Feature Selection for Intrusion Detection System Using Ant Colony Optimization

Mehdi Hosseinzadeh Aghdam[1], and Peyman Kabiri[2]
(Corresponding author:Mehdi Hosseinzadeh Aghdam)
Department of Computer Engineering and Information Technology, Payame Noor University (PNU)[1]
P.O. BOX 19395-3697, Tehran, Iran
School of Computer Engineering, Iran University of Science and Technology[2]
Tehran, 16846-13114, Iran
(Email: mhaghdam@iust.ac.ir)
(Received March 30, 2014; revised and accepted Jan. 16 & Mar. 4, 2015)

Abstract

Intrusion detection is a major research problem in network security. Due to the nonlinear nature of the intrusion attempts, unpredictable behavior of the network traffic and the large number of features in the problem space, intrusion detection systems represent a complicated problem area. Choosing effective and key features for intrusion detection is a very important topic in information security. The purpose of this study is to identify important features in building an intrusion detection system such that they are computationally efficient and effective. To improve the performance of intrusion detection system, this paper proposes an intrusion detection system that its features are optimally selected using ant colony optimization. The proposed method is easily implemented and has a low computational complexity due to use of a simplified feature set for the classification. The extensive experimental results on the KDD Cup 99 and NSL-KDD intrusion detection benchmark data sets demonstrate that the proposed method outperforms previous approaches, providing higher accuracy in detecting intrusion attempts and lower false alarm with reduced number of features.

Keywords: Ant colony optimization, feature selection, intrusion detection system

1 Introduction

Intrusion Detection Systems (IDSs) have become important and widely used tools for ensuring network security. In recent years, intrusion detection based on statistical pattern recognition methods has attracted a wide range of interest in response to the growing demand of reliable and intelligent IDSs, which are required to detect sophisticated and polymorphous intrusion attacks [7]. In general, IDSs are usually classified into two categories in the literature: signature-based intrusion detection and anomaly-based intrusion detection [33]. Signature-based intrusion detection, misuse detection, approach can reliably identify intrusion attacks in relation to the known signatures of discovered vulnerabilities. Therefore, well-known intrusions can be detected efficiently with a very low false positive rate. Intrusions are usually polymorphic, and evolve continuously. Signature-based detection will fail easily when facing unknown intrusions and emergent intervention of security experts is required to define signatures or accurate rules, which limits the application of the signature-based detection approach to build intelligent IDSs. Signature-based detection is commonly used in the design of commercial IDSs [38].

Anomaly-based IDS usually deals with statistical analysis and pattern recognition problems. It can detect zero day attacks if the classification model has the generalization capability to extract intrusion pattern and knowledge during the training period [33]. In the anomaly-based intrusion detection approach, the system builds models for normal behavior in the network traffic and detects any deviation from this model as a potential intrusion attempt. This approach commonly suffers from high false positive rate on classifying normal network traffic. Due to the dynamic nature of the network traffic, discovering boundaries between normal and abnormal behavior is a major difficulty in this approach [38]. To overcome the anomaly intrusion detection problem, the data mining [14], machine learning [25] and immune system [37] approaches have been proposed in recent years.

Considering the available computational power and due to the large amount of audit data with a large number of features that IDS needs to examine, even for a small network, traffic analysis is a difficult task. Audit data captures various features of the connections. For example, the audit data would show the network service on the destination (http, telnet, etc.), or the number of wrong fragments or length of the connection (second). Selected features in the problem space might be correlated, which

is difficult for humans to discover. An IDS operating in real-time needs a reduced amount of data for the processing. Some of these features are irrelevant and redundant and thus can be eliminated before the processing [7]. Most of these features are not applicable to intrusion detection; even some noise data may badly affect the performance of the process of detecting intrusions. Extra and noisy features can increase the computation time, and can have a negative impact on the accuracy of the IDS [28]. Hence, we need to select some representative features from the original feature space to reduce the dimensionality of feature space and improve the efficiency and performance of IDS.

Feature Selection (FS) is a commonly used step in machine learning, especially when dealing with a high dimensional space of features. The objective of FS is to simplify a data set by reducing its dimensionality and identifying relevant underlying features without sacrificing predictive accuracy. By doing that, it also reduces redundancy in the information provided by the selected features. In real world problems FS is a must due to the abundance of noisy, irrelevant or misleading features. FS is extensive and it spreads throughout many fields, including text categorization, data mining, pattern recognition, signal processing and intrusion detection [1, 2].

Recently, Ant Colony Optimization (ACO) is gaining popularity as a new approach to the FS [1, 13, 36]. Some literatures provided experimental results that illustrated ACOs superiority over conventional approaches, such as sequential search and the genetic algorithm [13]. Dorigo and colleagues introduced meta-heuristic optimization algorithm based on behavior of ants in the early 1990s. ACO is a new solution finding method in artificial intelligence called Swarm Intelligence (SI). In SI interaction of cooperative individuals is such that a problem-solving behavior emerges. SI is the property of a system whereby the collective behaviors of unsophisticated individuals interacting locally with their environment cause coherent functional global patterns to emerge. Insects such as ants and bees live in colonies. An individual can only do simple behavior on its own, while their colonial cooperative work represents a complex behavior [10]. ACO algorithm is inspired by social behavior of ant colonies. Although they have no sight, ants are capable of finding the shortest route between a food source and their nest by chemical materials called pheromone that they leave when moving [6].

ACO algorithm was originally applied to the travelling salesman problem and later on, it was successfully applied to other optimization problems such as the quadratic assignment problem [18], routing in telecommunication networks, graph coloring problems, scheduling, etc. This method is particularly attractive for FS as there seems to be no heuristic that can lead the search to the optimal minimal subset every time [1]. In this paper a novel intrusion detection system using ACO has been introduced. The classifier performance and the length of selected feature subset are adopted as heuristic information for ACO. Thus, the proposed method needs no prior knowledge of features. Also the classifier performance and the length of selected feature vector are considered for performance evaluation. Finally, the proposed method is applied to KDD Cup 99 and NSL-KDD, the new version of KDD Cup 99, data sets.

The rest of this paper is organized as follows. Section 2 outlines the related work about feature selection methods in intrusion detection. The proposed ACO-based method for IDS is described in section 3. Section 4 reports computational experiments. It also includes a brief discussion of the results obtained and finally the conclusion and future works are offered in the last section.

