A STUDY ON CLASSIFICATION TECHNIQUES FOR NETWORK INTRUSION DETECTION

Sophia Kaplantzis
Electrical & Comp Systems Eng
Monash University Australia
sophia.kaplantzis@eng.monash.edu.au

Nallasamy Mani
Electrical & Comp Systems Eng
Monash University Australia
nallasamy.mani@eng.monash.edu.au

ABSTRACT
Computer systems vulnerabilities such as software bugs are often exploited by malicious users to intrude into information systems. With the recent growth of the Internet such security limitations are becoming more and more pressing. One commonly used defense measure against such malicious attacks in the Internet are Intrusion Detection Systems (IDSs).

In this paper, we compare the ability of three classification techniques (k-means classifiers, neural networks and support vector machines) to perform for network intrusion detection applications. The results indicate that Support Vector Machines train in the shortest amount of time with an acceptable accuracy whilst Neural Networks exhibit high accuracy at the cost of long training times.

KEY WORDS
Artificial neural networks, K-means Classifier, Network intrusion detection, Support vector machines.

1 Introduction

The need for network security today has become a pressing issue. The dependence of the modern world on the cyber domain is apparent now more than ever, with many companies relying solely on web services as their major source of information and revenue. The modern internet user engages in a number of applications that require secure transactions when transferring information and funds, whether that is everyday internet banking, shopping or attaining access to password protected websites. As the number of secure transactions increase so does the number of sophisticated attacks on systems that store precious corporate and personal data. Symantec in a recent report [1] uncovered that the number of phishing attacks targeted at stealing confidential information such as credit card numbers, passwords and other financial information are on the rise, going from 9 million attacks in June 2004 to over 33 million in less than a year.

One solution to such problems may be to redesign protocols from scratch with security as the main driver, in order to prevent such attacks from happening in the first place. However, revamping the core of the internet is not likely to be feasible as there is an inadequacy of technical and economic resources which will allow for the alleviation of the vulnerabilities that hackers enjoy exploiting.

Another solution that has dominated the world of network security is the use of Intrusion Detection Systems (IDS) that identify attacks while they are occurring on a network, in an attempt to defend the network. It is therefore crucial that such systems are accurate in identifying attacks, quick to train and generate as few false positive alarms as possible.

In this paper, we will be presenting a comparative study of three classification techniques, in order to find the best performing classifier in terms of speed and accuracy for an intrusion detection system using pattern matching. These three techniques are namely: i) K-means nearest neighbour classifiers ii) Artificial Neural Networks iii) Support Vector Machines

The rest of the paper is structured as follows: *Section 2* gives a background into common intrusion detection methods. *Section 3* describes the four most common types of network attacks, which the classifiers will be required to identify. *Section 4* gives the basic principles of the three classifiers and their mathematical formulations. *Section 5* discusses how the classifiers were designed and trained to recognise network attacks. *Section 6* provides a summary of the results obtained and their significance for intrusion detection systems. *Section 7* concludes the paper with an insight into the future work of the authors.

2 Intrusion Detection Methods

This section discusses the two most common intrusion detection models used in network security today: misuse detection (a.k.a. pattern matching) and anomaly detection [2].

Misuse detection entails identifying and storing signatures of known intrusions and then matching the activities occurring on an information system to these signatures, in order to detect whether the system is undergoing an attack. The benefits of this technique are that the signatures are based on well known intrusive activity and hence the attacks detected are well defined. Other benefits include the simplicity of these systems and the ability to detect attacks immediately after installation. In contrast, the major drawback of misuse detection systems is the ability to manage state information effectively by updating the attack signatures as new attacks are published. These systems cannot detect unpublished attacks and are prone to circumventing false negative alarms. Also the cost of generating signatures for all known attacks is very costly.

Anomaly detection establishes a profile of the subject's normal activities (norm profile) and then compares activities on the information system to this norm behaviour. It then signals an intrusion when the observed activities, differ largely from those usually undertaken by the user. The major benefit of such a technique is that they can detect attacks that are unpublished; however such systems are complex and resource hungry as they are constantly generating logs and checking audit files. In this paper, we will be reviewing the performance of classifiers when trained to identify signatures of specific attacks. These attacks are highlighted in more detail in the following section.

