Intrusion Detection Using Ensemble of Soft Computing Paradigms

Srinivas Mukkamala[1], Andrew H. Sung[1,2] and Ajith Abraham[3]

{srinivas|sung}@cs.nmt.edu, ajith.abraham@ieee.org
[1]Department of Computer Science, [2]Institute for Complex Additive Systems Analysis, New Mexico Tech, Socorro, New Mexico 87801
[3]Department of Computer Science, Oklahoma State University, Tulsa, OK 74106

Abstract:

Soft computing techniques are increasingly being used for problem solving. This paper addresses using ensemble approach of different soft computing techniques for intrusion detection. Due to increasing incidents of cyber attacks, building effective intrusion detection systems (IDSs) are essential for protecting information systems security, and yet it remains an elusive goal and a great challenge. Two classes of soft computing techniques are studied: Artificial Neural Networks (ANNs) and Support Vector Machines (SVMs). We show that ensemble of ANN and SVM is superior to individual approaches for intrusion detection in terms of classification accuracy.

Keywords: Information system security, Intrusion detection, neural network, support vector machines, ensemble of soft computing paradigms

1 Introduction

This paper concerns intrusion detection and the related issue of identifying a good detection mechanism. Intrusion detection is a problem of great significance to critical infrastructure protection owing to the fact that computer networks are at the core of the nation's operational control. This paper summarizes our current work to build Intrusion Detection Systems (IDSs) using Artificial Neural Networks or ANNs [1], Support Vector Machines or SVMs [2] and the ensemble of different artificial intelligent techniques. Since the ability of a good detection technique gives more accurate results, it is critical for intrusion detection in order for the IDS to achieve maximal performance. Therefore, we study different soft computing techniques and also their ensemble for building models based on experimental data.

Since most of the intrusions can be uncovered by examining patterns of user activities, many IDSs have been built by utilizing the recognized attack and misuse patterns to develop learning machines [3,4,5,6,7,8,9]. In our recent work, SVMs are found to be superior to ANNs in many important respects of intrusion detection [9]; In this paper we will concentrate on using the ensemble of support vector machines and neural networks with different training functions to achieve better classification accuracies.

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 [10].

We performed experiments to classify each of the five classes (normal, probe, denial of service, user to super-user, and remote to local) of patterns in the DARPA data. It is shown that using the ensemble of different artificial intelligent techniques for classification gives good accuracies.

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 theoretical aspects of neural networks, support vector machines and the ensemble of artificial intelligent techniques. In section 4 we present the experimental results of neural networks, support vector machines and their ensemble. In section 5 we summarize our results and give a brief description of our proposed IDS architecture.

2 Intrusion Dataset

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 real environment, but being blasted with multiple attacks. For each TCP/IP connection, 41 various quantitative and qualitative features were extracted (11). Of this database a subset of 494021 data were used, of which 20% represent normal patterns.

The four different categories of attack patterns are:

a. Denial of Service (DOS) Attacks: A denial of service attack is a class of attacks in which an attacker makes some computing or memory resource too busy or too full to handle legitimate requests, or denies legitimate users access to a machine. Examples are Apache2, Back, Land, Mail bomb, SYN Flood, Ping of death, Process table, Smurf, Syslogd, Teardrop, Udpstorm.
b. User to Superuser or Root Attacks (U2Su): User to root exploits are a class of attacks in which an attacker starts out with access to a normal user account on the system and is able to exploit vulnerability to gain root access to the system. Examples are Eject, Ffbconfig, Fdformat, Loadmodule, Perl, Ps, Xterm.
c. Remote to User Attacks (R2L): A remote to user attack is a class of attacks in which an attacker sends packets to a machine over a network–but who does

not have an account on that machine; exploits some vulnerability to gain local access as a user of that machine. Examples are Dictionary, Ftp_write, Guest, Imap, Named, Phf, Sendmail, Xlock, Xsnoop.

d. Probing (Probe): Probing is a class of attacks in which an attacker scans a network of computers to gather information or find known vulnerabilities. An attacker with a map of machines and services that are available on a network can use this information to look for exploits. Examples are Ipsweep, Mscan, Nmap, Saint, Satan.