2 Related Work

Feature selection is a process that selects a subset of original features. The significance of FS can be viewed in two facets. The frontier facet is to filter out noise and eliminate irrelevant and redundant features. FS is compulsory due to the abundance of noisy, irrelevant or misleading features in a data set. Second, FS can be considered as an optimization problem for an optimal subset of features that better satisfy a desired measure [12]. Quality of the FS optimization can be measured using certain evaluation criteria. Since FS is a NP-hard problem, there is no practical solution to find its optimal feature subset [17]. A typical FS process includes *subset generation, subset evaluation, termination criteria and result validation.* Subset selection procedure implements a search method that selects feature subsets for evaluation based on a certain search strategy. It may start with empty subset, full subset, a selected feature subset or some random feature subset. Those methods that start with an initial subset usually select this feature subset using heuristic methods. These search methods include forward selection, backward elimination and forward/backward combination methods. The process of subset selection and evaluation is repeated until a given termination condition is satisfied. The selected best feature subset usually needs to be validated using a different test data set [17].

Volume of the network traffic data and the high dimensionality of the feature space are main causes for prohibitively high processing overhead. This is why they are major problems in IDS design. Therefore, FS is an important phase in IDS design. Variation of the normal traffic often hinders anomaly-based IDS ability to accurately model normal state of the operation of the network. In any intrusion attempt there are some behavioral patterns and interrelations that are unique and recognizable. Since these patterns are hidden within the irrelevant and redundant features it is often difficult to discover them. Eliminating the less relevant features can improve both the speed and the accuracy of the classification [29].

In accordance to the literature, approaches to FS can be divided into filters, wrappers and embedded approaches [12]. The filter model separates FS from learning

algorithm and selects feature subsets that are independent of any learning algorithm. It relies on various measures of the general characteristics of the training data such as distance, information, dependency, and consistency. In the wrapper approach feature subset is selected using the evaluation function based on the same learning algorithm that will be used later for learning [17]. In this approach the evaluation function finds the optimal feature subset using the subset generation procedure. Finally, comparing the result with best subset resulted from the previous iterations subset evaluation procedure only keeps the best subset. Subset evaluation procedure tests the best subset against the termination criteria to determine if the selection process should end. Although, wrappers may generate a better result, their execution cost is high and may encounter problem in dealing with feature spaces with the very high dimensions. This is due to the use of learning algorithms in the evaluation of subsets, some of which can encounter problems while dealing with a high dimensional space of features [4, 12]. If the FS and learning algorithm are interleaved then the FS approach is a kind of embedded approach [20].

Feature selection is an important issue in intrusion detection. Elimination of the insignificant features may enhance accuracy of the detection. By concentrating on the most important features execution speed of the process can be increased without significant effect on the accuracy of detection. Chebrolu et al. employed the effectiveness of IDS in terms of real-time performance and detection accuracy from the feature selection perspective [7]. In the reported work, features were selected using two methods, Bayesian network and classification and regression trees. Four different sets of features were derived and used in their ensemble method for IDS. In their experiments, KDD Cup 99 data set is used. A very high detection rate is reported in their experiments. Their approach uses only 5092 records for the training and 6890 records for the test. They have proposed no generic subset. Their reported results show that different feature subsets with different length are more effective in detecting various types of attacks [7].

Sung and Mukkamala proposed a well-known closed-loop FS method for SVM-based IDS, called SVM-RFE, which recursively eliminated one feature at a time and compared the resulting performance in each SVM test [28]. They also ranked six significant features [29]. They used three methods and compared the performance of these methods in terms of classification accuracy on the test data set. In the reported work, they used support vector decision function ranking, linear genetic programming and multivariate adaptive regression splines [29]. Sung et al. reported a new feature subset that provides accurate results for the detection of different types of attacks [30]. They have used genetic algorithm to maximize inter-class difference and minimize size of the subset. Mukkamala and Sung applied SVM-RFE method to the KDD Cup 99 data and performed the feature ranking for FS [22]. They ranked the features into three categories: important, secondary, and insignificant according to three main performance criteria: overall accuracy of classification, training time, and testing time. 19 important features were identified and used in the experiments. This heuristic-based technique is time consuming. Additionally, unknown attack types are not considered in the reported work since the same data set (kddcup.data-10-percent.gz) was applied to the experiment.

Zhang et al. considered the capability of rough set theory in coming up with classification rules in detecting the attacks [39]. They showed that rough set classification using GA can produce a high detection rate. Speed of the feature ranking in the reported work is fast. They did not report the selected features used for classification process. Ohn et al. adopted genetic algorithm to find optimal feature subset for SVM [23]. 31 features were used with radial kernel function in their experiment and a very high detection rate was obtained for the original KDD Cup 99 test data set (corrected.gz). Since their training data set was sampled from the full data set (kddcup.data.gz), the challenge of the problem was reduced. The reason is that the number of attack types in the original training data set (kddcup.data-10-percent.gz) is intentionally employed more than the test data set (corrected.gz). These two data sets can challenge the methods of detecting the unknown type attack.

From these reported works, we can conclude that some features are really significant in intrusion detection. Also, it has been proven that there is no single generic classifier that can best classify all the attack types. Instead, in some cases, specific classifier performs better than others. Thus, most of these works lead to an ensemble or fusion of multiple classifier IDS.

3 Proposed ACO-based Method for Intrusion Detection System

In the early nineties an algorithm called Ant System (AS) was proposed by Dorigo and colleagues as a novel nature-inspired meta-heuristic approach for the solution of combinatorial optimization problems. First, algorithm was applied to the traveling salesman problem. Recently, it was extended and/or modified both to improve its performance and to apply it to other optimization problems. Improved versions of AS include, among others, Ant Colony System (ACS), MAX-MIN, AS and AS-rank. An ant colony optimization algorithm is essentially a system based on individuals which simulate the natural behavior of ants, including mechanisms of cooperation and adaptation. The inspiring source of ACO is the foraging behavior of real ants [9]. The ACO algorithm is based on a computational paradigm inspired by real ant colonies and the way they function. The idea is to use several constructive computational ants. Based on the results of previous experiments stored in the ant dynamic memory structure, each ant is guided to the constructed solution. The paradigm is based on the observation made by ethol-

ogists about the medium used by ants to communicate information regarding shortest paths to food by means of pheromone trails. While an isolated ant moves practically at random, exploration, an ant encountering a previously laid trail can detect it and decide with high probability to follow it, exploitation, and consequently reinforces the trail with its own pheromone. What emerges is a form of autocatalytic process through which the more the ants follow a trail, the more attractive that trail becomes to be followed. The process is thus characterized by a positive feedback loop, during which the probability of choosing a path increases with the number of ants that previously chose the same path. The mechanism above is the inspiration for the algorithms of the ACO family [21].

Before the training phase, a FS phase may also be considered. The FS process identifies which features are more discriminative than the others. This has the benefit of generally improving system performance by eliminating irrelevant and redundant features. In general, FS is not very popular procedure in IDS. However, a few studies use different FS methods for their experiments. This implies that FS could improve some certain level of classification accuracy in IDS. Given a feature set of size N, the FS problem is to find a minimal feature subset of size S ($S < N$) while retaining the previous accuracy. Therefore, there is no concept of solution path in FS problem. A partial solution, i.e. subset, does not define any order among the components, i.e. features of the solution, and the next component to be selected is not necessarily influenced by the last component added to the partial solution [5, 15]. FS problem solutions are not necessarily of the same size. The first step in FS using ACO is to address the problem of redefining the way that the ACO representation graph is used.

