Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Network and Computer Applications 28 (2005) 167–182

Journal of NETWORK and COMPUTER APPLICATIONS

www.elsevier.com/locate/jnca

Intrusion detection using an ensemble of intelligent paradigms

Srinivas Mukkamala[a,*], Andrew H. Sung[a], Ajith Abraham[b]

[a]*Department of Computer Science, New Mexico Tech, Socorro, NM 87801, USA*
[b]*Department of Computer Science, Oklahoma State University, Tulsa, OK, USA*

Received 7 January 2004; accepted 7 January 2004

Abstract

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

Keywords: Computer security; Support vector machines; Network security

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 (ANNs) (Hertz et al., 1991), Support Vector Machines (SVMs) (Joachims, 1998), Multivariate Adaptive Regression Splines (MARS) (Friedman, 1991) and the ensemble of different soft computing techniques. Since the ability of a good detection technique gives more accurate results, it is

* Corresponding author.
E-mail address: srinivas@cs.nmt.edu (S. Mukkamala).

1084-8045/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jnca.2004.01.003

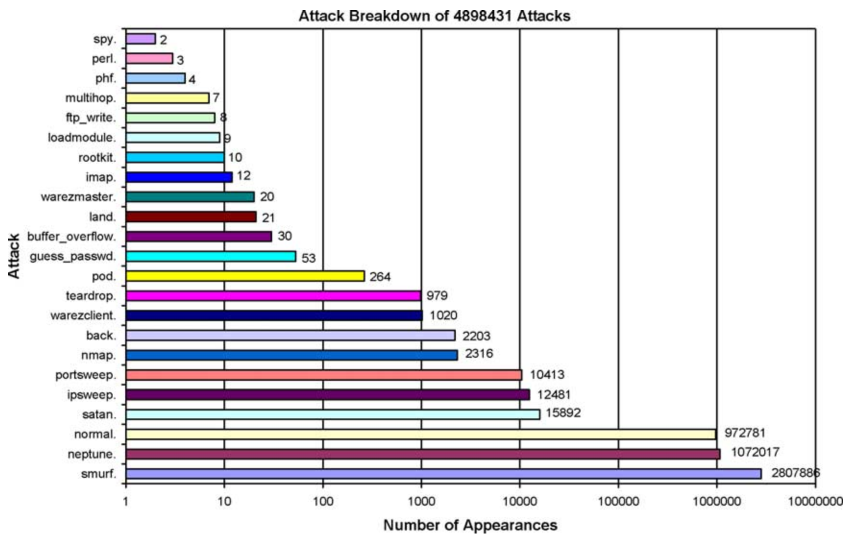

Fig. 1. Intrusion detection data distribution.

critical for intrusion detection in order for the IDS to achieve maximal performance. Therefore, we study different intelligent computing techniques and also their ensemble for building models based on DARPA intrusion detection data shown in Fig. 1.

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 that can classify a user's activity as normal or abnormal (attack). Several intelligent techniques including but not limited to ANNs, SVMs, Petri nets, and data mining techniques are being used to build IDSs. In our recent work, SVMs are found to be superior to ANNs in many important respects of intrusion detection (Mukkamala et al., 2002). In this paper, we will concentrate on using the ensemble of SVMs, MARS and ANNs 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 IDS evaluations by DARPA and is considered a benchmark for intrusion detection evaluations (Kendall, 1998; Webster, 1998).

We performed experiments to classify each of the five classes (normal, probe, denial of service (DoS), 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 related work in the field of intrusion detection is given in Section 2. A brief introduction to the data we used is given in Section 3. In Section 4 we describe the theoretical aspects of ANNs, SVMs, and MARS. Ensemble of soft computing techniques is described in Section 5. In Section 6 we present the experimental results of ANNs, SVMs, MARS and their ensemble. In Section 7, we summarize our results.

2. Related work

Identifying unauthorized use, misuse and attacks on information systems is defined as intrusion detection (Denning, 1987; Kumar and Spafford, 1994a). The most popular way to detect intrusions has been done by using audit data generated by operating systems and by networks. Since almost all activities are logged on a system, it is possible that a manual inspection of these logs would allow intrusions to be detected. It is important to analyze the audit data even after an attack has occurred, for determining the extent of damage occurred, this analysis helps in attack trace back and also helps in recording the attack patterns for future prevention of such attacks. An IDS can be used to analyze audit data for such insights. This makes IDS a valuable real-time detection and prevention tool as well as a forensic analysis tool.

