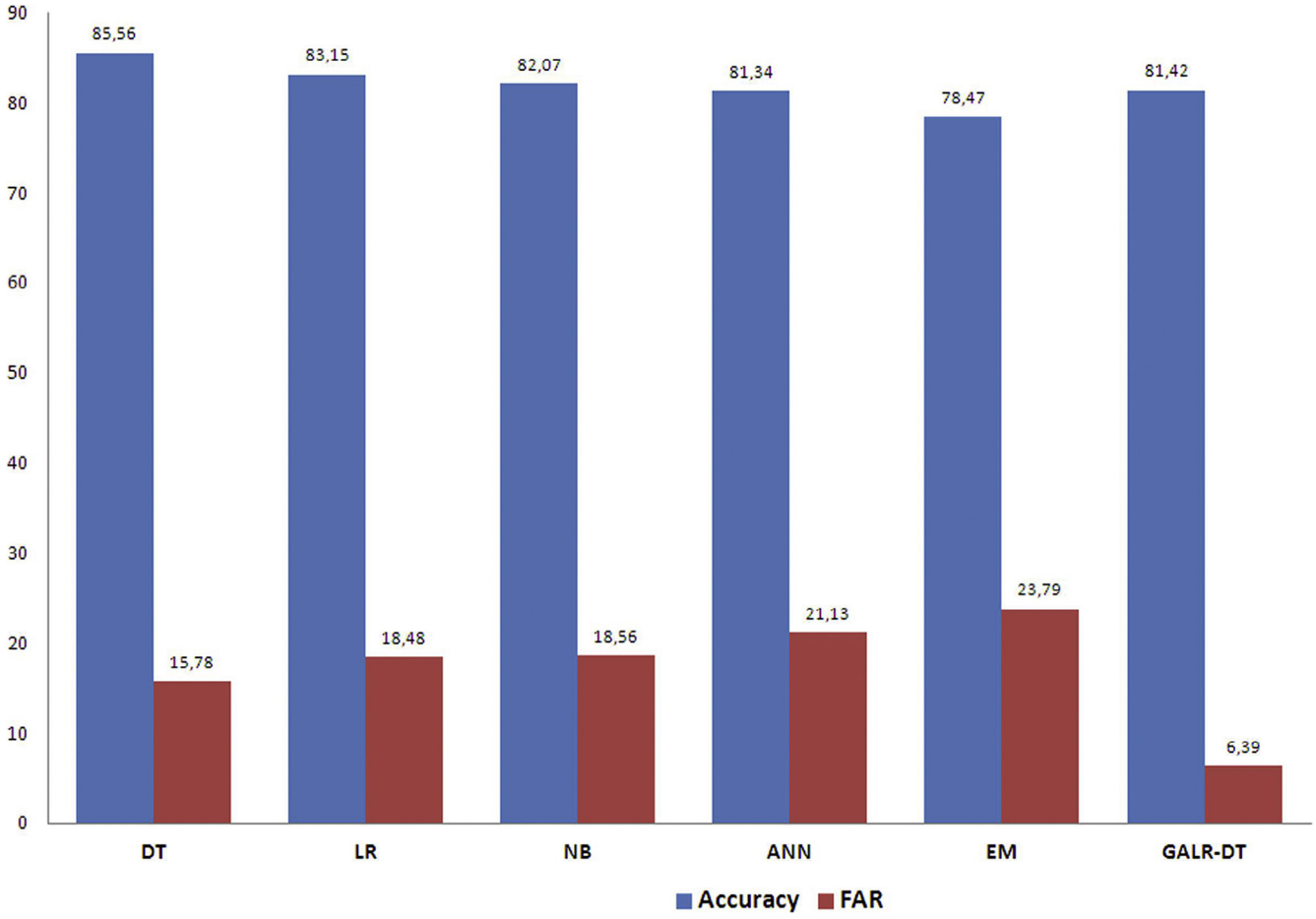

Fig. 8 – Accuracy and FAR of our proposed method compared to other IDSs using UNSW-NB15 dataset.

the lowest FAR with 6.39% and a good classification accuracy compared to the other mentioned approaches with a subset composed of 20 features. These findings led to the fact that the UNSW-NB15 dataset is more complex than the KDD99 dataset. Thus, we must try other approaches to improve the accuracy of classification for the new IDS benchmark.

