One-Dependence Estimators for Accurate Detection of Anomalous Network Traffic

Zubair A. Baig, Abdulrhman S. Shaheen, Radwan AbdelAal
Department of Computer Engineering
King Fahd University of Petroleum & Minerals
Dhahran 31261, Saudi Arabia

Abstract

Classification of Internet traffic as anomalous has remained an issue of concern for the network security research community in recent times. Advances in computing performance, in terms of processing power and storage, have allowed the use of resource-intensive intelligent algorithms, to detect intrusive activities, in a timely manner. Naïve Bayes is a statistical inference learning algorithm with promise for document classification, spam detection and intrusion detection. The attribute independence issue associated with Naïve Bayes has been resolved through the development of one dependence estimation algorithms such as Super-Parent One Dependence Estimators (SPODE) and Average One Dependence Estimator (AODE). In this paper, we propose the design of an intrusion detection system based on the classification capabilities of SPODE and AODE for accurate detection of anomalous network traffic. The performance of the proposed scheme is studied and analyzed on the KDD-99 intrusion benchmark data set, with significantly high detection rates of the two algorithms as opposed to results obtained through Naïve Bayes simulation.

1. Introduction

With the rapid growth of the Internet and readily available open source tools for launching malicious attacks, attempts to intrude computer networks have increased manifold. The sophisticated nature of contemporary attacks can cause heavy financial losses, in addition to affecting an organization's reputation, due to the downtime of the network and associated service-based applications. Detecting these attacks is an important step toward protection against such attacks.

Intrusion Detection Systems (IDS) have emerged as a defense platform, by acting as an intermediary for identifying potentially harmful network traffic, before it penetrates a computer network. Generally, network intrusion detection systems detect these attacks by monitoring ingoing and outgoing traffic, in order to identify possible outliers in the observed traffic patterns, where outliers can possibly be anomalous traffic. Intrusion detecion systems can be categorized into two types - a) Anomaly detectors, which detect deviations of network traffic behavior from predefined normal traffic profiles, and b) Misuse detectors, which attempt to find signatures of malicious patterns known to the IDS beforehand [11].

Naïve Bayes is a statistical learning tool which can be trained by introducing a data set of network traffic. The training process involves the generation of a set of probabilities, which are subsequently used by the algorithm for confirming the class of actual network traffic, observed in real-time. However, Naïve Bayes does not consider interdependencies between individual features of observed network traffic. In other words, the network traffic features are assumed to be independent of each other. This assumption of exclusiveness in the observed network traffic may invariably diminish the accuracy of the intrusion detection system.

Several other techniques have subsequently been proposed, to alleviate the attribute independence assumption of Naïve Bayes. These include - One Dependence Estimators (ODE), Super Parent One Dependence Estimators (SPODE), and Average One Dependence Estimators (AODE) [18][21]. These techniques provide varying types of attribute dependence assumptions, with the promise of enhancing the attack detection accuracy. We postulate that due to the attribute dependence assumptions and the ability to average various models of observed traffic, AODE stands out as a technique capable of accurately estimating the probabilities of observed network traffic to belong to a given class i.e. $normal$ or $anomalous$, with a high degree

of accuracy.

As part of this paper, we design and test two intrusion detection systems, namely, SPODE-based and AODE-based, for estimating the probabilities of feature values of observed network traffic, to be classified as normal or anomalous. We performed simulations on the NSL-KDD 99 data set [10], which is a common benchmark for testing the capabilities of network intrusion detection systems.

The paper is organized as follows. In Section 2, we perform a literature review of various intelligent techniques that exist to classify network traffic. We elaborate upon all steps that constitute our proposed intrusion detection system, in Section 3. In Section 4, we present our simulation results, and analyze our findings. Finally, we conclude and provide future directions for our work, in Section 5.

2. Related Work

In the field of intrusion detection, Artificial Neural Networks (ANNs) have been widely studied, implemented and tested. The ability of a given ANN to perform classification of observed network traffic is dependant on the nature of the algorithm, if we consider the type of network traffic that is introduced to the ANNs to be a standard benchmark.

