Hierarchical Graphs for Data Clustering

E.J. Palomo[1], J.M. Ortiz-de-Lazcano-Lobato[1], Domingo López-Rodríguez[2], and R.M. Luque[1]

[1] Department of Computer Science
E.T.S.I.Informatica, University of Malaga
Campus Teatinos s/n, 29071 – Malaga, Spain
{ejpalomo,jmortiz,rmluque}@lcc.uma.es
[2] Department of Applied Mathematics
E.T.S.I.Informatica, University of Malaga
Campus Teatinos s/n, 29071 – Malaga, Spain
dlopez@ctima.uma.es

Abstract. The self-organizing map (SOM) has been used in multiple areas and constitutes an excellent tool for data mining. However, SOM has two main drawbacks: the static architecture and the lack of representation of hierarchical relations among input data. The growing hierarchical SOM (GHSOM) was proposed in order to face these difficulties. The network architecture is adapted during the learning process and provides an intuitive representation of the hierarchical relations of the data. Some limitations of this model are the static topology of the maps (2-D grids) and the big amount of neurons created without necessity. A growing hierarchical self-organizing graph (GHSOG) based on the GHSOM is presented. The maps are graphs instead of 2-D rectangular grids, where the neurons are considered the vertices, and each edge of the graph represents a neighborhood relation between neurons. This new approach provides greater plasticity and a more flexible architecture, where the neurons arrangement is not restricted to a fixed topology, achieving a more faithfully data representation. The proposed neural model has been used to build an Intrusion Detection Systems (IDS), where experimental results confirm its good performance.

Keywords: Data clustering, self-organization, graph-based representation, hierarchical clustering, intrusion detection systems.

1 Introduction

Data clustering is an unsupervised learning method to find an optimal partitioning of the data set. This partitioning organizes the input data into clusters according to a similarity measure, where data belonging to one cluster are more similar than data belonging to different clusters. Generally speaking, unsupervised learning methods are specially useful when we have unlabeled input data, that is, the groups in which data are classified are unknown. The purpose of these methods is to discover groups in a data set based on similarity, where input data are usually represented as feature vectors of high dimensionality.

J. Cabestany et al. (Eds.): IWANN 2009, Part I, LNCS 5517, pp. 432–439, 2009.

The self-organizing map (SOM) is a widely used unsupervised neural network for clustering high-dimensional input data and mapping these data into a two-dimensional representation space [1]. However, it has some drawbacks. To begin with, the number and arrangement of neurons (network architecture) of the SOM is static and has to be established in advance. This task can be difficult because it is needed a prior study of the problem domain, specially when we have vectors with many features. Also, inherent hierarchical relations among input data are not represented on the map and can make difficult the interpretation of the data.

The growing hierarchical SOM (GHSOM) tries to face these problems derived from SOM. The GHSOM has a hierarchical architecture arranged in layers, where each layer is composed of different growing SOMs expanded from neurons of the upper layer maps and the number of neurons of each map are adaptively determined [2]. This way, the architecture of the GHSOM is established during the unsupervised learning process according to the input patterns. However, this neural model keep showing some difficulties related to its static topology. In fact, each map is initiated with 2x2 neurons, forcing the map to be a 2-D rectangular grid. The problem of this topology is that we need to preserve the rectangular grid and when the map grows, not only a neuron but a row or a column of neurons has to be inserted. As a result, a lot of neurons are inserted without necessity, being the amount of neurons far from optimal, especially when we have to insert neurons in very large maps. Moreover, there are wide spectrum application domains where a rectangular grid is not the most suitable topology to represent the input data. In order to make easy the identification of relations among input data and mirror its inherent structure, the map should capture the own topology of the data as faithfully as possible.

A graph-based representation space provides the enough flexibility to reflect different representations of the same domain (data and relations) and is easy to understand. Most of the related works which use SOMs with a graph-based representation, treat each neuron as a graph since they use this model for representing input data [3,4]. The goal is to cluster these graphs. Here the goal is different; we use graphs to represent the topology of a SOM. Thus, the representation space can totally adapt to the data structure.

In this paper, by using this graph-based representation in hierarchical self-organization, we propose a novel artificial neural network architecture based on the GHSOM called the growing hierarchical self-organization graph (GHSOG). This new model has the advantages of the GHSOM model and also provides a better and easy data representation thanks to a more flexible and adaptive topology and a lower number of neurons used in the representation.

