World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:6, 2008

Improved C-Fuzzy Decision Tree for Intrusion Detection

Krishnamoorthi Makkithaya, N. V. Subba Reddy, and U. Dinesh Acharya

Abstract**—**As the number of networked computers grows, intrusion detection is an essential component in keeping networks secure. Various approaches for intrusion detection are currently being in use with each one has its own merits and demerits. This paper presents our work to test and improve the performance of a new class of decision tree c-fuzzy decision tree to detect intrusion. The work also includes identifying best candidate feature sub set to build the efficient c-fuzzy decision tree based Intrusion Detection System (IDS). We investigated the usefulness of c-fuzzy decision tree for developing IDS with a data partition based on horizontal fragmentation. Empirical results indicate the usefulness of our approach in developing the efficient IDS.

Keywords**—**Data mining, Decision tree, Feature selection, Fuzzy c- means clustering, Intrusion detection.

I. Introduction

THE wide spread use of computer networks in today's society, especially the sudden surge in importance of e-commerce to the world economy, has made computer network security an international priority. Since it is not technically feasible to build a system with no vulnerabilities, intrusion detection has become an important area of research. Intrusion detection includes identifying a set of malicious actions that compromise the integrity, confidentiality, and availability of information resources. Traditional methods for intrusion detection are based on extensive knowledge of signatures of known attacks. Monitored events are matched against the signatures to detect intrusions. These methods extract features from various audit streams, and detect intrusions by comparing the feature values to a set of attack signatures provided by human experts. The signature database has to be manually revised for each new type of intrusion that is discovered. A significant limitation of signature-based methods is that they cannot detect emerging cyber threats, since by their very nature these threats are launched using previously unknown attacks. In addition, even if a new attack is discovered and its signature developed, often there is a substantial latency in its deployment across networks. These limitations of signature-based methods have led to an increasing interest in data mining based approaches, to build the detection models for IDS [5],[9],[11],[12].

Developing effective methods for the detection of intrusions and misuses therefore is essential for assuring system security. Despite the promise of better detection performance and generalization ability of data mining-based IDSs, there are some inherent difficulties in the implementation and deployment of these systems. We can group these difficulties into three general categories: accuracy, efficiency, and usability. An effective data mining-based IDS must address each of these three groups. Although there are tradeoffs between these groups, each can generally be handled separately.

Various approaches to intrusion detection are currently being in use with each one has its own merits and demerits. New means and ways that will minimize these shortcomings must, therefore, continuously be researched and defined [8].The objective of this study is to test and improve the performance of a new class of decision tree as explained in [10] based IDS. The C- fuzzy decision trees are classification constructs that are built on a basis of information granules fuzzy clusters. The way in which these trees are constructed deals with successive refinements of the clusters (granules) forming the nodes of the tree. When growing the tree, the nodes (clusters) are split into granules of lower diversity (higher homogeneity). In contrast to C4.5-like trees, all features are used once at a time, and such a development approach promotes more compact trees and a versatile geometry of the partition of the feature space.

The performance, robustness, and usefulness of classification algorithms are improved when relatively few features are involved in the classification. Thus, selecting relevant features for the construction of classifiers has received a great deal of attention. Several approaches to feature selection have been explored [14]. The purpose of this work also includes to identify best candidate feature sub set in building the c-fuzzy decision tree IDS that is computationally efficient and effective. We investigated the usefulness of c-fuzzy decision tree for developing IDS with a data partition based on horizontal fragmentation.

The rest of this paper is organized as follows: Section II gives an overview of *C*-fuzzy decision trees. The fragmentation and feature selection techniques are described

Krishnamoorthi Makkithaya, is with the Computer Science and Engineering Department, Manipal Institute of Technology, Manipal; Manipal University, India (phone: 0820-2924517; e-mail: k.moorthi@manipal.edu).

N. V. Subba Reddy., is with the Computer Science and Engineering Department, Manipal Institute of Technology, Manipal; Manipal University, India (e-mail: dr_nvsreddy@rediff.mail.com).

U. Dinesh Acharya. is with the the Computer Science and Engineering Department, Manipal Institute of Technology, Manipal; Manipal University, India (e-mail: dinesh.acharya@manipal.edu).

in Section III. Experimental results and comparisons are presented in Section IV. Finally, conclusion is given in Section V.