3 Networking Attacks

This section is an overview of the 4 major categories of networking attacks. Every attack on a network can comfortably be placed into one of these groupings [3].

Denial of Service (DoS): A DoS attacks is a type of attack in which the hacker makes a computing or memory resources too busy or too full to serve legitimate networking requests and hence denying users access to a machine e.g. apache, smurf, Neptune, ping of death, back, mail bomb, UDP storm etc. are all DoS attacks.

Remote to User attacks (R2L): A remote to user attack is an attack in which a user sends packets to a machine over the internet, which she or he does not have access to in order to expose the machines vulnerabilities and exploit privileges which a local user would have on the computer e.g. xlock, guest, xnsnoop, phf, sendmail dictionary etc

User to Root Attacks (U2R): These attacks are exploitations in which the hacker starts off on the system with a normal user account and attempts to abuse vulnerabilities in the system in order to gain super user privileges e.g. perl, xterm

Probing: Probing is an attack in which the hacker scans a machine or a networking device in order to determine weaknesses or vulnerabilities that may later be exploited so as to compromise the system. This technique is commonly used in data mining e.g. saint, portsweep, mscan, nmap etc. The classifiers in our research are required to identify attacks from all four groups as accurately as possible.

4 Classification Methods

Classifiers are tools that partition sets of data into different classifications on the basis of specified features in that data. In this section we give an introduction into the classification techniques and the mathematical representations used for network intrusion detection in our research.

4.1 K-Means classifiers:

The K-means nearest neighbour classifier is a simple and popular classifier that uses statistical properties and distance measures to cluster information into specific groupings. Although this method has been around since 1967, nearest neighbour techniques are still very competitive, grunting it out against some of the more modern and sophisticated models [4]. The main idea of this technique is to define k centroids or means one for each cluster (in the context of this paper we use 2 clusters, one for attacks and one for normal sessions). These centroids are associated with a training set initially; hence in our case the means of the attack data and the normal data can be obtained [5]. The classification is then performed by taking each individual point in a test set and associating it with the nearest centroid. This results in each point being assigned to a certain cluster and hence having been classified. The most common distance measures used in such clustering algorithms are the Euclidean and Manhattan. Mathematically, we can consider the k-means training algorithm as follows:

1. Consider D to be the training set consisting of m vectors, with x_i being a feature vector, y_i being the label for that feature vector and N being the dimension of the problem (i.e. the number of features in a vector) then:
   $D = \{(x_1, y_1), (x_2, y_2) \ldots (x_m, y_m)\}$
   where $x_i \in \Re^N$ and $y_i = \{l_1, l_2, \ldots, l_k\}$

2. It is then necessary to split the training data into k categories according to their label (i.e. all feature vectors with label l_1 are to form one grouping; all feature vectors with label l_2 are to form another grouping etc.).

3. The next step is simply to calculate the mean of each grouping, hence the "k means". This is done by applying the mean statistic for multiple dimensions. $Mean = 1/m * x_{ij}$ where $(i = 1, 2, \ldots, N)$ and $(j = 1, 2, \ldots, m)$

4. Once these means have been calculated and stored, all that needs to be done, is for every point x_i in the test set T, to determine which cluster it belongs to by comparing the distance from the point x$_i$ to every mean in the feature space. $T = x_1, x_2, \ldots, x_n$

   The shortest distance measure then determines the label for the test vector x_i.

The properties of this algorithm are that there will always be k clusters with at least one item in each and that these clusters will never overlap. Each item in a cluster is closest, distance wise, to that cluster's centroid than any other cluster near it.

4.2 Artificial Neural Networks

Neural networks are a uniquely powerful tool in multiple class classification, especially when used in such applications where formal analysis would be very difficult or even impossible, such as pattern recognition, nonlinear system identification and control.

Provided the neural network has been given sufficient time to train, the property of generalisation ensures that the network will be able to classify patterns that have never been seen before. The accuracy however of such classifications depends on a variety of parameters, ranging from the architecture of the actual neural network to the training algorithm of choice.

