Growing Recurrent Self Organizing Map

Özge Yeloğlu, *Student Member, IEEE, A. Nur Zincir-Heywood, Member, IEEE, Malcolm I. Heywood, Senior Member, IEEE*

***Abstract*— The growing Recurrent Self-Organizing Map (GRSOM) is embedded into a standard Self-Organizing Map (SOM) hierarchy. To do so, the KDD benchmark dataset from the International Knowledge Discovery and Data Mining Tools Competition is employed. This dataset consists of 500,000 training patterns and 41 features for each pattern. Unlike most of the previous methods, only 6 of the basic features are employed. The resulting model has a capability of detection (false positive) rate of 89.6% (5.66%), where this is as good as the data-mining approaches that uses all 41 features and twice as faster than a similar hierarchical SOM architecture.**

I. Introduction

Temporal sequence processing is a research area having applications in different fields varying from speech recognition to weather forecasting, signal processing, time series prediction and intrusion detection. In this work, we are concerned with the representation of time under the unsupervised learning paradigm where a static neural network (Self-Organizing Map) is provided with dynamic properties. For a neural network to be dynamic it must be given a memory. A traditional way of building short-term memory into the structure of a neural network that has no capacity for representing temporal dependencies is through the use of time delays, which can be implemented at the input layer of the network. That is to say, the temporal sequence is converted into a concatenated vector via a tapped delay line and training conducted without modification of the learning algorithm [1]. However, this approach has well-known drawbacks, one of the most serious ones being the difficulty in determining the proper delay line length. Within the specific context of Self Organizing Maps (SOM) two models have been previously proposed: the Temporal Kohonen Map (TKM) [12] and the Recurrent Self Organizing Map (RSOM) [9]. The two models are very different. A TKM associates all the temporal processing capacity with the trajectory of the best matching units. A RSOM explicitly includes recurrent connectivity into the neural output. This results in RSOM models having the capacity to explicitly capture temporal patterns in the original input, whereas the TKM concentrates on best matching unit sequence learning.

Manuscript received March 16, 2007. This work is supported in part by MITACS, CFI, SwissCom, TARA and NSERC Discovery grants.

Özge Yeloğlu is with Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 1W5, Canada (e-mail: yeloglu@cs.dal.ca)

A. Nur Zincir-Heywood is with Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 1W5, Canada (e-mail: zincir@cs.dal.ca, phone: 902-4943157; fax: 902-4921517)

Malcolm I. Heywood is with Faculty of Computer Science, Dalhousie University, Halifax, NS B3H 1W5, Canada (e-mail: mheywood@cs.dal.ca)

While RSOM is a promising structure in temporal sequence processing area, it is difficult to decide an appropriate network structure for a given problem. Since a fixed network structure is used in terms of number and arrangement of neurons, which has to be defined prior to training, this often leads to a significant degree of trial and error when deploying the model. Therefore, previously proposed growing neural network methods [4], [5], [6] motivate the idea of a Growing Recurrent Self-Organizing Map (GRSOM). The contribution of this work is to design a RSOM model that determines the number and arrangement of units during the unsupervised training process.

Performance of this work is demonstrated on the 1999 Knowledge Discovery and Data Mining Tools Competition dataset (KDD-99) [7] where this describes an intrusion detection problem. KDD-99 data set consists of approximately 500,000 training patterns of 41 features and provides the only *labeled* dataset for comparing IDS systems, which the authors are aware of. The previous works in this area are able to have detection (false positive) rates in the range of 89% (4.6%) to 98% (10%) whilst using all 41 connection features [2], [8].

This work only uses the six "Basic features of an Individual TCP connection" [7]. Six GRSOMs are built for each feature on the first layer. The standard SOM on the second level spatially correlates the outputs of the first level feature specific GRSOMs and third layer SOMs we selectively built for the worst case second layer neurons that respond to multiple connection types. Detection rate of the model on the test set is 89.6% while false positive rate is 5.66%. Moreover, our model is trained within matter of hours, as opposed to days when the standard SOM hierarchy is trained without the built-in memory feature.

The remainder of this paper is organized as follows. Section II provides the background and Section III provides the methodology of the work. Results are reported in Section IV and conclusions and future work are identified in Section V.