II. Overall Architecture of the Cluster based Decision Tree

The architecture of the cluster–based decision tree develops around fuzzy clusters that are treated as generic building blocks of the tree. The training data set X is clustered into C clusters so that the data points (patterns) that are similar are put together. These clusters are completely characterized by their prototypes (centroids). Tree growing start with them positioned at C top nodes of the tree structure. The way of building the clusters implies a specific way in which elements of X allocated to each cluster. In other words, each cluster comes with a subset of X, namely X1, X2 … Xc. The process of growing tree is guided by a certain heterogeneity criterion that quantifies the diversity of the data (with respect to the output variable y) falling under the given cluster (node) denoted by V!, V2,… ,Vc respectively (Fig. 1) The node with the highest value of the criterion is chosen and treated it as a candidate for further refinement. The process is repeated by selecting the most heterogeneous node out of all final nodes. The growth of the tree is carried out by expanding the nodes and building their consecutive levels that capture more details of the structure. It is noticeable that the node expansion leads to the increase in either the depth or width (breadth) of the tree. The pattern of the growth is very much implied by the characteristics of the data as well as influenced by the number of the clusters.

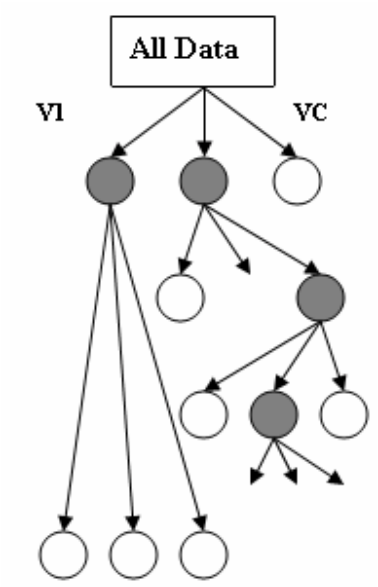

Fig. 1 Growing a decision tree by expanding nodes (which are viewed as clusters located at its nodes). Shadowed nodes are those with maximal diversity criterion and thus being subject to the split operation

The architecture of C-fuzzy decision tree for intrusion detection is similar to that in [10]. Once the tree has been constructed, we can use the tree to classify an unknown pattern or predict the corresponding output of the pattern. We traverse the tree by starting from the root node and then selecting a path according to the similarity calculations to move down from the current level to the next level until a leaf node is reached. Finally, the local linear model of the reached leaf node is used to calculate the output of the unknown pattern.

Fuzzy clustering is a core functional part of the overall tree. It builds the clusters and provides its full description. We confine ourselves to the standard fuzzy C-means (FCM), which is an omnipresent technique of information granulation. The description of this algorithm is well documented in the literature [13]. The FCM algorithm is an example of an objective-oriented fuzzy clustering where the clusters are built through a minimization of some objective function.

A. Heterogeneity Calculation

The purpose of the heterogeneity calculation for each leaf node in the current tree is to find out the node with highest diversity of the included patterns for splitting. Therefore, we use the same heterogeneity measure as that in [10]. The variability of the data in the output space at a node (Vi) is taken as a spread around the representation of the same node. In the next step the node with highest value of V say Vjmax selected for tree expansion.

III. Data Partitioning and Feature Selection

One of the key problems that arise in a great variety of fields, including pattern recognition and machine learning, is the so-called feature selection. Feature selection not only obtains better accuracy of the predictor but also reduces training and inference time. Thus, selecting relevant features for the construction of classifiers has received a great deal of attention. As computational cost depends upon the sample size, number of features, it is required some approaches to reduce sample size and number of features to make these IDS models work efficiently [11]. Domain knowledge can be systematically incorporated into the sample size reduction [12].

The accuracy by which the c-fuzzy decision tree classifies depends heavily on the performance of the FCM. The aim of FCM here is to cluster the network traffic data to find useful pattern to detect intrusion. Sometimes this may be considered as partitioning or division of data into groups using simple techniques to achieve the goal of clustering. Data partitioning in the preprocessing phase group the most similar data together so that it may help to understand the structure of data and relative importance of each attribute for the selection or classification process. In this section we explain our technique borrowed from distributed database system to partition the network traffic data for intrusion detection.. We are proposing a simple domain knowledge based fragmentation for data partition.

We analyzed network traffic data and observed following details which are very useful for data partition. A fundamental understanding of TCP/IP protocol is essential for using network based intrusion detection. Attackers often take the advantage of obscure aspects of specific protocols to execute their attacks. Decoding at the protocol level allows signatures

to be created that stipulate allowable and unallowable aspects of specific protocols. This type of signature allows for much broader rules, while still keeping a low false alert threshold. Attackers do not fully adhere to the protocol rules. TCP (Transmission Control Protocol) is the primary transport control protocol used on the Internet. It is basically a connection oriented service where as UDP (User datagram protocol) and ICMP (internet control message protocol) are connectionless service. UDP communications can be spoofed much easier than can TCP communications because UDP is a connectionless protocol and does not use sequence numbers. While UDP attacks are less common, they can also be more difficult to detect and decipher. UDP communications are session less. There are no flags to use as tell-tale indicators of malicious activity as with TCP communications. Unfortunately several severe attacks occur via UDP. This means that each transport layer protocol requires separate strategy and feature set to detect intrusion using network traffic data. Therefore, we believe fragmentation of the network traffic data based on different transport layer protocol will result in a better data set, resulting in effective classifiers.