Neural networks can be thought of simply as weighted directed graphs, with the nodes of the graph representing neurons that can either have an ”on” or an ”off” status. At each time instant all the ”on” nodes send an impulse along their outgoing arcs to their neighbour nodes. Subsequently, all nodes sum up their incoming inputs weighted according to each input arc, and if this sum exceeds a predefined threshold value, the nodes turn ”on” at the next time instant else they adopt the ’off” status. The outputs at the final stage of the network can then be read out to determine what class the input data belongs to. This procedure is repeated until the desired outcome is achieved. In more detail, each node computes some function f of the weighted sum of its inputs, which is referred to as the net input, with w_{ij} being the weight from neuron j to neuron i and b_i representing the bias value of the ith neuron. The function f is also known as the neurons activation function.

$Net = f(\Sigma_j w_{ij} x_j + b_i)$,
where $y_i = +1$, if $Net > 0$ ”on”
and $y_i = -1$, if $Net \leq 0$ ”off”

Neural networks use Hebbian learning to train. This is a method of weight adjustment based on the so called delta rule, which is used for adjusting the error amount between the actual and desired output to a minimum [6]. In overview, a neural network training algorithm can be thought of by following 3 steps, given in pseudo code manner:

1. Initialise the weights and biases to either zero or some small number.
2. Pick the learning rate between zero and one.
3. Until the stopping condition is met (i.e. the weights don’t change, a certain time limit has elapsed or a certain tolerance level has been met).

```
For each neuron do:
Calculate the output activation $y_i$
If $y_i = label_i$ don't change weights
Else if $y_i != label_i$ update the weights:
$w_i(new) = w_i(old) + (label_i - y) * x, \forall$ t
```

This algorithmic model is used, to produce feed-forward networks that can classify TCP data.

4.3 Support Vector Machines

The support vector machine is a binary classifier that can be applied to large data sets for pattern recognition. Similar to neural networks, support vector machines belong to the category of supervised learning. Being developed in 1995 by Vapnik and co-workers, the support vector machine is a relatively new machine learning formulation [7]. In general the support vector machine is built on linear discriminant Analysis and the Fisher discriminant [8] with the extension of mapping input data space to a higher dimension feature space, in order to increase the available degrees of freedom and hence the separability of classes. The generalisation capability of the support vector machine is based on the Structural Risk Minimisation principle [8], which states that no matter what the sample training state there is always an upper bound on the error of misclassification. Consider the training set D once again, including m feature vectors x_i of dimension N with respective labels y_i.

$D = \{(x_1, y_1), (x_2, y_2) \ldots (x_m, y_m)\}$
where $x_i \in \Re^N$ and $y_i = \{-1, 1\}$

It is necessary to define a non-linear mapping that transforms the input space to a higher dimensional space. This mapping is denoted by the symbol and hence the generic formula for a support vector machine:

$f(x) = \Sigma_{(i=1:m)} w_i \varphi_i(x) + b$

In order to find the boundary between the -1 and +1 classes it is necessary to solve the equation $f(x) = 0$. Note that the number of such lines is infinite. The points of interest in this model are the points closest to the separating line also known as support vectors.

In order to derive the best possible classifier, the idea is to maximise the distance of the support vectors from the separating line i.e. increase the separability between the classes. This is done by introducing the parameter C which determines the trade-off between the classification error and the generalisation capability of the classifier. When incorporating these parameters, the final decision function for a Support Vector Machine takes the form of

$f(x) = sign\{\Sigma_{(i=1:l)} \alpha_i y_i K(x_i, x) + b\}$

where i is a Lagrangrian multiplier and K is the kernel function of choice. [8], [9], [7].

5 Method

The data used for classification was part of the 1998 DARPA evaluation program and consisted of 11982 TCP/IP sessions, of which 5092 were used for training and the remaining 6890 for the evaluation of the classifiers performance. For more information about the contents of the data set and how it was split into training and test sets, the reader should refer to [3] and [10].

From each one of these sessions a set of 41 features was extracted that determined the stealthiness of the session, based on a priori knowledge of attack signatures. Such features included the duration of the session, the protocol types, the number of bytes sent from source to destination, number of connections to the same host [3]. The training set includes 22 different types of exploits that are either Denial of Service attacks, remote to user attacks, user to root attacks or probing attacks. The design and training procedures for each classification method employed are the following.