II. Background

In a previous work on the aforementioned, a shift register was utilized to embed the temporal relationship between incoming connections and a 3-layer SOM hierarchy was employed to separate normal from attack behavior [2]. However, for the first layer of the structure, fixed size SOMs were used. This implies that conservative estimates for the delay line depth are used to establish sufficient temporal properties for each feature. This results in a significant computational

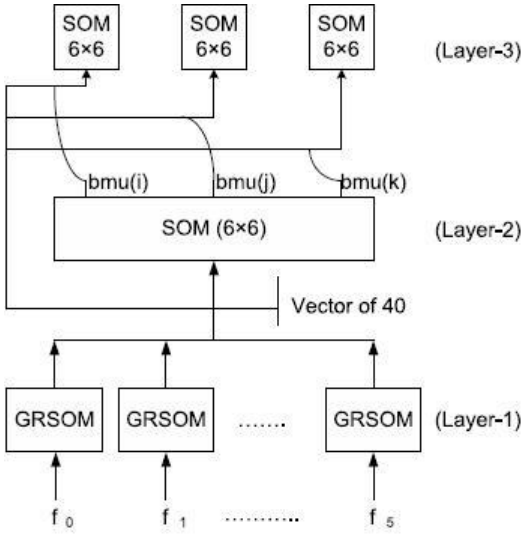

Fig. 1. 3 Layer GRSOM-SOM architecture

overhead and requires a post processing step to simplify the ensuing feature specific SOMs before building the second layer SOM. In this work, on the other hand, first layer maps take the form of growing RSOMs, which already have the ability to capture temporal patterns. To this end, we first describe the SOM and RSOM models as well as defining the proposed GRSOM architecture.

A. Self-Organizing Map

The principal goal of the SOM is to transform an incoming signal pattern of arbitrary dimension into a discrete one or two-dimensional map, where such a transformation is performed adaptively in a topologically ordered fashion [13]. The learning process is summarized as follows,

1) Establish map dimension;
2) Assign random numbers to the network weights, $w_{i,j}$;
3) Present an input pattern, x, in this case the distances of original inputs to the first level neurons;
4) Calculate the distance between the input, x, and each neuron weight w_j and identify the winning neuron as,

$$d = min_j||x - w_j|| \tag{1}$$

where $||.||$ is the Euclidean norm;

5) Adjust all weights in the neighborhood of the winning neuron as,

$$w_{i,j}(n+1) = w_{i,j}(n) + \gamma(n)h_i(n)(x_i(n) - w_{i,j}(n)) \tag{2}$$

where $h_i(n)$ is the value of the neighborhood function and $0 < \gamma(n)$ is the learning rate;

6) Repeat steps 2-4 until the convergence criterion is satisfied.

Once maps are trained, the best matching unit is used to facilitate the labeling of the higher level maps.

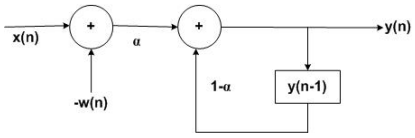

Fig. 2. Schematic picture of an RSOM unit

B. Recurrent Self-Organizing Map

The recurrent SOM is an extension to the Kohonen's SOM that enables neurons to compete to represent temporal properties in the data. RSOM is inspired by the Temporal Kohonen Map (TKM) [12]. In TKM, leaky integrators, which gradually lose their activity over time, are added into the outputs of the units in order to provide a short term memory mechanism. However, it is claimed in [9] that "The Temporal Kohonen Map [12] does not directly use the temporal contextual information of input sequences in weight updating at time t: The involvement of the previous inputs is only taken into account in the best matching unit searching". Therefore, the RSOM introduces leaky integrators (i.e. recurrent connectivity) to give a solution to this problem. These integrators are modeled in [9] with

$$y_i(n,\alpha) = (1-\alpha)y_i(n-1,\alpha) + \alpha(x(n) - w_i(n)) \tag{3}$$

where $w_i(n)$ is the weight vector, $x(n)$ is the input pattern, α is the memory coefficient and $y_i(n,\alpha)$ is the leaked difference vector for the unit i at step n while large α corresponds to short term memory whereas small values describe long term memory, or a slow activation decay. A schematic picture of an RSOM unit in [9] is shown in Fig. 2.

