Expert Systems with Applications 42 (2015) 2670–2679

Contents lists available at ScienceDirect

Expert Systems with Applications

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

A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems

Adel Sabry Eesa [a], Zeynep Orman [b,*], Adnan Mohsin Abdulazeez Brifcani [c]

[a] Computer Science Department, Faculty of Science, Zakho University, Duhok City, KRG, Iraq
[b] Department of Computer Engineering, Faculty of Engineering, Istanbul University, 34320 Avcilar, Istanbul, Turkey
[c] Department of IT, Duhok Technical Institute, Duhok Polytechnic University, Duhok City, KRG, Iraq

ARTICLE INFO

Article history:
Available online 15 November 2014

Keywords:
Feature selection
Cuttlefish algorithm
Intrusion detection systems
Decision trees
ID3 algorithm

ABSTRACT

This paper presents a new feature-selection approach based on the cuttlefish optimization algorithm which is used for intrusion detection systems (IDSs). Because IDSs deal with a large amount of data, one of the crucial tasks of IDSs is to keep the best quality of features that represent the whole data and remove the redundant and irrelevant features. The proposed model uses the cuttlefish algorithm (CFA) as a search strategy to ascertain the optimal subset of features and the decision tree (DT) classifier as a judgement on the selected features that are produced by the CFA. The KDD Cup 99 dataset is used to evaluate the proposed model. The results show that the feature subset obtained by using CFA gives a higher detection rate and accuracy rate with a lower false alarm rate, when compared with the obtained results using all features.

1. Introduction

Due to the expansion of computer networks, the number of hacking and intrusion incidents is increasing year by year as technology rolls out, which has made many researchers focus on building systems called intrusion detection systems (IDSs). These systems are used to protect computer systems from the risk of theft and intruders (Liao, Lin, Lin, & Tung, 2013). IDSs can be categorised as anomaly detection and misuse detection or signature detection systems (Depren, Topallar, Anarim, & Ciliz, 2005; Wang, Hao, Ma, & Huang, 2010). In anomaly detection, the system builds a profile of that which can be considered as normal or expected usage patterns over a period of time and triggers alarms for anything that deviates from this behaviour. On the other hand, in misuse detection, the system identifies intrusions based on known intrusion techniques and triggers alarms by detecting known exploits or attacks based on their attack signatures.

Dimensionality reduction is a commonly used step in machine learning, especially when dealing with a high dimensional space of features (Fodor, 2002; Van der Maaten, Postma, & van den Herik, 2008). Feature selection (FS) is a part of dimensional reduction which is known as the process of choosing an optimal subset of features that represents the whole dataset. FS has been used in many fields, such as classification, data mining, object recognition and so forth, and has proven to be effective in removing irrelevant and redundant features from the original dataset. Given a feature set of size n, the FS problem tries to find a minimal feature subset of size m ($m < n$) that enables the construction of the best classifier with high accuracy (Basiri, Ghasem-Aghaee, & Aghdam, 2008).

FS has been a fertile field of research and development since the 1970s, and it is used successfully in the IDSs domain. Stein, Chen, Wu, and Hua (2005) proposed a hybrid genetic-decision tree (DT) model. They used the genetic algorithm (GA) as a generator to produce an optimal subset of features, and then the produced features were used as an input for the DT that was constructed using the C4.5 algorithm. Bolon-Canedo, Sanchez-Marono, and Alonso-Betanzos (2011) proposed a new combinational method of discretization, filtering and classification which is used as an FS to improve the classification task, and they applied this method on the KDD Cup 99 dataset. Lin, Ying, Lee, and Lee (2012) presented an intelligent algorithm which was applied to anomaly intrusion detection. The paper proposed simulated annealing (SA) and support vector machine (SVM) to find the best feature subsets, while SA and DT were proposed to generate decision rules to detect new attacks. Tsang, Kwong, and Wang (2007) proposed an intrusion detection approach to extract accurate and interpretable fuzzy IF–THEN rules from network traffic data for classification. They also used a wrapper genetic FS to produce an optimal subset of features. Lassez, Rossi, Sheel, and Mukkamala (2008) proposed a new method for FS and

* Corresponding author.
E-mail addresses: adelsabryissa@gmail.com (A.S. Eesa), ormanz@istanbul.edu.tr (Z. Orman), president@dpu.ac (A.M.A. Brifcani).

http://dx.doi.org/10.1016/j.eswa.2014.11.009
0957-4174/

extraction by using the singular value decomposition paired with the notion of latent semantic analysis, which could discover hidden information to design signatures for forensics and eventually real-time IDSs. They used three automated classification algorithms (Maxim, SVM, LGP). Nguyen, Franke, and Petrovic (2010) presented a generic-feature-selection (GeFS) measure to find global optimal feature sets by using two methods: the correlation feature-selection (CFS) measure and the minimal redundancy-maximal-relevance (mRMR) measure. This approach is based on solving a mixed 0–1 linear programming problem by using the branch-and-bound algorithm, and the authors applied the proposed method to design IDSs. A hybrid model based on the information gain ratio and K-means is proposed by Neelakantan, Nagesh, and Tech (2011) to detect 802.11-specific intrusions. They used the information gain ratio as the FS and the K-means algorithm as the classifier. Mohanabharathi, Kalaikumaran, and Karthi (2012) proposed a new method which was a combination of the information gain ratio measure and the K-means classifier used for FS. The back-propagation algorithm was also used for the learning and testing processes. Datti and Lakhina (2012) compared the performance of two feature reduction techniques: principal component analysis and linear discriminate analysis. As a classifier, they used the back-propagation algorithm to test these techniques.

Since IDSs deals with a large amount of data, FS is a critical task in IDSs. In this paper, we propose an FS model based on the cuttlefish optimization algorithm (CFA) to produce the optimal subset of features. DT is also used as a classifier to improve the quality of the produced subsets of features. The rest of this paper is organised as follows: Section 2 presents an introduction and a brief overview of DT and CFA. The proposed feature-selection approach is discussed in Section 3. Section 4 reports on the experimental results of the proposed cuttlefish feature-selection approach and a brief discussion on the obtained results. Finally, the conclusions and future work are stated in Section 5.

