DFRWS
DIGITAL FORENSIC RESEARCH CONFERENCE

Audit Data Reduction Using Neural Networks and Support Vector Machines

By

Srinivas Mukkamala, Andrew Sung

From the proceedings of

The Digital Forensic Research Conference

DFRWS 2002 USA

Syracuse, NY (Aug 6th - 9th)

DFRWS is dedicated to the sharing of knowledge and ideas about digital forensics research. Ever since it organized the first open workshop devoted to digital forensics in 2001, DFRWS continues to bring academics and practitioners together in an informal environment.

As a non-profit, volunteer organization, DFRWS sponsors technical working groups, annual conferences and challenges to help drive the direction of research and development.

http:/dfrws.org

Feature Ranking and Selection for Intrusion Detection Systems Using Support Vector Machines

Srinivas Mukkamala, Andrew H. Sung
{srinivas, sung}@cs.nmt.edu
Department of Computer Science
New Mexico Institute of Mining and Technology
Socorro, New Mexico 87801

ABSTRACT

Intrusion detection is a critical component of secure information systems. This paper addresses the issue of identifying important input features in building an intrusion detection system (IDS). Since elimination of the insignificant and/or useless inputs leads to a simplification of the problem, faster and more accurate detection may result. Feature ranking and selection, therefore, is an important issue in intrusion detection.

Since support vector machines (SVMs) tend to scale better and run faster than neural networks with higher accuracy, we apply the technique of deleting one feature at a time to perform experiments on SVMs to rank the importance of input features for the DARPA collected intrusion data. Important features for each of the 5 classes of intrusion patterns in the DARPA data are identified.

It is shown that SVM-based IDSs using a reduced number of features can deliver enhanced or comparable performance. An IDS for class-specific detection based on five SVMs is proposed.

1. INTRODUCTION

This paper mainly addresses the issue of identifying important input features for intrusion detection. Science the ability to identify the important inputs and redundant inputs of a classifier leads directly to reduced size, faster training and possibly more accurate results, it is critical to be able to identify the important features of network traffic data for intrusion detection in order for the IDS to achieve maximal performance.

Since most of the intrusions can be uncovered by examining patterns of user activities, many intrusion detection systems have been built by utilizing the recognized attack and misuse patterns to develop learning machines [1,2,3,4,5,6,7,8,9]. In our earlier work, support vector machines (SVMs) are found to be superior to neural networks in many important respects of intrusion detection [10,11,12], so we will illustrate feature ranking use SVMs.

The data we used in our experiments originated from MIT's Lincoln Lab. It was developed for intrusion detection system evaluations by DARPA and is considered a benchmark for intrusion detection evaluations [13].

We performed experiments to rank the importance of input features for each of the five classes (normal, probe, denial of service, user to super, and remote to local) of patterns in the DARPA data. It is shown that using only the important features for classification gives good accuracies and, in certain cases, reduces the training time and testing time of the SVM classifier.

In the rest of the paper, a brief introduction to the data we used is given in section 2. In section 3 we describe the method of deleting one input feature at a time and the performance metrics considered for deciding the importance of a particular feature. In section 4 we present the experimental results of using SVMs for feature ranking. In section 5 we summarize our results and give a brief description of our proposed IDS architecture.

2. THE DATA

In the 1998 DARPA intrusion detection evaluation program, an environment was set up to acquire raw TCP/IP dump data for a network by simulating a typical U.S. Air Force LAN. The LAN was operated like a true environment, but being blasted with multiple attacks. For each TCP/IP connection, 41 various quantitative and qualitative features were extracted. Of this database a subset of 494021 data were used, of which 20% represent normal patterns.

Attack types fall into four main categories:

1. DOS: denial of service
2. R2L: unauthorized access from a remote machine
3. U2R: unauthorized access to local super user (root) privileges
4. Probing: surveillance and other probing

3. RANKING THE SIGNIFICANCE OF INPUTS

Feature selection and ranking is an important issue in intrusion detection. Of the large number of features that can be monitored for intrusion detection purpose, which are truly useful, which are less significant, and which may be useless? The question is relevant because the elimination of useless features (or audit trail reduction) enhances the accuracy of detection while speeding up the computation, thus improving the overall performance of an IDS. In cases where there are no useless features, by concentrating on the most important ones we may well improve the time performance of an IDS without affecting the accuracy of detection in statistically significant ways.