Soft computing techniques are being widely used by the IDS community due to their generalization capabilities that help in detecting know intrusions and unknown intrusions or the attacks that have no previously described patterns. Earlier studies have utilized a rule-based approach for intrusion detection, but had a difficulty in identifying new attacks or attacks that had no previously describe patterns (Lunt et al., 1992; Ilgun, 1993; Anderson et al., 1995; Porras and Neumann, 1997). Lately the emphasis is being shifted to learning by examples and data mining paradigms. Neural networks have been extensively used to identify both misuse and anomalous patterns (Debar and Dorizzi, 1992; Debar et al., 1992; Ryan et al., 1997; Cannady, 1998; Mukkamala et al., 2001). Recently kernel based methods, SVMs and their variants are being proposed to detect intrusions. Several researchers proposed data mining techniques to identify key patterns that help in detecting intrusions (Stolfo et al., 2000; Jianxiong and Bridges, 2000; Mukkamala and Sung, 2003). Distributed agent technology is being proposed by a few researchers to overcome the inherent limitations of the client-server paradigm and to detect intrusions in real time (Crosbie and Spafford, 1995; Prodromidis and Stolfo, 1999; Dasgupta, 1999; Helmer et al., 2003).

2.1. Misuse detection

The idea of misuse detection is to represent attacks in the form of a pattern or a signature so that the same attack can be detected and prevented in future. These systems can detect many or all known attack patterns (Kumar and Spafford, 1994b), but they are of little use for detecting naive attack methods. The main issues of misuse detection is how to build signatures that include possible signatures of attacks build a signature that includes all possible variations of the pertinent attack to avoid false negatives, and how to build signatures that do not match non-intrusive activities to avoid false positives.

2.2. Anomaly detection

The idea here is that if we can establish a normal activity profile for a system, in theory we can flag all system states varying from the established profile as intrusion attempts. However, if the set of intrusive activities is not identical to the set of anomalous activities, the situation becomes more interesting instead of being exactly the same, we find few interesting possibilities. Anomalous activities that are not intrusive are flagged as intrusive, though they are false positives. Actual intrusive activities that go undetected are called false negatives. This is a serious issue, and is far more serious than the problem of false positives. One of the main issues of anomaly detection systems is the selection of threshold levels so that neither of the above problems is unreasonably magnified. Anomaly detection is usually computationally expensive because of the overhead of keeping track of and possibly updating several system profiles. Recent proposed system Learning Rules for Anomaly Detection (LEARD) discovers relationships among attributes in order to model application protocols (Mahoney and Chan, 2003; Chan et al., 2003).

3. 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 US 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 (Lee and Stolfo, 2000). Of this database a subset of 494021 data were used, of which 20% represent normal patterns. The four different categories of attack patterns are as follows.

3.1. Probing

Probing is a class of attacks where an attacker scans a network to gather information or find known vulnerabilities. An attacker with a map of machines and services that are available on a network can use the information to look for exploits. There are different types of probes: some of them abuse the computer's legitimate features; some of them use social engineering techniques. This class of attacks is the most commonly heard and requires very little technical expertise. Different types of probe attacks are shown in Table 1.

3.2. Denial of service attacks

DoS is a class of attacks where an attacker makes some computing or memory resource too busy or too full to handle legitimate requests, thus denying legitimate users access to a machine There are different ways to launch DoS attacks: by abusing the computers

Table 1
Probe attacks

Attack type	Service	Mechanism	Effect of the attack
Ipsweep	Icmp	Abuse of feature	Identifies active machines
Mscan	Many	Abuse of feature	Looks for known vulnerabilities
Nmap	Many	Abuse of feature	Identifies active ports on a machine
Saint	Many	Abuse of feature	Looks for known vulnerabilities
Satan	Many	Abuse of feature	Looks for known Vulnerabilities