2. Introduction to DT and the cuttlefish optimization algorithm

2.1. Decision tree (DT)

DT is one of the most well-known machine learning techniques produced by Quinlan (Salzberg, 1994). DT has three main components: nodes, arcs, and leaves. Each node splits the instance space into two or more sub-spaces according to a certain discrete function of the input attribute values. The main node (root node) is also called the test node which has no incoming edges. Each arc out of a node is labelled with an attribute value and each leaf is labelled with a category or a class. The tree is constructed during a training phase by using the training data. In the test phase, each instance of the test data is classified by the navigation from the root of the tree down to a leaf, according to the outcome of the test data along the path. There are two popular algorithms which are used for constructing the DT: ID3 and C4.5 (Salzberg, 1994). In this paper we use the ID3 algorithm.

2.2. Cuttlefish algorithm (CFA)

In previous work, we produced a novel optimization algorithm called the CFA (Eesa, Abdulazeez, & Orman, 2013). The algorithm mimics the mechanisms behind a cuttlefish that are used to change its colour. The patterns and colours seen in cuttlefish are produced by reflected light from different layers of cells including chromatophores, leucophores and iridophores. The CFA considers two main processes: reflection and visibility. The reflection process is used to simulate the light reflection mechanism, while visibility is used to simulate the visibility of matching patterns. These two processes are used as a search strategy to find the global optimal solution. The diagram in Fig. 1 of cuttlefish skin, detailing the three main skin structures (chromatophores, iridophores and leucophores), two example states (a, b) and three distinct ray traces (1, 2, 3), shows the sophisticated means by which cuttlefish can change reflective colour (Eric et al., 2012).

CFA reorders these six cases shown in Fig. 1 to be as shown in Fig. 2. The formulation for finding the new solution (*newP*) using reflection and visibility is described in Eq. (1):

$$newp = reflection + visibility \tag{1}$$

For Cases 1 and 2 shown in Fig. 2, CFA uses the two processes reflection and visibility to find a new solution. These cases work as a global search using the value of each point to find a new area around the best solution with a specific interval. The formulations of these processes are described in Eqs. (2) and (3), respectively:

$$reflection_j = R * G_1[i].Points[j] \tag{2}$$

$$visibility_j = V * (Best.Points[j] - G_1[i].Points[j]) \tag{3}$$

where, G_1 is a group of cells, i is the ith cell in G_1, $Points[j]$ represents the jth point of the ith cell, *Best.Points* represents the best solution points, R represents the degree of reflection, and V represents the visibility degree of the final view of the pattern. R and V are found as follows:

$$R = random() * (r_1 - r_2) + r_2 \tag{4}$$

$$V = random() * (v_1 - v_2) + v_2 \tag{5}$$

where, *random*() function is used to generate random numbers between (0,1) and r_1, r_2, v_1, v_2 are four constant values specified by the user. As a local search, CFA uses Cases 3 and 4 to find the difference between the best solution and the current solution to

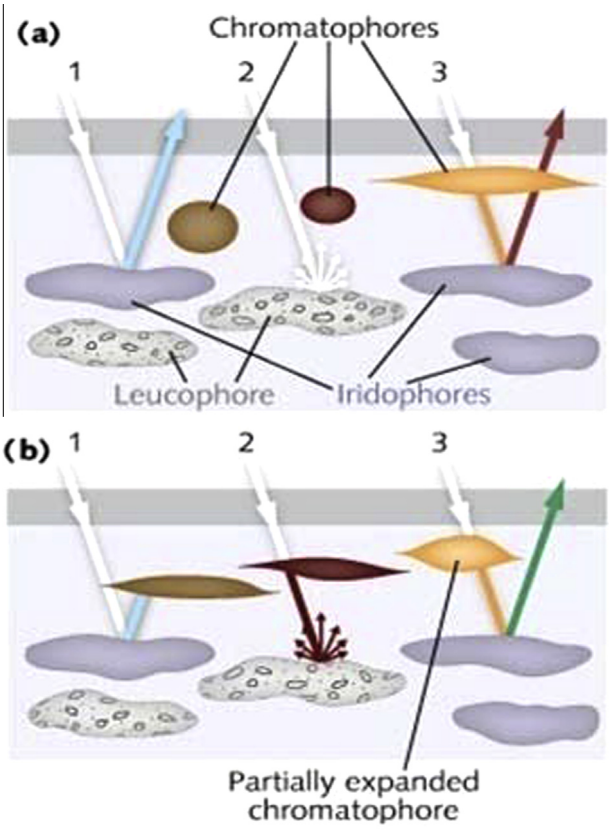

Fig. 1. Diagram of cuttlefish skin detailing the three main skin structures (chromatophores, iridophores and leucophores).

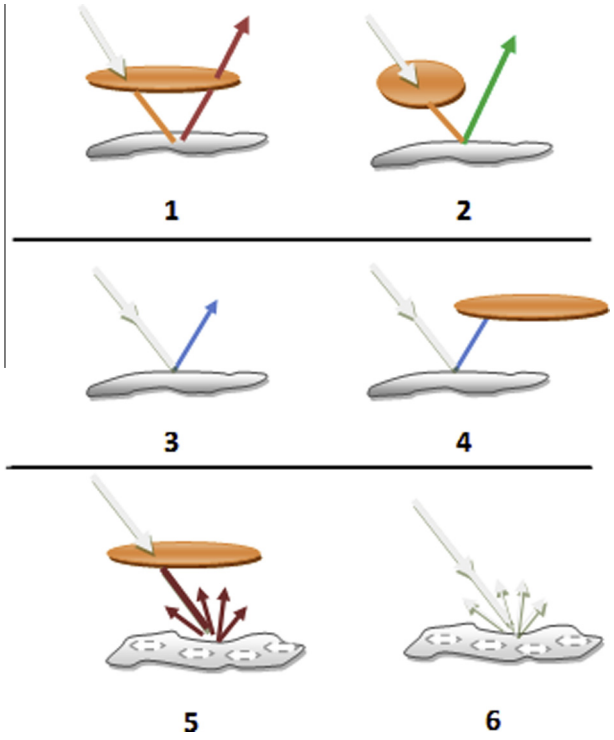

Fig. 2. Reorder of the six cases in Fig. 1.

produce an interval around the best solution as a new search area. The formula for finding the reflection is as follows:

$$reflection_j = R * Best.Point[j] \quad (6)$$

While the formulation for finding the visibility remains as in Cases 1 and 2.

For Case 5, the algorithm also uses this case as a local search, but this time the difference between the best solution points and the average value of the *Best* points is used to produce a small area around the best solution as a new search area.