As a future work, it is interesting to apply the GA-LR wrapper approach to extract the optimal subset of features for each class with a multi-objective approach which increase the accuracy of classification and decrease the misclassified instances. Furthermore, it seems promising to use multinomial LR (Czepiel, 2002) instead of binomial LR to predict all classes and not only for normal and attack classes.

REFERENCES

Aburomman AA, Reaz MBI. A novel svm-knn-pso ensemble method for intrusion detection system. Appl Soft Comput 2016;38:360–72.

Ahmed M, Mahmood AN, Hu J. A survey of network anomaly detection techniques. J Netw Comput Appl 2016;60:19–31.

Alba E, Garca-Nieto J, Jourdan L, Talbi E-G. Gene selection in cancer classification using pso/svm and ga/svm hybrid algorithms. In: IEEE congress on evolutionary computation, 2007 (CEC 2007). IEEE; 2007. p. 284–290.

Bahl S, Sharma SK. A minimal subset of features using correlation feature selection model for intrusion detection system. In: Proceedings of the second international conference on computer and communication technologies. Springer; 2016. p. 337–346.

Boln-Canedo V, Snchez-Maroo N, Alonso-Betanzos A. Feature selection for high-dimensional data. Berlin: Springer; 2016.

Buczak AL, Guven E. A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Commun Surv Tutor 2016;18(2):1153–76.

Bulajoul W, James A, Pannu M. Improving network intrusion detection system performance through quality of service configuration and parallel technology. J Comput Syst Sci 2015;81(6):981–99.

Chandrashekar G, Sahin F. A survey on feature selection methods. Comput Electr Eng 2014;40(1):16–28.

Chung YY, Wahid N. A hybrid network intrusion detection system using simplified swarm optimization (sso). Appl Soft Comput 2012;12(9):3014–22.

Costa KA, Pereira LA, Nakamura RY, Pereira CR, Papa JP, Falcão AX. A nature-inspired approach to speed up optimum-path forest clustering and its application to intrusion detection in computer networks. Inf Sci (Ny) 2015;294:95–108.

Czepiel SA. Maximum likelihood estimation of logistic regression models: theory and implementation; 2002. Available from: http://www.czep.net/stat/mlelr.pdf.

Dangelo G, Palmieri F, Ficco M, Rampone S. An uncertainty-managing batch relevance-based approach to network anomaly detection. Appl Soft Comput 2015;36:408–18.