3.1 Graph Representation

The main goal of ACO algorithm is to model a problem as the search for a minimum cost path in a graph. Here nodes can be considered as features, with the edges between them denoting the choice of the next feature. The search for the optimal feature subset is then an ant traversal through the graph where a minimum number of nodes, features, are visited that satisfies the traversal stopping criterion. Nodes are fully connected to allow any feature to be selected next. On the basis of this reformulation of the graph representation, the transition rules and pheromone update rules of standard ACO algorithms can be applied. In this case, pheromone and heuristic value are not associated with links. Instead, each feature has its own pheromone value and heuristic value [1].

3.2 Heuristic Information

Generally, the representation of heuristic value is the attractiveness of features and the basic ingredient of any ACO algorithm is a constructive heuristic for probabilistically constructing solutions. A constructive heuristic assembles solutions as sequences of features from the finite set of features. A subset construction starts with an empty subset. Then, at each construction step the current subset is extended by adding a feature from the set of features. A suitable heuristic desirability of traversing between features could be any subset evaluation. In proposed method classifier performance is mentioned as heuristic information for FS. In other words, the classifier accuracy of each feature on training set is considered as heuristic information for each feature. The heuristic information of traversal and node pheromone levels are combined to form the so called probabilistic transition rule, denoting the probability that ant k will include feature i in its subset at time step t:

$$P_i^k(t) = \begin{cases} \frac{[\tau_i(t)]^\alpha . [\eta_i]^\beta}{\sum_{u \in J^k} [\tau_u(t)]^\alpha . [\eta_u]^\beta} & if i \in J^k \\ 0 & otherwise \end{cases} \quad (1)$$

where J^k is the set of feasible features that ant k can be added to its subset; τ_i and η_i are respectively the pheromone value and heuristic information associated with feature i. α and β are two parameters that determine the relative weight of the pheromone value and heuristic information. The transition probability used by ACO is a balance between pheromone intensity (i.e. history of previous successful moves), τ_i, and heuristic information (expressing desirability of the move), η_i. This effectively balances the exploitation-exploration trade-off. The best balance between exploitation and exploration is achieved through proper selection of the parameters α and β. If α=0, no pheromone information is used, i.e. previous search experience is neglected. The search then degrades to a stochastic greedy search. If β=0, the attractiveness (or potential benefit) of moves is neglected.

3.3 Pheromone Update

Pheromone updating is an important part for working the ACO algorithm suitably. After all ants have completed their solutions, pheromone evaporation on all nodes is triggered using Equation (2) and then according to Equation (3) all ants deposit a quantity of pheromone, $\Delta\tau_i(t)$, on each node that they have used.

$$\tau_i(t) = (1-\rho)\tau_i(t) \quad (2)$$

$$\tau_i(t+1) = \tau_i(t) + \Delta\tau_i(t) \quad (3)$$

with

$$\Delta\tau_i(t) = \sum_{k=1}^{m} \Delta\tau_i^k(t) \quad (4)$$

where m is the number of ants at each iteration and $\rho \in (0,1)$ is the pheromone trail decay coefficient. The main role of pheromone evaporation is to avoid stagnation, that is, the situation in which all ants constructing the same solution. All ants can update the pheromone according to Equations (3,4). Where $\Delta\tau_i^k(t)$ is the amount

of pheromone deposited by ant k on node i at time step t:

$$\Delta\tau_i^k(t) = \begin{cases} \omega.\gamma(S^k(t)) + \phi.(n/|S^k(t)|) & if i \in S^k(t) \\ 0 & otherwise \end{cases} \quad (5)$$

where $S^k(t)$ is the feature subset found by ant k at iteration t, and $|S^k(t)|$ is its length. The pheromone is updated according to both the measure of the classifier performance, $\gamma(S^k(t))$, and feature subset length. ω and ϕ are two parameters that control the relative importance of classifier performance and feature subset length, $\omega \in [0,1]$ and $\phi = 1-\omega$. This formula means that the classifier performance and feature subset length have different significance for FS process. In our experiment we assume that classifier performance is more important than subset length, so they were set as $\omega = 0.7$, $\phi = 0.3$.

Algorithm 1 ACO-based Feature Selection Algorithm for IDS

1: Begin
2: Initialize all parameters, i.e. $\alpha, \beta, \rho, m, \tau_0, \phi, \omega, T$.
3: Let $t = 1$.
4: **for** Each node i **do**
5: $\tau_i(t) = \tau_0$.
6: **end for**
7: Place m ants, $k = 1, , m$. // Initialize a population of ants with random positions
8: **while** $t \leq T$ **do**
9: **for** Each ant $k = 1, ..., m$ **do**
10: $S^k(t) = \{\}$
11: **while** Ant is able to increase the detection rate **do**
12: From current node, select next node i using Equation (1).
13: Add node i to subset $S^k(t)$.
14: **end while**
15: Calculate the subset length $|S^k(t)|$.
16: Calculate the classifier performance $\gamma(S^k(t))$.
17: **end for**
18: **for** Each node i **do**
19: Apply pheromone evaporation using Equation (2).
20: Calculate $\Delta\tau_i(t)$ using Equations (4,5).
21: Update pheromone using Equation (3).
22: **end for**
23: $t = t + 1$
24: **end while**
25: Return the subset $S^k(t)$ with highest $\gamma(S^k(t))$ as the solution.
26: End