The formulas for finding reflection and visibility in this case are as follows:

$$reflection_j = R * Best.Points[j] \quad (7)$$

$$visibility_j = V * (Best.Points[j] - AV_{Best}) \quad (8)$$

where, AV_{Best} is the average value of the *Best* points. Finally, the CFA uses Case 6 as the random solutions. The general principle of the CFA is shown in Fig. 3.

3. New feature-selection approach based on CFA

3.1. Feature ranking and initialization

Connection records in KDD Cup 99 contain 41 features (Horng et al., 2011). We used the ranked method to rank these features based on their location, so acquiring *rankedArray* = {1,2,3,...,41}. The algorithm starts with a population *P* of *N* initial solutions generated randomly, $P = \{p_1, p_2, p_3, \ldots, p_N\}$. Each p_i is associated with two subsets: *selectedFeatures* and *unselectedFeatures*, where *selectedFeatures* $\subset$ *rankedArray*, *unselectedFeatures* $\subset$ *rankedArray*, and *selectedFeatures* $\cap$ *unselectedFeatures* = Ø. To illustrate, consider *selectedFeatures* = {4,2,5,1,3} and that its elements are selected randomly from *rankedArray; unselectedFeatures* is a set of all remaining features in *rankedArray*, so the *unselectedFeatures* will be equal to {6,7,...,41}.

After ranking and initializing the operators, the best solution of the population will be kept in both $AV_{Bestsubset}$ and *bestSubset*, where $AV_{Bestsubset}$ and *bestSubset* are two solutions containing the best subset of features (*selectedFeatures*) and the remaining features (*unselectedFeatures*). The size of *selectedFeatures* in *bestSubset* at any time should be one less than *selectedFeatures* presented in $AV_{Bestsubset}$. To do this, we can simply remove one feature from it. Then the search strategy will start with the work of the three cells layer using the main equation with the two main operations, *reflection* and *visibility*. Simulations of the six cases considered in Fig. 2 are described next.

3.2. Simulation of Cases 1 and 2

The simulation of these two cases is started by sorting the population *P* in descending order according to the fitness values. After that the *newSubset* will be generated from p_i, where $i = 1, 2, \ldots, k$. *k* is an integer number generated randomly between (0, and *N*/2). In the original algorithm, *R* represents the reflection degree used to find the stretch interval of the saccule when the muscles of the cell are contracted or relaxed, while *V* represents the degree of visibility of the final view of the matched pattern. Here, the main equation and the operator of *reflection* and *visibility* in the original algorithm for Cases 1 and 2 are modified as follows:

$$newSubset_i = Reflection_i \oplus Visibility_i \quad (9)$$

$$Reflection_i = random\,subset[R] \subset p_i.selectedFeatures \quad (10)$$

$$Visibility_i = random\,subset[V] \subset p_i.unselectedFeatures \quad (11)$$

where $Reflection_i$ and $Visibility_i$, are two subsets with size *R* and *V* their elements are produced randomly from *selectedFeatures* and *unselectedFeatures,* respectively. The value of *R* and *V* can be calculated as follows:

$$R = random(0,\ selectedFeatures.Size)$$

$$V = selectedFeatures.Size - R$$

The operator $\oplus$ represents the combination (union) of these two *subsets* to find the *newSubset*(*i*). To illustrate, consider p_i with *selectedFeatures* = $\{f_3, f_1, f_5, f_2, f_4\}$, and *unselectedFeatures* = $\{f_6, f_7, f_8, \ldots, f_{41}\}$. The calculation of *R*, *V*, *Reflection*, and *Visibility* is as follows:

$$R = random(0, 5) = 2,$$

where 5 is the size of *selectedFeatures*

Then:

$$V = 5 - 2 = 3$$

$Reflection_1$ is found as a subset with *R* random features such as:

$$Reflection_1 = \{f_1, f_4\},$$

where f_1, f_4 are two features selected randomly from *selectedFeatures*. In the same way $Visibility_1$ is found as a subset with *V* random features such as:

$$Visibility_1 = \{f_{15}, f_{30}, f_{22}\},$$

where f_{15}, f_{30}, and f_{22} are three features selected randomly from *unselectedFeatures*.

Then the *newSubset* is found as follows:

$$newSubset_1 = Reflection_1 \oplus Visibility_1 = \{f_1, f_4\} \oplus \{f_{15}, f_{30}, f_{22}\} = \{f_1, f_4, f_{15}, f_{30}, f_{22}\}$$

Fig. 4 describes the *shrink*, *reflection* and *visibility* processes of this example. We mention that, if the size of *selectedFeatures* is bigger than *unselectedFeatures* it may cause a problem, because the number of elements in *unselectedFeatures* is not enough to generate a