A. Fragmentation

The decomposition of global relation into fragments can be performed by applying two different types of fragmentation: horizontal fragmentation and vertical fragmentation. In all types of fragmentation, a fragment can be defined by an expression in a relational language, which takes global relation as operand and produces fragments as result. There are, however, some rules which must be followed when defining fragments. They are completeness, reconstruction condition, and disjoint condition.

Horizontal fragmentation consists of partitioning the tuples of a global relation into subsets, where each subset can contain data which have common properties. It can be defined by expressing each fragment as a selection operation on the global relation.

Horizontal fragments are defined using selection operation on global relation .Our proposed horizontal fragmentation is as follows:

We assumed the**Training-data** as the global relation and transport layer protocol feature is selected for horizontal fragmentation.

The horizontal fragmentation is defined as follows:

Training-data1= **SL** protocol="Tcp"
Training-data

Training-data2= **SL** protocol = "Udp"
Training-data

Training-data3= **SL** protocol = "Icmp"
Training-data

where **SL** is the selection operation on the relation **Training-data**.

The above fragmentation satisfies the completeness condition as "Tcp", "Udp", "Icmp" are the only possible values of the protocol attribute in the global relation Training-data. The reconstruction condition is easily verified, because it is always possible to reconstruct the Training-data through following operation:

Training-data = Training-data1 **UN** Training-data2 **UN** Training-data3.

where **UN** is the union operation. The disjoint condition is clearly verified.

Our proposed fragmentation makes data set for modified c-fuzzy decision tee and modified tree is as shown in the Fig. 2. This divides data set into three processes, where same program run on each subset to generate c-fuzzy decision tree. Each process results in independent decision tree and these trees can be used separately for training and testing.

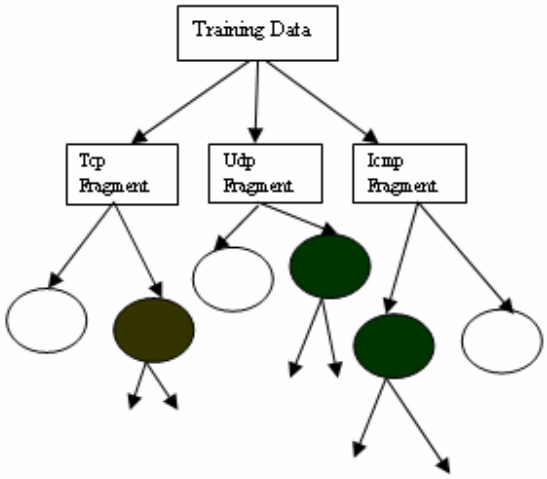

Fig. 2 Horizontal fragmentation based modified c-fuzzy decision tree

B. Feature Selection

Technology advances have made data collection easier and faster, resulting in larger, more complex datasets with many objects and dimensions. As the datasets become larger and more varied, adaptations to existing algorithms are required to maintain cluster quality and speed. Traditional clustering algorithms consider all of the dimensions of an input dataset in an attempt to learn as much as possible about each object described. In high dimensional data, however, many of the dimensions are often irrelevant. These irrelevant dimensions can confuse clustering algorithms by hiding clusters in noisy data. In very high dimensions it is common for all of the objects in a dataset to be nearly equidistant from each other, completely masking the clusters [14]. Feature selection methods have been employed somewhat successfully to improve cluster quality. These algorithms find a subset of dimensions on which to perform clustering by removing irrelevant and redundant dimensions.