Table 2
DoS attacks

Attack type	Service	Mechanism	Effect of the attack
Apache2	http	Abuse	Crashes httpd
Back	http	Abuse/Bug	Slows down server response
Land	http	Bug	Freezes the machine
Mail bomb	N/A	Abuse	Annoyance
SYN flood	TCP	Abuse	Denies service on one or more ports
Ping of death	Icmp	Bug	None
Process table	TCP	Abuse	Denies new processes
Smurf	Icmp	Abuse	Slows down the network
Syslogd	Syslog	Bug	Kills the Syslogd
Teardrop	N/A	Bug	Reboots the machine
Udpstrom	Echo/Chargen	Abuse	Slows down the network

legitimate features; by targeting the implementations bugs; or by exploiting the system's misconfigurations. DoS attacks are classified based on the services that an attacker renders unavailable to legitimate users. Some of the popular attack types are shown in Table 2.

3.3. User to root attacks

User to root exploits are a class of attacks where 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. Most common exploits in this class of attacks are regular buffer overflows, which are caused by regular programming mistakes and environment assumptions. Please refer to Table 3 for some of the attack types in this category.

3.4. Remote to user attacks

A remote to user (R2L) attack is a class of attacks where an attacker sends packets to a machine over a network, then exploits machine's vulnerability to illegally gain local access as a user. There are different types of R2U attacks: the most common attack in this class is done using social engineering. Some of the R2U attacks are presented in Table 4.

Table 3
User to super-user attacks

Attack Type	Service	Mechanism	Effect of the attack
Eject	User session	Buffer overflow	Gains root shell
Ffbconfig	User session	Buffer overflow	Gains root shell
Fdformat	User session	Buffer overflow	Gains root shell
Loadmodule	User session	Poor environment sanitation	Gains root shell
Perl	User session	Poor environment sanitation	Gains root shell
Ps	User session	Poor temp file management	Gains root shell
Xterm	User session	Buffer overflow	Gains root shell

Table 4
Remote to user attacks

Attack type	Service	Mechanism	Effect of the attack
Dictionary	Telnet, rlogin, pop, ftp, imap	Abuse feature	Gains user access
Ftp-write	Ftp	Misconfig.	Gains user access
Guest	Telnet, rlogin	Misconfig.	Gains user access
Imap	Imap	Bug	Gains root access
Named	Dns	Bug	Gains root access
Phf	Http	Bug	Executes commands as http user
Sendmail	Smtp	Bug	Executes commands as root
Xlock	Smtp	Misconfig.	Spoof user to obtain password
Xnsoop	Smtp	Misconfig.	Monitor key stokes remotely

4. Soft computing and hard computing paradigms

Soft computing was first proposed by Zadeh to construct new generation computationally intelligent hybrid systems consisting of neural networks, fuzzy inference system, approximate reasoning and derivative free optimization techniques. It is well known that the intelligent systems, which can provide human like expertise such as domain knowledge, uncertain reasoning, and adaptation to a noisy and time varying environment, are important in tackling practical computing problems. In contrast with conventional Artificial Intelligence (AI) techniques, which only deal with precision, certainty and rigor the guiding principle of hybrid systems is to exploit the tolerance for imprecision, uncertainty, low solution cost, robustness, partial truth to achieve tractability, and better rapport with reality.

4.1. Artificial neural networks

The ANN methodology enables us to design useful non-linear systems accepting large numbers of inputs, with the design based solely on instances of input–output relationships. ANN (in the present context, multilayer, feedforward type networks) consists of a collection of highly interconnected processing elements to perform an input–output transformation. The actual transformation is determined by the set of weights associated with the links connecting elements. The neural network gains knowledge about the transformation to be performed by iteratively learning from a sufficient training set of samples or input–output training pairs. A well-trained network can perform the transformation correctly and also possess some generalization capability.

Since multilayer feedforward ANNs are capable of making multiclass classifications, an ANN is employed to perform the intrusion detection, using the same training and testing sets as those for other connectionist paradigms. This Section briefly introduces to three best-performed neural network training algorithms for intrusion detection (Demuth and Beale, 2000).

4.1.1. Resilient back propagation (RP)

The purpose of the resilient backpropagation 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 (Riedmiller and Braun, 1993).