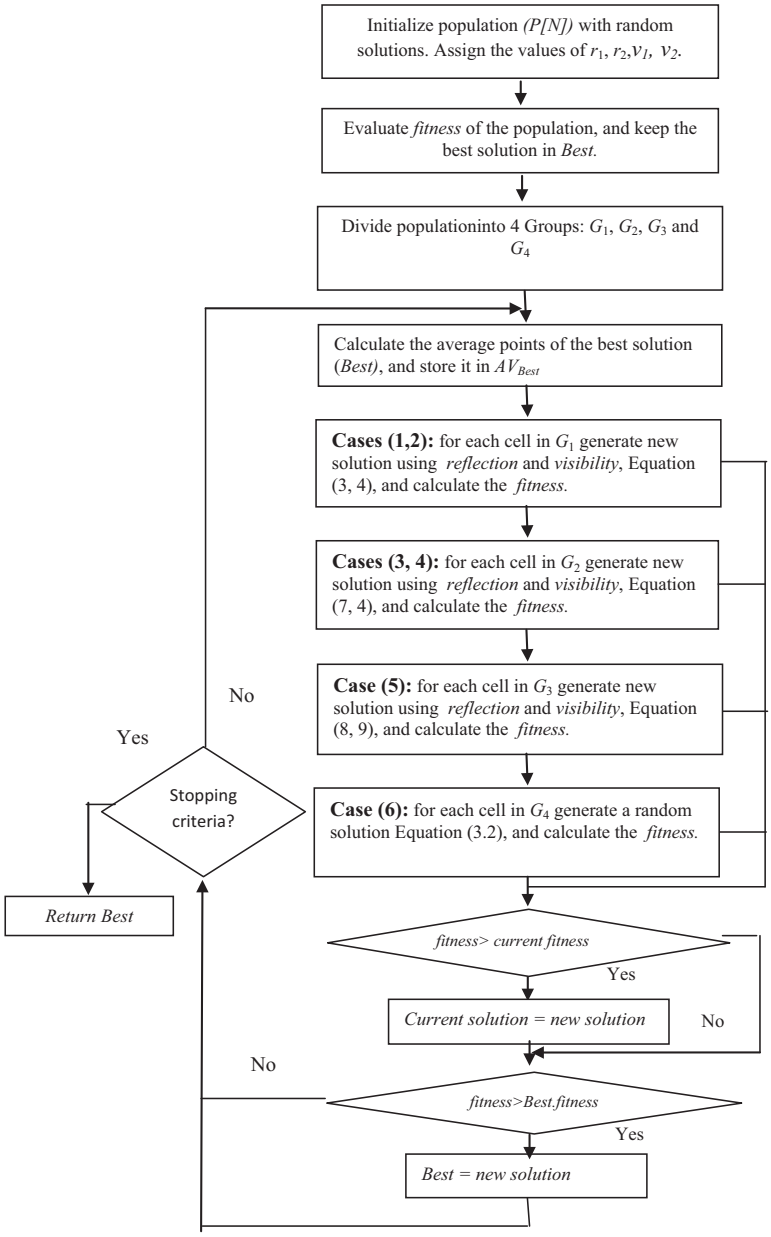

Fig. 3. General principle of CFA.

newSubset, in this case we can use original features to produce a *newSubset* without repetition of any element.

In short, Cases 1 and 2 use the best *k* solutions from the half best solution of population *P* to be a part of the new solution (new subset), by keeping some elements from the best solutions and completing them with some new elements that have a chance to be a part of the new solution.

3.3. Simulation of Cases 3 and 4

Iridophores cells are light reflecting cells which are assisting in organs concealment. That means the final reflected colour from Iridophores cells around the organs is very similar to the organs original colours. Therefore, we assumed that the organs are represented by the best solution (*bestSubset*), and the final reflected colour is represented by the new solution. The new solution colour should be very similar to the colour of the organs.

As a simulation to the similarity of the incoming colour and the reflected colour, we considered that the reflected colour is a subset produced by removing only one random feature from the incoming colour (*selectedFeatures*) whereas the visibility is considered as a single feature which is selected randomly from the *unselectedFeatures*. The combination between the reflection and the visibility will produce the new solution (*newSubset*). So the formulation of finding the new solution, reflection and the visibility are reformulated as follows:

$$Reflection = bestSubset\_selectedFeatures - bestSubset\_selectedFeatures[R] \quad (12)$$

$$Visibility = bestSubset\_unselectedFeatures\,[V] \quad (13)$$

where *R* represents the index of the feature that should be removed from *selectedFeatures*. *V* represents the index of the feature that should be selected from *unselectedFeatures*. The calculation of finding *R* and *V* are as follows:

$$R = random(0, bestSubset\_selected\ Features.Size)$$

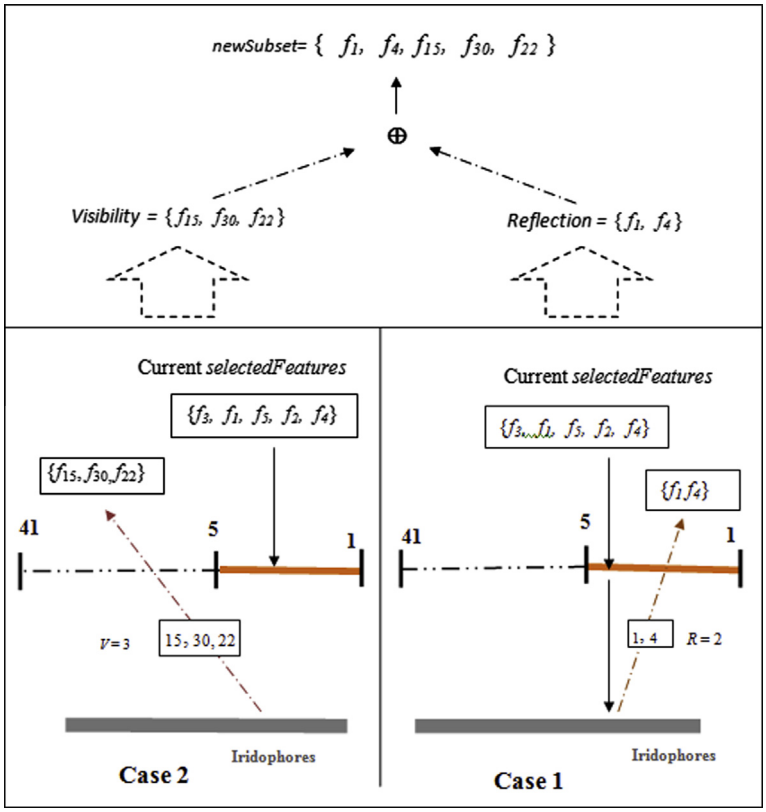

Fig. 4. Cases 1and 2, reflection and visibility processes.

$V = random(0, bestSubset\_unselected\ Features.Size)$

The *newSubset* is then calculated by using Eq. (9), by adding the feature represented by the visibility to the subset represented by the reflection. In another words, the *newSubset* is a set produced by exchanging the feature *R* with the feature *V*. For example, consider *bestSubset* with *selectedFeatures* = $\{f_3, f_1, f_2, f_4\}$, and *unselectedFeatures* = $\{f_5, f_6, f_7, \ldots, f_{41}\}$. The calculation of *R*, *V*, *Reflection*, and *Visibility* is as follows:

$R = random\,(0, 4) = 2,$

where 4 is the size of *selectedFeatures*.

Then

$Reflection = \{f_3, f_1, f_2, f_4\} - f_2 \quad //\text{removing } f_2 = \{f_3, f_1, f_4\},$

$V = random\,(5, 41) = 7,$

where 41 is the size of *unselectedFeatures*

$Visibility = f_7$

Feature f_2 will be added to *unselectedFeatures*, then *newSubset* is found as follows:

$$newSubset = Reflection \oplus Visibility = \{f_3, f_1, f_4\} + f_7 = \{f_3, f_1, f_4, f_7\},$$

Or simply, we can directly exchange f_2 with f_7. Fig. 5 detailing these processes clearly.