In order to check the performance of the GHSOG, an Intrusion Detection System (IDS) has been implemented using this new model. An IDS monitors network traffic to detect an attack or intrusion in a network environment. Some anomaly detection systems using data mining techniques such as clustering, support vector machines (SVM) and neural network systems have been proposed [5,6]. Many self-organizing models have been used to implement an IDS,

however they have many difficulties detecting a wide variety of attacks with low false positive rates [7].

The remainder of this paper is organized as follows. In Section 2, we first present a description of our new GHSOG model. In Section 3, an IDS is implemented with the GHSOG model, where data preparation and some experimental results are presented. Some remarks conclude this paper in Section 4.

2 Growing Hierarchical Self-Organizing Graph

The Growing Hierarchical Self-Organizing Graph is a neural network based on the GHSOM, where only one neuron is added when a self-organizing map should be expanded. This way, the topology that the tree nodes present is a graph instead of a grid. Initially, a segment composed of two neurons and a connection edge is used to represent a cluster. If more neurons are needed to capture the cluster knowledge accurately, then these neurons are added one by one, ensuring that each new neuron forms a triangle tile in the graph. See Figure 1.

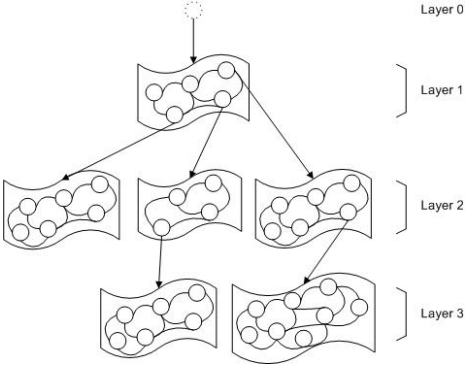

Fig. 1. A sample of the GHSOG model

Let $X = x_1, x_2, \ldots, x_L$ the training data set consisting of L samples, the starting point of the training algorithm is the computation of the *quantization error* of the whole training data set

$$qe_0 = \sum_{x_j \in X} \|w_0 - x_j\| \tag{1}$$

with w_0 the global mean. Using the quantization error instead of the mean quantization error gives the method the capacity of assigning more neurons in the map for more densely populated regions of the input space. The first self-organizing map is created at layer 1, i.e., at depth level 1 in the tree. This map consists of two neurons which are linked by a unique connection.

Then for each local graph, i.e. each self-organizing graph, a two phase learning process begins. The first phase carries out the training (and growing if required) of the local self-organizing graphs; the second one tries to expand that neurons which required a more thorough clustering of the data they represent. For that purpose a new graph in the next tree layer is created.

In the *local training and growing phase* each local graph is trained following the learning process proposed by Kononen in [1] except for the neighborhood function. In the proposed network it is considered that the neighbors of a neuron are all the neurons which are connected directly in the graph.

It should be noticed that the samples used to train each local graph are those belonging to the cluster corresponding to the father neuron in the tree, i.e., if f is the father neuron then the training data set for the child graph is $C_f \subseteq X$ where C_f is the subset of vectors of the input data that is mapped onto unit f.

After training the local graph, the learning algorithm checks if the graph must grow.

$$\frac{1}{|U_f|} \sum_{i \in U_f} qe_i < \tau_1 \cdot qe_f \tag{2}$$

where $qe_i = \sum_{x_j \in C_f} \|w_i - x_j\|$ is the absolute quantification error and U_f the neurons in the hierarchical tree which are part of the local graph, expanded from neuron f. The expression compares the mean quantization error of the graph with the quantization error of the corresponding neuron f in the upper layer. The parameter τ_1 allow the user to control the depth/shallowness of the obtained hierarchical structure. If the criterion is not satisfied then a new neuron is added and this phase starts again. This iterative process finishes when the expression holds true.

When the mean quantization error of the graph is too high and does not fulfill the stopping criterion, a new neuron is added to the graph in order to reduce that error. Therefore, we look for the neuron of the graph which contributes most to the quantization error. This neuron is named the *error neuron*

$$e = \arg\max_i \sum_{x_j \in C_i} \|x_j - w_i\| \tag{3}$$

A second neuron is needed to keep the triangle tile based structure of the local graphs. The selected neuron will be the most dissimilar neighboring neuron d of the error neuron e

$$d = \arg\max_i (w_e - w_i), w_i \in N_e \tag{4}$$

with N_e the set of neighbours neurons of the error neuron e. Finally the location of the new neuron in the input space, which coincides with the weight vector of the neuron is

$$w_{new} = \frac{qe_e \cdot w_e + qe_d \cdot w_d}{qe_e + qe_d} \tag{5}$$

Note that the most dissimilar neuron d of the error neuron e is the farthest of its neighbors and, therefore, most of the time the new neurons are placed near

the error neuron, which is the region in the input space where the quantization error is greater and help is needed.