4.1.2. Scaled conjugate gradient algorithm (SCG)

Moller (1993)) introduced the SCG 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

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

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 Moller (1993).

4.1.3. 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 (Bishop, 1995), the updated expression is as follows:

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

where

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

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 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 (Bishop, 1995).

4.2. Support vector machines

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 (Joachims, 2000). Training a 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 (Vapnik, 1995).

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

Subject to

$$\sum_{i=1}^{l} y_i \alpha_i \qquad \forall i : 0 \le \alpha_i \le C \tag{5}$$

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 Eq. (4) is the vector α^* for which Eq. (4) is minimized and Eq. (5) is fulfilled.

4.3. Multivariate adaptive regression splines

Splines can be considered as an innovative mathematical process for complicated curve drawings and function approximation. To develop a spline the X-axis is broken into a convenient number of regions. The boundary between regions is also known as a knot. With a sufficiently large number of knots, virtually any shape can be well approximated. While it is easy to draw a spline in two-dimensions by keying on knot locations (approximating using linear, quadratic or cubic polynomial, etc.), manipulating the mathematics in higher dimensions is best accomplished using basis functions. The MARS model is a regression model using basis functions as predictors in place of the original data. The basis function transform makes it possible to selectively blank out certain regions of a variable by making them zero, and allows MARS to focus on specific sub-regions of the data. It excels at finding optimal variable transformations and interactions, and the complex data structure that often hides in high-dimensional data (Friedman, 1991).

Given the number of records in most data sets, it is infeasible to approximate the function $y = f(x)$ by summarizing y in each distinct region of x. For some variables, two regions may not be enough to track the specifics of the function. If the relationship of y to some x's is different in three or four regions, for example, the number of regions requiring examination is even larger than 34 billion with only 35 variables. Given that the number of regions cannot be specified a priori, specifying too few regions in advance can have serious implications for the final model. A solution is needed that accomplishes the following two criteria:

- judicious selection of which regions to look at and their boundaries
- judicious determination of how many intervals are needed for each variable

Given these two criteria, a successful method will essentially need to be adaptive to the characteristics of the data. Such a solution will probably ignore quite a few variables (affecting variable selection) and will take into account only a few variables at a time (also reducing the number of regions). Even if the method selects 30 variables for the model, it will not look at all 30 simultaneously. Such simplification is accomplished by a decision tree at a single node, only ancestor splits are being considered; thus, at a depth of six levels in the tree, only six variables are being used to define the node.

4.3.1. MARS smoothing, splines, knots selection and basis functions

To estimate the most common form, the cubic spline, a uniform grid is placed on the predictors and a reasonable number of knots are selected. A cubic regression is then fit within each region. This approach, popular with physicists and engineers who want continuous second derivatives, requires many coefficients (four per region) to be estimated. Normally, two constraints, which dramatically reduce the number of free parameters, can be placed on cubic splines:

- curve segments must join,
- continuous first and second derivatives at knots (higher degree of smoothness)

Fig. 2 depicts a MARS spline with three knots. A key concept underlying the spline is the knot. A knot marks the end of one region of data and the beginning of another. Thus, the knot is where the behavior of the function changes. Between knots, the model could be global (e.g. linear regression). In a classical spline, the knots are predetermined and evenly spaced, whereas in MARS, the knots are determined by a search procedure. Only as many knots as needed are included in a MARS model. If a straight line is a good fit, there will be

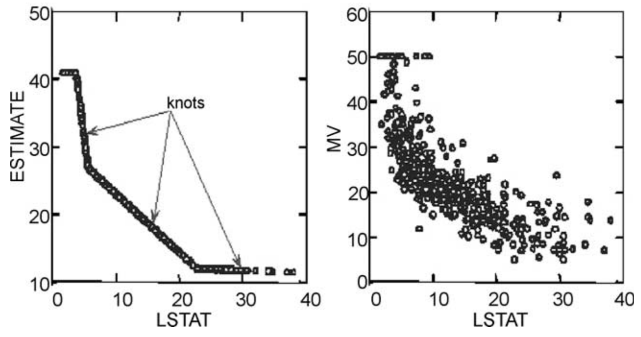

Fig. 2. MARS data estimation using spines and knots (actual data on the right).

no interior knots. In MARS, however, there is always at least one ‘pseudo’ knot that corresponds to the smallest observed value of the predictor (Steinberg et al., 1999).

