Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 6 (2011) 219–224

Procedia Computer Science

Complex Adaptive Systems, Volume 1
Cihan H. Dagli, Editor in Chief
Conference Organized by Missouri University of Science and Technology
2011- Chicago, IL

Multiple SOFMs Working Cooperatively In a Vote-based Ranking System For Network Intrusion Detection

Charlie Obimbo, Haochen Zhou, Ryan Wilson

School of Computer Science,University of Guelph, Guelph ON N1G 2W1, Canada

Abstract

Protection from hackers on networks is currently of great importance. Recent examples of victims include the recent repeated hacking of Sony PS3, which involved 24.6 million customer accounts being vulnerable, and the hacking of websites both including US and Canadian government sites. Thus there is a drear need for effective Intrusion Detection and Prevention systems. Anomaly intrusion detection is a popular method of detecting Intrusions on Computer Networks. In 2011, Wilson and Obimbo proved that the use of Self-Organized Feature Maps (SOFM) could be used to increase the performance on KDD-99 dataset. This paper introduces a vote-based ranking system for intrusion detection based on SOFM. The experimental results are promising and are an improvement in both Wilson and Obimbo's system and the Winning system of the KDD IDS Competition.

Keywords: IDS, KDD-99, SOFM, Ranking System

1. Introduction

Intrusion Detection and Prevention has become a pertinent part of network security. An example of a vulnerability capitalized on was the recent hacking of Sony Play-Station Network, which resulted in the alleged stealing of 24.6 million accounts [1]. In 1994, just in US alone, the cost of systems downtime to companies was placed by some estimates at $4 billion a year, with a loss of 37 million hours in worker productivity [2]. This numbers are most likely higher today. These losses mainly arise due to DoS, User to root and Remote to Local attacks, which are events that can be evaded with effective Intrusion Detection and Prevention Systems.

Artificial neural networks (ANNs) are popularly applied to Intrusion Detection Systems. However, their performance so far have not been satisfactory. Beghdad [3], reported low detection rates in many ANNs. Sabhnani and Serpen [4] even claimed that no machine learning algorithms could work on KDD99 for network intrusion detection [5]. Nevertheless, in 2010, Wilson and Obimbo [6] described a use of SOFMs for IDSs, and generated better results than the winner of KDD Cup competition. This renews people's view on applying ANNs on network intrusion, and makes future researches on using SOFMs for IDSs becomes feasible and sound.

This research focuses on developing a vote-based ranking system by using self-organizing feature maps (SOFMs) [7], which is one fully developed algorithm of ANNs, on improving anomaly detection for IDSs. In Wilson and

1877–0509 © 2011 Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
doi:10.1016/j.procs.2011.08.041

Obimbo's research [6], the SOFMs yielded a reasonably good classification rate, while achieving a low false positive rate. The SOFMs also have advantages on unsupervised training process, and identifying attack types.

The vote-based ranking system is a novel approach for detecting intrusion. This system creates several SOFMs trained differently instead of one to improve the precision on classification rate and reduce the false positive rate on each type of attacks. The idea behind this is SOFMs generate good results when using small dataset [8]. Therefore, each SOFM is trained using a relatively smaller dataset compared to the whole training data set (i.e. 10% of the data, but only from 2 to 3 specific types of attack, instead of all 5 of them). After training SOFMs, intrusion types are detected by a master system, which calculates ranks for each SOFM.

2. Related Work

2.1 Intrusion Detection Systems

An Intrusion Detection System can dynamically monitor the events happening in a system, and identify whether they are attacks or legitimate accesses [9]. IDSs can be categorized as Misuse Detectors, Anomaly Detectors, or a Hybrid of the two. This research deals with Anomaly Detection, and so we shall major on these. Anomaly detection uses the assumption that unexpected behaviour is evidence of an intrusion. They can further be categorized as *specification-based* (these are a set of rules from human experts which provide the basis of bias that help determine "good/normal" behaviour), and *behaviour-learning-based*, in which the Detection System automatically learns the behaviour of the system under normal operation.

Rule based IDSs such as Snort and Bro use human-crafted rules to determine known attacks[10,11], for example, virus signatures and requests to nonexistent services or hosts. However, anomaly detection systems such as SPADE [12], NIDES [13], PHAD [14], ALAD [15] compute (statistical) models for normal network traffic and trigger alarms when a large deviation from the norm is found.

