An Hybrid Architecture for Active and Incremental Learning: The Self-Organizing Perceptron (SOP) Network

Jean-François Hébert, Marc Parizeau
Laboratoire de vision et systèmes numériques
Université Laval
Ste-Foy (Qc), Canada, G1K 7P4
{jfhebert,parizeau}@gel.ulaval.ca

Nadia Ghazzali
Dept. de mathématiques et de statistique
Université Laval
Ste-Foy (Qc), Canada, G1K 7P4
ghazzali@mat.ulaval.ca

Abstract

This paper describes a new hybrid architecture for an artificial neural network classifier that enables incremental learning. The learning algorithm of the proposed architecture detects the occurrence of unknown data and automatically adapts the structure of the network to learn these new data, without degrading previous knowledge. The architecture combines an unsupervised self-organizing map with a supervised Perceptron network to form the Self-Organizing Perceptron network (SOP).

1 Introduction

Neural network training may be passive, active or incremental. Passive learning is a one shot process where training is conducted on all available data. In contrast, an active strategy allows some form of interaction between learner and teacher. For example, an active process can concentrate on specific regions of the input domain. Incremental learning differs from passive and active learning in that it does not assume that all input data are available a priori. The main reasons for wanting active and incremental learning in neural networks are: 1) to allow training by parts on huge data sets, 2) to enable learning of new knowledge, 3) to adapt to knowledge evolution, and 4) to reinforce current knowledge. In order to achieve incremental learning, however, it appears necessary that the number of free parameters in the network be allowed to change over time, which translates to adding both new neurons and new connections in order to build constructive neural networks [1, 2, 3, 4, 5].

The constructive architecture proposed in this paper combines a self-organizing neural network linked with a multi-layered feed-forward network. The idea is to take advantage of the modeling abilities of the self-organizing network for clustering the feed-forward network into specialized groups of neurons. This neural architecture can learn basic knowledge using an initial passive training, and subsequently, it can adapt its knowledge to learn new input data with an incremental learning strategy. The proposed strategy is based on local vigilance data used to model previous knowledge.

The rest of this paper is organized as follows. Section 2 first reviews the motivation behind incremental learning. Section 3 presents the proposed hybrid neural network architecture and describes its passive learning algorithm. Section 4 next defines the incremental learning strategy of this architecture. Finally, Section 5 shows some experimental results that confirm its efficiency.

2 Motivation

Multilayer Perceptrons (MLP) have been in used for many years for solving pattern classification problems, and have proven to perform well in the context of passive learning. However, the MLP have many defects that are well known and documented. For example, they are almost useless in the context of incremental learning since they tend to forget very quickly what they have learned previously when presented with new data. This is the so-called moving target problem: since neurons on a layer do not communicate with one another, each neuron decides independently which part of the classification problem it will tackle [1]. Another drawback of the MLP is its inability to draw closed separation surfaces in the input space. That unsuitable behavior has been mathematically proven by M. Gori and F. Scarselli [6], who conclude as a direct consequence that the MLP is definitively inadequate for application of pattern recognition requiring a reliable rejection criteria.

To overcome both the moving target problem and the open separation surfaces problem, a conceivable solution could be to specialize each hidden neuron by restricting them to act only within a localized region (a cluster) of the input space. Radial basis functions (RBF) networks [7, 8] were derived from that idea. Because of their local learning properties, the RBF-like networks seems a priori to be

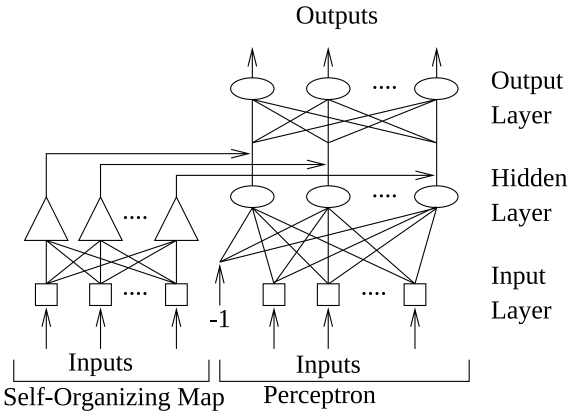

Figure 1: *SOP network architecture.*

better adapted than the MLP structure to achieve incremental learning. However, they also have significant defects which make them less attractive. In particular, the RBF-like networks usually require a very large number of neurons in order to learn a smooth transformation between the input-output space [9], and more training data are required to achieve similar precision to that of the MLP network [10]. In addition, the original RBF structure is not constructive, that is, the number of neurons is fixed over time and can not be increased (or reduced) during the training phase. Several authors [2, 11, 12] have proposed constructive extension to RBF networks. However, incremental learning possibilities are also limited for those networks since they usually consider that all input data are available a priori. Other constructive architectures also exist, such as the family of the ART networks [5], but within our current framework, we focused on feedforward architectures.

In conclusion, since the most currently used feedforward neural architecture are unusable for incremental learning, it appeared necessary to us to design a new network architecture specially adapted for incremental learning. This network is presented in the following sections.