A supervised learning algorithm, namely, the Multilayer Perceptron (MLP), is proposed in [12], for network traffic classification. MLP neural networks are trained by alteration of weights that are assigned to the interconnections between the neural network nodes. This is accomplished by using different functions during the training period for the algorithm, such as gradient-based optimization algorithm. When the network converges to the local minima of error, the output layer of the network shows the expected result when data is fed into the input layer. Two types of MLP-based intrusion detection techniques are applied in [4]. The first version operates as a stand-alone service, whereas the second technique uses rule-based expert systems. The scheme uses nine features of observed network traffic, which are observed from the training data set. The input layer has nine neurons, and the output layer has two neurons, with all layers being fully connected. A Sigmoid function was used as transfer function between the neurons. The proposed scheme was tested using 10,000 data elements, i.e. parameters from observed traffic, with 1000 used for training, and the remaining for testing the performance of the algorithm.

Self-Organizing Maps (SOM) group data into clusters based on the similarity index. Subsequent to building a map using training data, data can be conveniently classified as normal or anomalous, based on its precise location on the map. SOMs have been used for designing intrusion detection systems, as given in [7][8][17].

Both MLPs and SOMs have been known to have sluggish performances, and take a long period of time for being trained/re-trained on a given data set.

The Random Neural Network (RNN) model operates as either a feed-forward or a fully recurrent network of neurons [13][14]. RNNs are known for their capability to generalize sophisticated data sets, even when the training dataset is small in size. The network also achieves fast learning due to its computational efficiency in its weight update process. RNN was used by Oke et. al. in [20] for detecting distributed denial of service (DDoS) attacks.

Bayesian learning is a supervised learning technique, which uses probabilistic models and apriori knowledge to classify observations [15]. Prior knowledge is incorporated naturally into a Bayesian network, and all uncertainty is handled in a consistent manner. Moreover it is possible to learn model structures and readily compare between model classes. Several different classes of Bayesian networks exist in the literature, namely, Naïve-Bayes [3], Tree augmented Naïve-Bayes (TANs) [2], Augmented Naïve-Bayes (BANs) [21], Bayesian multi-nets, and general Bayesian networks (GBNs) [6]. Such networks have been extensively studied in various disciplines such as text recognition [6], SPAM detection [9], intrusion detection [3], etc. Bayesian networks in general suffer from the assumption of independence between various features (attributes) of a data set, thus making them less accurate. Several schemes have been proposed to overcome this shortcoming, including Lazy Bayesian Rule (LBR) [23] and Super-Parent Tree Augmented Networks [2]. Although, these techniques demonstrate high performance, they suffer from high computational cost.

AODE [21], with accuracy and performance comparable to LBR and Super-Parent TAN, solves the problem of independence by averaging all models generated by one dependence estimators. AODE also solves the problem of large computation overhead, which was evident in the performance of its predecessor classifiers. It is also known to have low variance, and is suitable for incremental learning [6].

In this paper, we propose using both SPODE-based as well as AODE-based classifiers for detecting network intrusion, and analyzing their performance on the KDD-99 dataset. The NSL-KDD dataset is a reduced version of original KDD-99 dataset, and consists of the same set of features as KDD-99, i.e. 41 features and one class attribute. The class attribute has a total of 21 classes that represent the following four types of attacks: Probe attacks, User to Root (U2R) attacks, Remote to Local (R2L) attacks and Denial of Service attacks. This data set has a binary class attribute. Also, it has a reasonable number of training and test instances which make it practicable for training and testing simulations, to compare and benchmark different intrusion

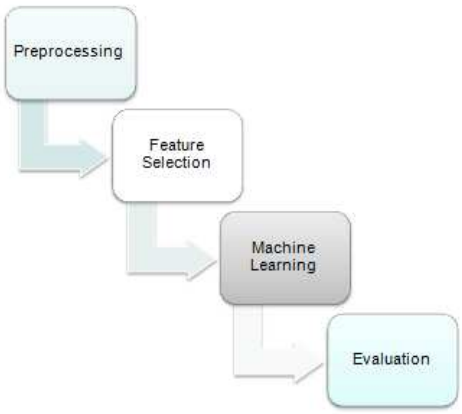

Figure 1. IDS implementation flowchart

detection techniques.

3. Proposed Intrusion Detection Scheme

In this section, we discuss our proposed scheme for intelligent classification of network traffic. The proposed scheme operates in four phases, as highlighted in Figure 1:

1. Preprocessing
2. Feature Selection
3. Machine Learning
4. Testing or Evaluation

In the following subsections, we describe the above steps of the intrusion detection process.