3.4. Simulation of Case 5

In this case, the leucophore cells work as a mirror. The cells will reflect the predominant wavelength of light in the environment. In this case, the light is coming through chromatophore cells with specific colour. The outgoing light is very similar to the light coming from the chromatophore cells. In order to cover the similarity between the incoming colour and the outgoing colour, we proposed that the values of the incoming colour and the reflection be equal to *selectedFeatures* in $AV_{Bestsubset}$ and the outgoing colour be a new subset produced from the *selectedFeatures* in $AV_{Bestsubset}$ by removing one feature from it. While the visibility represents the feature *i* that should be removed from *selectedFeatures*. These operations make the matched pattern very similar to the original pattern that appears in the environment. In this way, we can produce *R* new subsets, the value of *R* is equal to the *size* of *selectedFeatures* each subset representing the matched pattern by removing one feature from *selectedFeatures* at each time. The two equations for finding the reflection and the visibility and the main equation are modified as follows:

$$newSubset_i = Reflection - Visibility_i \tag{14}$$

$$Reflection = AV_{Bestsubset}\_selectedFeatures \tag{15}$$

$$Visibility_i = AV_{Bestsubset}\_selectedFeatures\ [i] \tag{16}$$

where, *i* represents the index of the features that should be removed from *selectedFeatures*: $i = \{1, 2, \ldots, R\}$, and *R* is the size of *selectedFeatures*.

To illustrate, consider $AV_{Bestsubset}$ with *selectedFeatures* = $\{f_2, f_6, f_1, f_9, f_{20}\}$. *R* equals to the size of *selectedFeatures* = 5 and the five produced *newSubsets* are:

$$\begin{aligned} newSubsets[5] = \{ & \{f_2, f_6 f_1, f_9\}, \quad //removing \quad f_{20} \\ & \{f_2, f_6 f_1, f_{20}\}, \quad //removing \quad f_9 \\ & \{f_2, f_6 f_9, f_{20}\}, \quad //removing \quad f_1 \\ & \{f_2, f_1 f_9, f_{20}\}, \quad //removing \quad f_6 \\ & \{f_6, f_1 f_9, f_{20}\} \quad //removing \quad f_2 \\ \} & \end{aligned}$$

and

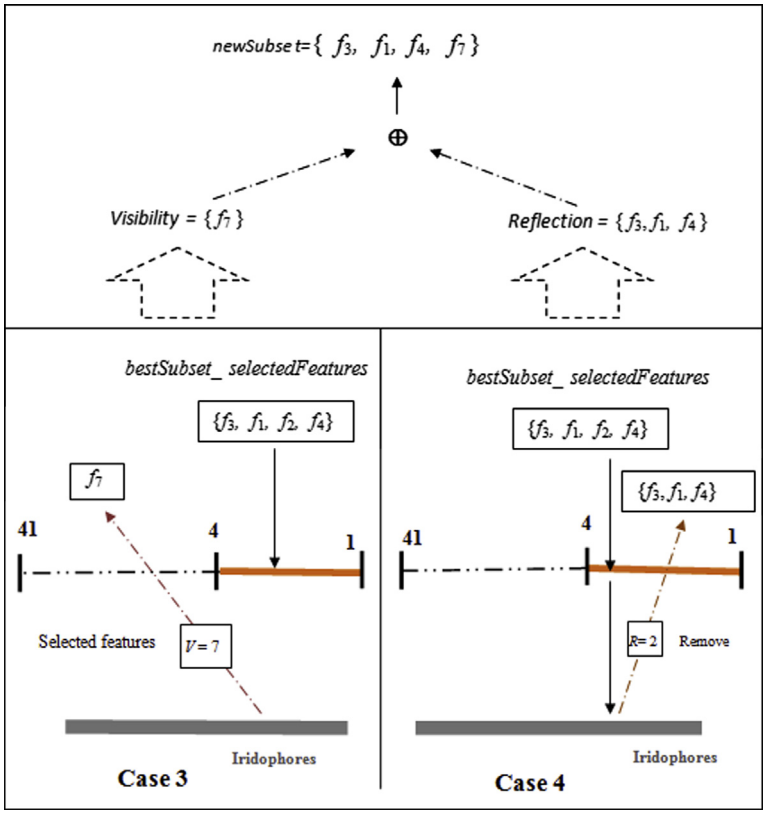

Fig. 5. Cases 3 and 4, reflection and visibility processes.

$newSubset_1 = newSubsets[1] = \{f_2, f_6, f_1, f_9\}$ *and etc.*

Based on this example, the $AV_{Bestsubset}$ is used to produce five new subsets by removing one feature from it each time. Fig. 6 clearly describes this operator.

We want to highlight that we have two best subsets of features, the $AV_{Bestsubset}$ and *bestSubset*. The *bestSubset* is produced from $AV_{Bestsubset}$, and the size of *selectedFeatures*associated with it is one less than the size of *selectedFeatures* in $AV_{Bestsubset}$.

3.5. Simulation of Case 6

In this case, the leucophore cells will just reflect the incoming light from the environment. This operator allows the cuttlefish to blend itself into its environment. As a simulation, one can assume that any incoming colour from the environment will be reflected as it can be represented by any random solution.

In the initial algorithm, this case is used to generate random solutions. Also, we use this case as a random generator process to generate random solutions. The number of generations is equal to m, where $m = N - k$. k is a random number which was previously generated in Cases 1 and 2. The new generation will be started at the location k after sorting the population P in descending order. If the new generated solution is better than the current solution, then the current solution is replaced with the new solution. The process of random solutions is the same as that used with the initialization process which is described in Section 3.1.

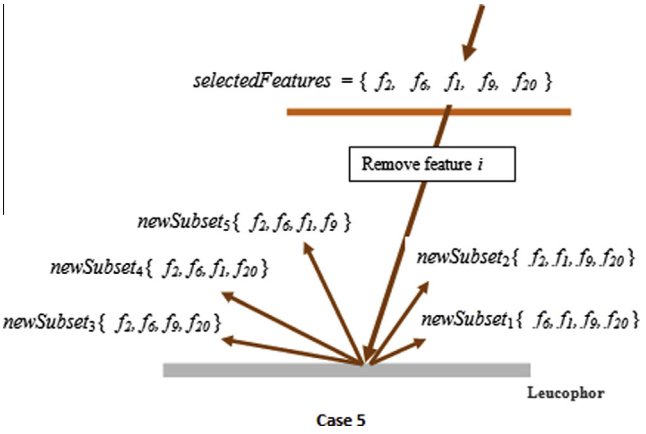

Fig. 6. Producing the new subsets operator from Av_Bestsubset, Case 5.