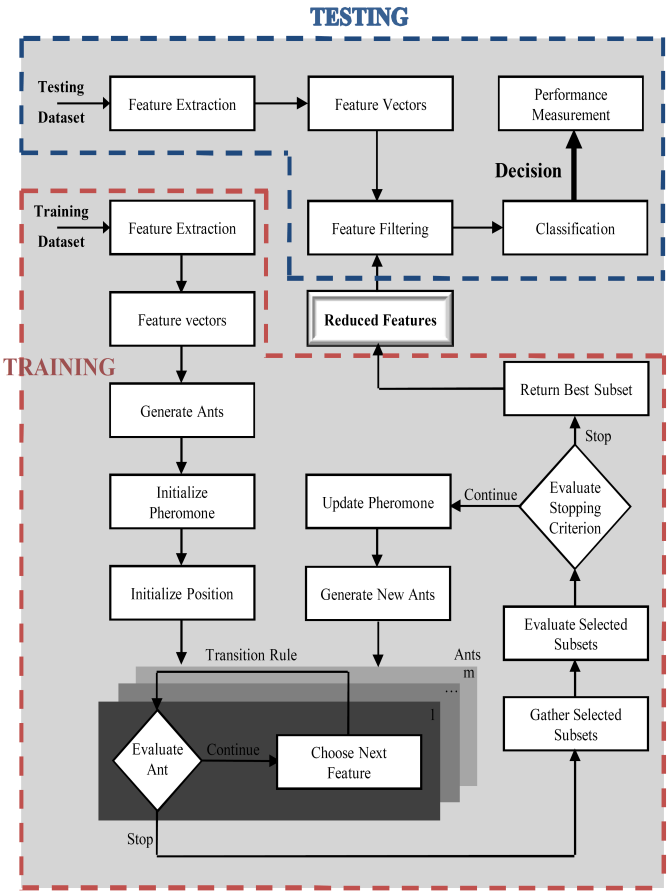

Figure 1: Overall process of proposed ACO-based IDS

3.4 Solution Construction

The overall process of ACO feature selection can be seen in Figure 1. The process starts by generating a number of ants which are then placed randomly on the graph i.e. each ant begins with one random feature. Alternatively, the number of ants to place on the graph may be set equal to the number of features within the data set; each ant starts path construction at a different feature. From these initial positions, they traverse nodes, features, probabilistically until a traversal stopping criterion is satisfied. The resulting subsets are collected and then evaluated. If an optimal subset is found or the algorithm is executed a certain number of times, then the process halts and outputs the best feature subset encountered. If none of these conditions occurred, then the pheromone is updated, a new set of ants are created and the process iterates once more.

In the proposed approach an IDS consists of several essential parts including feature extraction and FS. After preprocessing of compressed raw (binary) TCP dump data of 7 weeks of network traffic (DARPA 98), feature extraction is used to transform the input TCP dump data into a feature set, feature vector. FS is applied to the feature set to select more informative features and to reduce the dimensionality of the problem space. ACO algorithm is used to explore the space of all subsets of given feature set. The performance of selected feature subsets is measured by invoking an evaluation function with the corresponding reduced feature space and measuring the specified classification result. The best feature subset found

is then output as the recommended set of features to be used in the actual design of the IDS.

4 Experiments and Results

The following sections describe the data sets and implementation results.

4.1 Data sets

Since 1999, KDD Cup 99 data set from UCI repository is widely used as the benchmark data set for IDS evaluation [34]. The KDD Cup 99 contained 4,898,431 and 311,029 records in the training set and test set, respectively. Each of record contains 41 features and is labeled as either normal or an attack, with exactly one specific attack type. In our experiments, we apply its 10% training data set consisting of 494021 connection records for training. This data set is prepared by Stolfo et al. and is built based on the data captured in DARPA 98 IDS evaluation program [27]. The DARPA 98 intrusion detection evaluation program was prepared and managed by MIT Lincoln Labs. Lincoln Labs set up an environment to acquire nine weeks of raw TCP dump data for a LAN simulating a typical U.S. Air Force LAN. DARPA 98 is about 4 gigabytes of compressed raw (binary) TCP dump data of 7 weeks of network traffic, which can be processed into about 5 million connection records, each with about 100 bytes. The two weeks of test data have around 2 million connection records. All attacks fall in one of the following four categories:

- **Denial of Service (DoS):** Is an attempt to consume network resources in such a way that their services become limited or unavailable for the legitimate users.
- **User to Root (U2R):** Is an attack in which the attacker starts accessing a normal user account on a machine and gains root access to the machine by exploiting vulnerabilities.
- **Remote to Local (R2L):** Occurs when an attacker does not have an account on a remote system, but who has the ability to send packets to a system over a network and exploits vulnerabilities to gain local access as a user of that system.
- **Probing:** An attack in which the attacker scans network to collect information about its systems for the apparent purpose of circumventing its security controls.

KDD Cup 99 features can be classified into three groups:

1) **Basic features:** this category encapsulates all the attributes that can be extracted from a TCP/IP connection. Most of these features leading to an implicit delay in detection.
2) **Traffic features:** this category includes features that are computed with respect to a window interval.
3) **Content features:** Most of the DoS and Probing attacks have many intrusion frequent sequential patterns, this is due to the fact that these attacks establish many connections to the host(s) in a very short period of time. Unlike these attacks, the R2L and U2R attacks donot have any intrusion frequent sequential patterns. The R2L and U2R attacks are embedded in the payload of the packets, and normally include only a single connection. To identify these kinds of attacks, some relevant features are needed to identify suspicious behavior in the packet payload. These features are called content features [31].

Conducting a thorough analysis of the recent research trend in anomaly detection, one will encounter several machine learning methods reported to have a very high detection rate of 98% while keeping the false alarm rate at 1%. However, when we look at the state of the art IDS solutions and commercial tools, there is few products using anomaly detection, and practitioners still think that it is not a mature technology yet. Tavallaee et al. studied the details of a research in anomaly detection and considered various aspects such as learning and detection approaches, training data sets, testing data sets, and evaluation methods. Their study shows that there are some inherent problems in the KDD Cup 99 data set [31], which, is widely used as one of the few publicly available data sets for network-based anomaly detection systems.

The first important deficiency in the KDD Cup 99 data set is the large amount of redundant records. They found approximately 78% and 75% of the records are duplicated in the training and testing sets, respectively. This large number of redundant records in the training set will cause learning algorithms to be biased towards the more frequent records, and thus prevent it from learning infrequent records which are usually more harmful to networks such as U2R attacks. The existence of these repeated records in the test set, on the other hand, will cause the evaluation results to be biased by the methods which have better detection rates on the frequent records [31].

Tavallaee et al. applied 21 learned machines to analyze the difficulty level of the records in KDD Cup 99 data set. They labeled the records of the entire train and test sets, which provide 21 predicted labels for each record. All the 21 methods applied on the data set classified about 98% of the records in the train set and 86% of the records in the test set correctly. Tavallaee et al. reported these statistics on both KDD Cup 99 train and test sets since they have found similar results presented in many papers, random parts of the KDD Cup 99 training set are used as test sets. As a result, these papers obtain about 98% detection rate. Even applying the KDD test set will result in having a minimum detection rate of 86%, which makes the comparison of IDSs quite difficult since they all vary in the range of 86% to 100% [31]. In this paper, both the KDD Cup 99 and more recent and revised version

of KDD Cup 99 data set, NSL-KDD data set, are used for the experimentation. NSL-KDD data set is publicly available for the researchers and does not suffer from any of the mentioned problems [35]. Additionally, the number of NSL-KDD records in the train and test sets is more reasonable. This advantage makes it a good choice to run the approaches on the complete KDD Cup 99 data set without the need to randomly select a small portion.

