GMDH-based networks for intelligent intrusion detection

Zubair A. Baig a,*, Sadiq M. Sait a, AbdulRahman Shaheen a

a Department of Computer Engineering, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia
b Center for Communications and IT Research, King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia

A R T I C L E   I N F O

Article history:
Received 21 April 2012
Received in revised form
12 March 2013
Accepted 18 March 2013
Available online 22 April 2013

Keywords:
Network intrusion detection
GMDH
Feature ranking
Machine learning

A B S T R A C T

Network intrusion detection has been an area of rapid advancement in recent times. Similar advances in the field of intelligent computing have led to the introduction of several classification techniques for accurately identifying and differentiating network traffic into normal and anomalous. Group Method for Data Handling (GMDH) is one such supervised inductive learning approach for the synthesis of neural network models. Through this paper, we propose a GMDH-based technique for classifying network traffic into normal and anomalous. Two variants of the technique, namely, Monolithic and Ensemble-based, were tested on the KDD-99 dataset. The dataset was preprocessed and all features were ranked based on three feature ranking techniques, namely, Information Gain, Gain Ratio, and GMDH by itself. The results obtained proved that the proposed intrusion detection scheme yields high attack detection rates, nearly 98%, when compared with other intelligent classification techniques for network intrusion detection.

© 2013 Elsevier Ltd. All rights reserved.

1. Introduction

The ever-expanding nature of Internet traffic accompanied with convenient availability of open source tools to launch malicious attacks has placed a demand for better network intrusion detection systems to detect such attacks accurately, so as to initiate subsequent countermeasures. Malicious attacks launched against an organization's computing infrastructure may cause huge financial losses. Accurate detection is an important first step towards securing a computer network. Rapid detection will allow the victim network to trigger appropriate countermeasures to reduce the effects of these attacks. A victim may range from a critical server operating to serve a client-base, to an entire infrastructure network. The attacks themselves vary in type and scopes of their abilities, from trojan horses to report back stolen information to the attacker, to distributed intensity-driven attacks such as Denial of Service (DoS). While the former tends to operate in the background and study the behavior of a machine, with the intent of stealing sensitive information, the latter attack type involves the participation of multiple attacker machines (which may be unaware of their participation in the attack), to send high volumes of traffic to the victim machine in a short interval of time. As a result, traffic will aggregate at the victim's end beyond its ability to process such inflow of high magnitude, consequently causing it to be incapacitated from providing further services.

* Corresponding author. Tel.: +966544078080.
E-mail addresses: baig.zubair@gmail.com, zbaig@kfupm.edu.sa (Z.A. Baig), sadiq@kfupm.edu.sa (S.M. Sait), shaheen@kfupm.edu.sa (A. Shaheen).

0952-1976/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.engappai.2013.03.008
signatures do not exist in the signatures database. Therefore, to maintain a high degree of accuracy in intrusion detection, an updated version of the attack signature database needs to be introduced to the misuse detector, on a frequent basis, for retraining. Misuse detectors are known for their high degree of accuracy and efficiency in attack detection (Barbara et al., 2001; Stolfo et al., 2001).

In this paper, the problem of network intrusion detection is addressed through classification of network traffic as either normal or anomalous. We introduce the use of Group Method for Data Handling (GMDH)-based networks (Ivakhnenko, 1966) for intelligent classification of network traffic. The GMDH technique has been found to hold promise in the field of intelligent computing. Such a technique for data classification based on established input–output relationships of a dataset, has been applied to diverse application domains, such as educational testing, pattern recognition, spam email classification (Abdel-Aal, 2005), and even for intrusion detection (Onwubolu and Sharma, 2008). Unlike regression-based techniques, the GMDH technique does not require user intervention for specifying the model relationship or the architecture of the neural network a priori. In addition, it performs well even with fewer training parameters, and yields high accuracies (Agarwal, 1999; Montgomery and Drake, 1991).

The scheme proposed in this paper operates in two phases. During phase 1, selection of the most appropriate network traffic features is performed, and during phase 2, the network traffic is classified as being either normal or anomalous through the use of GMDH-based networks, tested on both ranked as well as the entire feature set. The GMDH network models are built at various levels of complexity and their attack classification performance is studied for the KDD-99 dataset (Kayacik et al., 2005). The ranking of network traffic features, which are 41 in number for the KDD-99 dataset, is done based on three statistical ranking techniques, namely, Information Gain, Gain Ratio, and GMDH. Our proposed scheme selects the top m common features from the ranked feature lists generated by these three techniques. These features are subsequently introduced as input to the GMDH network for generation of models for network traffic classification. The KDD-99 dataset has a total of 22 attack types, and 1 normal type, as labels (y_i) for each data sample x_i. The GMDH models are generated based on a variation of the model complexity, defined through the Complexity Penalty Multiplier (CPM) parameter. The resulting models are then subject to an unlabeled segment of the dataset, to test the classification accuracy of the generated models.

