DOI 10.1007/s10115-005-0195-8
Springer-Verlag London Ltd. © 2005
Knowledge and Information Systems (2005) 8: 498–509

Knowledge and Information Systems

Short Paper

A supervised clustering algorithm for computer intrusion detection

Xiangyang Li[1], Nong Ye[2]

[1]Department of Industrial and Manufacturing Systems Engineering, University of Michigan—Dearborn, Dearborn, MI, USA
[2]Department of Industrial Engineering, Arizona State University, Tempe, AZ, USA

Abstract. We previously developed a clustering and classification algorithm—supervised (CCAS) to learn patterns of normal and intrusive activities and to classify observed system activities. Here we further enhance the robustness of CCAS to the presentation order of training data and the noises in training data. This robust CCAS adds data redistribution, a supervised hierarchical grouping of clusters and removal of outliers as the postprocessing steps.

Keywords: Classification; Clustering; Intrusion detection

1. Introduction

Signature recognition learns signature patterns of intrusive (and normal) activities from training data, and then in detection, matches these signatures with the observed incoming data. Signature-recognition algorithms face great challenges in computer-intrusion detection. First, activity data from a computer system can easily contain millions of records per day. In addition, each record may have hundreds of data fields. Thus, an algorithm to learn signature patterns in such data must be scalable. Second, patterns of normal and intrusive activities very likely change over time, and new forms of attacks emerge everyday. Hence, a data-mining algorithm must have the incremental learning ability to update signature patterns as more training data become available. Last, the distribution for normal and intrusive data may be unclear.

Data-mining techniques, such as decision trees, association rules, artificial neural networks and Bayesian networks, have been used as signature-recognition algorithms for intrusion detection (Axelsson 2000). However, in many cases, they are not capable

Received 4 March 2004
Revised 5 November 2004
Accepted 20 November 2004
Published online 21 April 2005

of learning signature patterns in a scalable, incremental manner. Moreover, several of them, including Bayesian networks, require the understanding of domain knowledge or data distribution.

Addressing the above challenge, we have developed an innovative algorithm, called clustering and classification algorithm—supervised (CCAS) (Li and Ye 2002). CCAS is based on supervised clustering for learning patterns of normal and intrusive activities and instance-based learning to classify observed activities. Clustering relies very little on the distribution of data, suitable for intrusion detection. Like other incremental data-mining algorithms, CCAS shows sensitivity to the presentation order of training data. Recently, grid-based and density-based methods have been used to overcome this problem (Ester et al. 1998; Harsha and Choudhary 1999; Zhang 1997). Built on concepts from these clustering methods, together with several innovative concepts, we develop a robust extension of CCAS in this paper.

2. Original CCAS algorithm

Each data record has attribute vector X in p dimensions and a target class Y. Here we consider only numeric attributes. A cluster L is represented by the centroid coordinates XL of all the data points in it, the number of data points NL, and its class YL. A weighted Euclidean distance is used to calculate the distance from a data point D to a cluster L,

$$d(D, L) = \sqrt{\sum_{i=1}^{p} (X_i - XL_i)^2 r_{iY}^2},$$

where X_i and XL_i are, respectively, the coordinates of D's and L's centroid on the ith dimension and r_{iY} is the correlation coefficient between predictor X_i and target variable Y. The distance between two clusters is calculated similarly by replacing the data point's coordinates with the cluster centroid's coordinates.

The core of CCAS is a grid-based incremental supervised clustering. Each dimension is divided into a set of intervals within the range defined by the minimum and maximum values of data points, separating the space into cubic cells. Grid configuration could use different numbers of unequal intervals in different dimensions. We simplify our study by using the same number of equal grid intervals for all dimensions. At this research stage, we pick the best setting of this parameter from experimentation and expertise. Using a heuristic, the original CCAS clusters the training data points one by one based on the distance as well as the target class information.