The best matching unit (bmu) b at step n is searched by

$$||y_b(n,\alpha)|| = min_{i \in V}||y_i(n,\alpha)|| \tag{4}$$

where V is the set of all neurons comprising the RSOM. Then, the map is adopted with a modified Hebbian training rule as

$$w_i(n+1) = w_i(n) + \gamma(n)h_{i,b}(n)y_i(n,\alpha) \tag{5}$$

where $h_{i,b}(n)$ is the value of the neighborhood function and $0 < \gamma(n) \leq 1$ is the monotonically decreasing learning rate factor. A common choice for the neighborhood function is a Gaussian

$$h_{i,b}(n) = exp\{-||r_i - r_b||^2/2\sigma(n)^2\} \tag{6}$$

where r_i are the map coordinates of the unit i and r_b are the map coordinates of the *bmu*. The width of the Gaussian bell is controlled by $\sigma(n)$ which is normally reduced as the learning processes.

In comparison to Kohonen's original TKM, the RSOM provides a more accurate model for temporal behavior of a single input. Conversely, the TKM provides a simple mechanism for memory, but accepts multiple input features per neuron. In this work, we propose to decouple the combined properties of the TKM by (1) introducing a growing version of the RSOM and (2) using the standard SOM to provide the spatial correlation process over multiple layers, Fig. 1.

III. METHODOLOGY

In this work, a growing behavior is added to the RSOM. Fig. 3, summarizes the adaptive model for the proposed GRSOM algorithm. Training of this model starts with only one neuron and new neurons are added until the desired accuracy is reached. For every unit that is newly added, the weight and the leaked difference vectors are initialized to zero. Then, an input from the training data is presented to the new neuron and weight and leaked difference vector values are adapted (Fig. 3 Step 3). The next input is presented to the neuron until the leaked difference value of the neuron is under a threshold, which is decided at the beginning of training. If the number of neurons is less than the desired maximum number of neurons, then a new neuron is added to the network until the end of the training data is reached (Fig. 3 Step 5). If the number of neurons is more than the desired maximum number of neurons, training is stopped and pruning algorithm starts (Fig. 3 Step 6). During the pruning phase, the neurons responding to the same or very similar data are deleted from the network and only one of them is kept to respond to the corresponding data. After pruning, training continues as long as there is more data for training or less neurons than desired.

1) i = 1; n = 0; MaxNeurons = 100;
2) neuron(i) $\leftarrow$ ($w_i(n)$ = 0, $y_i(n,\alpha)$ = 0);
3) Update $w_i(n)$ and $y_i(n,\alpha)$ using (3) and (1) respectively, n++;
4) IF ($||y_b(n,\alpha) \geq \beta_1||$)
   a) THEN Step 3
   b) ELSE i++;
5) IF $i < MaxNeurons$
   a) THEN Step 2;
6) $\forall j,k < MaxNeurons$
   a) IF $dist_{j \neq k}(w_j, w_k) < \beta_2$
      i) THEN delete neuron j, i- -;

Fig. 3. GRSOM algorithm

In short, the proposed system GRSOM begins with a single neuron and adds neurons until the desired accuracy is achieved. This in return, avoids the algorithm to begin by specifying a priori number of neurons and a tapped delay line approach, both of which are problem specific and therefore requires trial and error.

In addition to the growing behavior, a single pass algorithm is used for this work, which is also different than the original RSOM model. In [9], at each learning cycle a time t, $t \geq s$ is randomly selected for sampling a subsequence $\{x_{(t-s+1)}, x_{(t-s)}, ..., x_{(t)}\}$ of s samples to be presented to the RSOM starting at sample $x(t-s+1)$. However, GRSOM is a single pass algorithm, which starts the learning process from the first sample in dataset and terminates at the end of the sequence. This is a very important property as it means that the algorithm scales well and is applicable to online training.

IV. RESULTS

In this section, we describe the dataset used for training and testing phases of the network structure, training procedure and evaluation of the proposed architecture.