K-means Classifiers: This classifier was employed due to its conceptual simplicity and to see how it would perform on a problem of this scale (41 dimensions 5092 times over). It is known that K-mean classifiers perform the best when dealing with a data set that is evenly clustered [11]. There was always a probability that the above described data set might fit into this category, and therefore the nearest neighbour approach may well have been the most appropriate solution to the problem. Therefore 2-mean classifiers were developed, using Euclidean, squared Euclidean and Manhattan distance measures. The training phase of the K-mean classifier developed can be thought of simply as being the separation of the data points, belonging to the training set, into the classes ”attack” and ”normal” and computing the mean vector for each cluster. A simple non-iterative program was developed for the purpose of this uncomplicated classifier [5].

Artificial Neural Networks: In the search for an optimal neural network classifier to fit the nature of the intrusion detection data, three neural networks all different in size and structure were trained and evaluated.

1. *Network 1* (41-20-20-20-1) takes 41 inputs, has 3 hidden layers with 20 neurons each and has one output neuron, specifying whether the 41 inputs specify an attack or not (i.e. an output of +1 specifies an attack and an output of -1 a normal session).

2. *Network 2* accordingly has structure 41-40-40-1. It is simpler than Network 1 in the sense that is has overall fewer layers, however these layers have double the neurons in any of Network 1’s hidden stages.

3. *Network 3* has configuration 41-25-20-1 and is of all three networks the simplest in terms of hidden layers and neurons per layer.

Each network was trained using a variety of training algorithms. However, *resilient backpropagation* and *adaptive learning rate* proved to the best in terms of accuracy and training speeds. For more information on these algorithms, see [5].

During training of all three neural networks the following three clauses applied: i) All three networks had target error rate set to 0.001, but this goal was never met in the time needed to conduct 10,000 epochs. ii) The performance measure used was the mean square error, between outputs and the pre-specified labels in the training set. iii) The train accuracy had to do with, how close each network could classify the point in its training set, and the test accuracy had to do with the classification power of the trained network for unseen data. Clearly, the training accuracy should always be higher than the test accuracy, but the test accuracy is of more importance when rating a classifier because it states its ability to correctly classify examples it has never come across, i.e. how effectively it has been trained.

Support Vector Machines The SVMs were trained and evaluated using a program called D2C (data to classification), which uses an optimisation algorithm, enabling the system to produce results faster [8], [9].

The SVMs were trained and evaluated using the D2C program and the only information required from the user were a training set, a test set and the parameter C value. Also the kernel chosen for this program is a Gaussian Kernel, with =0.005 [5].

6 Results and Discussion

In this section, we present the performance of each classification technique and discuss the significance of these results. The results presented in the following sections were acquired on a Pentium4 CPU, 3.2 GHz, and 2GB of RAM PC.

K-Means classifiers: The classification accuracy results obtained for the different distance measures used are summarised in *Table1*. These results were expected when considering the complexity of the data set as opposed to the simplicity of this distance classifier. The K-means classifiers performed poorly with accuracies not exceeding 18%.

Artificial Neural Networks: The results of the training and testing processes for the three neural networks architectures are summarised in *Tables 2*, *3* and *4* respectively. From these results it is evident that resilient backpropagation dramatically outperforms adaptive learning rate for all three architectures i.e. gives a lower error rate/higher accuracy and slightly lower training times.

The training graphs for the three neural networks that depict the convergence of the neural networks to the desired performance level, using resilient propagation, are presented in *Figures 1*, *2* & *3* respectively. Out of all three networks *Network 2* was the most effective trained neural network with a very high accuracy percentage of nearly 96%. We can see by contrasting the networks that for this problem, a network of medium structural complexity is best suited. *Network 2* has fewer hidden layers than *Network 1*, but more neurons per layer than *Network 3*, hence yielding the best results.