3.6. Parts assembling in complete algorithm

Fig. 7 describes the block diagram of the main steps of the proposed feature-selection algorithm. These steps are described as follows:

Initialization: the algorithm starts with a population P of N initial solutions generated randomly, $P\ [N] = cells[N] = \{p_1, p_2, p_3, \ldots, p_N\}$. Each p_i is associated with two subsets: *selectedFeatures* and *unselectedFeatures*. The DT classifier will evaluate each *selectedFeatures* in each p_i, the best solution will be kept in both $AV_{Bestsubset}$ and *bestSubset*. As we mentioned before, the size of *bestSubset* is less than $AV_{Bestsubset}$ by one element, so we have to remove one element from *bestSubset* randomly.

Cases 1 and 2: In these cases, the best k of population P will be used to generate new subsets. This can be done by sorting the P in descending order based on the *fitness* values and choosing the first k subsets from P, where k is a random number between $(1, N/2)$, N is the size of P. After that, the new subsets will be generated from each subset in k subsets using *Reflection* set and *Visibility* set. Where

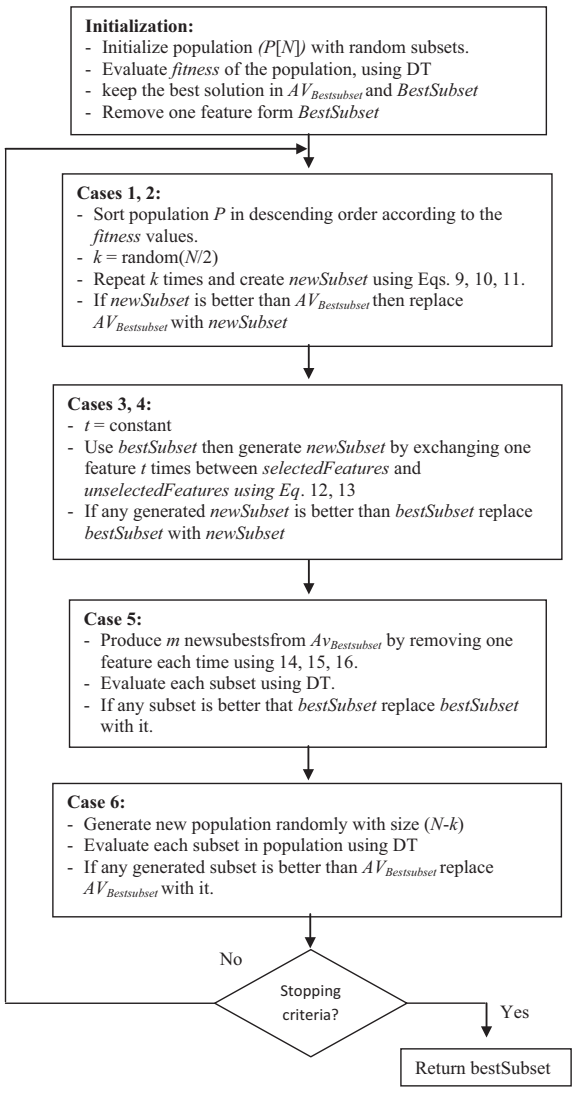

Fig. 7. Block diagram of the proposed feature selection model based on CFA.

$Reflection_i$ is a set with R features selected randomly from *selectedFeatures* that is associated with p_i, $i = \{1,2,\ldots,k\}$, and $Visibility_i$ is a set with V features selected randomly from *unselectedFeatures* which is also associated with p_i. The combination (union) of $Reflection_i$ and $Visibility_i$ will produce a new subset. The DT will evaluate the new produced subset. If there is any new generated subset better than $AV_{Bestsubset}$, $AV_{Bestsubset}$ will be replaced with it.

Cases 3 and 4: In these two cases, a single feature-exchanging operator is used to produce new solutions from *bestSubset*. A random feature is selected from *selectedFeatures* to be exchanged with another random feature selected from *unselectedFeatures*, where *selectedFeatures* and *unselectedFeatures* are the two subsets associated with *bestSubset*. This operator is repeated t times, where t is an integer constant value; its value is specified by the user. If any new produced solution is better than *bestSubset*, replace *bestSubset* with it.

Case 5: In this case, the $AV_{Bestsubset}$ will be used to produce R subsets of features by removing one element from $AV_{Bestsubset}$ each time. The number of generations (R) is equal to the size of $AV_{Bestsubset}$. Each of the generated group will be evaluated by the DT, if any new subset is better than *bestSubset*, the *bestSubset* will be replaced with that new subset.

Case 6: After best k solutions are used in Cases 1 and 2 to generate new solutions. In this case the remaining solutions of P will be generated randomly. This operator is the same as in the initialization step. The $AV_{Bestsubset}$ will be replaced with the best new generated solution if the new solution is better than it, after DT is used to evaluate the new generation.

The work of these cases is repeated until the stop criterion, such as number of iterations, is satisfied. The pseudo code of the algorithm is described in Fig. 8.

4. Experiments and results

4.1. Evaluation criteria

The proposed model is evaluated based on three performance measures: The detection rate (DR), False Positive Rate (FPR), and Accuracy Rate (AR) (Chen, Cheng, & Hsieh, 2009).

$$DR = \frac{No.\ of\ attacks\ that\ are\ correctly\ classified\ as\ attack}{Total\ No.\ of\ attacks\ in\ the\ test\ dataset} * 100\% \quad (17)$$

$$FPR = \frac{No.\ of\ normal\ that\ are\ uncorrectly\ classified\ as\ attack}{Total\ No.\ of\ normal\ in\ the\ test\ dataset} * 100\% \quad (18)$$

$$AR = \frac{No.\ of\ correctly\ classified\ as\ instances}{Total\ No.\ of\ instances\ in\ the\ test\ dataset} * 100\% \quad (19)$$

Higher values of DR and AR, and lower values of FPR show better classification performance for IDSs.

4.2. Fitness function

The validation and the quality of each subset of features are supported by the DT classifier. DT will suggest which subset of features is better according to the fitness defined in Eq. (20).

$$Fitness = \alpha * DR + \beta * (1 - FPR) \quad (20)$$

This equation means that the DR and FPR have different importance based on α and β, where $\alpha \in [0,1]$ and $\beta = 1 - \alpha$, are two parameters referring to the importance of the DR quality and FPR. In this experiment, we proposed that the quality of DR is more important than FPR and we set $\alpha = 0.7$, $\beta = 0.3$.