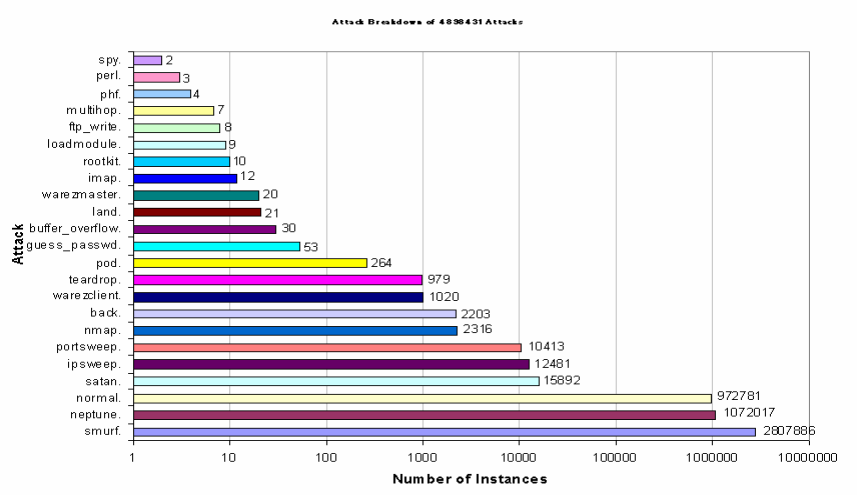

Figure 1: Intrusion detection data distribution

3 Connectionist Paradigms

Connectionist models "learn" by adjusting the interconnections between layers. When the network is adequately trained, it is able to generalize relevant output for a set of input data.

3.1. Artificial Neural Networks (ANNs)

The artificial neural network (ANN) methodology enables us to design useful nonlinear systems accepting large numbers of inputs, with the design based solely on instances of input-output relationships.

a. **Resilient Back propagation (RP)**

The purpose of the resilient back propagation training algorithm is to eliminate the harmful effects of the magnitudes of the partial derivatives. Only

the sign of the derivative is used to determine the direction of the weight update; the magnitude of the derivative has no effect on the weight update. The size of the weight change is determined by a separate update value. The update value for each weight and bias is increased by a factor whenever the derivative of the performance function with respect to that weight has the same sign for two successive iterations. The update value is decreased by a factor whenever the derivative with respect that weight changes sign from the previous iteration. If the derivative is zero, then the update value remains the same. Whenever the weights are oscillating the weight change will be reduced. If the weight continues to change in the same direction for several iterations, then the magnitude of the weight change will be increased [12].

b. **Scaled Conjugate Gradient Algorithm (SCG)**

Moller [13] introduced the scaled conjugate gradient algorithm as a way of avoiding the complicated line search procedure of conventional conjugate gradient algorithm (CGA). According to the SCGA, the Hessian matrix is approximated by

$$E''(w_k)p_k = \frac{E'(w_k + \sigma_k p_k) - E'(w_k)}{\sigma_k} + \lambda_k p_k \qquad (1)$$

where E' and E'' are the first and second derivative information of global error function $E(w_k)$. The other terms p_k, σ_k and λ_k represent the weights, search direction, parameter controlling the change in weight for second derivative approximation and parameter for regulating the indefiniteness of the Hessian. In order to get a good quadratic approximation of E, a mechanism to raise and lower λ_k is needed when the Hessian is positive definite. Detailed step-by-step description can be found in [13].