Support Vector Machines: A summary of the test performance and test accuracy results obtained by varying the C parameter of the SVM is shown in *Table 5*. We can see that the Support Vector Machine performs consistently for the training set provided. SVMs will always return the same classifier no matter where you start (in terms of initial weight and bias values). The training time for each case was approximately 100secs, outdoing the training times for the neural networks 33 times. This speed in training is due to the fact that Support Vector Machines abide by a set of rules during training. This is known as heuristic learning and is the fastest means of training available to machine learning. Although *Neural Network 2* gives a higher accuracy than the Support Vector Machine, one must weigh the importance of time and accuracy for the application in question. It is very important for the intrusion detection system to train as fast as possible in such cases where a new signature is published (in which case SVMs are far superior) and it is equally as important for the system to recognize an occurring attack as accurately as possible (in which case *Neural Network 2* is superior). Finally, it is important to highlight that the results depend very much so on the nature of the data. Any change in the data space, defines an entire new problem. Hence the best solution for a problem is never known beforehand. There is always a need to experiment, in order to fit the solution as best as possible to the nature of the problem.

7 Conclusion

Three types of classifiers were developed to classify TCP/IP intrusion data to recognise whether a system is under attack. Among all the classifiers tested, Neural Network 2 and the SVM delivered highly accurate results, within similar levels of performance. The major issue is that the Support Vector Machine trained in a significantly shorter time than Neural Network 2 (100secs compared to 55mins) and this is rather important in cases where retraining must be done quickly e.g. when a new attack is published. In contrast however, the Support Vector Machine is only a binary classifier and cannot be used to identify the exact class an attack belongs to, unlike neural networks. This may be a drawback in systems that need to identify the class of the attack in order to apply defence measures. Avenues for further investigation include intrusion detection methods for wireless sensor networks. The challenging aspect with this topic will be to devise a robust and effective technique to make such vulnerable networks secure, given their many limitations.

Acknowledgements

The authors would like to thank Mr. Daniel Lai of Monash University, for his assistance with the section on Support Vector Machines and for allowing the use of his D2C optimization program. In addition, a special thanks to Mr. Srinivas Mukkamala of the Department of Computer Science of the New Mexico Institute of Mining and Technology for providing a processed intrusion data set.

References

[1] "Symantec internet security threat report highlights rise in threats to confidential information (symantec.com)," Accessed: 2005, May 10. Available: `http://www.prdomain.com/companies/s/symantec/newsreleases/200503mar/symantec_internet_20050321.htm`.

[2] R. Durst, T. Champion, B. Witten, E. Miller, and L. Spagnuolo, "Testing and evaluating computer intrusion detection systems," *Commun. ACM*, vol. 42, no. 7, pp. 53–61, 1999.

[3] A. Sung and S. Mukkamala, "Identifying important features for intrusion detection using support vector machines and neural networks," in *Symposium on Applications and the Internet*, pp. 209–216, 2003.

[4] "A tutorial on clustering algorithms," Accessed: 2004, October 1. Available: `http://www.elet.polimi.it/upload/matteucc/Clustering/tutorial_html/kmeans.html`.

[5] S. Kaplantzis, "Classification techniques for network intrusion detection," 4th year thesis project, Monash University, 2004.

[6] P. Picton, *Neural networks*. Grassroots series (Palgrave (Firm)), Basingstoke: Palgrave, 2nd ed., 2000.

[7] V. N. Vapnik, *Statistical learning theory*. Adaptive and learning systems for signal processing, communications, and control, New York: Wiley, 1998.

[8] D. Lai and N. Mani, "Support vector machines and linear stationary models," conversion report, Monash University, 2003.

[9] D. Lai, "Welcome to daniel lai," Accessed: 2004, September 21. Available: `http://www-personal.monash.edu.au/~dlai`.

[10] "Darpa intrusion detection evaluation (mit lincoln laboratory)," Accessed: 2004, October 1. Available:`http://www.ll.mit.edu/IST/ideval/`.

[11] P. Jeong and S. Nedevschi, "Efficient and robust classification method using combined feature vector for lane detection," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 15, no. 4, pp. 528–537, 2005.

[12] W. Allen and G. Marin, "On the self-similarity of synthetic traffic for the evaluation of intrusion detection systems," in *Symposium on Applications and the Internet*, pp. 242–248, 2003.