The feature ranking and selection problem for intrusion detection is similar in nature to various engineering problems that are characterized by

- Having a large number of input variables $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ of varying degrees of importance; i.e., some elements of $\mathbf{x}$ are essential, some are less important, some of them may not be mutually independent, and some may be useless or noise
- Lacking an analytical model or mathematical formula that precisely describes the input-output relationship, $\mathbf{y} = \boldsymbol{F}(\mathbf{x})$
- Having available a finite set of experimental data, based on which a model (e.g. neural networks) can be built for simulation and prediction purposes

Due to the lack of an analytical model, one can only seek to determine the relative importance of the input variables through empirical methods. A complete analysis would require examination of all possibilities, e.g., taking two variables at a time to analyze their dependence or correlation, then taking three at a time, etc. This, however, is both infeasible (requiring 2^n experiments!) and not infallible (since the available data may be of poor quality in sampling the whole input space). In the following, therefore, we apply the technique of deleting one feature at a time [14] to rank the input features and identify the most important ones for intrusion detection using SVMs.

3.1 Methodology for Ranking Importance

We first describe the input ranking methodology: One input feature is deleted from the data at a time, the resultant data set is then used for the training and testing of the classifier. Then the classifier's performance is compared to that of the original classifier (based on all features) in terms of relevant performance criteria. Finally, the importance of the feature is ranked according to a set of rules based on the performance comparison. The methodology is summarized as follows:

```
1. compose the training set and the testing set;
   for each feature do the following
2. delete the feature from the (training and testing) data;
3. use the resultant data set to train the classifier;
4. analyze the performance of the classifier using the test set, in terms of the selected
   performance criteria;
5. rank the importance of the feature according to the rules;
```

3.2 Performance Metrics

To rank the importance of the 41 features (of the DARPA data) in an SVM-based IDS, we consider three main performance criteria: overall accuracy of (5-class) classification; training time; and testing time. Each feature will be ranked as "important", "secondary", or "insignificant", according to the following rules that are applied to the result of performance comparison of the original 41-feature SVM and the 40-feature SVM:

Rule set:

1. ***If*** *accuracy* decreases ***and*** training time increases ***and*** testing time decreases, ***then*** the feature is important
2. ***If*** *accuracy* decreases ***and*** training time increases ***and*** testing time increases, ***then*** the feature is important
3. ***If*** *accuracy* decreases ***and*** training time decreases ***and*** testing time increases, ***then*** the feature is important
4. ***If*** *accuracy* unchanges ***and*** training time increases ***and*** testing time increases, ***then*** the feature is important
5. ***If*** *accuracy* unchanges ***and*** training time decreases *and* testing time increases, ***then*** the feature is secondary
6. ***If*** *accuracy* unchanges ***and*** training time increases ***and*** testing time decreases, ***then*** the feature is secondary

7. ***If*** *accuracy* unchanges ***and*** training time decreases ***and*** testing time decreases, ***then*** the feature is unimportant
8. ***If*** *accuracy* increases ***and*** training time increases ***and*** testing time decreases, ***then*** the feature is secondary
9. ***If*** *accuracy* increases ***and*** training time decreases ***and*** testing time increases, ***then*** the feature is secondary
10. ***If*** *accuracy* increases ***and*** training time decreases ***and*** testing time decreases, ***then*** the feature is unimportant

Our performance-based methodology is an extension of the feature ranking technique of [14] and it has the advantages of linear time complexity (i.e., requiring only $O(n)$ experiments) and being generally applicable (i.e., regardless of the type of the learning machine used), and tunable (i.e., the rules can be iteratively tuned to improve performance). The rule set above leads to a 3-class ranking of the input significance (important, secondary, and unimportant); however, we also note that another advantage of the methodology is that it can be applied to obtain a ranking into more than 3 classes by designing the rules accordingly.

Because SVMs are only capable of binary classifications, we will need to employ five SVMs for the five-class identification problem in intrusion detection [15,16.17,18]. But since the set of important features may differ from class to class, using five SVMs becomes an advantage rather than a hindrance, i.e., in building an IDS using five SVMs, each SVM can use only the important features for that class which it is responsible for making classifications.