The work of Wilson et al [16] describes an intrusion detection system that is most similar to our work. Their IDS is based on generating a Self-Organizing Feature Map based on known intrusions, which are categorized as Probe, User-to-root, Remote-to-local, and Denial-of-Service, and also normal data, based on the KDD data set. They use the Euclidean distance to determine the Hotspot Vectors, which are then used to categorize the various categorize the various payload provided. They also do Vector-pruning (see Section 4), to eliminate the least significant hotspots, and this reduces false-positives. The results obtained in Wilson et. al.'s work show a significant improvement from the KDD Cup winner's in the category of User-to-Root, being able to detect 63% correctly, compared to 15% for the KDD Cup winner.

2.2 KDD-99 Dataset

In 1999, Knowledge Discovery and Data Mining (KDD) Cup competition was held for building a network intrusion detector on KDD-99 dataset. Since that time, KDD-99 has become a benchmark for IDSs.

The KDD-99 dataset is derived from DARPA 1998 dataset. DARPA-Lincoln datasets are collected by MIT's Lincoln laboratory, which is under the DARPA ITO and Air Force research Laboratory sponsorship, to test the capability of different intrusion detection techniques. In the DARPA 1998 dataset, the data was collected from victim UNIX hosts for nine weeks, and the total dataset is about 4 gigabytes of compressed binary TCP dump file [17]. These connections belong to 38 different attacks which each falls into one of the following categories: Denial of Service Attack (DoS), Probing Attack (Probe), Remote to Local Attack (R2L), and User to Root Attack (U2R).

In 1999, the TCP dump files were pre-processed for the International KDD Cup competition as the IDSs benchmark. In the TCP dump file, packet information transfer into connections. "A connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows from a source IP address to a target IP address under some well defined protocol [18,19]." And totally 41 features are generated in each connection.

KDD-99 dataset has three components, which are described in Table 1. 10% KDD-99 is only used for the training purpose, and this dataset includes 22 attack types. It can be viewed as simple version of the Whole KDD-99. The Corrected KDD-99 has different statistical distributions from either 10% KDD-99 or Whole KDD-99, and it contains 14 additional attacks. So the Corrected KDD-99 is employed for testing.

Table 1: Basic Characteristics of the KDD-99 Dataset

Dataset	DoS	Probe	U2R	R2L	Normal
10% KDD	391458	4107	52	1126	97278
Corrected KDD	229853	4166	70	16347	60593
Whole KDD	3883370	41102	52	1126	972780

3. Self-organizing Feature Maps

Self-organizing Feature Maps (SOFMs) are also called Kohonen's Maps, because the theory of SOFMs was first raised by Finnish academic Dr. Teuvo Kohonen [7]. SOFMs are typical unsupervised neural networks. The concept of "supervised" and "unsupervised" is based on whether the results or outputs can be predicted or not. Supervised training process always contains input pattern and a desired output, and adjusts the network to more likely generate desired results. In contrast, unsupervised training process does not have desired output; the network has to find out the best way to represent input vectors. The rest of this section will describe the details of SOFMs' working principle.

3.1 Overview

SOFMs contain two layers: one layer for the input vector, and the other a feature map for outputs or clusters. Each neuron in the feature map is completely connected with input vectors by weights. Figure 1, shows a basic architecture of a SOFM.

3.2 Initialization

There are three factors that need to be determined before training SOFMs.

1. **The size of feature map**. Choosing a suitable size of map can improve the accuracy of output by reducing the overlap area [7]. The nodes in the output layer adheres to the rectangular topology which easily maps to Cartesian plane.
2. **Iteration time.** Though training the feature map for a long time might give a better accuracy, one has to determine the "optimal" training time for the relatively desired accuracy.
3. **Selection of a Neighbourhood function:** a neighbourhood function needs to be selected that adjusts the nodes around a candidate node. The neighbourhood should shrink with time. The most common neighbourhood function for SOFMs is the Gaussian curve.

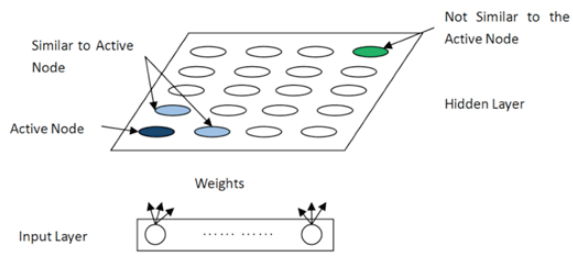


Fig. 1: SOM Neural Network

3.3 Training

After the initial setup, the SOFMs start to learn. And the training process contains three phases: competition, cooperation, and adaption. The three phases proceed iteratively until a certain criterion (like number of iterations) is met.