The cluster structure represents the patterns of normal and intrusive activities. We classify a new data point by comparing new data points with these clusters. For the binary target variable, we assign a continuous value falling in [0, 1], describing the closeness to the two target classes. We calculate the distance-weighted average of the target values of the k nearest clusters as the target value Y of a new data point D,

$$W_j = 1/d^2(D, L^j)$$
$$Y = \sum_{j=1}^{k} YL^j W^j \Big/ \sum_{j=1}^{k} W^j,$$

```
For each data point D in training data
    Find the grid cell of D
    For each existing cluster L_i
        If D and L_i are in the same grid cell
            determine whether L_i is the nearest cluster to D
    End
    If there is a nearest cluster L_n and its class label is same as D
        incorporate D into L_n
        update the centroid coordinates
        update the number of points in L_n
    Else create a new cluster containing only D
        the label of this cluster is same as the class of D
End
```

Fig. 1. The supervised clustering procedure

	Incremental learning	Scalability	Robustness
Grid-based clustering	✓	✓	✓
Redistribution			✓
Supervised grouping		✓	✓
Outlier removal			✓

Fig. 2. Problems addressed by the steps in the robust CCAS

where W_j is the weight for the jth nearest cluster. The target value of this cluster is YL_j.

3. Robust CCAS

3.1. Postprocessing

At any instant during the supervised clustering, the cluster structure considers only the data points processed so far, reflecting a local view on training data. Presented with the data points in a different order, normal and intrusive clusters could grow differently if data points nearest in space do not come successively. Natural clusters of the same class may merge into larger clusters due to such unusual input order. Dividing the data space into grid cells and allowing the formation of clusters only within grid cells help alleviate this problem. We further strengthen CCAS with several postprocessing steps, listed with the problems they address in Fig. 2.

(1) Redistribution of data points is a common way to remedy the localisation in incremental clustering, as used in Zhang (1997). All training data points are clustered again using the produced clusters as seeds. When a seed cluster with the same class label is found to be the nearest to the incoming data point, this cluster is replaced

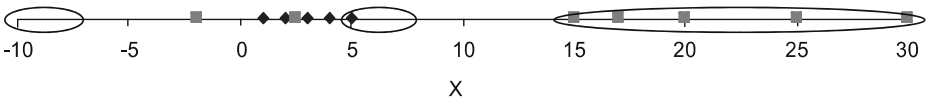

Fig. 3. An illustration of the supervised hierarchical grouping algorithm in one dimension

with a new cluster, of which this data point is the centroid. We consider that the initial cluster structure is not the reliable reflection of data distribution and it functions as another facility to limit the growth of clusters here. We allow new clusters to emerge and thus allow bigger adjustment to the cluster structure.

This redistribution process can be repeated many times. Classification performance improves with repetition but at additional computation cost. The cluster structure becomes stable after several passes. Our experiments show that usually one round of redistribution is sufficient.

(2) In the grid-based clustering, a natural cluster may correspond to several produced clusters falling into neighbouring grid cells. Hence, we employ a hierarchical grouping procedure to regroup these clusters. This algorithm is different from the traditional hierarchical clustering in that it combines a pair of clusters into one larger cluster only when they not only are closest to each other but also have the same class. A single linkage method (Jain and Dubes 1988) is used in determining the distance between larger clusters. The distance between two clusters is defined as the shortest distance between any two points belonging to the respective clusters.

(3) Clusters that have few data points may represent noises in data samples and can be removed. The threshold on the minimum number of data points could be based on the average number of data points in clusters and be different for different classes. However, this threshold is closely dependent on specific training data. For example, there may be very few instances of certain attack. To keep the signatures of this type of attack, the threshold for this attack type should be very small. We set the threshold to 2 in our study, a very conservative number.

The robust CCAS supports the incremental update of cluster structure. The redistribution is incremental. After the supervised grouping, we still could incorporate new available data points one by one, with or without the use of a grid. Another advantage of the robust CCAS is to support the local adjustment of the cluster structure, attributed to the cluster representation and working procedures. Each step functions independently, linked by the clusters.

3.2. Workflow

The clustering and postprocessing steps can be flexibly arranged. Figure 4 shows the five phases with the corresponding actions in our application. Phase 1 calculates the correlation coefficients in the distance measure. We apply the supervised grouping in phase 5 again to get a more compact cluster structure.

Basically, we may use all the produced clusters after each phase to calculate the target value of a new data point. However, this is not appropriate after the grid-based clustering in which grid cells limit the formation of clusters. Therefore, after phases 2 and 3, we use only the clusters in the grid cell of a data point. Grid cells play no role in phases 4 and 5, and then we use all the clusters in classification.

