Evolutionary Learning Program's Behavior in Neural Networks for Anomaly Detection

Sang-Jun Han, Kyung-Joong Kim, and Sung-Bae Cho

Dept. of Computer Science, Yonsei University,
134 Shinchon-dong, Sudaemoon-ku, Seoul 120-749, Korea
{sjhan,kjkim,sbcho}@cs.yonsei.ac.kr

Abstract. Learning program's behavior using machine learning techniques based on system call audit data is effective to detect intrusions. Among several machine learning techniques, the neural networks are known for its good performance in learning system call sequences. However, it suffers from very long training time because there are no formal solutions for determining the suitable structure of networks. In this paper, a novel intrusion detection technique based on evolutionary neural networks is proposed. Evolutionary neural networks have the advantage that it takes shorter time to obtain superior neural network than the conventional approaches because they learn the structure and weights of neural network simultaneously. Experimental results against 1999 DARPA IDEVAL data confirm that evolutionary neural networks are promising for intrusion detection.

1 Introduction

In host-based anomaly detection, the idea of learning program's behavior has been studied and used actively by many researchers. It considers normal behavior from the point of individual program. Profiles for program's behavior are built and the behaviors which deviate from the profile significantly are recognized as attacks. Machine learning methods have been used to profile program's behavior because it can be viewed as a binary classification problem which is one of the traditional problems in pattern classification. Especially, in previous researches, neural network showed the performance superior to other techniques. However, profiling normal behavior requires very long time due to the huge amount of audit data and computationally-intensive learning algorithm. Moreover, to apply neural network to real world problems successfully, it is very important to determine the topology of network, and the number of hidden nodes which are proper for the given problem, because the performance hinges upon the structure of network. Unfortunately, although many works on designing the domain-specific network structure automatically, there is no absolute solution [1] and typically the network structure is designed by repeating trial and error cycles on the basis of the experiences of working on similar problem. A.K. Ghosh who showed the best performance against the pubic benchmark data trained 90 neural networks in total for each program: 10, 15, 20, 25, 30, 35, 40, 50, and 60 hidden nodes and 10 networks for each number of hidden nodes. Then a neural network which showed best performance against the validation data was selected [2]. Therefore it takes a very long time to build normal behavior model and it is the vital drawback of neural network-based intrusion detection technique.

In this paper, we employ evolutionary neural network (ENN) to overcome the shortcoming of the conventional intrusion detection technique based on neural network. ENN does not require trial and error cycles for designing the network structure and the near optimal structure can be obtained automatically. Due to these advantages of ENN, we can get better classifier in shorter time. We examine the proposed method through experiments with real audit data and compare the result with that of other methods.

2 Intrusion Detection with Evolutionary NNs

Fig. 1 illustrates the overall architecture of ENN-based intrusion detection technique. We use system call-level audit data provided by BSM (Basic Security Module) of Solaris operating system. Preprocessor monitors the execution of specified programs and generates system call sequences by programs. GA modeler builds normal behavior profiles using ENN. One neural network is used per one program. New data are input to the corresponding neural network. If the evaluation value exceeds the pre-defined threshold, the alarm is raised.

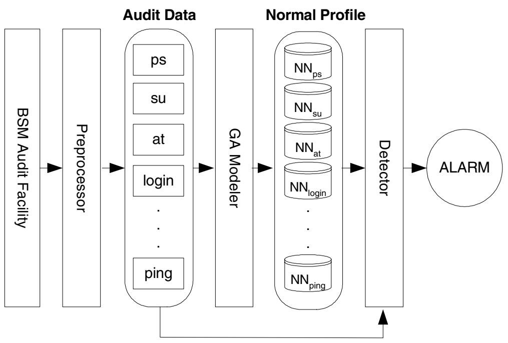

Fig. 1. Overall architecture of the proposed technique.

2.1 Modeling Normal Behavior

Our ENN has L input nodes because the system call sequence S_t which is generated at time t with window length L is used as input. There are two output nodes which represent normal and attack behavior respectively. 10 input nodes are used because we have set the window length as 10. There are 15 hidden nodes among which the connectivity is determined by evolutionary algorithm. Anomaly detector uses only attack-free data in training phase, but to train the supervised learner like neural network, the data labeled as attack are also needed.

For this reason, we have generated the artificial random system call sequences and used them as intrusive data. The training data is generated by mixing real normal sequences and artificial intrusive sequences in the ratio of 1 to 2. In this way, we can obtain the neural network which classifies all system call sequences except given normal sequence as attack behavior.