Finding the one best knot in a simple regression is a straightforward search problem: simply examine a large number of potential knots and choose the one with the best R^2. However, finding the best pair of knots requires far more computation, and finding the best set of knots when the actual number needed is unknown is an even more challenging task. MARS finds the location and number of needed knots in a forward/backward stepwise fashion. A model which is clearly overfit with too many knots is generated first; then, those knots that contribute least to the overall fit are removed. Thus, the forward knot selection will include many incorrect knot locations, but these erroneous knots will eventually (although this is not guaranteed), be deleted from the model in the backwards pruning step.

5. Ensemble of intelligent paradigms

Optimal linear combination of neural networks has been investigated and has found to be very useful (Hashem, 1995). 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, MARS, ANN (RP), ANN (SCG) and ANN (OSS) are a_n, b_n, c_n, d_n and e_n, respectively, and the corresponding desired value is x_n. Our task is to combine a_n, b_n, c_n, d_n, and e_n so as to get the best output value that maximizes the classification accuracy. The following ensemble approach was used. We used the majority voting approach in which the detected class is the one where the majority of networks agreed. The approach is depicted in Fig. 3.

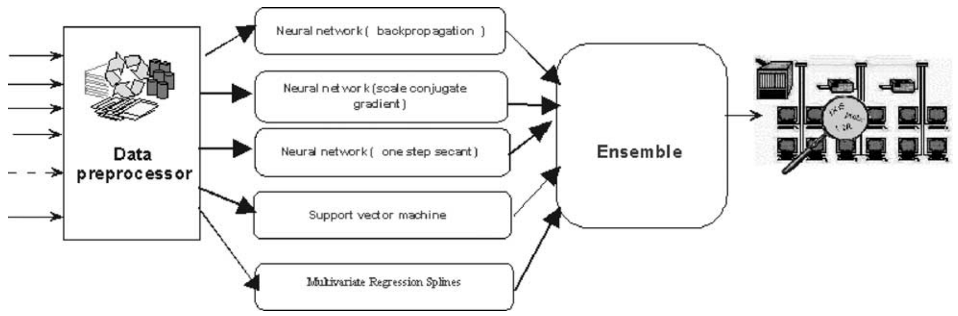
Data preprocessor
Neural network (backpropagation)
Neural network (scale conjugate gradient)
Neural network (one step secant)
Support vector machine
Multivariate Regression Splines
Ensemble

Fig. 3. Ensemble approach to combine intelligent paradigms for IDS.

6. Experiments

In our experiments, we perform five-class classification. The (training and testing) data set contains 11,982 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 class 1, probe belongs to class 2, DoS 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 (11,982) is used for testing different intelligent techniques.

6.1. Experiments using neural networks

The same data set described 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, DoS 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 multilayer feedforward networks are capable of multiclass classifications, we partition the data into five classes (Normal, Probe, DoS, and User to Root and Remote to Local). 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%.

The top-left entry of Table 5 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%.

Table 5
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	

Table 6
Performance of SVMs for five-class classifications

Class	Training time (s)	Testing time (s)	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

6.2. Experiments using support vector machines

The data set described in Section 4 is being used to test the performance of SVMs. 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 five-class 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 radial bias function (RBF) 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 6 summarizes the results of the experiments.

6.3. Experiments using MARS

We use five basis functions and selected a setting of minimum observation between knots as 10. The MARS training mode is being set to the lowest level to gain higher accuracy rates. Five MARS models are employed to perform five-class classifications (normal, probe, DoS, user to root and remote to local). 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. Table 7 summarizes the results of MARS.