3.3.1 Competition

Competition is the first phase of training process. It generates the candidate node in the feature map. The activation of each neuron is calculated by equation (1), where X represents m-dimension input vector, and w_j is weights between input and output layer. Initially, the value of weights is randomly set.

$$\text{Activation} = w_j^T \cdot X; \quad X = [x_1, x_2, \ldots, x_m]; \quad w_j = [w_{j1}, w_{j2}, \ldots, w_{jm}], j = 1, \ldots, l \tag{1}$$

The node with the largest activation becomes the centre of the topological neighbourhood. It is called winner, or candidate node. Then the training process enters into cooperation phase.

3.3.2 Cooperation

In this phase, the neighbourhood around the winner is identified by the neighbourhood function. The specialized Gaussian Curve is used for SOFMs:

$$h_{j,i} = e^{-\frac{d_{ij}^2}{2\sigma^2}} \tag{2}$$

where j is one of a set of cooperating nodes (neighborhood), i is the winning node, d_{ij} is the distance between i and j, σ is the parameter which controls the width of the neighborhood.

Shrinking the neighborhood allows large areas to form earlier during training, and eventually become more defined. This process is controlled by adjusting σ during training. And the formula of decay function for σ is showed in (3), where n represents time, σ_0 is the initial width, and p_1 is a constant between 0 and 1.

$$\sigma(n) = \sigma_0 e^{-\frac{n}{p_1}} \tag{3}$$

3.3.3 Adaption

Once the neighbourhood has found, learning rate and weights connected on the neighbourhood nodes need to be adjusted. The weights are updated by:

$$w_j(n+1) = w_j(n) + \alpha \times h_{j,i(X)} \times \left(X - w_j(n)\right) \tag{4}$$

where α is learning rate, X is the input, h is the neighbourhood nodes, and j is the winning node. With repeated presentations the weights tend to follow the distribution of input vectors. The learning rate can maintain the same during training, or decrease by time. Exponential decay is not generally optimal, but it is normally sufficient.

4. Methodology

This section will describe the adjustments for input data, the working principle of ranking system, and the methods that used in experiments.

4.1 Data Preprocessing

For generating better results, the data need to be normalized. Large value of attribute in input vector can greatly affect the competition in SOFM. Therefore, before putting data into SOFM, normalization should be processed to make each attributes in vector are equally treated in SOFM. The normalization function is chosen the same one in Wilson and Obimbo's paper [6]. Moreover, not all the features in KDD-99 become dimensions of input vectors. The redundant features (such as the duration time of a connection) are omitted. Therefore, the last 37 features in each row of dataset are analyzed in experiments.

4.2 Training the SOFMs

Training one SOFM is simple. First, read the data from dataset row by row; second, calculate or generate candidate node; third, adjust the neighbourhood; fourth, return to first step and read a new row, until the end of dataset. The details in these steps are described in Section 3.

In this research, 10 different SOFMs are trained using the same parameters. Each SOFM is just trained two types of attack on it. After the training finished, the training dataset compared the feature map again. This time, each SOFM has a matrix to track the times of hits and the type of intrusion on certain node. These matrixes will be used for classification.

4.3 Hotspot Vectors

The track matrix in each SOFM will determine the hotspots in the feature map. This helps in the classification process and reduces the complexity of the maps. Each hotspot creates a vector with Cartesian points. This method is similar to that used by Wilson and Obimbo [6], and it is also effective for multiple SOFMs. These new vectors are then compared to the ones already in the map, and calculations are done to determine overlapping vectors. The least influencing nodes are then eliminated from the map.

The number of hotspot vectors is showed in Table 2. Due to the overlapping nodes that may be eliminated, as described above, the totals in the table are equal to or less than the sum of the two.

Table 2: Hotspot Vectors in Each SOFM

	SOFM0		SOFM1		SOFM2
Total	299		244		320
DoS	79	Nor.	231	Nor.	319
Nor.	259	Pro.	52	R2L	39
	SOFM3		SOFM4		SOFM5
Total	26		345		273
R2L	24	Nor.	342	DoS	248
U2R	11	U2R	25	Pro.	64
	SOFM6		SOFM7		SOFM8
Total	47		146		62
Pro.	44	DoS	128	Pro.	44
U2R	7	R2L	26	R2L	29
	SOFM9				
Total	236				
DoS	232				
U2R	5				

4.4 Ranking System

After feature maps trained and hotspot vectors calculated, test data will go through each trained SOFM. The result from each SOFM is called vote. A vote indicated how close the input vector to a type of attack. The votes are measured by three factors: the distance between a hotspot vector, how many hits on the hotspot in training, and how many types of intrusion hits on it (in this research the most is two).