3.1. Preprocessing

Prior to the application of any training algorithm on a given data set, it is essential to convert all features (attributes) to a format that is intelligible by the classification algorithm. As a result, the effect of potential nullification of the impact of certain features on the outcome of the classification, is alleviated. In the NSL-KDD data set, all features of the data set take numeric values except three, namely, protocol_type, service, and flag. As part of the preprocessing phase, these features are converted into nominal values, so that the AODE training algorithm, which we use for network traffic classification, can operate on this data set, unaffected. The process of numeric to nominal conversion is achieved through discretization of the numeric values, using techniques such as equal frequency binning, wherein the frequency of occurrence of a certain data value defines the bin into which the data is placed, i.e. discretized.

3.2. Feature Selection for IDS

Subsequent to preprocessing of data, the features of the data set are identified as either being significant to the intrusion detection process, or redundant. This process is known as feature selection. Redundant features are generally found to be closely correlated with one or more other features. As a result, omitting them from the intrusion detection process does not degrade classification accuracy. In fact, the accuracy may improve due to the resulting data reduction, and removal of noise and measurement errors associated with the omitted features. Therefore, choosing a good subset of features proves to be significant in improving the performance of the system.

As part of the feature selection process for our proposed scheme, we use the following three techniques:

- *Group Method for Data Handling (or Abductive Networks):* Feature ranking using abductive networks [1], operates according to the predictive quality of features. The ranking process follows these steps:

  1. Change setting of model synthesis to select three inputs at a time. This method is used to restrict the number of selected features to three which effectively helps in ranking the features in groups of maximum size three.
  2. Remove selected features to force the model to select from the remaining less predictive features.
  3. Repeat the process until all features are selected or no features can be selected any more with changing model complexity in steps of varying CPM values, where CPM is defined as the complexity penalty multiplier.

  After ranking the features, a subset is selected by using the top ranked features one at a time, for as long as the accuracy of the selected model keeps increasing. When the accuracy of the model starts to drop, the model is considered to be overfitted, at which point the feature selection process is stopped.

- *Information Gain:* In this method, the features are filtered to create the most prominent feature subset before the start of the learning process. Attributes are individually ranked on the basis of separation of classes of the training data elements i.e. individual rows of the data set. The attribute ranks, with respect to the class, are calculated using the following formula:

$$Information\ gain = (D_x) - (D_{-x}) \qquad (1)$$

  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:

$$entropy = \sum_{k=1}^{n} p_k log p_k \quad (2)$$

- *Gain Ratio:* is a modification of information gain that solves the issue of bias towards features with a larger set of values, exhibited by information gain. For example, if a data set contains the serial numbers of grocery customers, then the information gain of the customer serial number (considered to be distinct for all customers) will be high, and it will be used in the decision-making process. This bias degrades the accuracy of learning algorithms. Gain Ratio is large when the attribute values are evenly spread, and is small when an attribute has a single value. For a given feature x and a feature value of y, it is calculated as follows:

$$GainRatio(y,x) = \frac{gain(y,x)}{intrinsic\ info(x)} \quad (3)$$

where,

$$intrinsic\ info(x) = -\sum \frac{\mid S_i \mid}{\mid S \mid} * log_2 \frac{\mid S_i \mid}{\mid S \mid} \quad (4)$$

$|S|$ is the number of possible values a feature x can take, and $|S_i|$ is the number of actual values of feature x.

3.3. Machine Learning

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 machine learning scheme. Training is performed on a subset of the data, and the performance of the resulting models is verified on the remaining parts of the data set. Naïve Bayes is a supervised learning algorithm that relies on probabilistic models and apriori knowledge for classifying data. It uses statistical inferences, with an assumption of attribute independence, where an attribute of a given data sample is a property of the sample with a given value. For instance, some of the attributes of a sample of network traffic may include - the arrival rate of the traffic, the number of data packets that use the TCP protocol, etc. However, assuming attribute independence does not effectively quantify relationships between real life data samples. Naïve Bayes has been tested to differentiate between different classes of a given data set for - a)classifying text documents, b) detecting spam email, and c) for network intrusion detection [2][21].

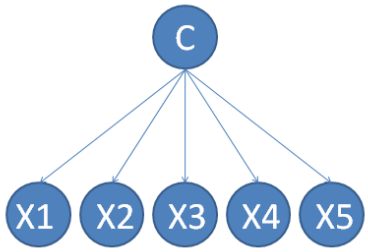

Figure 2. Naive Bayesian

On the contrary, a new breed of Naïve Bayes variants have emerged of late, to resolve the attribute independency issue. One dependence estimators were initially defined by Keogh et al. in [18]. As can be seen from Figure 3, individual attributes X_n belonging to class C are dependent on each other based on a given relationship function.