It should be mentioned that the test set is not from the same probability distribution as the training set, and it includes unknown attack types that do not exist in the training set that makes it more realistic. The data sets contain a total number of 22 training attack types, with an additional 17 types in the test data set [19]. In Table 2, distribution of different attack types in the KDD Cup 99 and NSL-KDD data sets are listed.

In the KDD Cup 99 and NSL-KDD data sets there are 41 features (listed in Table 3) suggested for each record.

4.2 Performance Measures

The False Positive Rate (FPR) is defined as the number of normal records that are incorrectly detected as intrusions divided by the total number of normal records. The detection rate is defined as the number of intrusion records classified by the IDS divided by the total number of intrusion records present in the test data set. These are good performance measures, since they measure what percentage of intrusions the system is able to detect and how many incorrect classifications are made in the process. Usually True Positive Rate (TPR) and false positive rate are used for performance measurement. TPR also known as Detection Rate (DR) or sensitivity or Recall and FPR also known as the False Alarm Rate (FAR). They showed in the following equations:

$$Recall = TPR = \frac{TP}{TP+FN} \quad (6)$$

$$Precision = \frac{TP}{TP+FP} \quad (7)$$

$$FDR = \frac{FP}{TN+FP} \quad (8)$$

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (9)$$

where TP, TN, FP, FN are the numbers of true positives, true negatives, false positives and false negatives, respectively. Another commonly used measure is F-measure that is defined in Equation (10) [26].

$$F-measure = \frac{2 \times Recall \times Precision}{(Recall + Precision)} \quad (10)$$

4.3 Results

A series of experiments was conducted to show the utility of proposed method. All experiments are executed on a machine with Intel(R) Core(TM) i7 CPU 3.2 GHz and 4 GB of RAM. The operating system was Windows 7 Professional. All tested models were implemented on MATLAB R2011b. Various values were tested for the parameters of the proposed method. The results show that the highest performance is achieved by setting the parameters to values as follow: $\alpha = 0.4, \beta = 0.6, \rho = 0.2$, the initial pheromone intensity of each feature is equal to 1 ($\tau_0 = 1$) the number of ant in every iteration is 50 (m=50) and the maximum number of iterations is 100 (T=100). These values were empirically determined in our preliminary experiments; however, we make no claims that these are optimal values. Parameter optimization is still a topic for future research.

Many papers have focused on improving detection rates of the IDSs using efficient classifiers, i.e. this is a quiet difficult approach. This paper puts forward a modified ACO-based FS algorithm aiming at building a classifier to detect intrusion attempts. In the experiments, the whole training set and test set were applied. Each record in the training set or the test set consisted of 41 features. Initially the proposed FS algorithm is used to select important features for each type of the previous discussed attacks. Later on, a classification system was used to classify the attacks were the results are reported. Each feature value is standardized using Equation (11) and then unimportant features are removed, i.e., leaving only the important features, as listed in Table 4. For each type of attacks, using selected features classification of the attacked is performed and results are compared with those using all 41 features. Finally, using the results of the comparisons, their performance in detecting attacks is evaluated.

$$StandardScore = \frac{X-\mu}{\sigma} \quad (11)$$

where X is a feature value to be standardized, μ is the mean of feature values and σ is the standard deviation of the feature values.

Results of classification using the proposed method are reported in Tables 5 and 6. In each class (normal or attack), each row shows the performance of the proposed method and baseline approach (using all features) in detecting attacks. Experimental results show that FS phase of the process improves the detection rate. In the normal class, studying input features with regard to the output shows that there is no linear relation between the input features and output. Therefore, comparing it versus the baseline approach, implementing the proposed method has significantly improved accuracy of the classification models. The best performance of this system, in terms of its accuracy, is reported to be 98.90%, with only 2.59% false positive rate.

Several experiments are performed to compare the two different IDSs. Results show that an IDS combined with the proposed ACO-based algorithm has higher detection rates in detecting attacks than the baseline approach. Figure 2 shows the true positive rate and the false positive rate for the proposed method as we change the number of selected features. The effect of selecting different fea-

Table 1: KDD Cup 99 and NSL-KDD data sets

Data set name		# Instances	Normal	DoS	U2R	R2L	PROBE
KDD Cup 99	Training set	494021	97278	391458	54	1124	4107
	Testing set	311029	60593	229853	2636	13781	4166
NSL-KDD	Training set	25192	13449	9234	11	209	2289
	Testing set	22544	9711	7458	533	2421	2421

Table 2: The distribution of attack types in the KDD Cup 99 and NSL-KDD

Attack category	Attack type	KDD Cup 99		NSL-KDD	
		Training set kddcup.data10percent	**Testing set** corrected.gz	**Training set** KDDTrain+20Percent	**Testing set** KDDTest+
Denial of Service (DoS)	Neptune	107201	58001	8282	4657
	Smurf	164091	280790	529	665
	Pod	264	87	38	41
	Teardrop	979	12	188	12
	Land	21	9	1	7
	Back	2203	1098	196	359
	Apache2	-	794	-	737
	Udpstorm	-	2	-	2
	Process-table	-	759	-	685
	Mail-bomb	-	5000	-	293
User to Root (U2R)	Buffer-overflow	30	22	6	20
	Load-module	9	2	1	2
	Perl	3	2	0	2
	Rootkit	10	13	4	13
	Spy	2	-	1	-
	Xterm	-	13	-	13
	Ps	-	16	-	17
	Http-tunnel	-	158	-	133
	Sql-attack	-	2	-	2
	Worm	-	2	-	2
	Snmp-guess	-	2406	-	331
Remote to Local (R2L)	Guess-password	53	4367	10	1231
	Ftp-write	8	3	1	3
	Imap	12	1	5	1
	Phf	4	2	2	2
	Multihop	7	18	2	18
	Warezmaster	20	1602	7	944
	Warezclient	1020	-	181	-
	Snmpgetattack	-	7741	-	178
	Named	-	17	-	17
	Xlock	-	9	-	9
	Xsnoop	-	4	-	4
	Send-mail	-	17	-	14
Probe	Port-sweep	1040	354	587	157
	IP-sweep	1247	306	710	141
	Nmap	231	84	301	73
	Satan	1589	1633	691	735
	Saint	-	736	-	319
	Mscan	-	1053	-	996

