Hybrid Feature Selection for Modeling Intrusion Detection Systems

Srilatha Chebrolu, Ajith Abraham and Johnson P Thomas

Department of Computer Science, Oklahoma State University, USA
ajith.abraham@ieee.org, jpt@okstate.edu

Abstract. Most of the current Intrusion Detection Systems (IDS) examine all data features to detect intrusion or misuse patterns. Some of the features may be redundant or contribute little (if anything) to the detection process. The purpose of this study is to identify important input features in building an IDS that is computationally efficient and effective. We investigated the performance of two feature selection algorithms involving Bayesian Networks (BN) and Classification and Regression Trees (CART) and an ensemble of BN and CART. Empirical results indicate that significant input feature selection is important to design an IDS that is lightweight, efficient and effective for real world detection systems. Finally, we propose an hybrid architecture for combining different feature selection algorithms for real world intrusion detection.

1 Introduction and Related Research

Intrusion Detection Systems (IDS) have become important and widely used tools for ensuring network security. Since the amount of audit data that an IDS needs to examine is very large even for a small network, analysis is difficult even with computer assistance because extraneous features can make it harder to detect suspicious behavior patterns [7][4]. Complex relationships exist between the features, which are difficult for humans to discover. IDS must therefore reduce the amount of data to be processed. This is very important if real-time detection is desired. Reduction can occur by data filtering, data clustering and feature selection. The purpose of data filtering is to reduce the amount of data directly handled by the IDS. Some data may not be useful to the IDS and thus can be eliminated before processing. Clustering can be performed to find hidden patterns in data and significant features for use in detection. Clustering can also be used as a reduction technique by storing the characteristics of the clusters instead of the actual data. In complex classification domains, features may contain false correlations, which hinder the process of detecting intrusions. Further, some features may be redundant since the information they add is contained in other features. Extra features can increase computation time, and can have an impact on the accuracy of IDS. Feature selection improves classification by searching for the subset of features, which best classifies the training data [8].

In the literature a number of work could be cited wherein several machine learning paradigms, fuzzy inference systems and expert systems, were used to develop IDS [4][5]. Authors of [8] have demonstrated that large number of features is unimportant and may be eliminated, without significantly lowering the performance of the IDS. Very little scientific efforts are diverted to model efficient IDS feature selection. IDS task is often modeled as a classification problem in a machine-learning context.

2. Feature Selection and Classification Using AI Paradigms

2.1. Bayesian Learning and Markov Blanket Modeling of Input Features

The Bayesian Network (BN) is a powerful knowledge representation and reasoning algorithm under conditions of uncertainty. A Bayesian network B = (N, A, Θ) is a Directed Acyclic Graph (DAG) (N, A) where each node n ∈ N represents a domain variable (e.g. a dataset attribute or variable), and each arc $a_i \in$ A between nodes represents a probabilistic dependency among the variables, quantified using a conditional probability distribution (CP table) $\theta_i \in \Theta$ for each node n_i. A BN can be used to compute the conditional probability of one node, given values assigned to the other nodes. Markov Blanket (MB) of the output variable *T,* is a novel idea for significant feature selection in large data sets [9]. *MB (T)* is defined as the set of input variables such that all other variables are probabilistically independent of *T*. A general BN classifier learning is that we can get a set of features that are on the Markov blanket of the class node. The Markov blanket of a node *n* is the union of *n*'s parents, *n*'s children and the parents of *n*'s children [2]. This subset of nodes shields *n* from being affected by any node outside the blanket. When using a BN classifier on complete data, the Markov blanket of the class node forms feature selection and all features outside the Markov blanket are deleted from the BN.

2.2. Classification and Regression Trees Learning and Modeling Input Features

The Classification and Regression Trees (CART) methodology is technically called as binary recursive partitioning [1]. The process is binary because parent nodes are always split into exactly two child nodes and recursive because the process is repeated by treating each child node as a parent. The key elements of CART analysis are a set of rules for splitting each node in a tree; deciding when tree is complete and assigning a class outcome to each terminal node. As an example, for the DARPA intrusion data set with 5092 cases and 41 variables, CART considers up to 5092 times 41 splits for a total of 208772 possible splits. For splitting, Gini rule is used which essentially is a measure of how well the splitting rule separates the classes contained in the parent node. Splitting is impossible if only one case remains in a particular node or if all the cases in that node are exact copies of each other or if a node has too few cases. Instead of attempting to decide whether a given node is terminal or not, the algorithm proceeds by growing trees until it is not possible to grow them any further. Once the algorithm has generated a maximal tree, it examines smaller trees obtained by pruning away branches of the maximal tree. Feature selection is done based on the contribution the input variables made to the construction of the decision tree. Feature importance is determined by the role of each input variable either as a main splitter or