[13] A. Bivens, C. Palagiri, R. Smith, B. Szymanski, and M. Membrechts, "Network based intrustion detection using neural networks," 2002.

[14] C. Manikopoulos and S. Papavassiliou, "Network intrusion and fault detection: a statistical anomaly approach," *Communications Magazine, IEEE*, vol. 40, no. 10, pp. 76–82, 2002.

[15] S. Mukkamala and A. Sung, “A comparative study of techniques for intrusion detection,” in *Tools with Artificial Intelligence, 2003. Proceedings. 15th IEEE International Conference on*, pp. 570–577, 2003.

Table 1. Accuracy results for the k-means classifiers

Distance Measure	**Accuracy**
Euclidean	17.40%
Squared Euclidean	17.40%
Manhattan	17.59%

Table 2. Results for Neural Network 1 (41-20-20-1)

Algorithm	**Adaptive Learning Rate**	**Resilient Backpropagation**
Train Accuracy	32.46%	85.07%
Test Accuracy	24.33%	81.65%
Performance	0.235996	0.00443572
Time to train	55mins	50mins

Table 3. Results for Neural Network 2 (41-40-40-1)

Algorithm	**Adaptive Learning Rate**	**Resilient Backpropagation**
Train Accuracy	55.22%	96.05%
Test Accuracy	51.30%	93.58%
Performance	0.2156318	0.00327674
Time to train	1hr	55mins

Table 4. Results for Neural Network 3 (41-25-20-1)

Algorithm	**Adaptive Learning Rate**	**Resilient Backpropagation**
Train Accuracy	32.57%	79.46%
Test Accuracy	23.32%	63.25%
Performance	0.666477	0.00327674
Time to train	50mins	50 mins

Table 5. Results for the Support Vector Machine

C Parameter	**Accuracy**	**Performance**
0.01	82.692%	0.69
0.1	82.6392%	0.692319
1	82.6339%	0.68
10	82.9824%	0.68
100	82.9824%	0.680703
100	82.9824%	0.68

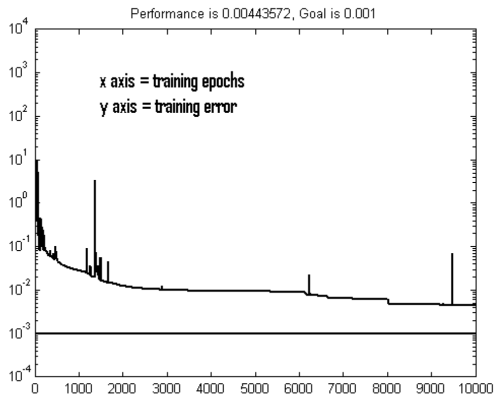

Figure 1. Training convergence of network 1 to a 0.001 error goal using resilient back propagation

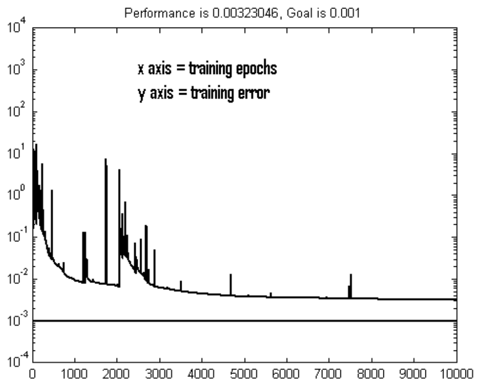

Figure 2. Training convergence of network 2 to a 0.001 error goal using resilient back propagation

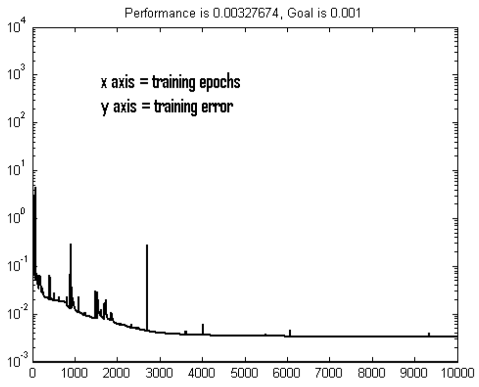

Figure 3. Training convergence of network 3 to a 0.001 error goal using resilient back propagation