Table 3: Lists of features in KDD Cup 99 and NSL-KDD data sets

No.	Feature name	Description	Type
1	Duration	Length of the connection (second)	Continuous
2	Protocol-type	Type of protocol, e.g. tcp, udp, etc.	Discrete
3	Service	Network service on the destination, e.g., http, telnet, etc.	Discrete
4	Flag	Normal or error status of the connection	Discrete
5	Src-bytes	Number of data bytes from source to destination	Continuous
6	Dst-bytes	Number of data bytes from destination to source	Continuous
7	Land	1 if connection is from/to the same host/port; 0 otherwise	Discrete
8	Wrong-fragment	Number of wrong fragments	Continuous
9	Urgent	Number of urgent packets	Discrete
10	Hot	Number of hot indicators	Discrete
11	Num-failed-logins	Number of failed login attempts	Discrete
12	Logged-in	1 if successfully logged in; 0 otherwise	Discrete
13	Num-compromised	Number of compromised condition	Discrete
14	Root-shell	1 if root shell is obtained; 0 otherwise	Discrete
15	Su-attempted	1 if su root command attempted; 0 otherwise	Discrete
16	Num-root	Number of root accesses	Discrete
17	Num-file-creations	Number of file creation operations	Discrete
18	Num-shells	Number of shell prompts	Discrete
19	Num-access-files	Number of operations on access control files	Discrete
20	Num-outbound-cmds	Number of outbound commands in an ftp session	Discrete
21	Is-host-login	1 if the login belongs to the hot list; 0 otherwise	Discrete
22	Is-guest-login	1 if the login is a guest login; 0 otherwise	Discrete
23	Count	Number of connections to the same host as the current connection in the past two seconds	Discrete
24	Srv-count	Number of connections to the same service as the current connection in the past two seconds	Discrete
25	Serror-rate	Percent of connections that have SYN errors	Discrete
26	Srv-serror-rate	Percent of connections that have SYN errors	Discrete
27	Rerror-rate	Percent of connections that have REJ errors	Discrete
28	Srv-rerror-rate	Percent of connections that have REJ errors	Discrete
29	Same-srv-rate	Percent of connections to the same services	Discrete
30	Diff-srv-rate	Percent of connections to different services	Discrete
31	Srv-diff-host-rate	Percent of connections to different hosts	Discrete
32	Dst-host-count	Count for destination host	Discrete
33	Dst-host-srv-count	Srv-count for destination host	Discrete
34	Dst-host-same-srv-rate	Same-srv-rate for destination host	Discrete
35	Dst-host-diff-srv-rate	Diff-srv-rate for destination host	Discrete
36	Dst-host-same-src-port-rate	Same-src-port-rate for destination host	Discrete
37	Dst-host-srv-diff-host-rate	Diff-host-rate for destination host	Discrete
38	Dst-host-serror-rate	Serror-rate for destination host	Discrete
39	Dst-host-srv-serror-rate	Srv-serror-rate for destination host	Discrete
40	Dst-host-rerror-rate	Rerror-rate for destination host	Discrete
41	Dst-host-srv-rerror-rate	Srv-serror-rate for destination host	Discrete

Table 4: Selected features out of the 41 features

Category	# Feature	Selected features
Normal	5	Urgent, Num-failed-logins, Count, Rerror-rate, Dst-host-srv-diff-host-rate (9, 11, 23, 27, 37)
DoS	4	Duration, Flag, Root-shell, Dst-host-srv-diff-host-rate (1, 4, 14, 37)
U2R	4	Service, Dst-bytes, Count, Serror-rate (3, 6, 23, 25)
R2L	3	Count, Srv-count, Diff-srv-rate (23, 24, 30)
Probe	8	Protocol-type, Duration, Hot, Logged-in, Num-compromised, Num-access-files, Diff-srv-rate, Dst-host-diff-srv-rate (2, 4, 10, 12, 13, 19, 30,35)

Table 5: Proposed IDS performance on the KDD Cup 99 data set

Category	All features					Proposed IDS				
	Normal	DoS	U2R	R2L	Probe	Normal	DoS	U2R	R2L	Probe
#Correctly detected	45954	183928	29	537	2800	59024	229347	2465	13667	3110
# Miss detected	14639	45925	2607	13244	1366	1569	506	171	114	1056
Precision	77.74	87.86	46.77	82.23	6.68	69.60	81.66	6.53	13.07	86.41
Recall (TPR)	75.84	80.02	1.10	3.9	67.21	97.41	99.78	93.51	99.17	74.65
F-measure	76.78	83.76	2.15	7.45	12.15	81.19	89.82	12.21	23.10	80.10

tures for the all attack classes in each step of the proposed method is depicted in Figure 2. The maximum difference between true positive rate and false positive rate is in the 5th step. Hence the first five features are selected for modeling the system.

A comparison between the test results for the proposed method versus other machine learning methods tested on the KDD Cup 99 test set are presented in Table 7. It can be stated that all the machine learning algorithms tested on this data set offered an acceptable level of detection performance for Normal, DoS and Probe attacks but they did not have good performance on the U2R and R2L types. The proposed method shows better TPR for DoS, U2R and R2L attacks and offers an acceptable level of detection rate for Normal and Probe attacks. There are 18729 records of various new attacks in the KDD Cup 99 test set, which have never appeared in the training set. These new attack records make an IDS trained by a training set hard to achieve good performance for test set. Experiments show that the proposed method shows an acceptable detection rate for detecting new attacks.

Considering the reported results, ACO is faster in locating the optimal solution. In general, it can find the optimal solution within tens of iterations. If exhaustive search is used to find the optimal feature subset in the KDD Cup 99 data set, there will be tens of billions of candidate subsets, which, makes the search nearly impossible. Using ACO, the optimal solution is found after 100th iterations. ACO has powerful exploration ability; it is a gradual searching process that approaches optimal solutions. The execution time for the ACO is mainly affected by the dimensionality of the problem (number of the features), and the size of the data. ACO can search in