4. EXPERIMENTS

Support vector machines are used for ranking the importance of the input features, taking above mentioned performance metrics and the rule set into consideration [15]. Once the importance of the input features was ranked, the classifiers were trained and tested with only the important features. Further, we validate our methodology by comparing the performance of the classifier using all input features to that using the important and the secondary features; and we also compare the performance of a classifier using the union of the important features for all fives classes.

(Because SVMs are only capable of binary classifications, we will need to employ five SVMs for the five-class identification problem in intrusion detection. But since the set of important features may differ from class to class, using five SVMs becomes an advantage rather than a hindrance, i.e., in building an IDS using five SVMs, each SVM can use only the important features for that class which it is responsible for making classifications.)

4.1 Support Vector Machines

Our results are summarized in the following tables. Table 1 gives the performance results of the five SVMs for each respective class of data. Table 2 through Table 6, each containing the results of 41 experiments, give the performance statistics of the SVM with 40 features. Table 7 shows the results of SVMs performing classification, with each SVM using as input the important features for all five classes. Table 8 shows the results of

SVMs performing classification, with each SVM using as input the union of the important features for all five classes. Table 9 shows the result of SVMs performing classification, with each SVM using as input the important and secondary features for each respective class.

Table1: Performance of SVMs using 41 features

Class	Training Time (sec)	Testing Time (sec)	Accuracy
Normal	7.66	1.26	99.55%
Probe	49.13	2.10	99.70%
DOS	22.87	1.92	99.25
U2Su	3.38	1.05	99.87
R2L	11.54	1.02	99.78

Table2: Class 1, Normal

Feature deleted	Training Time (sec)	Testing Time (sec)	Accuracy
None	7.66	1.26	99.55
1.	10.19	1.11	99.51
2.	6.56	1.46	99.55
3.	9.06	1.47	99.48
4.	9.96	1.08	99.55
5.	33.11	1.62	99.19
6.	7.56	1.79	98.75
7.	7.11	1.43	99.55
8.	8.33	1.41	99.55
9.	8.37	1.37	99.55
10.	8.68	1.35	99.55
11.	7.49	1.33	99.55
12.	8.01	1.38	99.55
13.	7.14	0.81	99.55
14.	8.00	1.46	99.55
15.	9.81	1.43	99.55
16.	8.15	1.04	99.55
17.	8.12	1.47	99.55
18.	7.36	1.30	99.55
19.	8.00	1.12	99.55
20.	8.15	1.38	99.55
21.	7.98	1.42	99.55
22.	8.12	1.43	99.55
23.	7.65	1.34	99.56
24.	7.29	1.30	99.55
25.	8.32	1.35	99.55
26.	7.71	1.30	99.55
27.	7.73	1.38	99.55
28.	7.90	1.47	99.55
29.	7.81	1.39	99.55
30.	7.57	1.38	99.55
31.	7.11	1.30	99.55
32.	6.17	1.26	99.55
33.	8.53	1.51	99.48
34.	7.23	1.48	99.55
35.	6.96	1.35	99.55
36.	10.19	1.36	99.55
37.	6.74	1.33	99.55
38.	8.17	1.43	99.55
39.	7.75	1.32	99.55
40.	7.20	1.45	99.55
41.	9.38	1.43	99.55

Table3: Class 2, Probe

Feature deleted	Training Time (sec)	Testing Time (sec)	Accuracy
None	49.13	2.10	99.70
1.	58.93	2.01	99.70

2.	44.07	1.79	99.70
3.	51.00	2.19	99.61
4.	62.42	1.85	99.70
5.	75.67	1.97	98.14
6.	51.03	1.17	99.52
7.	51.62	1.98	99.70
8.	55.34	1.88	99.72
9.	53.05	1.99	99.70
10.	46.29	2.00	99.70
11.	45.68	1.96	99.70
12.	53.18	1.95	99.70
13.	55.27	1.95	99.70
14.	50.67	1.92	99.70
15.	49.50	2.07	99.70
16.	47.61	2.16	99.70
17.	49.38	1.93	99.70
18.	50.28	1.91	99.70
19.	50.33	1.94	99.70
20.	48.61	1.93	99.70
21.	50.40	1.89	99.70
22.	51.50	1.96	99.70
23.	49.00	2.63	99.46
24.	42.86	1.97	99.61
25.	52.40	1.95	99.71
26.	52.42	1.99	99.71
27.	62.51	2.05	99.71
28.	71.80	1.91	99.71
29.	45.95	1.78	99.70
30.	46.62	2.00	99.70
31.	46.35	1.93	99.70
32.	31.89	1.82	99.67
33.	50.90	1.83	99.62
34.	47.64	1.30	99.70
35.	49.49	1.87	99.70
36.	47.39	1.97	99.70
37.	48.19	2.03	99.70
38.	57.51	1.85	99.71
39.	52.54	1.94	99.71
40.	56.45	1.98	99.70
41.	51.66	1.71	99.70