The contributions of this paper can be outlined as follows:

- Introduction of three prominent statistical ranking techniques, to identify the most relevant network traffic features for the dataset,
- Proposal of a GMDH network-based approach for classification of network traffic into normal or anomalous, and
- Analysis of the simulation results obtained when the proposed scheme is tested on unlabelled network traffic data.

The remainder of this paper is organized as follows. Section 2 discusses related work found in the literature for intelligent network intrusion detection. GMDH-based networks are elaborated upon in Section 3. In Section 4, a detailed description of the proposed intrusion detection technique is provided. Section 5 provides the simulation results together with a detailed analysis and insight. Finally, the concluding remarks are stated in Section 6.

2. Literature Review

Neural networks and Artificial intelligence (AI) techniques have been widely employed for the detection of anomalous traffic in computer communication networks. In this section we summarize some relevant work available in the literature.

An artificial neural network (ANN) consists of a collection of processing elements that are highly interconnected and provide the necessary structure for classifying inputs into expected outputs. They provide the potential to identify and classify network activity based on limited, incomplete, and nonlinear data sources (Cannady, 1998). The neural network performs generalization of malicious attacks for imprecise and uncertain information (Moradi and Zulkernine, 2004), which gives it additional ability to detect novel attacks. Neural network structures have been applied in building anomaly intrusion detection systems and the two most common architectures are the Self-Organizing Maps (SOMs) and its variants, and the Multilayer Perceptron (MLP) (Tavallaee et al., 2009; Mitrokotsa and Doulgeris, 2005).

Self-Organizing Map (SOM) is an unsupervised learning algorithm employed to group similar data into clusters. It is a data visualization technique that produces a low dimensional topological map to help understand the original high dimensional data. Once the neural network is trained, the map converges to a stationary distribution and shows a clear separation between normal traffic and attack traffic. The output neurons are considered as the counts for normal and attack traffic points. After building the map using training data, future connections can be quickly classified as normal or anomalous based on their location in the map. Kayacik et al. (2003), Depren et al. (2005), and DeLooze (2006) used SOM in their IDS research.

Kohonen’s Emergent Self-Organizing Maps is also popularly known as a winner-take-all unsupervised neural network. It is unsupervised because there is no target vector which requires the administrator to label the clusters into normal cluster and attack clusters. This approach has advantage of combining machine learning and visualization techniques. However, KSOmS has limited number of neurons in the order of tens, which is not enough for analysis of large datasets with a large number of features. Emergent Self-Organizing Maps produce topological maps that illustrate the intra-data similarity. The map will represent the network traffic in data points in clusters which help to classify it into normal or anomalous depending on the position of its best match cluster. Valleys will have the data points that belong to same class of traffic. Borders will have some points that can be classified to the nearest matched valley.

The MLP is a supervised learning algorithm which uses a feed-forward structure to solve the classification problem. MLP neural networks are trained by manipulating the weights of the neural network connections. The network weights are updated by using different functions during the training period, such as the gradient-based optimization algorithm. When the network converges to the local minima of error, the output layer of the network will show the expected result. Faragoun and Boukelf (2006) propose a hybrid method of the k-means algorithm and MLP. The k-means algorithm is used to group the input data into a number of clusters (22 in their case, based on the number of attacks provided in KDD99). The distances between the centers of clusters and input data points are calculated, and only the most discriminating samples that cover the maximum region of each class are selected for the learning process. The selected samples are then presented to the MLP network for division into four classes of attacks, namely, DoS, Probing, U2R, and R2L.

An MLP for misuse detection was proposed by Cannady (1998) that uses two configurations. The first is a stand alone one, and the second uses a rule-based expert system (Cannady, 1998).
The proposed scheme uses nine traffic features as input to the MLP, which means the input layer should contain nine neurons. The MLP network consists of three layers: (i) the input layer with nine neurons, (ii) the hidden layer, and (iii) the output layer with two neurons, with all layers being fully connected. The Sigmoid function is used as a transfer function between the neurons. The author uses 10,000 data points, with 90% of it being used for training, and the remainder for testing.

The Random Neural Network (RNN) model (Gelenbe, 1990, 1989) has also been used successfully for a wide range of applications. It comes in two architectures, namely, feed-forward, or a fully recurrent architecture. RNNs have strong generalization capabilities, even when the training data set is relatively small compared to the actual testing data. The model also achieves fast learning due to its computational simplicity for the weight updating process. RNN was used by Oke and Loukas (2007) for DDoS attack detection. It was used in conjunction with statistical variables like maximum likelihood, Hurst parameter, and Entropy. Hurst parameter gives network traffic self-similarity while Entropy shows how much data is contained in the traffic, that differentiates significantly between normal traffic and anomalous traffic (Oke and Loukas, 2007). In Flegel and Meier (2004), a novel 1-class Support Vector Machine (SVM) has been proposed, specifically designed for handling intrusion detection features, wherein a single sphere is used for representing the class of normal or anomalous connections, with all outliers labeled as being in the opposite class. A quarter-sphere approach was also defined and both the single and the quarter sphere techniques were tested on the KDD-99 dataset. The performance of both approaches under varying anomaly ratios was reported, with the highest average accuracy reported as approximately 90%, at the cost of a 10% false alarm rate, as obtained from the illustrated ROC curves.