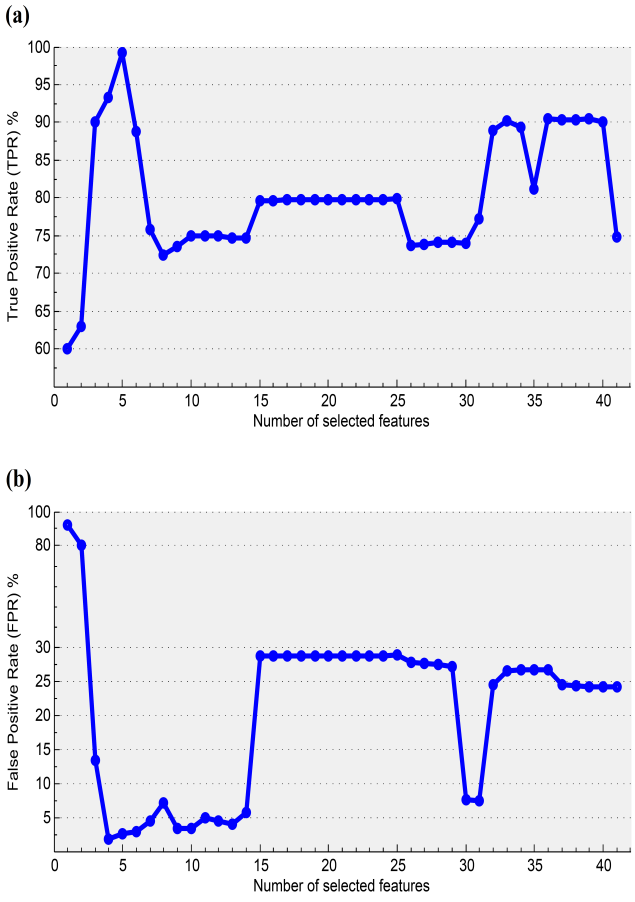

Figure 2: (a) TPR is depicted in each step and (b) FPR is shown in each step

Table 6: Proposed IDS performance on the NSL-KDD data set

Category	All features					Proposed IDS				
	Normal	DoS	U2R	R2L	Probe	Normal	DoS	U2R	R2L	Probe
#Correctly detected	9455	5131	28	454	1434	9483	5613	392	597	1667
# Miss detected	256	2327	505	1967	987	228	1845	141	1824	754
Precision	65.46	87.34	68.29	90.80	85.10	61.31	79.22	8.17	93.14	85.79
Recall (TPR)	97.36	68.80	5.25	18.75	59.23	97.65	75.26	73.55	24.66	68.86
F-measure	78.29	76.97	9.75	31.08	69.85	75.33	77.19	14.71	39.00	76.40

Table 7: Detection rate per record of KDD Cup 99 for the different algorithms performances on the test data set with corrected labels of KDD Cup 99 data set

Model	Normal(%)	DoS(%)	U2R(%)	R2L(%)	Probe(%)	Accuracy	FPR
Proposed method	97.41	99.78	93.51	99.17	74.65	98.9	2.59
PLSSVM [3]	95.69	78.76	30.7	84.85	86.46	Not reported	4.3
Clustering feature [11]	99.3	99.5	19.7	28.8	97.5	95.7	0.7
ESC-IDS [32]	98.2	99.5	14.1	31.5	84.1	95.3	1.9
KDDwinner [24]	99.5	97.1	13.2	8.4	83.3	91.8	0.5
KDD99 runner-up [16]	99.4	97.5	11.8	7.3	84.5	91.5	0.6

the feature space until the optimal solution is found. ACO comprises a very simple concept, and the ideas can be implemented in a few lines of computer code. It requires only primitive mathematical operators, and is computationally inexpensive in terms of both memory requirements and speed. Each ant has a separate memory. In ACO, each ant that passes the optimum solutions are tagged so that other ants can use them and knowledge of good solutions is retained by all ants.

5 Conclusions and Future Works

This paper addresses the problem of dimensionality reduction using ACO in intrusion detection problem area. ACO has the ability to converge quickly. It has a strong search capability in the problem space and can efficiently find minimal feature subset. Experimental results demonstrate a competitive performance. More experimentation and further investigation into this technique is required. The pheromone trail decay coefficient and pheromone amount have an important impact on the performance of ACO. Selection of the parameters is proved to be problem-dependent. The deposited pheromone expresses the quality of the corresponding solution. Evaporation becomes more important for more complex problems. If $\rho = 0$, i.e. no evaporation, the algorithm does not converge. If pheromone evaporates too much (a large ρ is used), the algorithm often converged to sub-optimal solutions. In many practical problems, it is difficult to select the best ρ without trial-and-error. α and β are also key factors in ACO for FS. For large data sets, to speed up the calculation of FS process, a parallel algorithm can be implemented.

The proposed method uses ACO algorithm and a simple classifier (nearest neighbor classifier) to select important features and a trained classifier to identify any kind of new attacks. Tests and comparisons are performed on KDD Cup 99 and NSL-KDD data sets, the test sets contains 17 kinds of different attacks. The proposed method reduced the number of features by approximately 88% and the detection error reduced by around 24% using KDD Cup 99 test data set. The proposed method will significantly reduce both the memory size and the CPU time required for intrusion detection by reducing number of the features used for the detection. This shows that the proposed method is very reliable for intrusion detection. Results indicate that the proposed ACO-based detection method outperforms other methods since it can provide better and more robust representation of the data. This is due to the fact that it can accurately detect a broader range of attacks using smaller number of features.

As for the future work, intention is to apply the proposed intrusion detection method using complicated classifiers to improve its performance and to combine the proposed method with other population-based algorithms. Analyzing packet payload is recently attracting lots of attention and many researchers report works carried-out in this area. It is notable that feature selection for the payload-based intrusion detection is not mature yet. Intension will be to extract and selection appropriate features from the packet payload to improve the detection rate.

Acknowledgments

This study was supported by the Iran University of Science and Technology. The authors gratefully acknowledge the anonymous reviewers for their valuable comments.

References

[1] M. H. Aghdam, N. Ghasem-Aghaee, and M. E. Basiri, "Application of ant colony optimization for feature selection in text categorization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC'08)*, pp. 2872–2878, 2008.

[2] M. H. Aghdam, J. Tanha, A. R. Naghsh-Nilchi, and M. E. Basiri, "Combination of Ant Colony Optimization and Bayesian Classification for Feature Selection in a Bioinformatics Dataset," *Journal of Computer Science and System Biology*, vol. 2, pp. 186–199, 2009.

[3] F. Amiri, M. M. R. Yousefi, C. Lucas, A. Shakery, and N. Yazdani, "Mutual information-based feature selection for intrusion detection systems," *International Journal of Network and Computer Applications*, vol. 34, pp. 1184–1199, 2011.

[4] J. Bins, *Feature Selection from Huge Feature Sets in the Context of Computer Vision*, Ph.D. dissertation, Colorado State University, 2000.

[5] C. Blum, and M. Dorigo, "The hyper-cube framework for ant colony optimization," *IEEE Transaction on Systems, Man, and Cybernetics*, Part B, vol. 34, no. 2, pp. 1161–1172, 2004.

[6] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, New York, 1999.

[7] S. Chebrolu, A. Abraham, and J. P. Thomas, "Feature deduction and ensemble design of intrusion detection systems," *International Journal of Computers and Security*, vol. 24, pp. 295–307, 2005.

[8] P. S. Chung, C. W. Liu, and M. S. Hwang, "A Study of Attribute-based Proxy Re-encryption Scheme in Cloud Environments," *International Journal of Network Security*, vol. 16, no. 1, pp. 1–13, 2014.

[9] M. Dorigo, and C. Blum, "Ant colony optimization theory: A survey," *Theoretical Computer Science*, pp. 243–278, 2005.

[10] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, John Wiley & Sons, London, 2005.

[11] S. J. Horng, M. Y. Su, Y. H. Chen, T. W. Kao, R. J. Chen, J. L. Lai, C. D. Perkasa, "A novel intrusion detection system based on hierarchical clustering and support vector machines," *International Journal of Expert Systems with Applications*, vol. 38, no. 1, pp. 306–313, 2011.