Table 7
Performance of MARS on test dataset

Class	Accuracy (%)
Normal	96.08
Probe	92.32
DoS	94.73
U2Su	99.71
R2L	99.48

Table 8
Performance comparison of testing for five-class classifications

Class	Accuracy (%)						
	SVM	RP	SCG	OSS	Ensemble of ANN	MARS	Ensemble of ANN, SVM and MARS
Normal	98.42	99.57	99.57	99.64	99.64	99.71	99.71
Probe	98.57	92.71	85.57	92.71	98.14	56.42	99.85
DoS	99.45	97.47	72.01	91.78	99.61	96	99.97
U2Su	64.00	48.00	0.00	16.00	56.00	40.00	76.00
R2L	97.33	95.73	98.57	97.15	99.46	98.75	100.00
Overall	98.85	97.09	80.89	93.64	99.3	92.75	99.82

6.4. Experiments using ensemble of intelligent paradigms

Different intelligent 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 8 summarizes the test results achieved for the five-class classification using three different neural networks, SVMs, multivariate regression splines and the ensemble of all the different paradigms.

7. Summary and conclusions

Our research has clearly shown the importance of using ensemble approach for modeling IDSs. 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, MARS and ANNs in the important respect of classification accuracies for all the five classes.
- If proper intelligent paradigms are chosen, their ensemble might help in gaining 100% classification accuracies.
- SVMs outperform MARS and 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 five 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 US Department of Defense and NSF IASP capacity building grant is gratefully acknowledged.

References

Anderson D, Lunt TF, Javitz H, Tamaru A,, Valdes A. Detecting unusual program behavior using the stastistical component of the next-generation intrusion detection expert system (NIDES). SRI-CSL-95-06, Menlo Park, CA: SRI International; 1995.

Bishop CM. Neural networks for pattern recognition. Oxford Press; 1995.

Cannady J. Artificial neural networks for misuse detection. National Information Systems Security Conference; 1998. p. 368–81.

Chan PK, Mahoney M, Arshad M. Learning rules and clusters for anomaly detection in network traffic. Managing cyber threats: issues, approaches and challenges, Dordrecht: Kluwer; 2004. In press.

Crosbie M, Spafford EH. Defending a computer system using autonomous agents. Technical Report CSD-TR-95-022; 1995.

Dasgupta D. Immunity-based intrusion detection system: a general framework. Proceedings of 22nd National Information Systems Security Conference (NISSC); 1999. p. 147–60.

Debar H, Dorizzi B. An application of a recurrent network to an intrusion detection system. Proceedings of the International Joint Conference on Neural Networks; 1992. p. 78–83.

Debar H, Becke B, Siboni D. A neural network component for an intrusion detection system. Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy; 1992. p. 240–50.

Demuth H, Beale M. Neural network toolbox user's guide. Natick, MA: MathWorks, Inc; 2000.

Denning D. An intrusion-detection model. IEEE Trans Software Engng 1987;SE-13(2):222–32.

Friedman JH. Multivariate adaptive regression splines. Anal Stat 1991;19:1–141.

Hashem S. Optimal linear combination of neural networks. Neural Network 1995;10(3):792–994.

Helmer G, Wong J, Honavar V, Miller L. Lightweight agents for intrusion detection. J Syst Software 2003; 109–22.

Hertz J, Krogh A, Palmer RG. Introduction to the theory of neural computation. Reading, MA: Addison-Wesley; 1991.

Ilgun K. USTAT: a real-time intrusion detection system for UNIX. Proceedings of the 1993 Computer Society Symposium on Research in Security and Privacy, Oakland, California, May 24–26, Los Alamitos, CA: IEEE Computer Society Press; 1993. pp. 16–29.

Jianxiong L, Bridges SM. Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection. Int J Intell Syst 2000;15(8):687–704.

Joachims T. Making large-scale SVM learning practical. LS8-Report, University of Dortmund; 1998.

Joachims T. SVMlight is an implementation of support vector machines (SVMs) in C. University of Dortmund, Collaborative Research Center on Complexity Reduction in Multivariate Data (SFB475); 2000, http://ais.gmd.de/~thorsten/svm_light.

Kendall K. A Database of computer attacks for the evaluation of intrusion detection systems. Master's Thesis. Massachusetts Institute of Technology; 1998.

Kumar S, Spafford EH. An application of pattern matching in intrusion detection. Technical Report CSD-TR-94-013, Purdue University; 1994a.

Kumar S, Spafford EH. A pattern matching model for misuse intrusion detection. In Proceedings of the 17th National Computer Security Conference; 1994b. p. 11–21.