De la Hoz E, de la Hoz E, Ortiz A, Ortega J, Martínez-Álvarez A. Feature selection by multi-objective optimisation: application to network anomaly detection by hierarchical self-organising maps. Knowl Based Syst 2014;71:322–38.
De la Hoz E, De La Hoz E, Ortiz A, Ortega J, Prieto B. Pca filtering and probabilistic som for network intrusion detection. Neurocomputing 2015;164:71–81.
Depren O, Topallar M, Anarim E, Ciliz MK. An intelligent intrusion detection system (ids) for anomaly and misuse detection in computer networks. Expert Syst Appl 2005;29(4):713–22.
Eesa AS, Orman Z, Brifcani AMA. A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems. Expert Syst Appl 2015;42(5):2670–9.
Elhag S, Fernández A, Bawakid A, Alshomrani S, Herrera F. On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on intrusion detection systems. Expert Syst Appl 2015;42(1):193–202.
Fessi BA, Benabdallah S, Boudriga N, Hamdi M. A multi-attribute decision model for intrusion response system. Inf Sci (Ny) 2014;270:237–54.
Gan X-S, Duanmu J-S, Wang J-F, Cong W. Anomaly intrusion detection based on pls feature extraction and core vector machine. Knowl Based Syst 2013;40:1–6.
Garcia-Teodoro P, Diaz-Verdejo J, Maciá-Fernández G, Vázquez E. Anomaly-based network intrusion detection: Techniques, systems and challenges. Comput Sec 2009;28(1):18–28.
Garner SR. Weka: the Waikato environment for knowledge analysis. In: Proceedings of the New Zealand computer science research students conference. Citeseer; 1995. p. 57–64.
Ghorbani AA, Lu W, Tavallaee M. Network intrusion detection and prevention: concepts and techniques, vol. 47. Berlin: Springer Science & Business Media; 2009.
Ghosh P, Mitra R. Proposed ga-bfss and logistic regression based intrusion detection system. In: 2015 Third international conference on IEEE computer, communication, control and information technology (C3IT); 2015. p. 1–6.
Guyon I, Elisseeff A. An introduction to variable and feature selection. J Mach Learn Res 2003;3:1157–82.
Guyon I, Gunn S, Nikravesh M, Zadeh LA. Feature extraction: foundations and applications, vol. 207. Berlin: Springer; 2008.
Haider W, Hu J, Xie M. Towards reliable data feature retrieval and decision engine in host-based anomaly detection systems. In: 2015 IEEE 10th conference on industrial electronics and applications (ICIEA). IEEE; 2015. p. 513–517.
Haider W, Hu J, Slay J, Turnbull B, Xie Y. Generating realistic intrusion detection system dataset based on fuzzy qualitative modeling. J Netw Comput Appl 2017;87:185–92.
Han J, Kamber M, Pei J. Data mining: concepts and techniques. Amsterdam: Elsevier; 2011.
Huang C-L, Wang C-J. A ga-based feature selection and parameters optimization for support vector machines. Expert Syst Appl 2006;31(2):231–40.
James G, Witten D, Hastie T, Tibshirani R. An introduction to statistical learning, vol. 112. Berlin: Springer; 2013.
Kang S-H, Kim KJ. A feature selection approach to find optimal feature subsets for the network intrusion detection system. Cluster Comput 2016;1–9.
Karami A, Guerrero-Zapata M. A fuzzy anomaly detection system based on hybrid pso-kmeans algorithm in content-centric networks. Neurocomputing 2015;149:1253–69.
Katos V. Network intrusion detection: evaluating cluster, discriminant, and logit analysis. Inf Sci (Ny) 2007;177(15):3060–73.
Kavitha B, Karthikeyan S, Maybell PS. An ensemble design of intrusion detection system for handling uncertainty using neutrosophic logic classifier. Knowl Based Syst 2012;28:88–96.
KDD Cup 1999 data. The UCI KDD Archive, Information and Computer Science. Irvine, CA: University of California; 1999. Available from: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
Kim G, Lee S, Kim S. A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. Expert Syst Appl 2014;41(4):1690–700.
Kohavi R. Scaling up the accuracy of naive-Bayes classifiers: a decision-tree hybrid. In: KDD, Vol. 96, Citeseer, p. 202–207, 1996.
Kou G, Peng Y, Chen Z, Shi Y. Multiple criteria mathematical programming for multi-class classification and application in network intrusion detection. Inf Sci (Ny) 2009;179(4):371–81.
Liang H, Yan Y. Learning naive Bayes tree for conditional probability estimation. In: Conference of the Canadian Society for Computational Studies of Intelligence. Springer; 2006. p. 455–466.
Lin S-W, Ying K-C, Chen S-C, Lee Z-J. Particle swarm optimization for parameter determination and feature selection of support vector machines. Expert Syst Appl 2008;35(4):1817–24.
Lin S-W, Ying K-C, Lee C-Y, Lee Z-J. An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection. Appl Soft Comput 2012;12(10):3285–90.
Lin W-C, Ke S-W, Tsai C-F. Cann: an intrusion detection system based on combining cluster centers and nearest neighbors. Knowl Based Syst 2015;78:13–21.
Liu H, Motoda H. Computational methods of feature selection. Boca Raton (FL): CRC Press; 2007.
Louvieris P, Clewley N, Liu X. Effects-based feature identification for network intrusion detection. Neurocomputing 2013;121:265–73.
McHugh J. Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by Lincoln laboratory. ACM Trans Inform Syst Sec 2000;3(4):262–94.
Modi C, Patel D, Borisaniya B, Patel H, Patel A, Rajarajan M. A survey of intrusion detection techniques in cloud. J Netw Comput Appl 2013;36(1):42–57.
Mok MS, Sohn SY, Ju YH. Random effects logistic regression model for anomaly detection. Expert Syst Appl 2010;37(10):7162–6.
Moustafa N, Slay J. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In: Military communications and information systems conference (MilCIS). IEEE; 2015. p. 1–6.
Moustafa N, Slay J. The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set. Inform Sec J 2016;25(1–3):18–31.
Powers ST, He J. A hybrid artificial immune system and self organising map for network intrusion detection. Inf Sci (Ny) 2008;178(15):3024–42.
Rastegari S, Hingston P, Lam C-P. Evolving statistical rulesets for network intrusion detection. Appl Soft Comput 2015;33:348–59.
Shiravi A, Shiravi H, Tavallaee M, Ghorbani AA. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Comput Sec 2012;31(3):357–74.
Sindhu SSS, Geetha S, Kannan A. Decision tree based light weight intrusion detection using a wrapper approach. Expert Syst Appl 2012;39(1):129–41.
Singh R, Kumar H, Singla R. An intrusion detection system using network traffic profiling and online sequential extreme learning machine. Expert Syst Appl 2015;42(22):8609–24.