Once the first iterative phase has finished, the *neurons expansion phase* starts. In this step for each neuron of the graph we compute the checking

$$qe_i = \tau_2 \cdot qe_0 \tag{6}$$

where τ_2 is a user parameter which controls the desired quality for the data representation. If the expression holds true for all the neurons the training of the current local graph finishes and the process continues with another local graph (in the same layer of the tree or in deeper layers). Otherwise for each neuron which does not satisfies the criterion a new graph is created in the next layer of the tree. The new graph is a graph consisting of two neurons, whose weigth vectors are the average between the father vector and the first and second more similar neighbor vectors, respectively. If the father has only one neighbor, the second expanded neuron is a copy of the father.

3 Experimental Results

In order to prove the performance of the new GHSOG model, an IDS has been implemented, which detects a wide variety of attacks in a military network environment. GHSOG training and testing have been done with the KDD Cup 1999 benchmark data set created by MIT Lincoln laboratory. The purpose of this benchmark was to build a network intrusion detector capable of distinguishing between intrusions or attacks, and normal connections. To make easy the training and testing, the 10% training data set and the 10% testing data set are provided, which contain 494,021 and 311,029 connection records, respectively. In the training data set there are 22 attack types in addition to normal records whereas in the testing data set 15 new attack types are also present.

3.1 Data Preparation

In the KDD 1999 data set, each connection record has 41 features preprocessed from monitoring network packets. Among these features, 3 are qualitative: the protocol type, which can take the values TCP, UDP or ICMP; the service type, with 66 different values and the flag, with 11 possible values.

Most of the related works map these data into consecutive quantitative values so as to compute the similarities of the connections. However, since qualitative data have no an order associated, the use of the Euclidean distance is not appropriate for these types of data. To solve this problem, each qualitative feature has been replaced with a binary vector composed by as many binary features as different possible values that feature can take. Thus, the distance among qualitative values is always the same allowing the use of the Euclidean distance or other standard metric. This way, the number of features has been increased from 41 to 118.

3.2 Results

For our experiments, two data subsets have been randomly selected from the 10% training data set: S1 with 100,000 connection records and S2 with 169,000 connection records. The 22 attack types in addition to normal records are contained in both data subsets, and the distribution of each connection type reflects the own distribution of the 10% training data set. In order to train two neural networks with the two data subsets, 0.1 and 0.03 have been chosen as values for parameters τ_1 and τ_2, respectively. The resulting architecture of the first neural networks trained with S1 is shown in Fig. 1. The second one has 1 layer more, but with just a graph in that layer.

Most of the related works are only interested in knowing whether the input pattern is an anomaly or a normal connection record. Here, our interest is also to know whether one connection record type is identified as from its own type. We name this identification rate. Detection rate is the percentage of detected attacks and false positive rate is the percentage of normal connections records detected as attacks. The training results for S1 and S2 are given in Table 1. We can see that the detection rate is around the 96% for both data sets, but the false positive rate is higher for the first data subset.

Table 1. Training results for S1 and S2 data subsets

Training Set	Detection (%)	False Positive (%)	Identification (%)
S1	96.75	3.81	94.64
S2	96.44	1.42	95.40

Table 2. Testing results for GHSOGs trained with S1 and S2 data subsets

Training Set	Detection (%)	False Positive (%)	Identification (%)
S1	90.15	2.09	90.68
S2	91.48	4.51	90.57

After training, the two trained GHSOGs have been tested with the 10% KDD 1999 testing data set, which contains 311,029 connection records and 15 new attack types as mentioned above. Testing results are shown in Table 2. During the testing, we achieve similar detection rates for both GHSOGs but the false positive rate is higher in the second GHSOG. Note that the identification rate is higher than the detection rate in the first GHSOG. Although it can seem strange, this is due to the fact that normal connections are taken into account in addition to attack connections to compute the identification rate.