Table4: Class 3, Denial of Service

Feature deleted	Training Time (sec)	Testing Time (sec)	Accuracy
None	22.87	1.92	99.25
1.	21.76	1.87	99.23
2.	23.60	1.89	99.25
3.	17.88	2.03	99.10
4.	20.00	1.79	99.25
5.	39.57	1.61	97.55
6.	19.63	0.84	98.07
7.	23.76	1.87	99.25
8.	31.23	1.86	99.20
9.	23.80	1.78	99.25
10.	27.01	1.82	99.25
11.	22.03	1.86	99.25
12.	19.69	1.84	99.25
13.	21.30	1.93	99.25
14.	20.18	2.02	99.25
15.	18.76	1.89	99.25
16.	21.56	1.78	99.25
17.	22.98	2.09	99.25
18.	21.47	1.95	99.25
19.	20.79	1.97	99.25
20.	21.49	1.96	99.25
21.	21.75	1.94	99.25
22.	24.93	2.01	99.25
23.	23.94	3.01	98.58
24.	25.43	2.05	99.20
25.	21.70	1.80	99.19
26.	25.93	1.98	99.19
27.	24.21	1.41	99.20
28.	26.16	1.80	99.20
29.	29.99	1.93	99.25
30.	18.27	1.79	99.20
31.	19.85	1.79	99.25
32.	11.70	0.95	98.69
33.	44.19	1.74	99.19
34.	28.27	1.88	99.25
35.	28.94	1.75	99.22
36.	27.39	1.80	99.22
37.	22.40	1.86	99.25
38.	22.45	1.95	99.19
39.	23.81	1.92	99.20
40.	50.15	1.84	99.22
41.	25.36	2.03	99.19

Table5: Class 4, User to Root

Feature	Training	Testing	Accuracy

deleted	Time (sec)	Time (sec)	
None	3.38	1.05	99.87
1.	2.98	0.96	99.87
2.	3.35	0.98	99.87
3.	3.00	1.04	99.87
4.	3.21	1.04	99.87
5.	3.11	0.65	99.72
6.	1.99	0.18	88.81
7.	3.40	1.07	99.87
8.	3.43	1.10	99.87
9.	3.37	0.97	99.87
10.	3.69	0.97	99.87
11.	3.47	1.06	99.87
12.	3.36	0.99	99.87
13.	3.61	1.01	99.87
14.	3.12	1.02	99.87
15.	3.40	1.11	99.87
16.	3.57	1.14	99.87
17.	3.39	0.98	99.87
18.	3.46	1.07	99.87
19.	3.41	1.05	99.87
20.	3.35	1.10	99.87
21.	3.34	1.08	99.87
22.	3.26	1.07	99.87
23.	3.39	1.05	99.87
24.	3.32	1.07	99.87
25.	3.44	1.09	99.87
26.	3.38	1.06	99.87
27.	3.36	1.05	99.87
28.	3.23	1.00	99.87
29.	3.36	0.98	99.87
30.	3.42	0.98	99.87
31.	3.34	1.00	99.87
32.	3.95	0.92	99.84
33.	4.58	0.99	99.85
34.	3.36	1.02	99.87
35.	2.98	1.05	99.87
36.	3.50	1.05	99.87
37.	3.43	1.00	99.87
38.	3.79	1.05	99.87
39.	3.27	1.07	99.87
40.	3.36	0.99	99.87
41.	3.36	1.01	99.87