c. **One-Step-Secant Algorithm (OSS)**

Quasi-Newton method involves generating a sequence of matrices $G^{(k)}$ that represents increasingly accurate approximations to the inverse Hessian (H^{-1}). Using only the first derivative information of E [14], the updated expression is as follows:

$$G^{(k+1)} = G^{(k)} + \frac{pp^T}{p^T v} - \frac{(G^{(k)}v)v^T G^{(k)}}{v^T G^{(k)} v} + (v^T G^{(k)} v)uu^T \qquad (2)$$

where

$$p = w^{(k+1)} - w^{(k)}\ , v = g^{(k+1)} - g^{(k)}\ ,\ u = \frac{p}{p^T v} - \frac{G^{(k)}v}{v^T G^{(k)} v} \qquad (3)$$

and T represents transpose of a matrix. The problem with this approach is the requirement of computation and storage of the approximate Hessian matrix for

every iteration. The One-Step-Secant (OSS) is an approach to bridge the gap between the conjugate gradient algorithm and the quasi-Newton (secant) approach. The OSS approach doesn't store the complete Hessian matrix; it assumes that at each iteration the previous Hessian was the identity matrix. This also has the advantage that the new search direction can be calculated without computing a matrix inverse [14].

3.2 Support Vector Machines (SVMs)

The SVM approach transforms data into a feature space *F* that usually has a huge dimension. It is interesting to note that SVM generalization depends on the geometrical characteristics of the training data, not on the dimensions of the input space [15]. Training a support vector machine (SVM) leads to a quadratic optimization problem with bound constraints and one linear equality constraint. Vapnik shows how training a SVM for the pattern recognition problem leads to the following quadratic optimization problem [16].

$$\text{Minimize: } W(\alpha) = -\sum_{i=1}^{l} \alpha_i + \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j k(x_i, x_j) \qquad (4)$$

$$\text{Subject to } \sum_{i=1}^{l} y_i \alpha_i \qquad (5)$$
$$\forall i : 0 \le \alpha_i \le C$$

Where l is the number of training examples α is a vector of l variables and each component α_i corresponds to a training example (x_i, y_i). The solution of (4) is the vector α^* for which (4) is minimized and (5) is fulfilled.

3.2 Ensemble of Soft Computing Paradigms

Optimal linear combination of neural networks has been investigated and has found to be very useful **Error! Reference source not found.**. The optimal weights were decided based on the ordinary least squares regression coefficients in an attempt to minimize the mean squared error. The problem becomes more complicated when we have to optimize several other error measures. In the case of intrusion detection, our task is to design a classifier, which could give the best accuracy for each category of attack patterns. The first step is to carefully construct the different connectional models to achieve the best generalization performance for classifiers. Test data is then passed through these individual models and the corresponding outputs are recorded. Suppose the classification performance given by SVM, ANN (RP), ANN (SCG) and ANN (OSS) are a_n, b_n, c_n and d_n

respectively and the corresponding desired value is x_n. Our task is to combine a_n, b_n, c_n and d_n so as to get the best output value that maximizes the classification accuracy. The following ensemble approach was used. Determine the individual absolute error differences (example, $|x_n - a_n|$) and use the output value corresponding to the lowest absolute difference $\min |a_n - x_n|, |b_n - x_n|, |c_n - x_n|, |d_n - x_n|$. The approach is depicted in Figure 2.

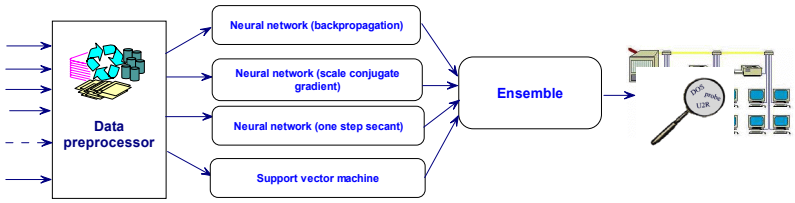

Figure 2. Ensemble approach to combine intelligent paradigms for IDS

4 Experiments

In our experiments, we perform 5-class classification. The (training and testing) data set contains 11982 randomly generated points from the five classes, with the number of data from each class proportional to its size, except that the smallest class is completely included. The normal data belongs to class1, probe belongs to class 2, denial of service belongs to class 3, user to super user belongs to class 4, remote to local belongs to class 5. A different randomly selected set of 6890 points of the total data set (11982) is used for testing different soft computing techniques.