4.3. Results

In our experiments we used the KDD Cup 99 dataset available on the website http://kdd.ics.uci.edu to measure the performance of the proposed cuttlefish feature-selection model. 10%KDD Cup 99 training dataset and test dataset contain about 494,020 and 311,028 connection records, respectively. This data is too large to use in such studies. For this reason, two subsets (train and test) dataset are extracted randomly and, to keep the proportion of each attack both in the training and test datasets, each attack is divided by 100. For example, the number of *ipsweep* attacks in the original training and test data is (1247, 306) while the number of these attacks in the extracted data is equal to (12, 3). Table 1 describes different attack types and their corresponding occurrence number in the training and test data, respectively. The number in the training data is 4947 and in the test data 3117, selected randomly for this experiment. From Table 1 Probing (41; 42) means that the number of records in the training dataset of the attack Probe is equal to 41 connection records, while the number of records in the test dataset of this attack is equal to 42 connection records.

The experiments were carried out using C# on a Dual-Core CPU 2.20 GHz laptop with 2 GB RAM. The obtained results shown in Figs. 9 and 10, and Table 2 are the average of ten independent runs

1. *Initialize population P[n] with random solutions; intialize t*
2. *Evaluate fitness of the population using DT*
   keep the best solution in $AV_{Bestsubset}$and in bestSubset.
3. *Remove one feature form bestSubset*
4. *While (stopping criterion is not met)*

1. Cases 1, 2:

Sort the population P in descending order according to the fitness values.
k = random(N/2)
for(i =0;i<k;i++)

{

R= random(p_i.selectedFeatures.Size)
$Reflection_i$[R]
V = p_i.selectedFeatures.Size – R;
$Visibility_i$[V]
newSubset =[];
chose R features randomly from p_i.selectedFeatures and add them to $Reflection_i$
chose V features randomly from p_i.unselectedFeatures and add them to $Visibility_i$
newSubset[] =$Reflection_i \oplus Visibility_i$
Evaluate newSubset using DT
If (newSubset is better than $AV_{Bestsubset}$)
$AV_{Bestsubset}$= newSubset

}

2. Cases3, 4:

for (i = 0;i<t;i++)

{

Use bestSubset and exchange one feature randomly between selectedFeatures and unselectedFeatures
If (newSubset is better than bestSubset)
bestSubset. = newSubset

}

3. Case 5:

for(i = 0; i< $AV_{Bestsubset}$.SelectedFeatures.Size; i++)
{
Create newSubset by removing feature i from $AV_{Bestsubset}$.SelectedFeatures
If (newSubset is better than bestSubset)
bestSubset. = newSubset
}

4. Case 6:

for(i = k;i<n;i++)
{
Generate newSubset randomly
evaluate newSubset using DT
If (newSubset is better than p_i)
P_i = newSubset;
if (p_i better than $AV_{Bestsubset}$)
$Av_{Bestsubset}$ = p_i
}

5. *End while*
6. *Return bestSubset*

Fig. 8. Pseudo code of the proposed feature selection based on CFA.

Table 1
Different attack types and their corresponding occurrence number respectively in the training and test dataset.

Normal(973;606)			
Probing (41; 42)	DoS(3915; 2299)	U2R(5; 10)	R2L(13; 160)
ipsweep(12;3), Mscan(0;11), Nmap(2;1) Portsweep(11;4) Saint(0;7), Satan(16;16).	apache2(0;8), back(22;11), land(0; 0), mailbomb(0;50), Neptune(1072;580), processtable(0;8), Pod(3;1), udpstorm(0;0), Smurf(2808;1641), Teardrop(10;0),	buffer_overflow(3;1), httptunnel(0;3), loadmodule(0;0), perl(0;0), rootkit(2;2), xterm(0;2), Ps(0;2), Sqlattack(0;0),	ftp_write(0;0), imap(0;0), guesspasswd(2;44), named(0;0), multihop(0;0), phf(0;0), sendmail(0;0), snmpgetattack(0;77), snmpguess(0;24), spy(0;0,) warezclient(10;0), worm(0;0), warezmaster(1;15), xsnoop(0;0), xlock(0;0),

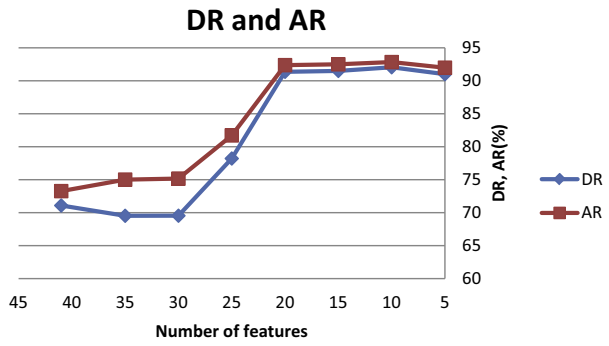

Fig. 9. DR, AR per number of selected features using proposed features selection based on CFA.

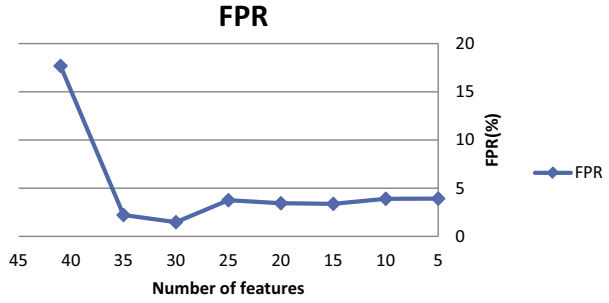

Fig. 10. FPR per number of selected features using proposed feature selection based on CFA.

Table 2
Results of proposed feature selection base on CFA.