A. KDD-99 Dataset

The KDD-99 dataset is based on the 1998 DARPA initiative to provide designers of intrusion detection systems (IDS) with a benchmark on which to evaluate different methodologies [10]. To do so, a simulation is made of a factitious military network consisting of three target machines. Additionally, there are three machines to spoof different IP addresses to generate traffic between different hosts. Finally, a sniffer is used to record all network traffic using the tcpdump format. Normal connections are designed to reflect traffic seen on military bases and attacks fall into one of five categories: (i) Denial of Service; (ii) User to Root; (iii) Remote to Local; (iv) Data and (v) Probe. It should be noted here that Remote to Local and User to Root represent content-based attacks, and may only be detected indirectly by the type of system developed in this work, since we only use 6 network based features.

In 1999, the original tcpdump files were preprocessed for utilization in the Intrusion Detection System benchmark of the International Knowledge Discovery and Data Mining Tools Competition [7]. To do so, packet information in the tcpdump files are summarized into connections by using the Bro IDS [7]. As a result of this process, nine basic features of an individual TCP connection are formed:

- Duration of a connection;
- Protocol type (such as TCP, UDP or ICMP)
- Service type (such as FTP, HTTP, Telnet);
- A status flag (summarizing the connection);
- Total bytes sent to the destination host;
- Total bytes sent to the source host;
- Whether source and destination addresses are the same or not;
- Number of wrong fragments;
- Number of urgent packets;

In addition to these nine features, each connection is also described in terms of an additional 32 derived features, falling into three categories,

- Content Features;
- Time-based Traffic Features;
- Host-based Traffic Features;

Protocol and Service types in the basic features are not derived i.e. they are estimated immediately as opposed to after a connection has completed. Moreover, last three features of the basic features are specific to certain attack types, hence these terms are ignored in this work. Thus, only the first six features of the basic features are used to establish our structure as we expect the GRSOM to capture the temporal characteristics of the data.

The KDD-99 data is composed of several components as seen in Table I. Only the 10% KDD data is used for the purpose of training as in the case of the International

TABLE I
BASIC CHARACTERISTICS OF THE KDD DATASET

Dataset Label	Total Normal	Total Attack
10% KDD	97,277	396,744
Corrected (Test)	60,593	250,436
Category Composition		
Category	10%	Corrected
Normal	97,277	60,593
DoS	391,458	229,853
Probe	4,107	4,166
R2L	1,126	16,347
U2R	52	70

TABLE II
GRSOM TRAINING PARAMETERS

Parameter	Value
Initial Learning Rate	0.2
Epoch Limit	3000
α	0.6
β_1	10^{-17}
β_2	0.0005
Maximum Number of Neurons	100

TABLE III
SOM TRAINING PARAMETERS

Parameter	Rough Training	Fine Tuning
Initial Learning Rate	0.5	0.05
Epoch Limit		4000
Neighborhood Parameters		
Initial Size	2	1
Function	Gaussian	
Relation	Hexagonal	

Knowledge Discovery and Data Mining Tools Competition [2]. One side effect of this data can be seen as that it actually contains more examples of attacks than normal connections. Moreover, the attack types are not represented equally, Denial of Service attacks form the majority of the attack instances. However, Corrected (Test) dataset has a significantly different statistical distribution than 10% KDD and additional unseen attacks. Even though KDD has its drawbacks [15], it is still the only publicly available IDS benchmarking dataset.

B. Hierarchical Model

A hierarchical GRSOM-SOM structure similar to the one used in an earlier work [2] is employed. Our motivation is to build a first layer consisting of GRSOMs instead of SOMs with tapped delay lines. Specifically, three layers are employed, Fig. 1. In the first level, individual GRSOMs are associated with each of the six basic TCP features. This provides a concise summary of the representative patterns of each feature. The second layer integrates the feature specific views provided by the first level GRSOMs into a single view. In this layer, the training set labels are associated with each pattern to label the best matching unit. The third layer is built for the second layer neurons, which win for multiple classes. Therefore, third layer SOMs are associated with specific neurons in the second layer. This also results in the third layer SOMs being trained over a small fraction of the data set.

C. Preprocessing

Preprocessing of the original data set has two basic functions, to provide a suitable numerical representation for the initial data and normalization of the data. Specifically, three of the basic features, Protocol type, Service type and Status Flag, are alphanumeric. Therefore, each instance of these features are mapped to sequential integer values. Numerical features, duration of Connection, total bytes sent to destination and total bytes sent to source host, are used unchanged. After representing all instances of features numerically, data normalization applied to the data using;

$$1/(1 + x_t)$$

where x is the input data at the time t. On the second level, each connection is characterized by its distance to the first level neurons. Therefore, the input data for the second layer is prepared by using (1), thus defining the distance of inputs to the first level neurons without losing the temporal behavior.