as a surrogate. Surrogate splitters are defined as back-up rules that closely mimic the action of primary splitting rules. Suppose that, in a given model, the algorithm splits data according to variable '*protocol_type*' and if a value for '*protocol_type*' is not available, the algorithm might substitute *'service'* as a good surrogate. Variable importance, for a particular variable is the sum across all nodes in the tree of the improvement scores that the predictor has when it acts as a primary or surrogate (but not competitor) splitter. Example, for node *i*, if the predictor appears as the primary splitter then its contribution towards importance could be given as $i_{importance}$. But if the variable appears as the n^{th} surrogate instead of the primary variable, then the importance becomes $i_{importance} = (p^n) * i_{improvement}$ in which *p* is the 'surrogate improvement weight' which is a user controlled parameter set between (0-1).

3. Experiment Setup and Results

The data for our experiments was prepared by the 1998 DARPA Intrusion Detection Evaluation program by MIT Lincoln Labs [6]. The data set contains 24 attack types that could be classified into four main categories namely *Denial of Service (DOS), Remote to User (R2L), User to Root (U2R)* and *Probing*. The original data contains 744 MB data with 4, 940,000 records. The data set has 41 attributes for each connection record plus one class label. Some features are derived features, which are useful in distinguishing normal connection from attacks. These features are either nominal or numeric. Some features examine only the connections in the past two seconds that have the same destination host as the current connection, and calculate statistics related to protocol behavior, service, etc. These are called same host features. Some features examine only the connections in the past two seconds that have the same service as the current connection and are called same service features. Some other connection records were also sorted by destination host, and features were constructed using a window of 100 connections to the same host instead of a time window. These are called host-based traffic features. R2L and U2R attacks don't have any sequential patterns like DOS and Probe because the former attacks have the attacks embedded in the data packets whereas the later attacks have many connections in a short amount of time. So some features that look for suspicious behavior in the data packets like number of failed logins are constructed and these are called content features. Our experiments have three phases namely data reduction, training phase and testing phase. In the data reduction phase, important variables for real-time intrusion detection are selected by feature selection. In the training phase, the Bayesian neural network and classification and regression trees constructs a model using the training data to give maximum generalization accuracy on the unseen data. The test data is then passed through the saved trained model to detect intrusions in the testing phase. The data set for our experiments contains randomly generated 11982 records having 41 features [3]. The 41 features are labeled in order as *A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, AA, AB, AC, AD, AF, AG, AH, AI, AJ, AK, AL, AM, AN, AO* and the class label is named as *AP*. This data set has five different classes namely *Normal, DOS, R2L, U2R* and *Probes.* The training and test comprises of 5092 and 6890 records respectively. All the IDS models are trained and tested with the same set of data. As the data set has five different classes

we perform a 5-class binary classification. The *Normal* data belongs to class 1, *Probe* belongs to class 2, *DOS* belongs to class 3, *U2R* belongs to class 4 and *R2L* belongs to class 5. All experiments were performed using an AMD Athlon 1.67 GHz processor with 992 MB of RAM.

3.1. Modeling IDS Using Bayesian Network

We selected the important features using the Markov blanket model and found out that 17 variables of the data set forms the Markov blanket of the class node as explained in Section 2.1. These 17 variables are *A, B, C, E, G, H, K, L, N, Q, V, W, X, Y, Z, AD* and *AF*. Further Bayesian network classifier is constructed using the training data and then the classifier is used on the test data set to classify the data as an attack or normal. Table 1 depicts the performance of Bayesian belief network by using the original 41 variable data set and the 17 variables reduced data set. The training and testing times for each classifier are decreased when 17 variable data set is used. Using the 17 variable data set there is a slight increase in the performance accuracy for *Normal* class compared to the 41 variable data set.