In Omwubolu and Sharma (2008), a hybrid differential evolution-GMDH technique for network intrusion detection is proposed. The study evaluated the ability of the differential evolution technique in selecting the most appropriate parameters of the GMDH model to be generated. The resulting model was used for classification of the DARPA dataset entries into normal or anomalous. In Wasiuki et al. (2005), a framework is proposed for agent-based network intrusion detection. The authors have attempted to use the self-organization ability of GMDH for pattern classification, when applied to data obtained from local network traffic by the Snort system. However, the results of tests conducted for the proposed scheme were not reported. In contrast, our proposed scheme does feature pre-processing and subsequent model generation on the resulting ranked features of the KDD-99 dataset. The following sections describe the proposed technique and its performance when simulated. Unlike conventional neural networks, GMDH generates models to depict generalization over a dataset without user intervention, and performs well even in the presence of a few independent variables. We provide an in-depth study of the GMDH technique in the following section.

3. GMDH-based networks

The original GMDH is a supervised inductive algorithm for construction of self-organizing models of optimal complexity based solely on the input–output relationships of a given dataset, without the need for user intervention. It introduces a higher-order polynomial to relate each input variable \( m \) of the dataset to a single output variable \( y \). The procedure adopted by the GMDH technique for evolving the polynomial so as to find an optimal model to represent the input–output relationship was said to follow the way nature evolves (Farlow, 1981). In order to solve higher order polynomials using traditional techniques such as regression, it would take a substantial amount of time to solve \( e \) equations with \( e \) unknowns. On the contrary, through GMDH-based model building, the computational overhead is substantially reduced, as the independent variables (i.e. features) that do not have a high correlation with the outputs are discarded during each iteration of the procedure. Through inductive learning, the algebraic and finite difference types of polynomial equations, several of these derived, are used for making predictions. On the contrary, the abductive GMDH method repairs the original dataset through the replacement of non-essential independent variables with better estimates, obtained during each iteration, to improve the quality of the model that best generalizes the input–output relationship of the dataset.

The abductive induction mechanism, is based on the self-organizing polynomial GMDH (Farlow, 1984). It uses mathematical functions for representing numerical knowledge derived from data, and uses artificial neural networks for learning functional models by subdividing complex problems into smaller and simpler ones. This variant of the original GMDH method was developed for inductively creating abductive network models (Abdel-Aal, 2005). It is a powerful supervised inductive learning approach for automatically synthesizing neural network models from input–output data relationships. It is based on the concept of abducting reasoning (Kim and Nelson, 1996), wherein, reasoning is performed from a set of general principles to specifics under uncertainty, through the use of numeric functions, measures, and abductive modeling through machine learning. The model that is formed post-training is a layered network of functional elements connected in a feed-forward manner.

The GMDH approach is a proven concept for iterated polynomial regression that can generate polynomial models for a given dataset using effective predictors. The iterative process involves using initially defined simple intra-data regression relationships, to derive more accurate representations in subsequent iterations of the technique. The number of independent variables, i.e. features that are combined for generating the appropriate models is varied in each step, and the technique is known to perform well even in the presence of a small subset of independent variables in the generated models.

3.1. Steps of execution

The algorithm selects the polynomial relationships and the input combinations that minimize the prediction error, during each iteration. This prevents exponential growth in the number of polynomial models generated. Iterations are stopped automatically at a point in time, when a balance between model complexity for accurate fitting of the training data, and model simplicity that allows it to generalize new data accurately, is achieved.

In the classical GMDH-based approach, abductive network models are constructed through the following steps (Farlow, 1984):

1. Data separation: The dataset is to be split into two parts, one for generation of the GMDH models and the other to test the accuracy in classification of the generated models.
2. Modeling: The independent variables (i.e. features) are considered two at a time, for calculation of the least squares polynomial. For a single GMDH node, only one independent variable is considered, and the polynomial equation is limited to the third degree, i.e.

\[
y = z_0 + z_1 x + z_2 x^2 + z_3 x^3
\]
where \( x \) is the input to the node, \( y \) is the output of the node and \( z_0, z_1, z_2 \) and \( z_3 \) are the node coefficients.

The double node GMDH implementation takes two inputs and the third-degree polynomial equation includes a cross term so as to consider the interaction between the two inputs, i.e.

\[
y = z_0 + z_1 x_i + z_2 x_j + z_3 x_i^2 + z_4 x_i x_j + z_5 x_j^2 + z_6 x_i^3 + z_7 x_j^3
\]

where \( x_i, x_j \) are the inputs to the node, \( y \) is the output of the node and \( z_0 \) through \( z_7 \) are the node coefficients.