D. Training

Learning parameters for the GRSOMs and their respective values are listed in Table II, whereas the parameters for the SOMs are listed in Table III. These parameters are repeated for every GRSOM and SOM comprising the hierarchy. In each case of SOM training, it is completed in two phases, rough training providing the general organization of the SOM and fine tuning of the neurons. The resulting hierarchy consists of 6 GRSOMs in the first layer, one for each of the first 6 basic features (Section III.A). The GRSOM algorithm returns RSOM networks consisting of 8, 4, 8, 8, 6 and 6 neurons respectively. This results in 40 inputs for the second layer 6×6 SOM, which specially correlates the feature specific GRSOM maps. After training the second level SOM, labeling takes place. For each connection in the training set, the corresponding label is given to the best matching unit in the second layer. The number of normal and attack connections that each best matching unit receives is kept for the purpose of building third layer SOMs. These 6×6 SOMs on the third layer are built for the second layer SOM neurons that wins for significant counts of more than one exemplar class, like Denial of Service and Probe or U2R and Denial of Service. This results in 3 SOMs being built on top of specific neurons of second layer. Fig. 3 and Fig. 4 summarize the counts of Denial of Service and Probe attacks respectively, where proportionally larger counts result in a greater area of hexagon being colored. It is clearly seen that neurons 5 and 17 are getting excited for both of these classes. However, when the third layer maps are built for these neurons, a better separation between these two classes is achieved as seen in Fig. 5 and Fig. 6.