A node is activated after an input vector is calculated by SOFM. This node will compare with the hotspot vectors which are near the node within a radius. The nearest hotspot vector will be selected. Then the vote is calculated by the formula (5), where *hits* represent certain type of attack that hits on this node, and *totalHits* is the number of total hits of the whole map. These votes collected from each SOFM are summarized as rank. The highest rank value on a particular type of intrusion represents the type of input. For instance, a raw data vector goes through the 10 SOFMs, and gets the highest rank on probe, and then this connection is probe attack.

$$\text{VoteValue} = \left(1 - \frac{distance}{radius}\right)\left(\frac{hits}{totalHits}\right) \tag{5}$$

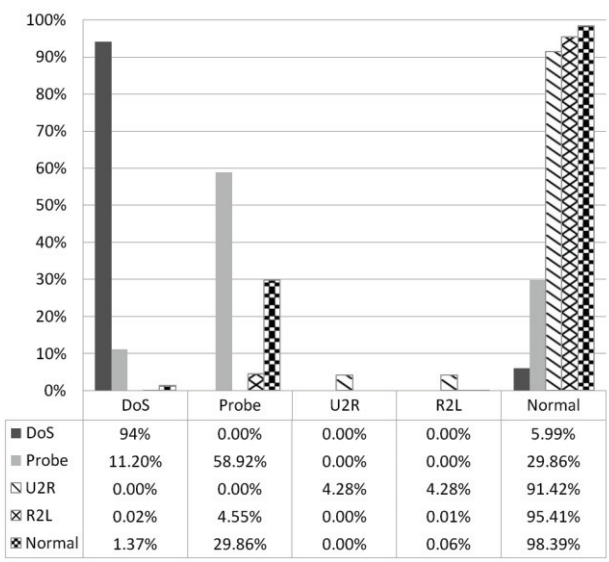
5. Experiments and Results

Two distinct experiments were carried out. They use different distance measurement and ranking systems, and the results show different emphasis on intrusion types.

5.1 Euclidean Distance Measurement

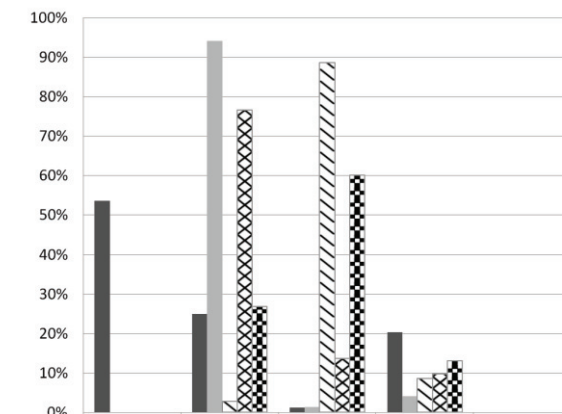
Euclidean distance is common method for SOFMs to check an input vector if it belongs to one cluster. The formula has stated before. Using this method is straight forward. The data in the testing dataset is imported row by row to a SOFM, calculated and compared with the existing map to find out the most similar node.

The distance between them computes the similarity between nodes. Then this node generates a vote on this SOFM. The same data gains votes from all SOFMs. The last step is check the intrusion type got from ranking system with the label on target input. Figures 2 and 3 show the results obtained using the various distance measurements. The columns of the tables below the graphs refer to the detectors and the rows are the actual intrusion types.



	DoS	Probe	U2R	R2L	Normal
DoS	94%	0.00%	0.00%	0.00%	5.99%
Probe	11.20%	58.92%	0.00%	0.00%	29.86%
U2R	0.00%	0.00%	4.28%	4.28%	91.42%
R2L	0.02%	4.55%	0.00%	0.01%	95.41%
Normal	1.37%	29.86%	0.00%	0.06%	98.39%

Fig. 2: Result of Using Euclidean Distance Measurement



	DoS	Probe	U2R	R2L	Normal
DoS	54%	24.89%	1.24%	20.25%	0.00%
Probe	0.05%	94.07%	1.32%	4.05%	0.00%
U2R	0.00%	2.85%	88.57%	8.57%	0.00%
R2L	0.00%	76.58%	13.68%	9.72%	0.00%
Normal	0.00%	26.87%	60.05%	13.06%	0.00%

Fig. 3: Result of Using Customized Distance Measurement

So in Figure 2 the 94% states that 94% of what the system categorized as DoS intrusion was correct, whereas below it, 11.20% of what the system categorized as DoS Intrusion were actually Probe attacks. The result, shown in Figure 2, is similar to the results of Wilson and Obimbo [6]. However, they got more than 60% detection rate on Probe and U2R attack. The large misclassification occurs on Normal. Like R2L, most of these attacks are viewed as Normal by the ranking system.