3. Evaluate: The models generated in the previous step are evaluated for each data point \( n \) of the training set \( N \), to construct a matrix \( Z \) of values generated when the obtained polynomial is used for evaluation of the data points, where, each column of \( Z \) represents the outputs generated when the polynomial of the previous step is used for classifying the \( N \) data points.

4. Replacement: The columns of the original variables \( X \) are replaced with those columns of \( Z \) which best predicted the output class \( y \). Specifically, the least square error \( d_j \) is computed as follows Ivakhnenko (1966):

\[
d_j^n = \frac{1}{t} \sum_{i=1}^{t} (y_i - z_{ij})^2
\]

where, \( t \) is the number of entries in the test data set.

5. Stopping criteria: The lowest value of \( d_j^n \) obtained from the previous step is checked to see if this value has decreased in magnitude from the previous iteration. If yes, continue with repetition of steps 2–4 for varying polynomial sizes, else stop execution.

### 3.2. Abductive network ensemble

Network ensemble is a learning approach where a set of network models, generated by the GMDH implementation based on varying complexities (defined through the CPM parameter), have their respective outcomes of individual data classification, merged, so as to attain higher degrees of classification accuracies. Each element of the network ensemble (or committee) is a GMDH model, trained on a mutually exclusive subset of the original training set. The resulting output of the classifier is generated through appropriate combination of the independent model outcomes of each committee member. The combination of the outputs of each ensemble member is achieved through the use of simple combination rules, such as

1. Simple majority vote: The categorical output of the classifier is a simple majority vote of the categorical output of each individual committee member. An odd number of members will ensure a clear bias towards one of two classes, as opposed to when even number of members constitute the committee.

2. Simple averaging of ensemble network members: In this method, the final output of the committee is computed based on the simple averaging of the outputs of the individual members, through the following relationship:

\[
z_i = \frac{1}{n} \sum_{i=1}^{n} y_i
\]

Through the testing of our scheme on ensembles of GMDH models, we obtained a set of results for comparison with monolithic GMDH test scenarios. The outcomes of our simulation exercise together with the analysis, is provided in Section 5.

### 3.3. Feature ranking

GMDH can also be used for ranking features of a given dataset. The feature ranking process is executed through the identification of the predictive quality of the data. The abductive learning
The features are filtered to create the most prominent feature subset before actual GMDH-based model generation is performed. The three feature ranking techniques constituting the proposed technique are summarized as follows:

1. **Information Gain**: is used to individually rank attributes based on class separation in the dataset rows. Attribute ranks can be calculated using Information Gain with respect to class based on the following formula:

   \[ \text{Information Gain} = (D_x) - (D_{-x}) \]

   where \( D_x \) is the information which includes attribute \( x \), and \( D_{-x} \) is information which excludes attribute \( x \). The value of \( D_{-x} \) is calculated as the average of each value that this particular attribute can take. The information itself is calculated using the entropy equation:

   \[ \text{entropy} = D_x = - \sum_{k=1}^{n} p_k \log p_k \]

   where \( p_k \) is the probability of occurrence of value \( k \) for feature \( x \), with the total number of distinct values of feature \( x \) being equal to \( n \).

2. **Gain Ratio**: is an improvement of the Information Gain technique that resolves the bias towards features which have a larger diversity of values. For example, if a dataset contains a diverse range of serial numbers of customers of a grocery store, then the Information Gain of the customer serial number will be high, and it will be used at the high level in decision trees. This bias degrades the ability of learning algorithms, such as decision trees, of generalization of new customers because the serial number will be considered on the top of the decision tree, as a result causing a skew in the accuracy in the recognition process. Information Gain Ratio corrects this shortcoming by taking the intrinsic information in terms of entropy of distribution of instance values, for a given attribute i.e. feature. The Gain Ratio is large when the data is evenly spread and is small when the data has a single value. It is calculated as following:

   \[ \text{Gain Ratio(Feature)} = \frac{\text{Information Gain}}{\text{Intrinsic Value}} \]

   where,

   \[ \text{Intrinsic Value} = - \sum_{v \in \text{values}(x)} \frac{|x \in S, \text{value}(x, a) = v|}{|S|} \log_2 \frac{|x \in S, \text{value}(x, a) = v|}{|S|} \]

   where \( S \) is the set of all samples of the dataset, \( x \) is a dataset sample, \( a \) is a feature of the dataset, \( \text{value}(a) \) is the set of all possible values of feature \( a \) of the dataset, and \( \text{value}(x, a) \) is defined as the value of feature \( a \) in the dataset sample \( x \). Information Gain is defined through Eq. (7). The numerator is the information we learn about the class. The denominator however, represents the information we learn about the attribute (feature), or in other words, the information necessary to specify the feature value of a particular attribute.

3. **GMDH** synthesizes optimized polynomial network structures through continuous iterations. Instead of using the two techniques for feature ranking mentioned above, a straightforward approach towards classifying the data is to use GMDH for feature ranking as well, prior to actual classification of the data. Feature ranking using abductive networks through the wrapper approach (Witten et al., 2001; Bello et al., 2008; Guyon, 2009) is done based on the predictive quality of the data, and consists of the following steps:

   a) Model synthesis to select three inputs (features) to the abductive network at any given time.
(b) Removal of selected features to force the model to select from the less-predictive remaining features.