However, the ODE algorithm suffers from large computation overhead associated with its hill climbing search algorithm for model selection. Super Parent One Dependence Estimator (SPODE) was another algorithm proposed to solve the attribute independence assumption of Naïve Bayes. As illustrated in Figure 4, this algorithm reduced the computation overhead associated with ODE, by allowing each attribute to depend on only one common attribute called the super parent attribute, besides the class attribute C. SPODE was proposed to address some of the issues associated with the regular ODE. The proposed scheme required all attributes of a given feature set to depend on one single parent, namely, the Super Parent. As a result, an annotated set of all possible parents in a given attribute class does not arise, and only one parent needs to be tagged. SPODE by itself has two modes of operation: 1) Full SPODE, and 2) Partial SPODE. A full SPODE outperforms the latter, as a result of improved performance due to reduced dependence on multiple parents. The SPODE estimator with a superparent x_p finds the maximum value of the probability of a given data element to belong to class y as follows:

$$P(y,x) = P(y,x_p)P(x|y,x_p) = P(y,x_p)\prod_{i=1}^{m} P(x_i|y,x_p) \quad (5)$$

The SPODE algorithm operates by establishing a mathematical relationship between the values of individual network traffic features given by x_i, as compared with the value of a super-parent, x_p. For a given output class y, the likelihood of a data-row from the network traffic dataset, x, to belong to this particular class, is dependant on the probability of the super-parent feature to belong to this output

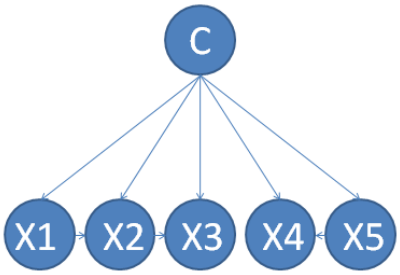

Figure 3. One Dependence Estimator

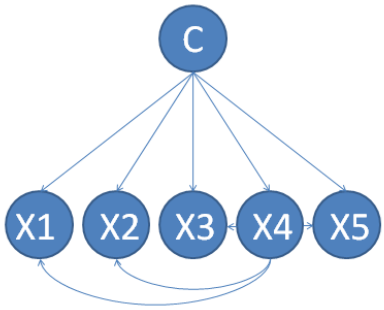

Figure 4. Super Parent One Dependence Estimator

class, given x. An alternate representation of this definition is to compute the probability of observing a feature value x_i, given the super-parent belongs to class y, times the probability of the super-parent x_p belonging to y. As can be observed from Equation 5, the super-parent class is essential in classifying any given data-row from the network traffic dataset into normal or anomalous, for the SPODE algorithm.

However, SPODE was also found to suffer from large computation overhead, required for model selection. Averaged One Dependence Estimators (AODE) were proposed to resolve the issues that were identified previously in the ODE and SPODE algorithms, by averaging all SPODE models that can be generated for a given set of attributes. As a result, the model generated proved to be a very accurate depiction of the data elements belonging to a given data set. [4, 5, 15, 16, 19, 22, 23]

The AODE algorithm averages over a space of alternative Bayesian models that have weaker independence assumptions than Naïve Bayes. The algorithm may give more accurate classification than Naïve Bayes on data sets with non-independent attributes. AODE was found to outperform both ODE and SPODE in terms of speed and accuracy, as elaborated upon by Webb et. al. in [21]. The training phase of the AODE algorithm operates by iterating through a given data set with k features, and generating a set of frequency vectors as follows:

1. $cfreq[y]$ - number of data elements belonging to a given class y
2. $afreq[k]$ - number of times a given data element is found to possess a value, iterated over all k features
3. $vfreq[x_i]$ - number of times the value x_i is encountered in the entire data set
4. $freq[y, x_i, x_j]$ - the frequency of simultaneous occurrence of two attribute values x_i and x_j for a given class y

During the testing phase of the AODE algorithm, the data elements or instances of the data set are introduced to the algorithm by hiding the class to which they belong. The task of the AODE classifier is to predict the probability of the data element to belong to each of the given classes. The higher probability is then used for deciding the class of the data element. These values are computed based on the following equations:

$$\forall i \in \ k \ , p = \hat{P}(x_i \wedge y) \tag{6}$$

$$\forall j \in k, if \ x_j \ \ is \ \ known, p = p \times \hat{P}(x_j | x_i \wedge y) \tag{7}$$

where, k is the total number of attributes, x_j is the value of a feature, $\hat{P}(x_i \wedge y)$ is the probability that the value x_i is observed given $x_i \in y$, and $\hat{P}(x_j | x_i \wedge y)$ gives the probability of observing the value x_j, given that the value $x_i \in y$.