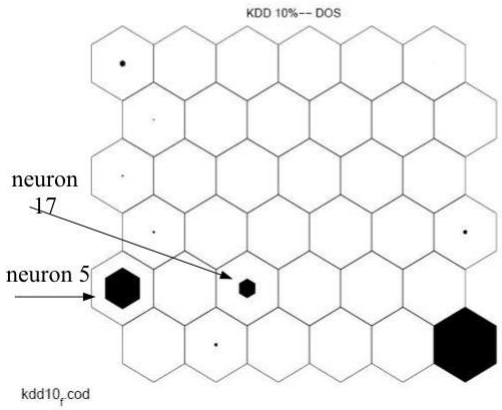

Fig. 4. Denial of Service hit histogram of the 2nd layer SOM

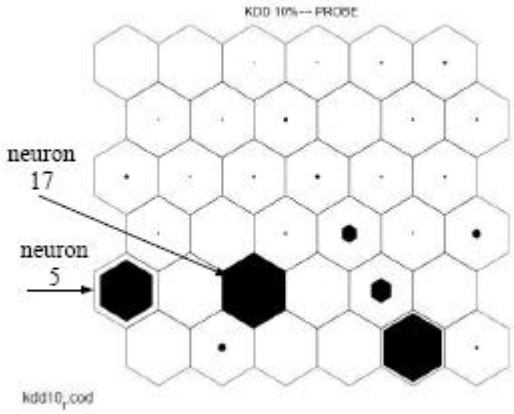

Fig. 5. Probe hit histogram of the 2nd layer SOM

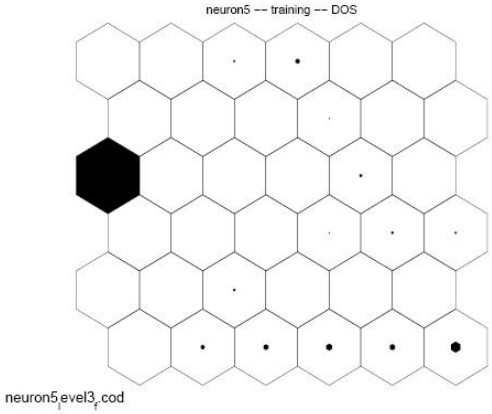

Fig. 6. Denial of Service hit histogram of the 2nd layer map's neuron 5

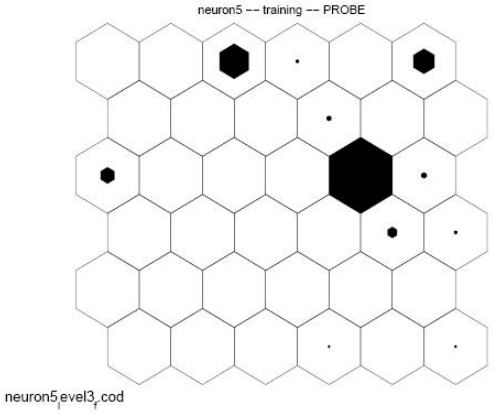

Fig. 7. Probe hit histogram of the 2nd layer map's neuron 5

E. Performance

Previous work with the standard SOM structure on the KDD data set has demonstrated that 33 hours are required for the training phase for the six first layer maps and a second layer map [2]. Given such an overhead, the DSS SOM algorithm was proposed for faster training [3]. However, to be able to give the ability of temporal behavior to SOMs, shift register was still in use for the first level. In this work, the built-in temporal behavior of GRSOM gives solutions to shift register parameterization problems in [2], whilst continuing to avoid any dependency on a priori provision of temporal features [3]. Needless to say it would be appropriate to employ DSS SOM in place of the second and third layer SOMs, Fig. 1. Moreover, performance of the classifier is evaluated in terms of false positive and detection rates, estimated as:

$$DetectionRate = 1 - \frac{NumberofFalseNegatives}{TotalNumberofAttackConnections}$$

$$FalsePositiveRate = \frac{NumberofFalsePositives}{TotalNumberofNormalConnections}$$

where False Positive (Negative) is the number of normal (attack) connections labeled as attack (normal).

Tables IV and V detail the performance on the training and test data sets respectively for two and three layer maps. Finally, performance of the two-layer and three-layer hierarchies on Corrected Test set is summarized in Table VI. It is clear that the larger classes of Normal and DoS are easily recognized by the system. Performance of Probe is also reasonable over 50% detection on test set, whereas, the content based nature of the U2R and R2L classes penalized the generalization of these classes on the test set. That is to say, the approach is only utilizing the first six basic features (non-content based features), performance on network-based attack categories will naturally be better than the content-based categories. Table VII provides a summary of previous results from alternative approaches using this data set. In comparison with these results, the GRSOM-SOM hierarchy appears to produce competitive results. Computationally,

TABLE IV

PERFORMANCE OF 2 LAYER AND 3 LAYER HIERARCHIES ON TRAINING DATA (10% KDD)

	Layer 2				
	Normal	DoS	Probe	U2R	R2L
Detection Rate	98.5%	99.2%	36.67%	0%	11%
False Positive Rate	1.08%	4.43%	0.0002%	0%	0.006%
	Layer 3				
	Normal	DoS	Probe	U2R	R2L
Detection Rate	98.5%	99.1%	85.78%	23.08%	15.36%
False Positive Rate	1.09%	1.86%	0.006%	0.002%	0.007%

TABLE V

PERFORMANCE OF 2 LAYER AND 3 LAYER HIERARCHIES ON KDD TEST

	Layer 2				
	Normal	DoS	Probe	U2R	R2L
Detection Rate	94.63%	96.59%	8.64%	0%	3.33%
False Positive Rate	9.38%	8.86%	0.034%	0%	0.008%
	Layer 3				
	Normal	DoS	Probe	U2R	R2L
Detection Rate	94.52%	96.37%	53.89%	10%	3.42%
False Positive Rate	9.38%	6.26%	0.26%	0.03%	0.008%

our hierarchy is faster than standard SOM hierarchy (15 hours versus 33 hours) with a slightly better detection rate. Moreover, while training of the second layer SOM takes 7 hours of total training time, training of 6 different GRSOMs of the first layer takes 8 hours, which is slightly more than an hour for each GRSOM. Naturally, all the data mining approaches utilize all 41 features of the data set (rows 1 to 4 of Table VII), while our hierarchy uses only 6.

V. CONCLUSIONS AND FUTURE WORKS

A hierarchical GRSOM and standard SOM approach to mining data sequences is proposed and demonstrated on International Knowledge Discovery and Data Mining Tools Competition intrusion detection benchmark [7]. The built-in sequence processing and the self-growing abilities of this model are pointed out specifically. In comparison to the data mining approaches previously proposed, this system provides competitive results whilst utilizing only a small subset of the feature set. Moreover, this approach is twice as fast as an approach based on the standard SOM model throughout. Also, the training of an SOM takes hours, while it only takes an hour for a GRSOM in our model. Furthermore, the single pass algorithm used in this work is a very important property as it means that the algorithm scales well and is applicable to online training.