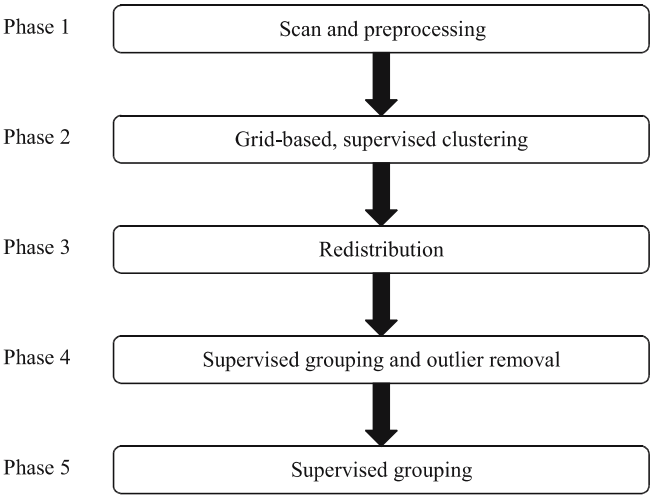

Fig. 4. Workflow of the robust CCAS

3.3. Computational complexity

Let M be the number of the produced clusters after each phase and N the number of data points. In grid-based supervised clustering, the upper bound on the computational complexity is $O(pNM)$ if we search the clusters sequentially, where p is the dimensionality of attributes. If we apply a more efficient cluster-storage structure, such as the one in Huang et al. (1992), the complexity can be reduced to $O(pN)$. For the supervised cluster grouping, the complexity has an upper bound, $O(M_I(M_I - 1)/2)$, on the number of pairwise distances of clusters, where M_I is the number of initial clusters. We inspect these distances in the hierarchical grouping beginning from the smallest one. The computation takes much less time because inspection terminates very soon when more and more distances are associated with clusters of different classes. The computational complexity of removing outliers is $O(M)$.

4. Evaluation experiments

We have applied the robust CCAS on one small data set and two large intrusion detection data sets, summarised in Fig. 5.

(1) THY data is used in an empirical comparison of decision tree, statistical and neural network classifiers (Lim et al. 2000). Taking advantage of its small size, we investigate the robustness of the robust CCAS to the presentation order of training data and the impact by the number of grid intervals. Classes 1 and 2 have much fewer representatives (93 and 191, respectively) than class 3 (3,488). The supervised grouping of clusters generates quite a few clusters for class 1 and 2, many of them containing only one data point. Therefore, we do not perform the outlier removal and Phase 5 because removal of outliers can be damaging.

(2) In 2000 data, we use an exponentially weighted moving-average (EWMA) technique to obtain the smoothed occurrence frequency distribution of 284 audit-

Data sets		THY data	2000 data	Kdd'99 data
Data type		Medical Diagnosis	Computer audit records for a multiple-stage (DDoS) attack	Network connection records for intrusion detection
# of records	Training	3772	Over 100,000	About 5,000,000
	Testing	3428	Over 100,000	Over 300,000
# of attributes	Numeric	6	284	34
	Nominal	15 (binary)	0	7 (transferred into binary)
Target variable		1:normal 2:hyperthyroid 3: subnormal functioning	0: normal 1: intrusive	0:normal, 1:probe, 2:DOS, 3:R2L, 4:U2R
Description		4 training presentation orders as original order with mixed classes(1), reversed original order(2), the order of classes 1, 2, 3 (3), and the order of classes 3, 2, 1 (4).	15 normal sessions and 7 attack sessions in testing data; 2 training presentation orders: original (1) and reversed (2) with mixed normal/attack sessions.	22 attack types in training data; 37 in testing data; attack types fall into 4 subcategories; normal and attack data are mixed.
Source		See [7]	2000 DARPA Intrusion Detection Evaluation Data (http://ideval.ll.mit.edu/)	KDD Cup'99 (http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html)

Fig. 5. Evaluation data sets

event types in Solaris operating system (Li and Ye 2002). Fifteen normal sessions and seven attack sessions are in the data stream from the host machine, called Mill, and 63 normal and 4 attack sessions in the other stream from machine Pascal. The sessions are arranged sequentially, i.e., the data points of different sessions are not mixed in time. We show the result using the Pascal data in training and the Mill data in testing. Although using them the other way shows similar performance, more attack sessions in testing make the performance change more distinctive. We use two input orders in training to examine the robustness of this algorithm. We also test its sensitivity to the grid parameter. We use 6 and 11 grid intervals for input order 1.