4.1 Experiments using Neural Networks

The same data set describe in section 2 is being used for training and testing different neural network algorithms. The set of 5092 training data is divided in to five classes: normal, probe, denial of service attacks, user to super user and remote to local attacks. Where the attack is a collection of 22 different types of instances that belong to the four classes described in section 2, and the other is the normal data. In our study we used two hidden layers with 20 and 30 neurons each and the networks were trained using RP, SCG and OSS algorithms.

The network was set to train until the desired mean square error of 0.001 was met. During the training process the goal was met at 303 epochs for SCG, 66 epochs for RP and 638 epochs for OSS.

As multi-layer feed forward networks are capable of multi-class classifications, we partition the data into 5 classes (Normal, Probe, Denial of Service, and User to Root and Remote to Local).We used the same testing data (6890), same network architecture and same activations functions to identify the best training function that plays a vital role for in classifying intrusions. Table 1 summarizes the results of three different networks: network using SCG performed with an accuracy of 95.25%; network using RP achieved an accuracy of 97.04%; network using OSS performed with an accuracy of 93.60%.

Table 1: Performance of Different Neural Network Training Functions

Training algorithm	No of Epochs Trial 1	No of Epochs Trial 2	Accuracy (%) Trail 1	Accuracy (%) Trail 1
RP	67	66	97.04	95.44
SCG	351	303	80.87	95.25
OSS	638	638	93.60	93.60

Table 2: Performance of the Best Neural Network Training Algorithm (RP)

	Normal	Probe	DoS	U2Su	R2L	%
Normal	1394	5	1	0	0	99.6
Probe	49	649	2	0	0	92.7
DoS	3	101	4096	2	0	97.5
U2Su	0	1	8	12	4	48.0
R2L	0	1	6	21	535	95.0
%	96.4	85.7	99.6	34.3	99.3	

The top-left entry of Table 2 shows that 1394 of the actual “normal” test set were detected to be normal; the last column indicates that 99.6 % of the actual “normal” data points were detected correctly. In the same way, for “Probe” 649 of the actual “attack” test set were correctly detected; the last column indicates that 92.7% of the actual “Probe” data points were detected correctly. The bottom row shows that 96.4% of the test set said to be “normal” indeed were “normal” and 85.7% of the test set classified, as “probe” indeed belongs to Probe. The overall accuracy of the classification is 97.04 with a false positive rate of 2.76% and false negative rate of 0.20 %.

4.2 Experiments using Support Vector Machines

The data set described in section 4 is being used to test the performance of support vector machines. Note the same training test (5092) used for training the neural networks and the same testing test (6890) used for testing the neural networks are being used to validate the performance.

Because SVMs are only capable of binary classifications, we will need to employ five SVMs, for the 5-clas classification problem in intrusion detection, respectively. We partition the data into the two classes of "Normal" and "Rest" (Probe, DoS, U2Su, R2L) patterns, where the Rest is the collection of four classes of attack instances in the data set. The objective is to separate normal and attack patterns. We repeat this process for all classes. Training is done using the RBF (radial bias function) kernel option; an important point of the kernel function is that it defines the feature space in which the training set examples will be classified. Table 3 summarizes the results of the experiments:

Table 3: Performance of SVMs for 5 class Classifications

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

4.3 Experiments using Ensemble of Soft Computing paradigms

Different soft computing paradigms are carefully constructed to achieve the best generalization performance for classifiers. Test data is then passed through these individual models and the corresponding outputs are recorded. Table 4 summarizes the test results achieved for the five-class classification using 3 different neural networks, support vector machines and the ensemble of all the four.