[12] R. Jensen, *Combining Rough and Fuzzy Sets for Feature Selection*, Ph.D. dissertation, School of Information, Edinburgh University, 2005.

[13] K. J. Lee, J., Joo, J., Yang, and V. Honavar, "Experimental comparison of feature subset selection using GA and ACO algorithm," in *Advanced Data Mining and Applications*, LNCS 4093, pp. 465–472, Springer, 2006.

[14] W. Lee, S. J. Stolfo, K. W. Mok, "Adaptive intrusion detection: a data mining approach," *International Journal Artificial Inteligence Review*, vol. 14, no. 6, pp. 533–567, 2000.

[15] G. Leguizamon, and Z. Michalewicz, "A New Version of Ant System for Subset Problems," *In Proceedings of IEEE Congress on Evolutionary Computation*, 1999.

[16] I Levin, "KDD-99 classifier learning contest LLSofts results overview," *International Journal of SIGKDD Explorations*, vol. 1, no. 2, pp. 67–75, 2000.

[17] H. Liu, and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, pp. 491–502, 2005.

[18] V. Maniezzo, A. Colorni, "The Ant System Applied to the Quadratic Assignment Problem," *IEEE Transaction on Knowledge and Data Engineering*, vol. 11, no. 5, pp. 769–778, 1999.

[19] MIT Lincoln Labs, *1998 DARPA Intrusion Detection Evaluation*, Feb. 2012. (`http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/index.html`)

[20] D. Mladeni, "Feature Selection for Dimensionality Reduction," in *Subspace, Latent Structure and Feature Selection, Statistical and Optimization, Perspectives Workshop (SLSFS'06)*, LNCS 3940, pp. 84–102, Springer, 2006.

[21] R. Montemanni, L. M. Gambardella, A. E. Rizzoli, and A. V. Donati, "A new algorithm for a Dynamic Vehicle Routing Problem based on Ant Colony System," in *Second International Workshop on Freight Transportation and Logistics*, pp. 27–30, 2003.

[22] S. Mukkamala, and A. H. Sung, "Feature ranking and selection for intrusion detection systems using support vector machines," in *Proceedings of the International Conference on Information and Knowledge Engineering*, pp. 503–509, 2002.

[23] S. Y. Ohn, N. H. Nguyen, D. S. Kim, and J. S. Park, "Determining Optimal Decision Model for Support Vector Machine by Genetic Algorithm," in *Computational and Information Science*, LNCS 3314, pp. 895–902, Springer, 2005.

[24] B. Pfahringer, "Winning the KDD 99 classification cup: bagged boosting," *International Journal of SIGKDD Explorations*, vol. 1, no. 2, pp. 65–66, 2000.

[25] M. Ramadas, S. Ostermann, and B. Tjaden, "Detecting anomalous network traffic with self-organizing maps," in *Proceedings of Sixth International Symposium on Recent Advances in Intrusion Detection*, pp. 36–54, 2003.

[26] C. J. Rijsbergen, *Information Retrieval (2nd ed.)*, London, UK: Butterworth, 1979.

[27] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, P. K. Chan, "Cost-based modeling for fraud and intrusion detection: Results from the jam project," in *In*

Proceedings DARPA Information Survivability Conference and Exposition (DISCEX'00), pp. 130–144, 2000.

[28] A. H. Sung, and S. Mukkamala, "Identifying important features for intrusion detection using support vector machines and neural network," in *Proceedings of International Symposium on Applications and the Internet*, pp. 209–216, 2003.

[29] A. H. Sung, and S. Mukkamala, "The Feature Selection and Intrusion Detection Problems," in *Proceedings of Advances in Computer Science - ASIAN: Higher-Level Decision Making. 9th Asian Computing Science Conference*, LNCS 3321, pp. 468–482, 2004.

[30] W. S. Sung, H. L. Chi, "Using Attack-Specific Feature Subsets for Network Intrusion Detection," in *Advances in Artificial Intelligence*, LNCS 4304, pp. 305–311, Springer, 2006.

[31] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data set," in *Proceeding of the IEEE Symposium on Computational Intelligence in Security and Defense Applications*, pp. 57–62, 2007.

[32] A. N. Toosi, and M. Kahani "A new approach to intrusion detection based on an evolutionary soft computing model using neuro-fuzzy classifiers," *International Journal of Computer Communications*, vol. 30, pp. 2201–2212, 2007.

[33] C. Tsang, S. Kwong, and H. Wang, "Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection," International Journal of Pattern Recognition, vol. 40, pp. 2373–2391, 2007.

[34] UCI, *KDD Cup 1999 Data*, The UCI KDD Archive Information and Computer Science University of California, Irvine, Oct. 2014. (`http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`)

[35] UNB, *NSL-KDD Data Set for Network-based Intrusion Detection Systems*, Mar. 2014. (`http://nsl.cs.unb.ca/NSL-KDD/`)

[36] S. M. Vieira, J. M. C. Sousa, and T. A. Runkler, "Fuzzy classification in ant feature selection," in *IEEE International Conference on Fuzzy Systems*, pp.1763–1769, 2008.

[37] P.D. Williams, K. P. Anchor, J. L. Bebo, G. H. Gunsch, and G. D. Lamont, "CDIS: towards a computer immune system for detecting network Intrusions," in *Proceedings of Fourth International Symposium on Recent Advances in Intrusion Detection*, pp. 117–133, 2001.

[38] S. X. Wu, and W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review," *International Journal of Applied Soft Computing*, vol. 10, pp. 1–35, 2010.

[39] L.H. Zhang, G. H. Zhang, L. Yu, J. Zhang, and Y. C. Bai, "Intrusion Detection Using Rough Set Classification," *International Journal of Zheijiang University Science*, vol. 5, no. 9, pp. 1076–1086, 2004.

Mehdi Hosseinzadeh Aghdam is a Ph.D. candidate in the computer engineering department at the Iran University of Science and Technology, he is also a lecturer at Payame Noor University (PNU). He graduated with a master's in computer engineering Artificial Intelligence from University of Isfahan (UI) in 2008. At UI, he worked on swarm intelligence-based method for feature selection. His main research interests are: Data Mining (Feature Selection, Recommendation Systems), Computational Intelligence, Pattern Recognition and Social Network Analysis.

Peyman Kabiri is assistant professor of computer engineering at the Iran University of Science and Technology (IUST). He received a B.Eng. degree in computer hardware engineering from IUST in 1992 and the M.Sc. and Ph.D. degrees in computer science at the Nottingham Trent University, UK, in 1996 and 2000 respectively. He is the director of the Intelligent Automation Laboratory at the Department of Computer Engineering-IUST. Dr. Kabiri has authored a number of publications including journal articles, book chapters, and conference papers. He is editor of a book in intrusion detection work area.