(3) The KDD Cup 1999 contest data contain features extracted from network traffic connections for the 1998 DARPA Intrusion Detection Evaluation Program. The number of grid intervals on each dimension is set to three after several experiments. In this study, we compare the accuracy of the robust CCAS with those contest participants.

By looking for the nearest cluster, we use the classic 1-nearest neighbour method for a categorical classification in calculating the confusion matrix. For the continuous predicted-target value, we perform the receiver operating characteristic (ROC) analysis. A false positive (false alarm) occurs when an event is predicted as intrusive but it is in fact normal. A false negative occurs when a truly intrusive event occurs without being signalled. If the target value is greater than the given signal threshold, the data record is signalled as intrusive, considered as normal otherwise. The hit rate is the ratio of the number of hits to the total number of the truly intrusive data records. The false-alarm rate is the ratio of false alarms to the total number of the truly normal data records. A ROC curve plots hit rates and false-alarm rates for various signal thresholds. The closer it is to the top left corner, with 100 % hit rate and 0 % false-alarm rate, of a chart, the better the performance.

In 2000 data, it is inappropriate to detect intrusions based on individual events for the audit sessions. The same audit event may be common in both normal and intrusive activities. We calculate a signal ratio for each session and plot the ROC curves on sessions, with details in Li and Ye (2002). A parameter a is used in signal ratio calculation.

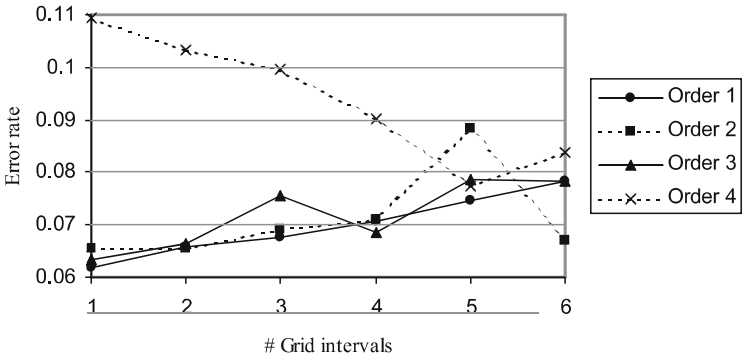

Fig. 6. Error rates versus the number of grid intervals for four input orders on THY data

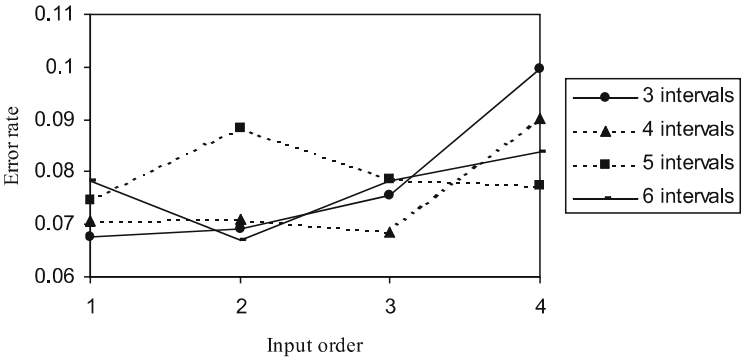

Fig. 7. Error rates versus input orders on THY data

5. Result analysis

For THY data, the overall performance of the robust CCAS is comparable with those classification algorithms in Lim et al. (2000), reporting error rates below 0.1. Figures 6 and 7 plot the change of error rate with different grid intervals and input orders. For input order 1–3, using one grid interval yields the best classification error rate. Compared with class 3, classes 1 and 2 are poorly represented, with very few data points in the training data set. The data points of three classes are well mixed in input order 1 and 2. Class 3 points are at the beginning of input 3. Using just the redistribution and supervised cluster grouping could achieve good performance. However, using more grid intervals, more data points of classes 1 and 2 are correctly classified. Input order 4 produces the worst performance without using grid intervals. The training data points there are organised in the order of classes 3, 2, and 1. The data records of the same class tend to group together, especially for class 3, which has the majority of data points in training. However, the performance is comparable with other input orders when using more grid intervals.