Tsai C-F, Hsu Y-F, Lin C-Y, Lin W-Y. Intrusion detection by machine learning: a review. Expert Syst Appl 2009;36(10):11994–2000.
UNSW-NB15 dataset; UNSW Canberra at the Australian Defence Force Academy, Canberra, Australia; 2015. Available from: https://www.unsw.adfa.edu.au/australian-centre-for-cyber-security/cybersecurity/ADFA-NB15-Datasets/.
Wu SX, Banzhaf W. The use of computational intelligence in intrusion detection systems: a review. Appl Soft Comput 2010;10(1):1–35.
Wu X, Kumar V, Quinlan JR, Ghosh J, Yang Q, Motoda H, et al. Top 10 algorithms in data mining. Knowl Inf Syst 2008;14(1):1–37.
Yu L, Liu H. Efficient feature selection via analysis of relevance and redundancy. J Mach Learn Res 2004;5:1205–24.
Zhang J, Zulkernine M, Haque A. Random-forests-based network intrusion detection systems. IEEE Trans Syst Man Cybern C Appl Rev 2008;38(5):649–59.
Zhang J, Li H, Gao Q, Wang H, Luo Y. Detecting anomalies from big network traffic data using an adaptive detection approach. Inf Sci (Ny) 2015;318:91–110.

Saoussen Krichen is a Full Professor and director of the Doctoral School of Higher Institute for Management of Tunis, University of Tunis from Tunisia. She is also head of the team "Decision Making" at the LARODEC laboratory. She works on big data analytics, single and multiobjective optimisation, dynamic optimisation, coalition formation and supply chain management. Her research team counts 25 Ph.D. students and 100 master students. She published numerous books in Wiley, Taylor & Francis and ISTE and more than 150 papers in impacted journals and international conferences. Her scientific partnership covers many countries as France (Kedge, Université Bordeaux I, Paris VI, Université de Strasbourg and École Centrale de Lille), Canada (Université Laval Quebec, HEC Montreal), Sultanate Oman (SQU, College of Arts and Applied Sciences, Dhofar university), Spain (University of Malaga) and Jordan (Applied Science University) and Palestine (Palestine Ahliya University College).

Chaouki Khammassi is a Ph.D. student and LARODEC laboratory member at the Higher Institute for Management of Tunis, University of Tunis from Tunisia. He is a computer science teacher at the Higher Institute of Commerce and Accounting of Bizerte, University of Carthage from Tunisia. He has a master degree in Enterprise Systems Engineering (Tempus Program) from the Higher Institute for Management of Tunis and a master degree in Computer Science and New Educational Technology from the Higher Institute of Education and Continuing Training, Virtual University of Tunis. He works on machine learning and data analytics. He worked in Access-Technologies Company and Millénia Engineering Company in Tunisia for developing software, PC maintenance, systems security and networks setup.