(c) Repetition of the process until all features are selected or no further features can be selected.

(d) Change model complexity in steps from small to large, if needed, to force the modeler to select the remaining features.

After feature ranking is completed, the top ranked features are considered for attack detection, one at a time as long as the accuracy of the selected model is non-decreasing. We stop when the accuracy drops, as an indication of model overfitting. In Section 5, we follow this procedure at different levels of model complexity, numbers of layers, and numbers of inputs, and study the corresponding effect on the attack detection process.

A comparison of the impact of all three techniques on the attack detection accuracy is provided in Section 5.

Subsequent to filtering and selection of the highest ranked features for the intrusion detection process, the reduced data set is used for training and evaluating the detection scheme.

5. Simulation results and analysis

This section describes the simulation performed for feature ranking based on the three statistical techniques defined in Section 3, and to build models of GMDH networks for network traffic classification. The simulator as such provides for simultaneous feature selection and model building on the dataset. The dataset itself was partitioned with 75% of it being used for training and the remainder 25% of unlabeled data used for testing the accuracy of the proposed technique. The GMDH networks were modeled at various levels of complexity, defined through the CPM (Complexity Penalty Multiplier) parameter. The value of CPM has an inverse effect on the complexity of the model generated (i.e. the number of levels of the model and the interconnections between the levels). Therefore, smaller CPM values will lead to more complex models as opposed to larger ones.

5.1. The dataset

For testing the accuracy of the GMDH models in distinguishing normal from anomalous traffic, the KDD-99 dataset (Tavallaee et al., 2009) was used. This dataset was originally derived from the raw DARPA network traffic. In the dataset, the network connection details that were obtained from the raw data were parsed into a vector with 41 distinct features. The processing of raw network connection data was carried out through the use of data mining and expert systems, so as to emulate a misuse-based network intrusion detection system. In addition, each attack type of the dataset was categorized into one of four categories, namely, Denial of Service, U2R, R2L, and Probing. Several intrusion detection schemes have been proposed in the past, with their corresponding performances being tested on the KDD-99 dataset (Yu, 2008; Sabhnani, 2003; Ahmad et al., 2008; Middlemiss and Dick, 2003; Zhang et al., 2011). The 41 features of the dataset constituting the feature vector are constituted of

- Nine basic and header features to depict the state of each connection.

We utilized the NominalToBinary Weka WEKA Data Mining Software filter for obtaining a binary feature set for the dataset, to facilitate GMDH model creation. As a result, each k-valued feature of the dataset was transformed to k binary features. For instance, if the feature \( \text{protocol}_{\text{IP}} \) can possess one of three different nominal values, namely, \( \{TCP, UDP, RTP\} \), this particular feature will be transformed into three distinct features, labeled as \( TCP, UDP, \) and \( RTP \), respectively. Each of these three newly generated features will be able to hold a binary value to represent either the presence or absence of the particular feature in a sample of the original dataset. As a result, for a dataset with \( k \) nominal features, with each feature capable of possessing one of \( l_k \) distinct values (3 in our case for the example above), the total number of transformed features that will be obtained is equal to: \( \sum_{k=1}^{3} l_k \). The number of features that were obtained post-transformation for the KDD-99 dataset is equal to 123. In order to reduce the total number of features to be used for intrusion detection, the three feature selection techniques elaborated upon earlier were applied to this transformed dataset.

5.2. Performance measures

The performance of intelligent classifiers may be measured using several metrics. The confusion matrix is one such visualization tool used for tabulating the overall performance of the classifier. Each row of the matrix represents the instances in a predicted class, while each column represents the instances in an actual class. The following measures are derived from the confusion matrix, and will be used for evaluating the proposed