There are many related works that have used self-organizing maps to implement an IDS. A hierarchical Kohonen net (K-Map) was proposed in [8], which consists of three static SOMs arranged in layers. They achieved 99.63% detection rate after testing, but taking into account several limitations. Although they

Table 3. Testing results for differents IDSs based on self-organization

	Detected(%)	False Positive(%)	Neurons
GHSOG	90.68	2.09	35
K-Map	99.63	0.34	144
SOM	97.31	0.042	400
SOM (DoS)	99.81	0.1	28800

used a data subset of 169,000 connection records with 22 attack types from the KDD Cup 1999 data set as we used, during the testing they used just 3 attack types, whereas we used 38 attack types, where 15 attack types were unknown. They also used a combination of 20 features that had to be established in advance and 48 neurons in each layer. In [9], a SOM trained on all the 41 features was chosen in order to compare results. A 97.31% detection rate was achieved but using 400 neurons. An emergent SOM (ESOM) for the Intrusion Detection process was proposed in [10]. They achieved detection rates between 98.3% and 99.81% and false positives between 2.9% and 0.1%. However, they just detected denial of service attacks (DoS), they used a pre-selected subset of 9 features and a number of neurons between 160x180 and 180x200. In addition, their best result (99.81% detection rate and 0.1% false positive rate) was just trained and tested with one DoS attack type, the smurf attack. These different results are summarized in Table 3. Note that our detection rates are lower than the rest of the proposed IDSs. Nevertheless, we provide a more simple architecture with less neuron, which was automatically generated during the training, and where the data hierarchical structure is easier to understand.

4 Conclusions

In this paper, a growing hierarchical self-organizing map (GHSOG) has been proposed. This novel neural network faces the limitations of the SOM related to its static architecture and the lack of representation of hierarchical relations among input data. The GHSOM model also faces these problems of the SOM, but keep maintaining some limitations: the static topology of the maps (2-D grids) and the big amount of neurons created without necessity. The proposed GHSOG has a topology based on graphs instead of 2-D rectangular grids, where the neurons are considered the vertices, and each edge of the graph represents a neighborhood relation between neurons. This graph-based representation provides greater plasticity and a more flexible and adaptive architecture, where the neurons arrangement is not restricted to a fixed topology, achieving a more faithfully data representation.

The performance of the GHSOG has been tested by implementing an Intrusion Detection System (IDS). To train and test the neural network, the KDD Cup 1999 benchmark data set has been used. During the training, two different data subsets has been selected from the 10% training KDD data set, with 100,000

and 169,000 connection records, respectively. The two generated GHSOG architectures were tested with the 10% training KDD data set, which contains 311,029 connection records. After testing, we achieved a 90.15% detection rate and a false positive rate of 2.09% as shown in Table 2. Comparison results are also provided. Although testing results cannot seem as good as in other related works, these just are limited to specific subsets of features and/or attacks, which were established in advanced and the number of neurons are higher than we used. Furthermore, our architecture was generated during the training adapting to input data, mirroring their hierarchical relations and achieving a more faithfully data representation.

Acknowledgements

This work is partially supported by Spanish Ministry of Science and Innovation under contract TIN-07362, project name Self-Organizing Systems for Internet.

References

1. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological cybernetics 43(1), 59–69 (1982)
2. Rauber, A., Merkl, D., Dittenbach, M.: The growing hierarchical self-organizing map: Exploratory analysis of high-dimensional data. IEEE Transactions on Neural Networks 13(6), 1331–1341 (2002)
3. Phuc, D., Hung, M.: Using som based graph clustering for extracting main ideas from documents. In: IEEE International Conference on Research, Innovation and Vision for the Future, pp. 209–214 (2008)
4. Hussin, M., Farra, M., El-Sonbaty, Y.: Extending the growing hierarchal som for clustering documents in graphs domain. In: IEEE International Joint Conference on Neural Networks, pp. 4028–4035 (2008)
5. Maxion, R., Tan, K.: Anomaly detection in embedded systems. IEEE Transactions on Computers 51(2), 108–120 (2002)
6. Tan, K., Maxion, R.: Determining the operational limits of an anomaly-based intrusion detector. IEEE Journal on Selected Areas in Communications 21(1), 96–110 (2003)
7. Ying, H., Feng, T.J., Cao, J.K., Ding, X.Q., Zhou, Y.H.: Research on some problems in the kohonen som algorithm. In: International Conference on Machine Learning and Cybernetics, vol. 3, pp. 1279–1282 (2002)
8. Sarasamma, S., Zhu, Q., Huff, J.: Hierarchical kohonenen net for anomaly detection in network security. IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics 35(2), 302–312 (2005)
9. DeLooze, L., AF DeLooze, L.: Attack characterization and intrusion detection using an ensemble of self-organizing maps. In: 7th Annual IEEE Information Assurance Workshop, pp. 108–115 (2006)
10. Mitrokotsa, A., Douligeris, C.: Detecting denial of service attacks using emergent self-organizing maps. In: 5th IEEE International Symposium on Signal Processing and Information Technology, pp. 375–380 (2005)