5.2 Novel Measurement Approaching

This experiment uses a novel concept to classify raw data. It still uses the same SOFMs and hotspots, but distance is not as a vote measurement any more. If we view feature map as universe, each node on it is a planet; two planets are close to each other does not mean one belongs to another, like the Earth and the Mars. The vote equation

uses (6), where *hitsNode* is the hits on the selected node. A node with more hits on it will be considered as the "nearest" node.

$$\text{VoteValue} = \left(\frac{hits}{hitNode}\right)\left(\frac{hits}{totalHits}\right) \tag{6}$$

The result is illustrated in Figure 3. It shows a great improvement on detecting Probe and U2R attacks, 94% and 88.5% respectively. However, the false positive is also unbelievably high -- 100%.

6. Conclusion

In the first experiment, if we do not change the distance measurement function, the result is similar to using only one trained SOFM [6]. The second experiment shows a great alteration on particular attack types -- Probe and R2L. It surely can be used to detect specific intrusions.

In the second experiment, comparing to Wilson and Obimbo's result [6], the ranking system improves the detection rate on Probe and U2R, and it also eliminates the overlap in hotspot vectors [6] due to only two intrusions data trained in each map. However, the low detection rate on DoS and R2L and the extremely high false positive are factors that need consideration in modification of the system, as the system cannot be used in it's current state. In future we hope to study what needs to be done to make it usable.

REFERENCES

[1] James Plafke. Sony Hack Not Done Yet: Sony Online Entertainment Compromised. http://www.geekosystem.com/sony-online-entertainment-hacked/, Mar. 2011. Accessed in May 2011.
[2] Louis Fried. Information security and new technology. Information Systems Management, 11(3):57–64, 1994.
[3] Rachid Beghdad. Critical study of neural networks in detecting intrusions. Computers and Security, 27:168–175, 2008.
[4] Maheshkumar Sabhnani and Gursel Serpen. Why machine learning algorithms fail in misuse detection on KDD intrusion detection data. Intelligent Data Analysis, 8:403–415, 2004.
[5] R. Lippmann, et al. The 1999 DARPA Off-Line Intrusion Detection Evaluation, Computer Networks 34(4) 579-595, 2000.
[6] Ryan Wilson and Charlie Obimbo. Self-organizing feature maps for user-to-root and remote-to-local network intrusion detection on the KDD cup 1999 dataset, in Proceedings of the World Congress on Internet Security (WorldCIS-2011), London, UK, Feb 2011, pp. 56-61.
[7] Tuevo Kohonen. Self-Organizing Maps. 2nd Ed. Springer 1997.
[8] Osama Abu Abbas. Comparison between data clustering algorithms. The International Arab Journal of Information Technology, 5(3): 320– 325, 2008.
[9] Hervé Debar, Marc Dacier, and Andreas Wespi. Towards a taxonomy of intrusion systems. Computer Networks, 31(8):805–822, 1999.
[10] http://www.snort.org/assets/168/LW-hakin9-custm-rules-2010.pdf.
[11] http://www-old.bro-ids.org/wiki/index.php/User_Manual:_Overview_of_Bro.
[12] J. Hoagland, SPADE, Silican Defense, http://www.silicondefense.com/software/ spice, 2000.
[13] H. S. Javits and A. Valdes. The NIDES statistical component: Description and justification. Technical report, SRI International, Computer Science Laboratory, 1993.
[14] M. Mahoney, P. K. Chan, Learning Nonstationary Models of Normal Network Traffic for Detecting Novel Attacks, Proc. SIGKDD 2002, 376-385.
[15] M. Mahoney. Network Traffic Anomaly Detection Based on Packet Bytes. Proc. ACM- SAC, pp. 346-350, 2003.
[16] Ryan Wilson. Self-organizing feature maps for user-to-root and remote-to-local network intrusion detection on the 1999 KDD cup dataset. Master's thesis, University of Guelph, Guelph, ON, Canada, 2011.
[17] KDD cup 1999 data. http://kdd.ics.uci.edu/ databases/kddcup99/kddcup99.html, 2009. Accessed on March 2011.
[18] Anirut Suebsing and Nualsawat Hiransakolwong. Feature Selection Using Euclidean Distance and Cosine Similarity for Intrusion Detection Model. Intelligent Information and Database Systems, Asian Conference on IEEE Computer Society, 86-91, 2009.
[19] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani. A detailed analysis of the KDD cup 99 data set. Proceeding of IEEE CISDA, pp 1 – 6, 2009. S