4. Simulation Analysis

In this section, we analyze the results obtained from simulations performed to test the effectiveness of our proposed AODE-based intrusion detection system. The results were quantified based on the following metrics, commonly used for evaluating intelligent classifiers:

- Accuracy = $\frac{TP+TN}{TP+TN+FN+FP}$
- Recall = $\frac{TP}{TP+FN}$
- Precision = $\frac{TP}{TP+FP}$
- Specificity = $\frac{TN}{TN+FP}$

- Detection rate = $\frac{total\ number\ of\ detected\ attacks}{Total\ Number\ of\ attacks} * 100\%$

where,
TP - is the number of actual positives classified correctly as true.
FP - is the number of negatives in the data set classified incorrectly as positives.
TN - is the number of negatives classified correctly as negatives, and
FN - is the number of positives classified incorrectly as negatives.

For all simulations, we trained and tested the performance of the classifiers on the KDD-99 dataset. The first set of simulations were performed to test the performance of Naïve Bayes using a training set of 15747 instances (data elements), and a test set of size 5249. As can be noted from Figure 5, an accuracy in detection of nearly 94.19% was observed, when all the features of the data set were included i.e. no feature selection algorithm was used. Subsequently, the same set of simulations were performed, with the three feature selection techniques discussed previously. It can be seen from the figure that the GMDH algorithm yielded an accuracy of 97.30% in the detection process. We further tested the detection scheme for the top 16 ranked features obtained through information gain, to obtain an accuracy of 92.11%. Finally, the top 16 ranked features through the gain ratio technique resulted in an accuracy of 94.01%. The false alarm rate was observed as 2.5% for the GMDH simulation, 0.8% for the gain ratio simulation, 0.7% for information gain, and 1%, when all features were used. In order of accuracy in attack detection, the GMDH-based feature selection approach proved to increase the detection rate accuracy more than the other approaches studied.

Precision-recall curves help support findings in data sets with large skew in the class distribution. In Figure 6, we evaluate the performance of the various feature selection techniques in terms of the yielded precision and recall values. The Naïve Bayes models built using GMDH ranked features outperformed the models built using top ranked gain ratio features in term of recall. On the other hand, Naïve Bayes models built using top ranked gain ratio features outperformed the NB models built using GMDH best features in terms of precision. It may be noted that the gain ratio technique outperformed other techniques in terms of both precision and recall, with a peak detection rate of 99.4%. The GMDH-based approach yielded a detection rate of 94.2%.

Subsequently, we performed simulations for the SPODE-based intrusion detection system. The data set and the number of training and testing instances were kept the same as the Naïve Bayes simulations. As can be seen from Figure 7, an accuracy of 97.7% was obtained when all data

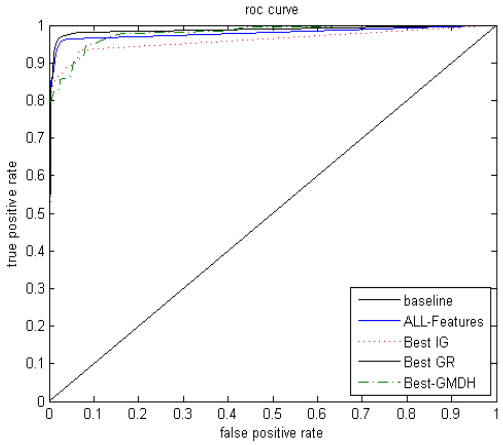

Figure 5. Comparison of the ROC curve for different feature sets used in building the NB model

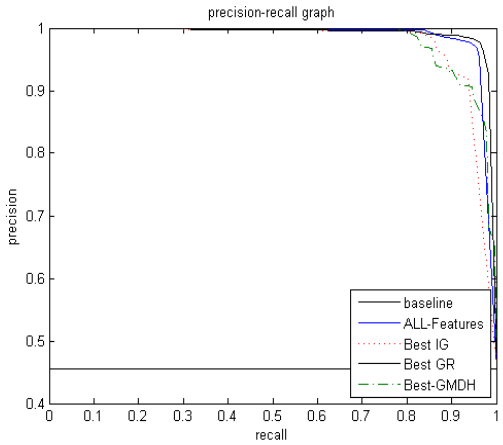

Figure 6. Comparison of the Precision-Recall curve for different NB models built using different feature selection techniques