Table 1. Performance of Bayesian Belief Network

Attack Class	**41 variables**			**17 variables**		
	Train (sec)	Test (sec)	Accuracy (%)	Train (sec)	Test (sec)	Accuracy (%)
Normal	42.14	19.02	99.57	23.29	11.16	**99.64**
Probe	49.15	21.04	**99.43**	25.07	13.04	98.57
DOS	54.52	23.02	**99.69**	28.49	14.14	98.16
U2R	30.02	15.23	**64.00**	14.13	7.49	60.00
R2L	47.28	12.11	**99.11**	21.13	13.57	98.93

3.2. Modeling IDS Using Classification and Regression Trees

We decided the important variables depending on the contribution of the variables for the construction of the decision tree. Variable rankings were generated in terms of percentages. We eliminated the variables that have 0.00% rankings and considered only the primary splitters or surrogates as explained in Section 2.2. This resulted in a reduced 12 variable data set with *C, E, F, L, W, X, Y, AB, AE, AF, AG* and *AI* as variables. Further the classifier is constructed using the training data and then the test data is passed through the saved trained model. Table 2 compares the performance of CART using the 41 variable original data set and the 12 variable reduced data set. Normal class is classified 100 percent correctly. Furthermore, the accuracies of classes U2R and R2L have increased by using the 12 variable reduced data set. It is also found that CART could classify accurately on smaller data sets. Further, we used the Bayesian reduced 17 variable data set (Section 3.1) to train CART and the CART reduced 12 variable dataset (Section 3.2) to train Bayesian network. As illustrated in Table 3 except R2L all other classes were classified well by the CART algorithm. Moreover, training and testing time for each class are greater for Bayesian network classifier compared to CART algorithm.

Table 2. Performance of classification and regression trees

Attack Class	41 variable data set			12 variable data set		
	Train (sec)	Test (sec)	Accuracy (%)	Train (sec)	Test (sec)	Accuracy (%)
Normal	1.15	0.18	99.64	0.80	0.02	**100.00**
Probe	1.25	0.03	**97.85**	0.85	0.05	97.71
DOS	2.32	0.05	**99.47**	0.97	0.07	85.34
U2R	1.10	0.02	48.00	0.45	0.03	**64.00**
R2L	1.56	0.03	90.58	0.79	0.02	**95.56**

Table 3. Performance of Bayesian and CART using reduced datasets

Attack Class	Bayesian with 12 variables			CART with 17 variables		
	Train (sec)	Test (sec)	Accuracy (%)	Train (sec)	Test (sec)	Accuracy (%)
Normal	20.10	10.13	98.78	1.03	0.04	**99.64**
Probe	23.15	11.17	99.57	1.15	0.13	**100.00**
DOS	25.19	12.10	98.95	0.96	0.11	**99.97**
U2R	11.03	5.01	48.00	0.59	0.02	**72.00**
R2L	19.05	12.13	**98.93**	0.93	0.10	96.62

Table 4. Performance of CART and Bayesian network using 19 variables

Class	Bayesian	CART
Normal	**99.57**	95.50
Probe	96.71	**96.85**
DOS	**99.02**	94.31
U2R	56.00	**84.00**
R2L	**97.87**	97.69

3.3. Feature Ranking Using Support Vector Machines

We also attempted to evaluate the performance of CART and Bayesian network using the reduced dataset (same input variables) given in [8]. Table 4 shows the performance comparisons of CART and Bayesian network using 19 variables. Except *U2R* the 17 and 12 variable dataset performs well for all the other classes.

3.4 Ensemble Approach Using Reduced Data Sets

In this approach we first construct the Bayesian network classifier and CART models individually to obtain a very good generalization performance. The ensemble approach is used for 12, 17 and 41 variable dataset and is illustrated in Figure 1. In the ensemble approach, the final outputs were decided as follows: Each classifier's output is given a weight (0-1 scale) depending on the generalization accuracy as given in

Section 3.1-3.2. If both classifiers agree then the output is decided accordingly. If there is a conflict then the decision given by the classifier with the highest weight is taken into account. Table 5 illustrates the ensemble results using the different data sets. From the results, we can conclude that ensemble approach gives better performance than the two individual separately used models. The ensemble approach basically exploits the differences in misclassification (by individual models) and improves the overall performance. Figure 2 illustrates the developed hybrid IDS model after summarizing all the empirical results. By using the hybrid model *Normal, Probe* and *DOS* could be detected with 100% accuracy and *U2R* and *R2L* with 84% and 99.47% accuracies respectively.