The main aim of the feature selection is to reduce the set of all features {W} to a smaller feature subset size m by selecting features ranked according to some goodness criteria. The selected m features then form a new feature vector and a corresponding reduced subset W′, where W'⊂W and {W'}<<{W}. A feature's discriminatory power is a useful

gauge of its goodness and is commonly ascertained using information gain (IG) score.

$$IG(X,Y) = \sum_{X=x} \sum_{Y=y} P(X=x, Y=y) [\log_2 P(X=x, Y=y) / P(X=x) P(Y=y)] \quad (1)$$

Here probabilities are estimated from S Training-data set. The information gain based ranking and selection of the features is the base line algorithm used in this paper and we will refer to it as BASE Fig. 3.

m = feature subset size
BASE
For each $w_i \in W$
Calculate IG score using S
Sort W in decreasing order of IG scores
$W' = \{w_1, w_2, \ldots w_m\}$
Return W'

Fig. 3 Feature selection with IG based ranking

After using Base for ordering the features in the first step we intended to find a small set of candidate features in the second step.. We used a simple sequential forward selection scheme as search strategy and used the performance of c-fuzzy decision tree as stopping criteria to select best candidate feature sub set. This performance indicator of c-fuzzy decision tree quantified by estimating diversity criterion as mentioned in[10]. The essence of the diversity criterion is to quantify the distribution of the data allocated to the given cluster so that higher dispersion of data results in higher values of the criterion [10].

The algorithm for candidate feature subset selection is as shown below.

1. Use Base to get ordered feature set
2. Select first two features from the ordered set and compute diversity criterion for c-fuzzy decision tree. V1
3. Add the next feature , construct the tree and calculate the diversity criterion. V2
4. If V1-V2 < E, or V2 starts increasing Stop, otherwise Repeat steps 3 & 4.
   Where E is selected by the user.

IV. Experimental Result

A. Data Set

The KDD Cup 1999 Intrusion detection contest data [6] (KDD cup 99 Intrusion detection data set) is used in our experiments. This data was prepared by the 1998 DARPA Intrusion Detection Evaluation program by MIT Lincoln Laboratory. Lincoln labs acquired nine weeks of raw TCP dump data. The raw data was processed into connection records, which consist of about 5 million connection records. The data set contains 24 attack types. These attacks fall into four main categories: DOS, Probe, u2r, and r2l. The data set has 41 attributes for each connection record plus one class label. The data set for our experiments contained 37015 records which were randomly generated from the MIT data set. Random generation of data include the number of data from each class proportional to its size. This data set is again divided into training data with 29612 records and testing data with 7403 records. The data is partitioned into the two classes of ''Normal'' and ''Attack'' patterns where Attack is the collection of four classes (Probe, DOS, U2R, and R2L) of attacks. The objective is to separate normal and attack pattern to make classifier into two class classifier [5]. Only the 34 numeric features are used in our experiments. We conducted a five fold experiment with random sub sampling to derive a classifier with training data and then to estimate accuracy of the classifier with testing data.

B. Result

In order to test the performance of our feature selection technique, we conducted experiment with training and testing data set. Base algorithm is used to get an ordered feature set based on information gain which is evaluated using Weka attribute selection InfoGain Attribute Eval filter available in Weka [3]. After ranking the features, subset selection algorithm is used to select best candidate feature subset. A fixed size (8 depth wise) c- fuzzy decision tree was constructed with feature subset as mentioned in the subset selection algorithm and the result is as shown in the Table 1. It is very clear from the Table1 that the diversity criterion decreases with addition of each feature initially but slows down or increases later. This is mainly due to performance degradation of FCM in high dimension, as data becomes very sparse and distance measures become increasingly meaningless. We experimentally selected (Table I) a subset with size equal to 10 and used the same for our further experiments.

TABLE I
COMBINED DIVERSITY CRITERION FOR DIFFERENT FEATURE SUBSET

No of features in the subset	Combined diversity criterion
2	319.92
3	298.94
4	280.95
5	255.63
6	247.40
7	233.64
8	225.53
9	206.55
10	**196.38**
11	211.38
12	222.23

We also conducted experiments with training and testing data to evaluate the performance of c-fuzzy decision tree with and without fragmentation. Sensitivity, specificity, and accuracy are calculated separately for both training ant testing data as shown below.

$$\text{sensitivity} = \frac{t\_pos}{pos} \quad (2)$$

$$\text{specificity} = \frac{t\_neg}{neg} \quad (3)$$

$$\text{accuracy} = \text{sensitivity}\ \frac{pos}{(pos+neg)} + \text{specificity}\ \frac{neg}{(pos+neg)} \quad (4)$$

Where t_pos is the number of true positives, t_neg is the number of true negatives, pos is the number of positives, and neg is the number of negative samples. As per the results shown in the Table II, and Table III it is very clear that the performance our fragmentation based c-fuzzy decision tree is better compared to original tree for intrusion detection. Specificity of ICMP fragmentation is very less, this may be due to very less number of normal connections in this fragment but overall accuracy is still improved because of better sensitivity.