Lee W, Stolfo SJ. A Framework for constructing features and models for intrusion detection systems. ACM Trans Inf Syst Security 2000;3(4):227–61.

Lunt T, Tamaru A, Gilham F, Jagannathan R, Jalali C, Neumann PG, Javitz HS, Valdes A, Garvey TD. A real time intrusion detection expert system (IDES)—Final Report. Menlo Park, CA: SRI International; 1992.

Mahoney M, Chan PK. An analysis of the 1999 DARPA/Lincoln laboratory evaluation data for network anomaly detection. Sixth International Symposium on Recent Advances in Intrusion Detection; 2003. p. 220–37.

Moller AF. A scaled conjugate gradient algorithm for fast supervised learning. Neural Networks 1993;(6): 525–33.

Mukkamala S, Sung AH. Feature selection for intrusion detection using neural networks and support vector machines. J Transport Res Board Natl Acad, Transport Res Record No 1822 2003;33–9.

Mukkamala S, Janowski G, Sung AH. Intrusion detection using neural networks and support vector machines. Proceedings of Hybrid Information Systems Advances in Soft Computing, Heidelberg: Physica/Springer; 2001. ISBN 3790814806, p.121–38.

Mukkamala S, Janoski G, Sung AH. Intrusion detection using neural networks and support vector machines. Proceedings of IEEE International Joint Conference on Neural Networks; 2002. p. 1702–07.

Porras A, Neumann PG. EMERALD: Event monitoring enabling responses to anomalous live disturbances. In: Proceedings of the National Information Systems Security Conference; 1997. p. 353–65.

Prodromidis L, Stolfo SJ. Agent-based distributed learning applied to fraud detection. Technical Report CUCS-014-99; 1999.

Riedmiller M, Braun H. A direct adaptive method for faster back propagation learning: the RPROP algorithm. Proceedings of the IEEE International Conference on Neural Networks, San Francisco; 1993.

Ryan J, Lin M-J, Miikkulainen R. Intrusion detection with neural networks. Advances in neural information processing systems 10, Cambridge, MA: MIT Press; 1997.

Steinberg D, Colla PL, Kerry M. MARS user guide. San Diego, CA: Salford Systems; 1999.

Stolfo J, Fan W, Lee W, Prodromidis A, Chan PK. Cost-based modeling and evaluation for data mining with application to fraud and intrusion detection. Results from the JAM Project by Salvatore; 2000.

Vapnik V. The nature of statistical learning theory. New York: Springer; 1995.

Webster SE. The development and analysis of intrusion detection algorithms. S.M. Thesis. Massachusetts Institute of Technology; 1998.

Zadeh LA. Roles of soft computing and fuzzy logic in the conception, design and deployment of information/intelligent systems. In: Kaynak O, Zadeh LA, Turksen B, Rudas IJ, editors. Computational intelligence: soft computing and fuzzy-neuro integration with applications; 1999. p. 1–9.

Srinivas Mukkamala is currently a doctoral candidate of the Computer Science Department of New Mexico Tech. He is currently working in the areas of information assurance and security, applications of soft computing and software security and has over 40 publications in the areas of information security. Srinivas Mukkamala received his B.E. in Computer Science and Engineering from University of Madras in 1999, M.S. in Computer Science form New Mexico Tech. He is currently a research associate and a student lead of the information assurance group at New Mexico Tech.

Andrew H. Sung is currently Professor and Chairman of the Computer Science Department of New Mexico Tech, and a founding coordinator of the school's new Information Technology Program. He is also the Associate Director for Education and Training of ICASA (Institute for Complex Additive Systems Analysis, a statutory research division of New Mexico Tech performing work on information technology, information assurance, and analysis and protection of critical infrastructures as complex interdependent systems). Andrew Sung received his B.S. in Electrical Engineering from National Taiwan University in 1976, M.S. in Mathematical Sciences from the University of Texas at Dallas in 1980, and Ph.D. in Computer Science from the State University of New York at Stony Brook in 1984. He joined New Mexico Tech in 1987, and served as the Computer Science department chair from 1988 to 1993, and again since January 2000. From 1984 to 1987, he was Assistant Professor of Computer Science at the University of Texas at Dallas.