Table6: Class 5, Remote to Local

Feature deleted	Training Time (sec)	Testing Time (sec)	Accuracy
None	11.54	1.02	99.78
1.	7.54	1.04	99.80
2.	8.79	1.23	99.78
3.	9.95	1.11	99.75
4.	8.56	1.26	99.78
5.	12.11	1.79	99.06
6.	16.52	0.63	98.88
7.	10.18	1.34	99.78
8.	9.59	1.31	99.78
9.	8.41	1.23	99.78
10.	9.30	1.32	99.78
11.	10.21	1.23	99.78
12.	9.48	1.33	99.78
13.	9.88	1.29	99.78
14.	8.84	1.22	99.78
15.	9.25	1.28	99.78
16.	8.89	1.20	99.78
17.	9.21	1.24	99.78
18.	9.60	1.30	99.78
19.	10.15	1.30	99.78
20.	10.68	0.99	99.78
21.	10.99	1.26	99.78
22.	10.88	1.26	99.78
23.	8.19	1.26	99.78
24.	7.67	1.22	99.72
25.	9.26	1.05	99.78
26.	10.11	1.30	99.78
27.	9.09	1.24	99.78
28.	9.10	1.23	99.78
29.	11.39	1.11	99.78
30.	10.64	1.26	99.78
31.	8.56	1.26	99.78
32.	11.55	1.05	99.80
33.	12.35	1.25	99.80
34.	10.59	1.14	99.78
35.	9.07	1.18	99.78
36.	9.22	1.22	99.78
37.	9.33	1.30	99.78
38.	8.98	0.95	99.78
39.	8.52	1.26	99.78

40.	8.98	1.11	99.78

41.	10.35	1.26	99.78

Table7: Performance of SVMs using important features

Class	No of Features	Training Time (sec)	Testing Time (sec)	Accuracy
Normal	25	9.36	1.07	99.59%
Probe	7	37.71	1.87	99.38%
DOS	19	22.79	1.84	99.22%
U2Su	8	2.56	0.85	99.87%
R2L	6	8.76	0.73	99.78%

Table8: Performance of SVMs using union of important features (30)

Class	Training Time (sec)	Testing Time (sec)	Accuracy
Normal	7.67	1.02	99.51%
Probe	44.38	2.07	99.67%
DOS	18.64	1.41	99.22%
U2Su	3.23	0.98	99.87%
R2L	9.81	1.01	99.78%

Table9: Performance of SVMs using important and secondary features

Class	No of Features	Training Time (sec)	Testing Time (sec)	Accuracy
Normal	39	8.15	1.22	99.59%
Probe	32	47.56	2.09	99.65%
DOS	32	19.72	2.11	99.25%
U2Su	25	2.72	0.92	99.87%
R2L	37	8.25	1.25	99.80%

5. SUMMARY & CONCLUSIONS

Comparing Table 1 with Tables 7, 8, and 9, we observe that

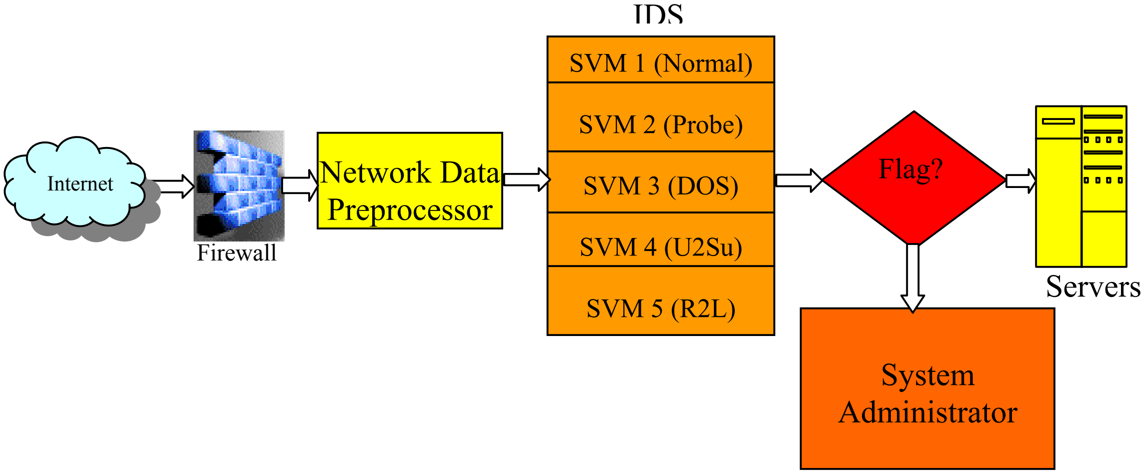
- The most important features for the two classes of ‘Normal’ and ‘DOS’ heavily overlap
- ‘U2Su’ and ‘R2L’, the two smallest classes representing the most serious attacks, each has a small number of important features and a large number of secondary features
- The performances of (a) using the important features for each class, Table 7, (b) using the union of important features, Table 8, and (c) using the union of