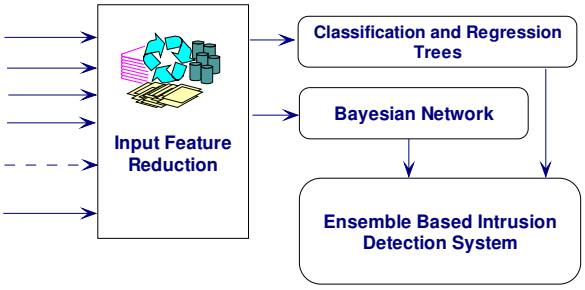

Fig. 1. Ensemble approach for IDS

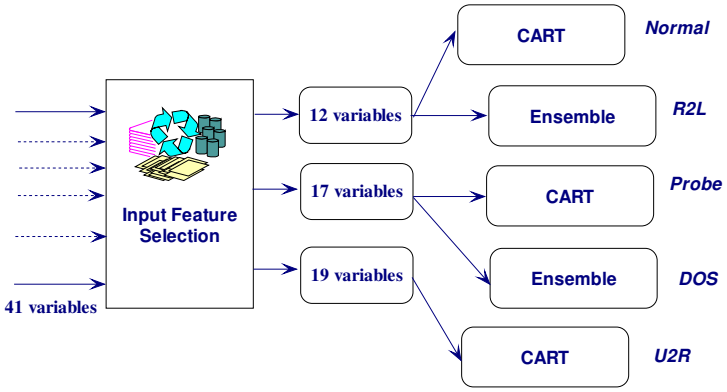

Fig. 2. Developed IDS model for different attack classes

Table 5. Performance of ensemble approach using different data sets

Class	12 variables	17 variables	41 variables
Normal	**100.00**	99.64	99.71
Probe	99.86	**100.00**	99.85
DOS	99.98	**100.00**	99.93
U2R	**80.00**	72.00	72.00
R2L	**99.47**	99.29	99.47

Conclusions

In this research we have investigated new techniques for intrusion detection and performed data reduction and evaluated their performance on the benchmark intrusion data. Our initial experiments using PCA/ICA to compress data was not successful (due to space limitations the results are not reported in this paper). We used the feature selection method using Markov blanket model and decision tree analysis. Following this, we explored general Bayesian Network (BN) classifier and Classification and Regression Trees (CART) as intrusion detection models. We have also demonstrated performance comparisons using different reduced data sets. The proposed ensemble of BN and CART combines the complementary features of the base classifiers. Finally, we propose a hybrid architecture involving ensemble and base classifiers for intrusion detection. From the empirical results, it is evident by using the hybrid model *Normal, Probe* and *DOS* could be detected with 100% accuracy and *U2R* and *R2L* with 84% and 99.47% accuracies respectively. Our future research will be directed towards developing more accurate base classifiers particularly for the detection of *U2R* type of attacks.

References

[1] Brieman L., Friedman J., Olshen R. and Stone C., Classification of Regression Trees. Wadsworth Inc., 1984.

[2] Cheng J., Greiner R., Kelly J., Bell D.A. and Liu W., Learning Bayesian Networks from Data: an Information-Theory Based Approach, The Artificial Intelligence Journal, Volume 137, Pages 43-90, 2002.

[3] KDD cup 99 Intrusion detection data set <http://kdd.ics.uci.edu/databases/kddcup99/kddcup.data_10_percent.gz>

[4] Lee W., Stolfo S. and Mok K., A Data Mining Framework for Building Intrusion Detection Models, In Proceedings of the IEEE Symposium on Security and Privacy, 1999.

[5] Luo J. and Bridges S. M., Mining Fuzzy Association Rules and Fuzzy Frequency Episodes for Intrusion Detection, International Journal of Intelligent Systems, John Wiley & Sons, Vol. 15, No. 8, pp. 687-704, 2000.

[6] MIT Lincoln Laboratory. <http://www.ll.mit.edu/IST/ideval/>

[7] Mukkamala S., Sung A.H. and Abraham A., Intrusion Detection Using Ensemble of Soft Computing Paradigms, Third International Conference on Intelligent Systems Design and Applications, Springer Verlag Germany, pp. 239-248, 2003.

[8] Sung A.H. and Mukkamala S., Identifying Important Features for Intrusion Detection Using Support Vector Machines and Neural Networks, Proceedings of International Symposium on Applications and the Internet (SAINT 2003), pp. 209-217, 2003.

[9] Tsamardinos I., Aliferis C.F.and Statnikov A., Time and Sample Efficient Discovery of Markov Blankets and Direct Causal Relations, 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, USA, ACM Press, pages 673-678, 2003.