TABLE II
PERFORMANCE OF C-FUZZY DECISION TREE WITH AND WITHOUT FRAGMENTATION (TRAINING DATA)

	C-Fuzzy Decision tree (size=8)	Tcp C-Fuzzy Decision tree (size=8)	Udp C-Fuzzy Decision tree (size=6)	Icmp C-Fuzzy Decision tree (size=6)
Sensitivity	.8355	.9336	.9215	.9890
Specificity	.9552	.9663	.9959	.3124
Accuracy	.8927	.9446	.9578	.9813

TABLE III
PERFORMANCE OF C-FUZZY DECISION TREE WITH AND WITHOUT FRAGMENTATION (TESTING DATA)

	C-Fuzzy Decision tree (size=8)	Tcp C-Fuzzy Decision tree (size=8)	Udp C-Fuzzy Decision tree (size=6)	Icmp C-Fuzzy Decision tree (size=6)
Sensitivity	.8422	.9256	.9315	.9745
Specificity	.9352	.9533	.9839	.3124
Accuracy	.8897	.9396	.9597	.9723

V. CONCLUSION

In this paper we presented a fragmentation based c-fuzzy decision tree model for Intrusion detection systems with a focus on improving the performance by reducing the number features and selecting more appropriate data set. For selection of compact feature sub set from the ordered feature set, we have used the performance of the c-fuzzy decision tree. It is evident from the results, our data partition and feature selection technique results in improved c-fuzzy decision tree to build an effective IDS. As a future work this could be easily extended as a multi class classifier to determine the different attack types. We used only information gain to order the feature set to demonstrate the effectiveness of our feature subset selection.. In future we would like to consider other feature ranking techniques to select best feature candidate subset.

REFERENCES

[1] Dana Elena Ilea, Paul F. Whelan, Ovidiu Ghita Vision Systems Group, School of Electronic Engineering "Characterization of Clustering Algorithms for Colour Image Segmentation", www.vsg.dcu.ie/papers/optim_2006_dana.pdf
[2] Dian-Rong Yang, Leu-Shing Lan* , and Shih-Hung Liao Department of Electronics Engineering National Yunlin University of Science and Technology, "A New Fuzzy Clustering Method with Controllable Membership Characteristics", 2006 IEEE International Conference on Fuzzy Systems , Vancouver, BC, Canada July 16-21, 2006.
[3] I.H. Witten and E.Frank. Data Mining- Practical Machine Learning Tools and Techniques with Java Implementation. Morgan Kaufman San Francisco 2000.
[4] Hsin-Wei Chiu, Chen-Sen Ouyang and Shie-Jue Leee Member IEEE, Abdul Manon Ahmed, "Improved C-Fuzzy decision Trees", 2006 IEEE International Conference on Fuzzy Systems , Vancouver, BC, Canada July 16-21, 2006.
[5] Jonatan Gomez and Dipankar Dasgupta, "Evolving Fuzzy classifiers for intrusion detection.", Proceedings of the 2002 IEEE workshop on Information Assurance, NY June 2001.
[6] KDD-Data, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[7] Krishnamoorthi, N.V. Subba Reddy, U.Dinesh Acharya, *CSE Department MIT Manipal* A Two Stage Hybrid Model for Intrusion Detection. 14th International Conference on Advanced Computing and Communication ADCOM 2006. NITK Surathkal
[8] Norbik Bashah, Shanmugam Bnaranidharan, Abdul Manon Ahmed, "Hybrid Intelligent Intrusion detection system", Trasactions on Engineering, Computing and Technology V6 June 2005.
[9] Ramesh Agarwal, Mahesh V. Joshi, " Pnrule: A new framework for learning classifier models in data mining, A case study in network intrusion detection",. Technical report RC 21719, IBM research report, Computer Science/ Mathematics, April 2000.
[10] Witold Pedrycz, Fellow, IEEE, and Zenon A. Sosnowski, Member, IEEE, "C–Fuzzy Decision Trees", IEEE Transactions on Systems, Man, and Cybernetics —Part C: Applications and Reviews, VOL. 35, NO. 4, November 2005.
[11] Wenke Lee and Salvatore J. Stolfo, Philip K. Chan. Real Time Data Mining- based Intrusion Detection. http://www.ncsu.edu/faculty/lee/project/id.html/
[12] Wenke Lee and Salvatore J. Stolfo, Combining Knowledge Discovery and Knowledge Engineering to build IDSs. http://www.ncsu.edu/faculty/lee/project/id.html
[13] J. C. Bezdek, Pattern Recognition with Fuzzy Objective Functions, New York: Plenum, 1981.
[14] Huan Liu, Lei Yu , " Toward Integrating Feature selection Algorithms for Classification and Clustering", IEEE Transaction on Knowledge and data Engineering VOl 17 , No 4, April 2005.