No. of features	FPR%	DR%	AR%	Fitness%
41	17.685	71.087	73.267	74.455
35	2.21488	69.526	75.013	78.004
30	1.471	69.538	75.167	78.235
25	3.752	78.212	81.714	83.623
20	3.438	91.362	92.372	92.922
15	3.372	91.500	92.500	93.070
10	3.900	92.051	92.837	93.265
5	3.917	91.000	91.986	92.524

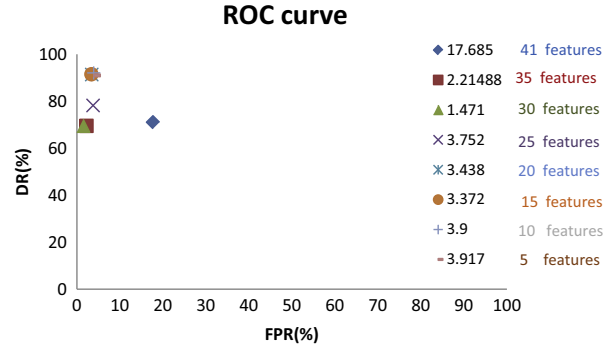

Fig. 11. ROC Curve of the intrusion detection using proposed feature selection based on CFA.

for each selected subset in terms of DR, FPR, and AR. The results show that whenever the number of features is decreased, the AR and DR are increased. In addition, the cuttlefish feature-selection model gives much better performance in all cases when compared with the results using all 41 features. From Table 2, we can see that the best obtained results are obtained when the number of features is equal or less than 20 features spatially with AR and DR. Also it is clearly seen that using 5 features results much better when compared with using the whole 41 features. With 5 features, we obtain AR = 91.986, DR = 91, FPR = 3.917 whereas with 41 features: AR = 73.267, DR = 71.087, FPR = 17.685. These numbers explain that the use of 5 features is much better than using 41 features. The ROC curve shown in Fig. 11 describes the performance of our model in terms of DR and FPR.

5. Conclusions

In this study, we have investigated the combination model of CFA and DT for feature selection for intrusion detection and evaluated its performance based on the benchmark KDD Cup 99 intrusion data. Firstly, we have modified the CFA to be used as a feature selection tool. Then, we used DT classifier as measurement on the generated features. Empirical results reveal that the produced features are performed the DR and AR especially when the number of produced features was equal or less than 20 features. In general whenever the number of features is decreased, the AR and DR are increased.

The value of FPR is not performed during the experiments. It is remained balancing between 3.3 and 3.92. This is because there are some instances of attacks in the test dataset that are never appeared in the train dataset, such as (Mscan, Saint, apache2, mailbomb, processtable, snmpgetattack, snmpguess).

The investigation of using CFA as a rule generator for IDS can be suggested as a future work. Moreover, the use of other techniques such as support vector machines, neural networks, clustering methods instead of using DT remains an open issue. Comparisons of feature selection techniques will also provide clues for constructing more effective models for intrusion detection.

References

Basiri, M. E., Ghasem-Aghaee, N., & Aghdam, M. H. (2008). Using ant colony optimization-based selected features for predicting post-synaptic activity in proteins. *Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, Lecture Notes in Computer Science, 4973*, 12–23.

Bolon-Canedo, V., Sanchez-Marono, N., & Alonso-Betanzos, A. (2011). Feature selection and classification in multiple class datasets: An application to KDD Cup 99 dataset. *Expert Systems with Applications, 38*(5), 5947–5957.

Chen, R.-C., Cheng, K.-F., & Hsieh, C.-F. (2009). Using rough set and support vector machine for network intrusion detection. *International Journal of Network Security and its Applications (IJNSA), 1*(1), 1–13.

Datti, R., & Lakhina, S. (2012). Performance comparison of features reduction techniques for intrusion detection system. *International Journal of Computer Science and Technology (IJCST), 3*(1), 332–335.

Depren, O., Topallar, M., Anarim, E., & Ciliz, M. K. (2005). An intelligent intrusion detection system (IDS) for anomaly and misuse detection in computer networks. *Expert Systems with Applications, 29*(4), 713–722.

Eesa, A. S., Abdulazeez, A. M., & Orman, Z. (2013). Cuttlefish algorithm – a novel bio-inspired optimization algorithm. *International Journal of Scientific and Engineering Research, 4*(9), 1978–1986.

Eric, K., Lydia, M. M., Roger, T. H., Patrick, B. D., Rajesh, R. N., Eric, F., et al. (2012). Biological versus electronic adaptive coloration: how can one inform the other. *Journal of the Royal Society Interface*. http://dx.doi.org/10.1098/rsif.2012.0601.

Fodor, I. K. (2002). *A survey of dimension reduction techniques*. Center for Applied Scientific Computing. Lawrence Livermore National Laboratory.

Horng, S.-J., Su, M.-Y., Chen, Y.-H., Kao, T.-W., Chen, R.-J., & Lai, J.-L. (2011). A novel intrusion detection system based on hierarchical clustering and support vector machines. *Expert Systems with Application, 38*(1), 306–313.

Lassez, J.-L., Rossi, R., Sheel, S., & Mukkamala, S. (2008) Signature based intrusion detection using latent semantic analysis. In *Neural networks, 2008. IJCNN 2008 (IEEE world congress on computational intelligence)* IEEE International Joint Conference.

Liao, H.-J., Lin, C.-H. R., Lin, Y.-C., & Tung, K.-Y. (2013). Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications, 36*(1), 16–24.

Lin, S.-W., Ying, K.-C., Lee, C.-Y., & Lee, Z.-J. (2012). An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection. *Applied Soft Computing, 12*(10), 3285–3290.

Mohanabharathi, R., Kalaikumaran, T., & Karthi, S. (2012). Feature selection for wireless intrusion detection system using filter and wrapper model. *International Journal of Modern Engineering Research (IJMER), 2*(4), 1552–1556.

Neelakantan, N. P., Nagesh, C., & Tech, M. (2011). Role of feature selection in intrusion detection systems for 802.11 networks. *International Journal of Smart Sensors and Ad Hoc Networks (IJSSAN), 1*(1), 98–101.

Nguyen, H. T., Franke, K., & Petrovic, S. (2010). Towards a generic feature-selection measure for intrusion detection. In *International conference on pattern recognition*.

Salzberg, S. L. (1994). Book review: C4.5: Programs for machine learning by J. Ross Quinlan. *Machine Learning, 16*, 235–240.

Stein, G., Chen, B., Wu, A. S., & Hua, K. A. (2005). Decision tree classifier for network intrusion detection with GA-based feature selection. In *Proceedings of the 43rd annual southeast regional conference*. New York: ACM.

Tsang, C.-H., Kwong, S., & Wang, H. (2007). Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection. *Pattern Recognition, 40*(9), 2373–2391.

Van der Maaten, L. J. P., Postma, E. O., & van den Herik, H. J. (2008). *Dimensionality reduction: A comparative review*. Elsevier.

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