<table>
<thead>
<tr>
<th>Rank</th>
<th>GMDH</th>
<th>Information Gain</th>
<th>Gain Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td>38</td>
<td>36</td>
</tr>
<tr>
<td>2</td>
<td>36</td>
<td>39</td>
<td>32</td>
</tr>
<tr>
<td>3</td>
<td>62</td>
<td>63</td>
<td>45</td>
</tr>
<tr>
<td>4</td>
<td>45</td>
<td>62</td>
<td>59</td>
</tr>
<tr>
<td>5</td>
<td>67</td>
<td>66</td>
<td>58</td>
</tr>
<tr>
<td>6</td>
<td>69</td>
<td>36</td>
<td>71</td>
</tr>
<tr>
<td>7</td>
<td>66</td>
<td>67</td>
<td>72</td>
</tr>
<tr>
<td>8</td>
<td>72</td>
<td>68</td>
<td>39</td>
</tr>
<tr>
<td>9</td>
<td>73</td>
<td>71</td>
<td>62</td>
</tr>
<tr>
<td>10</td>
<td>59</td>
<td>45</td>
<td>14</td>
</tr>
<tr>
<td>11</td>
<td>71</td>
<td>72</td>
<td>63</td>
</tr>
<tr>
<td>12</td>
<td>74</td>
<td>58</td>
<td>38</td>
</tr>
<tr>
<td>13</td>
<td>58</td>
<td>59</td>
<td>19</td>
</tr>
<tr>
<td>14</td>
<td>60</td>
<td>56</td>
<td>11</td>
</tr>
<tr>
<td>15</td>
<td>70</td>
<td>32</td>
<td>66</td>
</tr>
<tr>
<td>16</td>
<td>14</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>17</td>
<td>32</td>
<td>14</td>
<td>67</td>
</tr>
<tr>
<td>18</td>
<td>61</td>
<td>65</td>
<td>68</td>
</tr>
<tr>
<td>19</td>
<td>2</td>
<td>69</td>
<td>41</td>
</tr>
<tr>
<td>20</td>
<td>19</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>56</td>
<td>64</td>
<td>56</td>
</tr>
<tr>
<td>22</td>
<td>57</td>
<td>57</td>
<td>23</td>
</tr>
<tr>
<td>23</td>
<td>63</td>
<td>74</td>
<td>24</td>
</tr>
<tr>
<td>24</td>
<td>64</td>
<td>11</td>
<td>16</td>
</tr>
<tr>
<td>25</td>
<td>3</td>
<td>73</td>
<td>6</td>
</tr>
<tr>
<td>26</td>
<td>65</td>
<td>60</td>
<td>17</td>
</tr>
<tr>
<td>27</td>
<td>68</td>
<td>61</td>
<td>10</td>
</tr>
<tr>
<td>28</td>
<td>11</td>
<td>20</td>
<td>25</td>
</tr>
<tr>
<td>29</td>
<td>20</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>30</td>
<td>28</td>
<td>4</td>
<td>21</td>
</tr>
</tbody>
</table>

- Thirteen content-based features derived from network traffic payload. These features were constructed to identify U2R and R2L attacks.
- Ten host-based header features constructed over a 100 s time window to detect slow probes (Denial of Service) attacks.
- Ten temporal header features constructed over a 2 s time window, and
scheme:

\[
\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP} \tag{9}
\]

\[
\text{Recall} = \text{true positive rate} = \frac{TP}{TP + FN} \tag{10}
\]

\[
\text{Precision} = \frac{TP}{TP + FP} \tag{11}
\]

\[
\text{Specificity} = \frac{TN}{TN + FP} \tag{12}
\]

\[
\text{Detection rate} = \frac{\text{Attacks detected}}{\text{Total Number of attacks}} \times 100\% \tag{13}
\]

where TP, true positive is the number of normal test samples classified correctly. FP, false negative is the number of normal test samples classified as attacks. TN, true negative is the number of attack test samples classified correctly. FN, false negative is the number of attack test samples classified as normal.

The Receiver Operating Characteristic (ROC) area is used for weighing the performance of the GMDH classifier on the input feature set, through the following defined levels:

- 1.0: perfect prediction
- 0.9: excellent prediction
- 0.8: good prediction
- 0.7: mediocre prediction
- 0.6: poor prediction
- 0.5: random prediction
- < 0.5: poor prediction.

Precision–Recall (PR) curves, often used in Information Retrieval (Manning and Schutze, 2000; Raghavan et al., 1989), have been cited as an alternative to ROC curves for tasks with a large skew in the class distribution (Bunescu et al., 2005; Goadrich et al., 2004). An important difference between ROC space and PR space is the visual representation of the curves. Looking at PR curves can expose differences between algorithms that are not apparent in ROC space.

5.3. Feature ranking results and analysis

Simulations performed to test the proposed scheme can be divided into two phases. During Phase 1, the three feature ranking techniques defined in Section 3, were implemented to rank the features of the dataset. For running simulations based on selected features, the commonly occurring features in the three lists of ranked features, are selected (see Table 1). These selected features are then introduced to the abductive network during Phase 2 of the scheme, for building generic models to represent the dataset (i.e. training), and for subsequent classification of unlabeled data, i.e. testing of the dataset to quantify the accuracy in attack detection.

5.4. Monolithic abductive models

For monolithic abductive models, we ran the simulation using all features from the training set, and different CPM values, i.e. \(CPM = 0.1, 0.5, 1, 2, 5\). It may be noted that all features, regardless of their ranking, were introduced to the simulator, incrementally. A total of 65 abductive network models were built, with each model consisting of four layers and varying CPM values. In Table 2, the attack detection and the false alarm rates are illustrated for five synthesized GMDH models with varying CPM values. It is evident from the findings that the accuracy of the synthesized models remains consistent around 97.6%, unaffected by the variation of the CPM value. The false alarms associated with the scheme can be seen to remain constant at 2.2%, for varying values of CPM. The simulation was run a second time with the number of features selected, with five layer abductive networks.
network layers set to five and with $CPM=1$. Fig. 2 shows that the model stabilizes both in terms of the detection rate as well as the false alarm rates, beyond $k = 20$, when a total of five layers of an