There are several genotype representations methods for neural network such as binary, tree, linked list, and matrix representation. We have used a matrix-based genotype representation because it is straightforward to implement and easy to apply genetic operators. When N is the total number of nodes in a neural network including input, hidden, and output nodes, the matrix is $N \times N$ whose entries consist of connection links and the corresponding weights. In this model, each neural network uses only forward links. In the matrix, upper right triangle (see Fig. 2) has connection link information and lower left triangle describes the weight values corresponding to the connection link information. The number of hidden nodes can be varied within the maximum number of hidden nodes in the course of genetic operations.

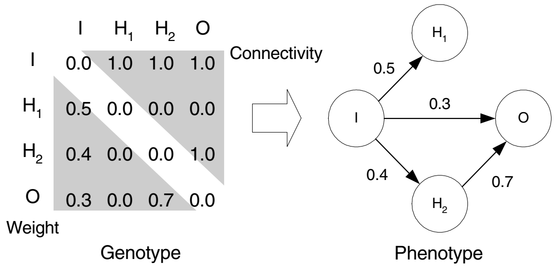

Fig. 2. An example of genotype-phenotype mapping.

Crossover and mutation operator is used as genetic operators and the fitness is calculated as the recognition rate for the training data. The rank-based selection in which the individuals' selection probabilities are assigned according to the individuals' rank based on the fitness evaluation function values is used.

2.2 Anomaly Detection

For accurate intrusion detection, it is important to recognize the temporal locality of abnormalous events, not the fluctuation of the output value [2]. High output values of attack node for very short time should be ignored because it is not sufficient to decide if that process is attack. To do that it is required to consider the previous output values as well as the current output values. For this purpose, we define a new measure of abnormality that has a leaky integrator. When o_t^1 denotes the output value of attack node, o_t^2 denotes the output value of normal node, and w_1, w_2, w_3 denote the weights to these values, the raw evaluation score r_t is calculated as follws:

$$r_t = w_1 \cdot r_{t-1} + w_2 \cdot o_t^1 + w_3 \cdot o_t^2 \tag{1}$$

It retains the evaluation value of past evaluation with decay and we get higher abnormality of current process as the output value of attack node is higher and the output value of normal node is lower. In this way, we can measure the abnormality of program's behavior robustly to short fluctuation and recognize the temporal locality of abnormal behaviors.

We define threshold and check whether its abnormality is exceeds it, to determine whether current process is attack or not. However, the decision boundaries vary from program to program because the different neural network is used to evaluate the different program behavior. Thus, applying a threshold to overall neural network is not feasible. To solve this problem we have normalized the raw evaluation values statistically. First, we test the training data using the trained neural network and we calculate the mean and variance of r_t. Then, under assumption of that r_t is normally distributed, we transform r_t to corresponding value in standard normal distribution R_t. When m is the mean of r_t and d is the standard deviation against the training data, the normalized evaluation value R_t is calculated as follows:

$$R_t = eval(S_t) = \frac{r_t - m}{d} \tag{2}$$

If R_t exceeds the pre-defined threshold, current process is considered as attack.

3 Experiments

3.1 Experimental Settings

To verify the proposed method, we have used the 1999 DARPA intrusion evaluation data set [4]. In this paper, our experiments are focused on detecting U2R attack attempts to gain root privilege by privileged program misuse. Thus, we monitors only SETUID privileged programs which are the target of most U2R attacks. This data set consists of five weeks of audit data. 1-3 week data are for training and 4-5 week data are for testing. We have used 1 and 3 weeks data which do not contain any attacks for training neural networks and 4 and 5 week data are used for testing. The test data contain 11 instances of 4 types of U2R attacks.

Population size is 20 and the maximum generation number is 100. Crossover rate is 0.3 and mutation rate is 0.08. The neural network which has the highest fitness is selected and used for testing.