Figure 8 shows the session-based ROC analysis for using 11 grid intervals and input order 1 on 2000 data. Three different a parameters generate three curves in each chart. The performance after phase 2 or phase 3 is not satisfactory. After phase 4 or phase 5, all seven attack sessions are detected without false alarms. We perform ROC

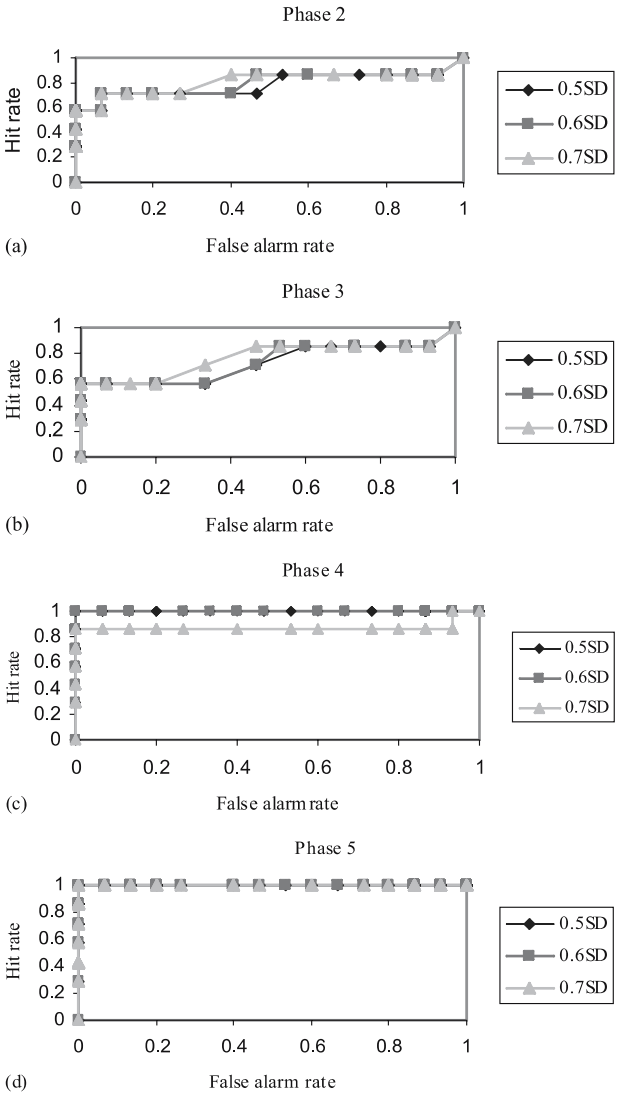

Fig. 8. ROC curves for 2000 data using 11 grid intervals and input order 1

analysis for input order 2 with 11 grid intervals, shown in Fig. 9. Input order shows impact, because the ROC curves for this input order are worse than input order 1 after phases 2 and 3. The following postprocessing steps improve the performance. After phase 5, for two a, we again capture all attack sessions without generating false alarms.

Figure 10 shows the ROC analysis using six grid intervals with input order 1. There is a slight difference between six grid intervals and 11 grid intervals after phase 2 for the same input order. However, after phase 5, they produce the same detection performance. The robust CCAS shows robustness to the grid parameter to some extent.