Table 4: Performance Comparison of Testing for 5 class Classifications

Class	SVMs Accuracy (%)	RP Accuracy (%)	SCG Accuracy (%)	OSS Accuracy (%)	Ensemble Accuracy (%)
Normal	98.42	99.57	99.57	99.64	99.71
Probe	98.57	92.71	85.57	92.71	99.86
DoS	99.11	97.47	72.01	91.76	99.95
U2Su	64	48	0	16	76
R2L	97.33	95.02	98.22	96.80	99.64

5 Summary and Conclusions

Our research has clearly shown the importance of using ensemble approach for modeling intrusion detection systems. An ensemble helps to indirectly combine the synergistic and complementary features of the different learning paradigms without any complex hybridization. Since all the considered performance measures could be optimized such systems could be helpful in several real world applications.

- A number of observations and conclusions are drawn from the results reported:
- The ensemble approach out performs both SVMs and ANNs in the important respect of classification accuracies for all the five classes.
- If proper soft computing paradigms are chosen, their ensemble might help in gaining 100% classification accuracies.
- SVMs outperform ANNs in the important respects of scalability (SVMs can train with a larger number of patterns, while would ANNs take a long time to train or fail to converge at all when the number of patterns gets large); training time and running time (SVMs run an order of magnitude faster); and prediction accuracy.
- Resilient back propagation achieved the best performance among the neural networks in terms of accuracy (97.04 %) and training (67 epochs).

We note, however, that the difference in accuracy figures tend to be very small and may not be statistically significant, especially in view of the fact that the 5 classes of patterns differ in their sizes tremendously. More definitive conclusions can only be made after analyzing more comprehensive sets of network traffic data.

Acknowledgements

Support for this research received from ICASA (Institute for Complex Additive Systems Analysis, a division of New Mexico Tech) and a U.S. Department of Defense IASP capacity building grant is gratefully acknowledged. We would also like to acknowledge many insightful conversations with Dr. Jean-Louis Lassez and David Duggan that helped clarify some of our ideas.

References

[1] Hertz J., Krogh A., Palmer, R. G. (1991) "Introduction to the Theory of Neural Computation*," Addison –Wesley.*

[2] *Joachims T. (1998)* "*Making Large-Scale SVM Learning Practical,"* LS8-Report, University of Dortmund, LS VIII-Report.

[3] Denning D. (Feb. 1987) "An Intrusion-Detection Model," *IEEE Transactions on Software Engineering*, Vol.SE-13, No 2.
[4] Kumar S., Spafford E. H. (1994) "An Application of Pattern Matching in Intrusion Detection," *Technical Report CSD-TR-94-013*. Purdue University.
[5] Ghosh A. K. (1999). "Learning Program Behavior Profiles for Intrusion Detection," *USENIX*.
[6] Cannady J. (1998) "Artificial Neural Networks for Misuse Detection," *National Information Systems Security Conference*.
[7] Ryan J., Lin M-J., Miikkulainen R. (1998) "Intrusion Detection with Neural Networks," *Advances in Neural Information Processing Systems* 10, Cambridge, MA: MIT Press.
[8] 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.
[9] Mukkamala S., Janoski G., Sung A. H. (2002) "Intrusion Detection Using Neural Networks and Support Vector Machines," *Proceedings of IEEE International Joint Conference on Neural Networks*, pp.1702-1707.
[10] http://kdd.ics.uci.edu/databases/kddcup99/task.htm.
[11] J. Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Philip K. Chan "Cost-based Modeling and Evaluation for Data Mining With Application to Fraud and Intrusion Detection," *Results from the JAM Project by Salvatore.*
[12] Riedmiller, M., and H. Braun, "A direct adaptive method for faster back propagation learning: The RPROP algorithm," *Proceedings of the IEEE International Conference on Neural Networks*, San Francisco, 1993.
[13] Moller A F, A Scaled Conjugate Gradient Algorithm for Fast Supervised Learning, Neural Networks, Volume (6), pp. 525-533, 1993.
[14] Bishop C. M, *Neural Networks for pattern recognition*, Oxford Press, 1995.
[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] Vapnik V. The Nature of Statistical Learning Theory. Springer-Verlag, New York, 1995.
[17] Hashem, S., Optimal Linear Combination of Neural Networks, Neural Network, Volume 10, No. 3. pp. 792-994, 1995.