3.2 Results

Comparison of Training Time. The time required for training general MLP and ENN is compared. The training program was run on the computer with the dual Intel Pentium Zeon 2.4GHz processor, 1GB RAM, and Sun Solaris 9 operating system and the average time was taken. In the case of MLP, the number of hidden nodes varied from 10 and 60 and for each number of hidden

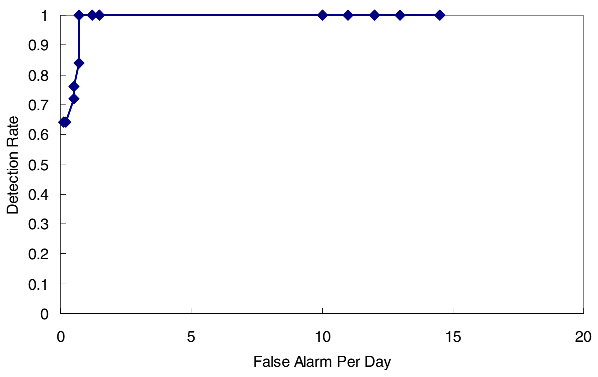

Fig. 3. Intrusion detection performance of ENN.

nodes, 10 networks were trained. Total 90 networks were trained. Error back propagation algorithm was iterated until 5000 epoch. ENN has the maximum 15 hidden nodes and the population of 20 neural networks was evolved to the 100th generation. Both neural network has 10 input nodes and 2 output nodes and are trained with the training data of *login* program which consists of 1905 sequences.

The conventional approach which repeats trial-and-error cycle requires about 17 hours 50 minutes. However, in case of evolutionary neural network, it takes 1 hour 14 minutes. Evolutionary approach can reduce the learning time as well as it has advantage that the near optimal network structure can be obtained.

Comparison of Detection Performance. Fig. 3 depicts the detection/false alarm plot which illustrates the intrusion detection performance of the proposed method. It produces 0.7 false alarms at 100% detection rate. In 1999 DARPA IDEVAL, the method which showed the best performance at detecting U2R attacks is the work of A.K. Ghosh *et at.* that learns system call sequences with Elman recurrent neural network [5]. It showed 3 false alarms at 100% detection rate [3]. The performance of ENN is superior to that of Elman network. This result illustrates that ENN can find more optimal neural network than the conventional neural network which has static and regular structure.

Comparsion of Network Structure. Table 1 compares ENN trained with *ps* program's behavior and general MLP in terms of network structure. Both have the same number of nodes: 10 input nodes, 15 hidden nodes, and 2 output nodes. The total number of connections does not differ much. However, ENN has more various types of connection including connection types which do not exist in MLP such as connections from input node to output node and from hidden node to hidden node. In the work of A.K. Ghosh *et al.*, they improved the performance by retaining context information between samples with recurrent topology. On the other hand, ENN attempts to increase learnable samples by forming non-regular and complex network structure.

Table 1. Comparison of network structure.

(a) ENN

From\To	Input	Hidden	Output
Input	0	86	15
Hidden	0	67	19
Output	0	0	0

(b) MLP

From\To	Input	Hidden	Output
Input	0	150	0
Hidden	0	0	30
Output	0	0	0

4 Conclusion

This paper proposes an evolutionary neural network approach for improving the performance of anomaly detection technique based on learning program's behavior. The proposed method cannot only improve the detection performance, but also reduce the time required for training because it learns the structure and weights of neural network simultaneously. The experimental result against 1999 DARPA IDEVAL which is superior to previous works verifies the proposed method. As future work, it is needed to find the network structure which is good for intrusion detection by analyzing the evolved structures. For more accurate modeling, we can employ multiple expert neural networks which are evolved with speciation and combine them.

Acknowledgement

This paper was supported by Brain Science and Engineering Research Program sponsored by the Korean Ministry of Science and Technology.

References

1. X. Yao, "Evolving Artificial Neural Networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423-1447, 1999.
2. A. K. Ghosh, A. Schwartzbard, and M. Schatz, "Learning Program Behavior Profiles for Intrusion Detection," *Proceedings of the 1st USENIX Workshop on Intrusion Detection and Network Monitoring*, pp. 51-62, Santa Clara, CA, April, 1999.
3. A. K. Ghosh, C. C. Michael, and M. A. Schatz, "A Real-Time Intrusion Detection System Based on Learning Program Behavior," *Proceedings of the Third International Symposium on Recent Advances in Intrusion Detection*, pp. 93-109, 2000.
4. MIT Lincoln Laboratory, "DARPA Intrusion Detection Evaluation," Available from http://www.ll.mit.edu/IST/ideval/index.html.
5. R. Lippmann, J. Haines, D. Fried, J. Korba, and K. Das, "The 1999 DARPA Off-Line Intrusion Detection Evaluation," *Computer Networks*, vol. 34, no. 4, pp. 579-595, 2000.