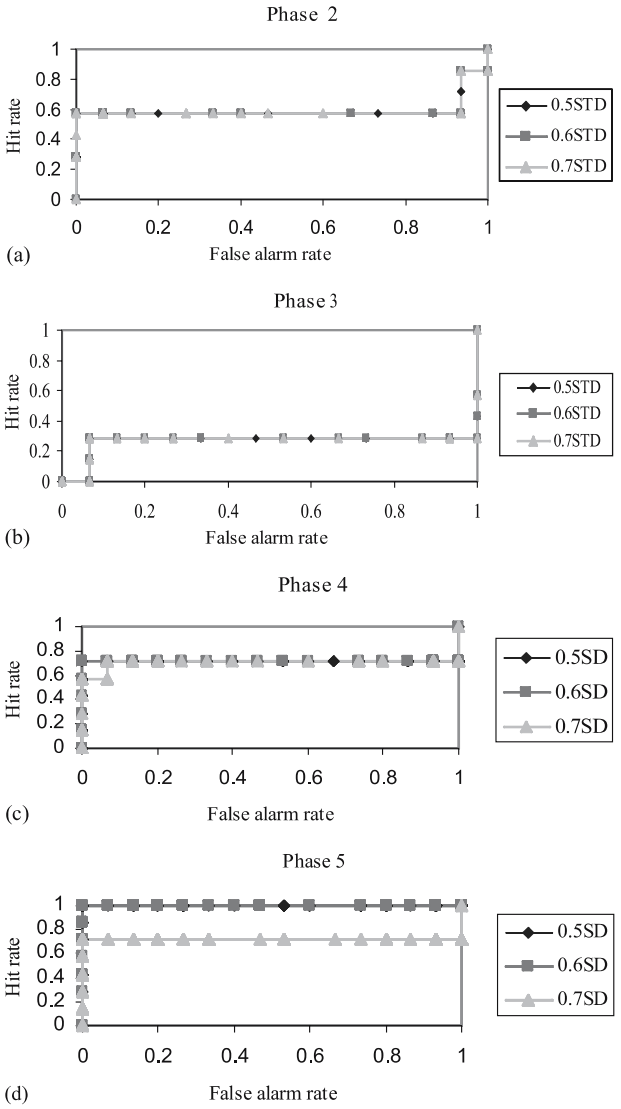

Fig. 9. ROC curves for 2000 data using 11 grid intervals and input order 2

Generally, finer grids may produce better performance because they lead to more clusters and thus allow important patterns of data points to be captured. However, too small grid intervals force natural clusters into smaller clusters. In special cases, this may generate poor cluster structures.

Figure 11 shows the ROC curves for KDD'99 data. The best classification performance is improved after all postprocessing steps, closer to 90 % hit rate with near 0 % false alarm rate. The KDD Cup 1999 applies cost weights to confusion matrix cells to obtain an average cost per data record. The lower this average cost, the better the classification performance. The winning technique produces the average cost of 0.2331, with 0.5 % false alarm rate and 91.8 % hit rate. The best 17 participating

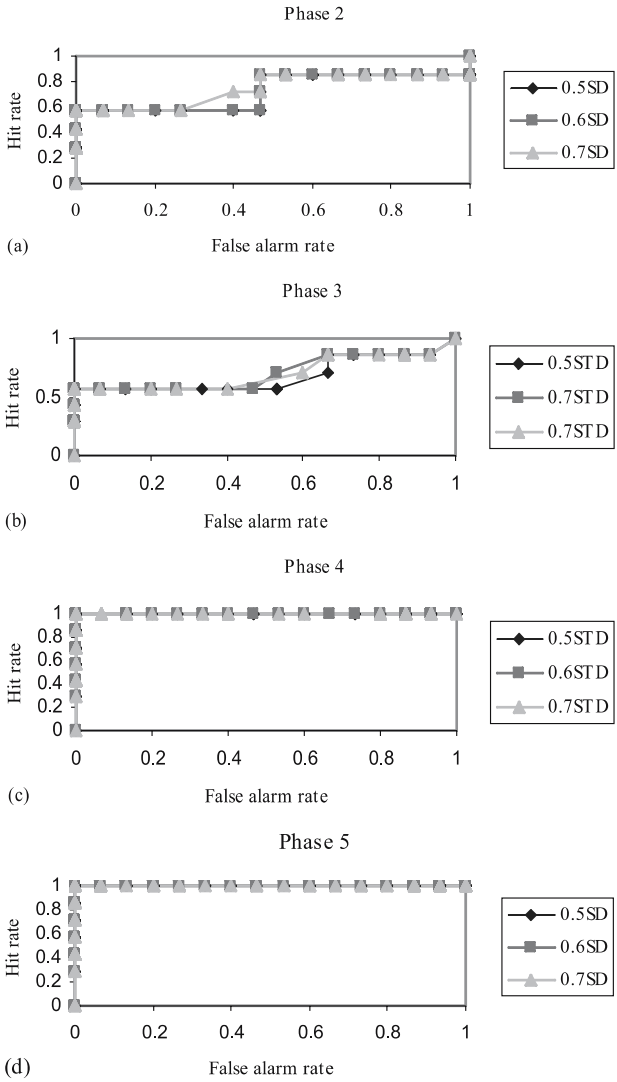

Fig. 10. ROC curves for 2000 data using 6 grid intervals and input order 1

algorithms have average costs from 0.2331 to 0.2684. Figure 12 shows the confusion matrices and calculated average cost, hit rate and false-alarm rate for the robust CCAS. The average cost is improved to be 0.2445 after phase 5, comparable with the best participants. The false-alarm rate is about 0.9 % and the hit rate is 91 %.