Future work will naturally consider the application of this hierarchical model to additional datasets of similar attributes with document and DNA sequencing.

TABLE VI

PERFORMANCE OF 2 AND 3 LAYER HIERARCHY ON KDD TEST

	Detection Rate	False Positive Rate	Training Time(sec)
Level 2	89%	5.37%	54,550
Level 3	89.6%	5.66%	59,735

TABLE VII

PREVIOUS RESULTS ON KDD TEST USING UNSUPERVISED LEARNING

Technique	Detection Rate	False Positive Rate
Data-Mining [8]	70-90%	2%
Clustering [14]	93%	10%
SVM [14]	98%	10%
K-NN [14]	91%	8%
Standard SOM hierarchy [2]	89%	4.6%

REFERENCES

[1] P. Lichodzijewski, A.N. Zincir-Heywood, M.I. Heywood, "Host-Based Intrusion Detection Using Self-Organizing Maps", *IEEE International Joint Conference on Neural Networks*, pp. 1714-1719, May 2002.

[2] H.G. Kayacik, A.N. Zincir-Heywood, M.I. Heywood, "On the Capability of an SOM based Intrusion Detection System", *IEEE-INNS International Joint Conference on Neural Networks*, pp. 1808-1813, 2003.

[3] S.T. Sarasamma, Q.A. Zhu, J. Huff, "Hierarchical Kohonen Net for Anomaly Detection in Network Security", *IEEE Transactions in Systems, Man and Cybernetics, Part B: Cybernetics*, vol.35(2), pp. 302-312, April 2005.

[4] M. Dittenbach, D. Merkl, A. Rauber, "The Growing Hierarchical Self-Organizing Map", *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, vol. 6, pp. 15-19, July 2000.

[5] J. Zhou, Y. Fu, "Clustering High-Dimensional Data Using Growing SOM", *Advances in Neural Networks ISNN 2005*, Springer (2005), vol.3497, pp. 63-68.

[6] Y. Liu, D. Tian, B. Li, "A Wireless Intrusion Detection Method Based on Dynamic Growing Neural Network", *Computer and Computational Sciences, 2006. IMSCCS '06. First International Multi-Symposiums on Computer and Computational Sciences (IMSCCS'06)*, vol. 2, pp. 611-615, April 2006.

[7] S. Hettich, S.D. Bay, *The UCI KDD Archive*, Irvine, CA: University of California, Department of Information and Computer Science, http://kdd.ics.uci.edu, 1999.

[8] W. Lee, S. Stolfo, K. Mok, "A Data Mining Framework for Building Intrusion Detection Models", *Proceedings of the 1999 IEEE Sypmposium on Security and Privacy*, pp. 120-132,1999.

[9] M. Varsta, J. Heikkonnen, J. D. R. Millan, "Context Learning with the Self-Organizing Map", *Proceedings of Workshop on Self-Organizing Maps*, Helsinki University of Technology, pp. 197-202, 1997.

[10] *The Intrusion Detection Off-Line Evaluation Plan*, MIT Lincoln Lab., Information Systems Technology Group, http://www.ll.mit.edu/IST/ideval/docs/1998/id98-eval-ll.txt, March 1998.

[11] T. Kohonen, *Self-Organizing Maps*,3rd Ed., Springer-Verlag, ISBN 3-540-67921-9, 2000.

[12] G. J. Chappell, J. G. Taylor, "The Temporal Kohonen Map", Neural Networks, vol. 6, pp. 441-445, 1995.

[13] S. Haykin, *Neural Networks*, 2nd Ed., Prentice Hall, pp. 446,1999.

[14] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, S. Stolfo, "A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data", in *Applications of Data Mining in Computer Security*, Chapter 4, D. Barbara and S. Jajodia (editors), Kluwer, ISBN 1-4020-7054-3, 2002.

[15] J. McHugh, "Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory", *ACM Transactions on Information and System Security*, 3(4), pp. 262-294, 2001.