Table 2

<table>
<thead>
<tr>
<th>CPM</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
<th>TP</th>
<th>FAR</th>
<th>DR</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>57</td>
<td>2769</td>
<td>65</td>
<td>2358</td>
<td>0.022</td>
<td>0.976</td>
</tr>
<tr>
<td>0.5</td>
<td>57</td>
<td>2769</td>
<td>65</td>
<td>2358</td>
<td>0.022</td>
<td>0.976</td>
</tr>
<tr>
<td>1</td>
<td>60</td>
<td>2777</td>
<td>57</td>
<td>2355</td>
<td>0.020</td>
<td>0.975</td>
</tr>
<tr>
<td>2</td>
<td>56</td>
<td>2773</td>
<td>61</td>
<td>2359</td>
<td>0.021</td>
<td>0.976</td>
</tr>
<tr>
<td>5</td>
<td>57</td>
<td>2769</td>
<td>65</td>
<td>2358</td>
<td>0.022</td>
<td>0.976</td>
</tr>
</tbody>
</table>

Table 3

Outcomes of simulation done to study the effect of synthesizing models based on the top-ranked 14 and 20 features on the attack detection process.

<table>
<thead>
<tr>
<th>No. of features</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
<th>TP</th>
<th>FAR</th>
<th>DR</th>
</tr>
</thead>
<tbody>
<tr>
<td>14</td>
<td>112</td>
<td>2753</td>
<td>81</td>
<td>2303</td>
<td>0.028</td>
<td>0.953</td>
</tr>
<tr>
<td>20</td>
<td>74</td>
<td>2775</td>
<td>59</td>
<td>2341</td>
<td>0.020</td>
<td>0.969</td>
</tr>
</tbody>
</table>

Table 4

Performance results of different network models synthesized using top 14 features selected using different feature selection algorithms.

<table>
<thead>
<tr>
<th>Model</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
<th>TP</th>
<th>FAR</th>
<th>DR</th>
</tr>
</thead>
<tbody>
<tr>
<td>GMDH</td>
<td>112</td>
<td>2753</td>
<td>81</td>
<td>2303</td>
<td>0.028</td>
<td>0.953</td>
</tr>
<tr>
<td>Information gain</td>
<td>152</td>
<td>2712</td>
<td>122</td>
<td>2263</td>
<td>0.043</td>
<td>0.937</td>
</tr>
<tr>
<td>Gain Ratio</td>
<td>59</td>
<td>2602</td>
<td>232</td>
<td>2356</td>
<td>0.082</td>
<td>0.975</td>
</tr>
</tbody>
</table>

Table 5

Performance results of ensemble network individual models synthesized using $CPM=1$.

<table>
<thead>
<tr>
<th>Model</th>
<th>FN</th>
<th>TN</th>
<th>FP</th>
<th>TP</th>
<th>FAR</th>
<th>DR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monolithic</td>
<td>60</td>
<td>2777</td>
<td>57</td>
<td>2355</td>
<td>0.020</td>
<td>0.975</td>
</tr>
<tr>
<td>Majority vote ensemble</td>
<td>63</td>
<td>2773</td>
<td>61</td>
<td>2352</td>
<td>0.021</td>
<td>0.973</td>
</tr>
</tbody>
</table>

Table 6

Performance comparison of various intelligent techniques for network intrusion detection.

<table>
<thead>
<tr>
<th>Intrusion detection scheme</th>
<th>False alarm rate (%)</th>
<th>Attack detection rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCC (Shyu et al., 2003)</td>
<td>2</td>
<td>96.07</td>
</tr>
<tr>
<td>GMDH</td>
<td>2.25</td>
<td>97.72</td>
</tr>
<tr>
<td>AODE (Baig et al., 2011)</td>
<td>0.01</td>
<td>99.54</td>
</tr>
<tr>
<td>NB</td>
<td>1.08</td>
<td>88.55</td>
</tr>
<tr>
<td>MLP (Sabhnani and Serpen, 2003)</td>
<td>3.5</td>
<td>93</td>
</tr>
</tbody>
</table>

Fig. 5. Comparison of the Precision–Recall curve for two abductive network classifiers synthesized using 14 and 20 top ranked GMDH features.

Fig. 6. Comparison of the Precision–Recall curve for three abductive network classifiers: network models synthesized using top 14 GMDH, Gain Ratio, and Information Gain-ranked features.

Fig. 7. Comparison of the Precision–Recall curve for two abductive network classifiers: the optimum monolithic model when $CPM=1$, and a three member network ensemble based on majority voting.

Fig. 8. Comparison of the Precision–Recall curve for two abductive network classifiers: the optimum monolithic model when $CPM=1$, and a three member network ensemble based on majority voting.

Fig. 9. Comparison of the Precision–Recall curve for two abductive network classifiers: the optimum monolithic model when $CPM=1$, and a three member network ensemble based on majority voting.