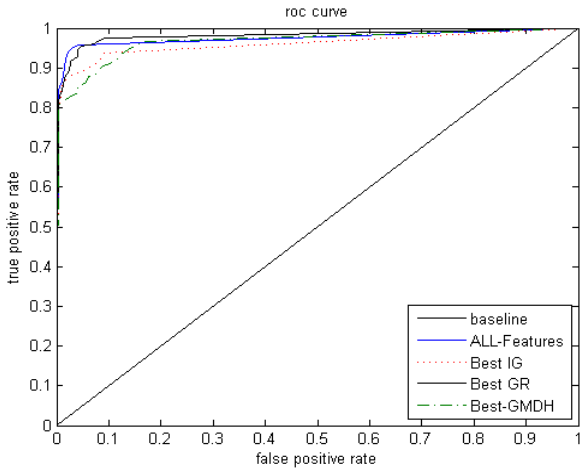

Figure 7. Comparison of the ROC curve for different SPODE models built using different feature selection techniques

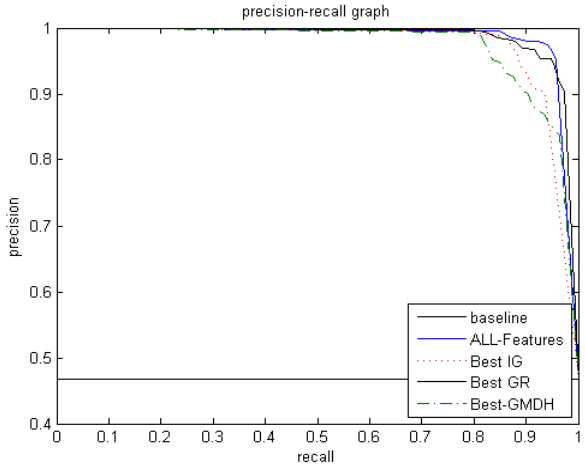

Figure 8. Comparison of the Precision-Recall curve for different SPODE models built using different feature selection techniques

set features were used. Subsequently, we tested the effect of using the top most prominent features selected by the GMDH algorithm, to obtain an accuracy of 96.98%, in the attack detection process. Then, we ran the simulation using the top 16 ranked features as obtained from the information gain technique, to acquire an accuracy in detection of 97.9%. Finally, we used the gain ratio technique to enlist the top 16 features, which yielded an accuracy of 97.86%. The corresponding false alarm rates were observed to be: 0.4% for gain ratio, 0.6% for information gain, 1.2% for GMDH, and 0.2% when all features were used.

Our next set of simulations were directed towards testing the effectiveness of our proposed AODE-based intrusion detection system. The data set and the number of training and testing instances were kept the same as the Naïve Bayes simulations. As observed from Figure 9, an accuracy of 99.7% was obtained when all data set features were used. Subsequently, we tested the effect of using the top most prominent features selected by the GMDH algorithm, to obtain an accuracy of 97.98%. Then, we ran the simulation using the top 16 ranked features as obtained from the information gain technique, to acquire an accuracy in detection of 99.4%. Finally, we used the gain ratio technique to enlist the top 16 features, which yielded an accuracy of 99.46%. The corresponding false alarm rates were observed to be: 0.5% for gain ratio, 0.6% for information gain, 1.2% for GMDH, and 0.1% when all features were used.

In Figure 10, we illustrate the precision and recall values for the detection scheme, for various feature selection algorithms used. Information gain and gain ratio were found to yield high precision and recall values. GMDH yielded

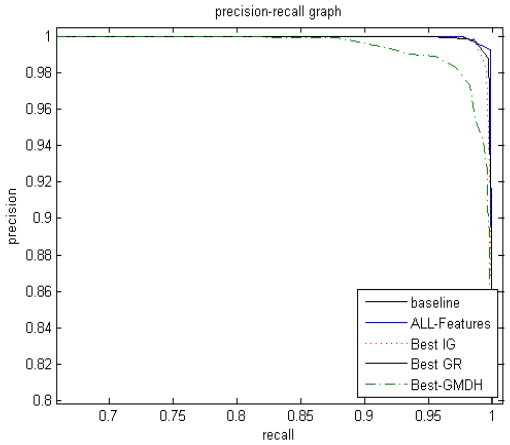

Figure 9. Comparison of the Precision-Recall curve for different AODE models built using different feature selection techniques