6. Conclusion

We present the robust CCAS—a scalable and incremental data-mining algorithm. The testing results show that the robust CCAS makes significant improvement in

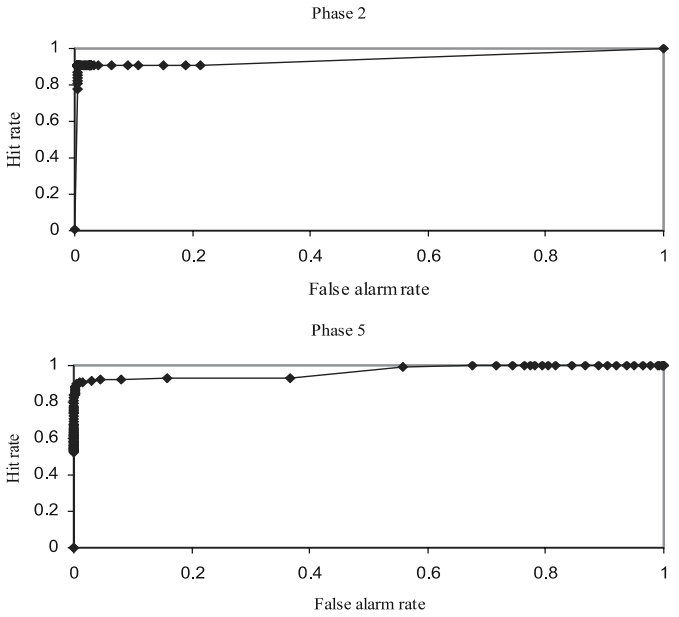

Fig. 11. ROC curves for KDD'99 data

Phase 2

		Predicted					
		0	1	2	3	4	Correct rate
Actual	0	60127	332	120	5	9	0.992
	1	885	3079	187	1	14	0.739
	2	6574	183	223061	0	35	0.97
	3	114	5	87	16	6	0.07
	4	15363	19	22	6	779	0.048
	Correct rate	0.724	0.851	0.998	0.571	0.924	
Cost per example:		**0.2488**	**False alarm rate:**		**0.008**	**Hit rate:**	**0.908**

Phase 5

		Predicted					
		0	1	2	3	4	Correct rate
Actual	0	60035	380	163	1	14	0.991
	1	985	3033	128	2	18	0.728
	2	6607	236	222962	0	48	0.97
	3	111	2	87	21	7	0.092
	4	14820	25	269	50	1025	0.063
	Correct rate	0.727	0.825	0.997	0.284	0.922	
Cost per example:		**0.2445**	**False alarm rate:**		**0.009**	**Hit rate:**	**0.910**

Fig. 12. Confusion matrices for KDD'99 data

detection performance and robustness to input order of training data. One important future research topic will be investigating the adaptive grid setting.

Acknowledgements. This work is sponsored by the Air Force Office of Scientific Research (AFOSR) under grant F49620-99-1-001.

References

Axelsson S (2000) Intrusion detection systems: a survey and taxonomy. Report, Dept of Computer Engineering, Chalmers University of Technology, Goteborg, Sweden

Ester M, Kriegel HP, Sander J, Wimmer M, Xu X (1998) Incremental clustering for mining in a data warehousing environment. Proc 24th VLDB conference, New York, USA

Harsha SG, Choudhary A (1999) Parallel subspace clustering for very large data sets. Technical report CPDC-TR-9906-010, Northwestern University, Evanston, Illinois, USA

Huang C, Bi Q, Stiles R, Harris R (1992) Fast full search equivalent encoding algorithms for image compression using vector quantization. IEEE Trans Image Process 1(3):413–416

Jain AK, Dubes RC (1988) Algorithms for clustering data. Prentice Hall

Li X, Ye N (2002) Grid- and dummy-cluster-based learning of normal and intrusive clusters for computer intrusion detection. Qual Reliabil Eng Int 18(3)

Lim TS, Loh WY, Shih YS (2000) A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. Mach Learn J 40:203–228

Zhang T (1997) Data clustering for very large datasets plus applications. Ph.D. Thesis, Department of Computer Science, University of Wisconsin–Madison, Madison, Wisconsin, USA

Correspondence and offprint requests to: Xiangyang Li, Department of Industrial and Manufacturing Systems Engineering, University of Michigan—Dearborn, Dearborn, MI 48128, USA. Email: xylum@umich.edu