Fig. 10. Comparison of the Precision–Recall curve for two abductive network classifiers: the optimum monolithic model when $CPM=1$, and a three member network ensemble based on majority voting.

Fig. 11. Comparison of the Precision–Recall curve for two abductive network classifiers: the optimum monolithic model when $CPM=1$, and a three member network ensemble based on majority voting.

abductive network (for $CPM=1$) are used for classification. It may be noted that with $k = 1$, the detection rate was found to be 100%, albeit with a very high false alarm rate. It is therefore impractical to have a scheme implementation wherein only a single feature is used for classification.

In Fig. 3, the ROC curve is illustrated as a performance measure for the abductive models, for varying CPM values. From these figures, it is evident that the CPM value does not have an effect on the attack detection rate. The area under the curve is nearly 99% for all cases. Fig. 4 shows the Precision–Recall curves for different abductive networks synthesized using varying model complexity values. The observed results are again unaffected by varying CPM values.
values. However, higher CPM values proved to slightly improve the performance as compared to lower CPM values, in terms of recall.

The simulation was also run with varying numbers of top-ranked features. When the top 14 and 20 ranked features were selected, with $CPM=1$, and the number of GMDH layers set as 4, the results were not as good as those obtained when using the full feature set, but were comparable to a certain extent. However, it was noticed that the training time for the reduced feature set simulation run was much less than the time required for running on the full feature set. The time required for training, i.e., abductive network model building, was found to improve with decreasing numbers of features. When all 123 features were used for training, the simulator took an estimated 1805 s for model building, whereas, with 14 features selected, the training time reduced to 589 s (Table 2).

Table 3 shows the results of using the top 14 and 20 commonly ranked features by all three techniques from Section 3. For the same simulation, Fig. 5 illustrates the precision-recall curves. As may be observed from both Table 3 and Fig. 5, the detection rate showed a slight degradation in performance, reaching a maximum of only 96.9% when 20 features are selected, as opposed to 97.9% when all features are selected (from the previous subsection results). For the 14-feature case, the false alarm rate was found to be 2.8%, whereas if 20-features are selected, the false alarm rate drops down to 2.0%.

5.5. Abductive networks for top-ranked features

After performing simulation runs with all features selected, we synthesized abductive networks using the top 14 selected features, ranked by the three feature selection algorithms outlined in Section 3. The resulting networks are compared based on the precision-recall curve, as shown in Fig. 6. The area under the curve for the GMDH selected features, is 0.993, whereas the area under the curve for the abductive network model synthesized using Gain Ratio-selected features is 0.990, and the area under the curve for the abductive network model synthesized using Gain for the GMDH selected features, is 0.993, whereas the area under the curve was found to be 0.9963, whereas for the monolithic model, wherein the closest model to a given input was selected without having the need for a committee of classifiers for deciding the outcome of the attack detection process, was found to be 0.993.

Table 5 provides a comparison of the results obtained through network ensembles against monolithic networks. The attack detection rate for the monolithic network was found to be 97.5% as compared to a 97.3% rate for the ensemble network. In addition, the false alarm rates for both approaches were comparable, at 2.0% and 2.1%, respectively. Therefore, it may be conclusively stated that the effect of an ensemble networks on improving the performance of the proposed approach for intrusion detection, is insignificant.

5.7. Performance comparison

Table 6 shows that the performance of our proposed scheme falls second only to Averaged One-Dependence Estimator (AODE) in terms of attack detection rates and fourth in terms of false alarm rates. Although the false alarms generated by AODE, Naive Bayes and Principal Component Analysis (PCA) are less than those generated by our proposed scheme, the attack detection rate is only second to AODE. We can therefore infer from the findings that the scheme proposed in this paper is closely comparable to the best known schemes for network intrusion detection, through intelligent classification.

6. Conclusions

Abductive learning methods have been found to hold promise in the field of intelligent computing. Through this paper, a two-phased approach towards classifying network traffic into normal and anomalous, was proposed. The technique operates through the identification of the most significant features of the KDD-99 dataset during phase 1. The feature ranking process is performed based on three techniques, namely, Information Gain, Gain Ratio, and GMDH. Subsequently, these top-ranked features are introduced to the simulator for modeling of abductive networks. These models help classify the traffic data of the KDD-99 dataset into either normal or anomalous. Simulation results of the monolithic abductive network models with and without feature selection, were analyzed. In addition, the effect of varying GMDH model complexities (defined through the CPM parameter) on the performance of the scheme was analyzed. It was found that ranking and subsequent selection of ranked features improved the performance of the scheme in terms of improved attack detection rates and reduced false alarm rates, as opposed to when all features of the dataset were used. In addition, the training time significantly reduced with decreasing numbers of features. A similar set of simulations were performed for ensemble abductive networks. It was observed that ensemble networks based on majority voting yielded insignificant improvements in performance over monolithic abductive networks.

Acknowledgments

The authors wish to acknowledge the continued support for research provided by King Fahd University of Petroleum & Minerals.
This research work was conducted as part of research project no. NSTIP-11-INF1658-04.

References