important and secondary features for each class, do not show significant differences, and are all similar to that of using all 41 features

- Using the important features for each class gives the most remarkable performance: the testing time decreases in each class; the accuracy increases slightly for one class 'Normal', decreases slightly for two classes 'Probe' and 'DoS', and remains the same for the two most serious attack classes.

Our ongoing experiments include making 23-class (22 specific attacks and normal) feature identification using SVMs and neural networks, for designing a cost-effective and real time intrusion detection tool.

We propose a five SVM based intrusion detection architecture, where we use the set of important features for each class that are responsible for making classifications.



7. REFERENCES

1. Denning D (Feb 1987) An Intrusion-Detection Model. IEEE Transactions on Software Engineering, Vol.SE-13, No 2.
2. Kumar S, Spafford EH (1994) An Application of Pattern Matching in Intrusion Detection. Technical Report CSD-TR-94-013. Purdue University.
3. Ghosh AK. (1999). Learning Program Behavior Profiles for Intrusion Detection. USENIX.
4. Cannady J. (1998) Artificial Neural Networks for Misuse Detection. National Information Systems Security Conference.
5. Ryan J, Lin M-J, Miikkulainen R (1998) Intrusion Detection with Neural Networks. Advances in Neural Information Processing Systems 10, Cambridge, MA: MIT Press.
6. Debar H, Becke M, Siboni D (1992) A Neural Network Component for an Intrusion Detection System. Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy.
7. Debar H, Dorizzi B (1992) An Application of a Recurrent Network to an Intrusion Detection System. Proceedings of the International Joint Conference on Neural Networks, pp.78-83.

8. Luo J, Bridges SM (2000) Mining Fuzzy Association Rules and Fuzzy Frequency Episodes for Intrusion Detection. International Journal of Intelligent Systems, John Wiley & Sons, pp.687-703.

9. Cramer M, et. al. (1995) New Methods of Intrusion Detection using Control-Loop Measurement. Proceedings of the Technology in Information Security Conference (TISC) '95, pp.1-10.

10. S Mukkamala, G Janowski, A H. Sung Monitoring Information System Security, Proceedings of the 11th Annual Workshop on Information Technologies & Systems (December 2001), pp.139-144.

11. S Mukkamala, G Janowski, A H. Sung Intrusion Detection Using Neural Networks and Support Vector Machines Proceedings of IEEE International Joint Conference on Neural Networks 2002 (Hawaii, May 2002), pp.1702-1707.

12. Srinivas Mukkamala, A H. Sung (2002) Comparison of Neural Networks and Support Vector Machines in Intrusion Detection Workshop on Statistical and Machine Learning Techniques in Computer Intrusion Detection, June 11-13, 2002 http://www.mts.jhu.edu/~cidwkshop/abstracts.html

13. http://kdd.ics.uci.edu/databases/kddcup99/task.htm.

14. Sung AH (1998) Ranking Importance of Input Parameters Of Neural Networks. Expert Systems with Applications, pp.405-411.

15. Joachims T (2000) SVMlight is an implementation of Support Vector Machines (SVMs) in C. http://ais.gmd.de/~thorsten/svm_light. University of Dortmund. Collaborative Research Center on "Complexity Reduction in Multivariate Data" (SFB475).

16. Joachims T (1998) Making Large-Scale SVM Learning Practical. LS8-Report, University of Dortmund, LS VIII-Report.

17. Vladimir VN (1995) The Nature of Statistical Learning Theory. Springer, Berlin Heidelberg New York.

18. Joachims T (2000) Estimating the Generalization Performance of a SVM Efficiently. Proceedings of the International Conference on Machine Learning, Morgan Kaufman.