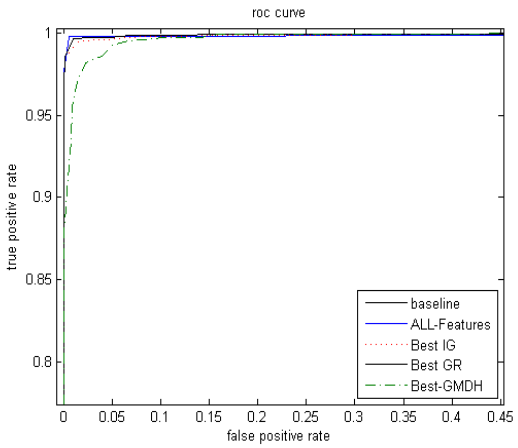

Figure 10. Comparison of the ROC curve for different AODE models built using different feature selection techniques

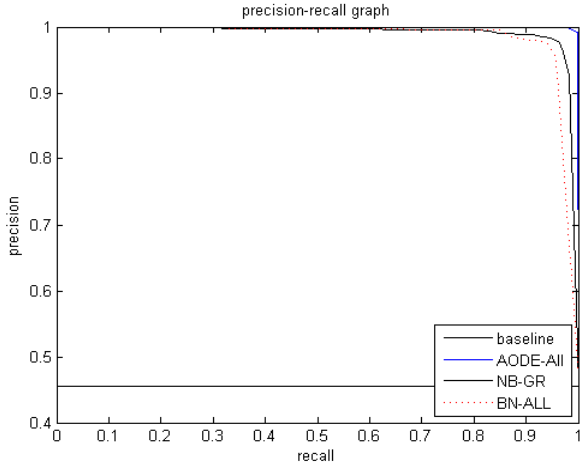

Figure 11. Comparison of the ROC curve for the best SPODE, AODE model and best NB model

a lower precision value, without much affect on the recall value.

In Figures 11 and 12, we compare the best AODE models and the best Naïve Bayes models by plotting the AODE data points when no feature selection algorithm was used, and the Naïve Bayes model when gain ratio was used. It may be noticed from these figures that the AODE-based models outperform Naïve Bayes both in terms of detection rate and false positive rates. The precision and recall values also allude towards the supremacy of the performance of AODE over Naïve Bayes for intrusion detection, both in terms of precision as well as recall values.

5. Conclusions

In this paper, we One Dependence Estimator-based intrusion detection systems for classification of network traffic. The proposed schemes intelligently classifies network traffic based on the attributes (features) of the network traffic. Naïve Bayes does not accurately detect network intrusions, as it does not take into account existing dependencies, that may exist between the features of the data set. SPODE resolves this issue by having a single feature identified as a super-parent, upon whom all other features depend. As a results, a dependancy graph is generated to establish inter-feature relationships. It was observed that such an approach improves the accuracy of the detection process significantly. An enhancement to SPODE, namely, AODE resolves the attribute independence assumption by ensuring that dependencies between various attributes in a given data set are averaged for various models. For the NSL-KDD data set, our proposed intrusion detection scheme outperforms Naïve Bayes in terms of accuracy in attack detection, lowered false alarm rates, and improved precision and recall values, as observable from the simulation results. In particular, the SPODE classified detected 97.8% of all attack traffic, whereas the AODE classifier successfully detected 99.3% of the attacks, with a false positive rate of 0.1%, as compared to a detection rate of 97.3% and a false positive rate of 1%, exhibited by the Naïve Bayes classifier. As part of our future work, we intend to extend the simulation of our proposed scheme on other data sets, for benchmarking and for validating the effectiveness of our proposed intrusion detection scheme.

6. Acknowledgements

The authors wish to acknowledge the continuing support and facilities provided by King Fahd University of Petroleum and Minerals to conduct research.

7. References

[1] R. Abdel-Aal. GMDH-based feature ranking and selection for improved classification of medical data. *Journal of Biomedical Informatics*, 38(6):456–468, 2005.

[2] N. Amor, S. Benferhat, and Z. Elouedi. Naive bayes vs decision trees in intrusion detection systems. In *Proceedings of the 2004 ACM symposium on Applied computing*, page 424. ACM, 2004.

[3] D. Barbara, N. Wu, and S. Jajodia. Detecting novel network intrusions using bayes estimators. In *First SIAM Conference on Data Mining*. Citeseer, 2001.

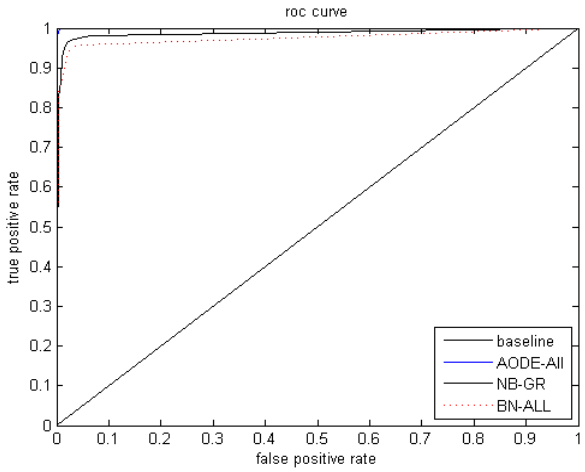

Figure 12. Comparison of the Precision-Recall curve for the best SPODE, AODE model and best NB model

[4] J. Cannady. Artificial neural networks for misuse detection. In *National information systems security conference*, pages 368–81. Citeseer, 1998.

[5] J. Cheng and R. Greiner. Learning bayesian belief network classifiers: Algorithms and system. *Advances in Artificial Intelligence*, pages 141–151.

[6] J. Cheng, R. Greiner, J. Kelly, D. Bell, and W. Liu. Learning Bayesian networks from data: an information-theory based approach. *Artificial Intelligence*, 137(1-2):43–90, 2002.

[7] L. DeLooze. Attack characterization and intrusion detection using an ensemble of self-organizing maps. In *2006 IEEE Information Assurance Workshop*, pages 108–115, 2006.

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

[9] E. El-Alfy and R. Abdel-Aal. Using GMDH-based networks for improved spam detection and email feature analysis. *Applied Soft Computing*, 2009.

[10] C. Elkan. Results of the KDD'99 classifier learning. *ACM SIGKDD Explorations Newsletter*, 1(2):64, 2000.

[11] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A Geometric Framework for Unsupervised Anomaly Detection Detecting Intrusion in Unlabeled Data. 2002.

[12] K. Faraoun and A. Boukelif. Neural networks learning improvement using the k-means clustering algorithm to detect network intrusions. *International Journal of Computational Intelligence*, 3(2):161–168, 2006.

[13] E. Gelenbe. Random neural networks with negative and positive signals and product form solution. *Neural Computation*, 1(4):502–510, 1989.

[14] E. Gelenbe. Stability of the random neural network model. *Neural Computation*, 2(2):239–247, 1990.

[15] Z. Ghahramani and M. Beal. Propagation algorithms for variational Bayesian learning. *Advances in neural information processing systems*, pages 507–513, 2001.

[16] D. Jurafsky, J. Martin, and A. Kehler. *Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition*. MIT Press, 2000.

[17] H. Kayacik, A. Zincir-Heywood, and M. Heywood. On the capability of an SOM based intrusion detection system. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, volume 3, 2003.

[18] E. Keogh and M. Pazzani. Learning augmented Bayesian classifiers: A comparison of distribution-based and classification-based approaches. In *Proceedings of the seventh international workshop on artificial intelligence and statistics*, pages 225–230. Citeseer, 1999.

[19] C. Manning and H. Schlutze. *Foundations of statistical natural language processing*. MIT Press, 2000.

[20] G. Oke and G. Loukas. A denial of service detector based on maximum likelihood detection and the random neural network. *The Computer Journal*, 50(6):717, 2007.

[21] G. Webb, J. Boughton, and Z. Wang. Not so naive bayes: Aggregating one-dependence estimators. *Machine Learning*, 58(1):5–24, 2005.

[22] Y. Yang, K. Korb, K. Ting, and G. Webb. Ensemble selection for superparent-one-dependence estimators. *AI 2005: Advances in Artificial Intelligence*, pages 102–112.

[23] Z. Zheng and G. Webb. Lazy Bayesian rules. *Deakin University Computing Technical Report TR C*, 98, 1998.