3 Self-Organizing Perceptron (SOP)

The basic idea behind the Self-Organizing Perceptron is to take advantage of the modeling abilities of an already trained unsupervised Self-Organizing Neural Network (SONN; see Section 3.1) to cluster a multilayer Perceptron network into specialized groups of neurons. In contrast with RBF-like networks, where the outputs of the unsupervised network serve as the inputs of a classifying algorithm, the SOP network rather uses the self-organizing network outputs to balance the activation signals for the output layer of a Perceptron network. Indeed, as shown in Figure 1, each neuron $c_1, c_2, \ldots, c_q$ of a SONN is linked with a specific neuron located on the last hidden layer of a Perceptron network. The clusterisation is achieved through the computation for each neuron of the SONN of a selection factor p_{c_i}, $i = 1, 2, \ldots, q$ which balances in regard to an input exemplar $\xi \in \mathbb{R}^n$ the output of the ith hidden neuron of the Perceptron. Given an input datum ξ, its selection factor $p_{c_i}(\xi)$ for neuron c_i is computed as:

$$p_{c_i}(\xi) = \exp\left(-\frac{||\omega_i - \xi||^2}{\sigma_i^2}\right), \tag{1}$$

where $\omega_i \in \mathbb{R}^n$ is the position of neuron c_i in the input space, $\sigma_i \in R^+$ is a parameter which controls the sphere of influence for neuron c_i, and $||\cdot||$ denotes the Euclidean vectorial norm. The resulting selection factors are then used to balance the activation signal a_j, of the j^{th} output neuron in the MLP:

$$a_j = \sum_{i=1}^{q} p_{c_i}\, w_{ji}\, s_i, \tag{2}$$

where s_i is the output signal provided by the i^{th} hidden neuron of the MLP and w_{ji} is the synaptic weight between that hidden neuron and the j^{th} output neuron.

The SOP network can be passively trained with an hybrid learning process. First, an unsupervised learning algorithm is used to determine the position and the sphere of influence of the self-organized neurons. A supervised learning algorithm such as the classic backpropagation algorithm then serves to adapt the different synaptic weights of the Perceptron network. Note that the backpropagation equations are slightly modified to take into account the influence of the selection factors.

Both equations (1) and (2) ensure that input data located near each other in the input vector space will tend to activate the same group of neurons in the MLP during the supervised training of the SOP network. In contrast with standard backpropagation training, the use of selection factors to control network outputs allows to freeze the weight adjustments for inactive parts of the MLP. It is that feature that leads to a possible incremental learning for the SOP network.

3.1 Choice of the self-organizing network

Basically, the goal of the unsupervised self-organizing network for the SOP network is to generate a mapping from an original high-dimensional input space to a lower-dimensional topological structure which preserves the neighborhood relations contained in the input data. That mapping requires that:

- Data near each other in the input space should be assigned to neighboring neurons in the topological structure.

- Dense regions of the input space should be assigned to more neurons.

Kohonen's Self-Organizing Map (SOM) [13], Fritzke's Growing Cell Structures (GCS) [2], and Fritzke's Growing Neural Gas Network (GNG) [14] are three popular self-organized networks using a similar learning algorithm, but having a very different structure. The SOM uses a fixed rectangular topological map which is user-defined at the beginning of the learning process. In contrast, the GCS is a constructive network formed of a set of k-dimensional hypertetrahedrons[1] which grows over time. The GNG network is also a constructive network, but in contrast with the SOM and the GCS networks, it imposes no explicit constraints on the organization of the neuron map. Rather, a competitive Hebbian learning rule is defined to continuously updated the map. Each neuron $c \in \mathcal{C}$ of a SOM, a GCS, or a GNG is fully connected by a weight vector $\omega_c \in I\!R^n$ to each input vector $\xi \in I\!R^n$.

In regard to their learning process, the three networks use the same basic principle which states that at each iteration, an input datum attracts toward him the best-matching neuron $b \in \mathcal{C}$ and its set of topological neighbors $N_b \subset \mathcal{C}$. At each iteration, the updated weight vectors ω_c^{new} are computed as follows:

$$\omega_c^{\text{new}} = \omega_c^{\text{old}} + \epsilon_c(\xi - \omega_c^{\text{old}}) \tag{3}$$

where ϵ_c is an adaptation strength function such that:

$$\epsilon_c = \begin{cases} \epsilon_b & \text{if } c = b \\ \epsilon_n & \text{if } c \in N_b \\ 0 & \text{otherwise (i.e. } c \notin N_b \cup \{b\}) \end{cases} \tag{4}$$

and ϵ_b, and ϵ_n are two adaptation strength factors. In Kohonen's algorithm, both ϵ_b and ϵ_n decrease over time while they are kept constant in Fritzke's algorithms. The growth process of the GCS and of the GNG consists in the insertion of a new neuron each time a constant number λ of adaptation steps are completed using Equation (3) (see [2] et [14] for details).

The simulation results illustrate in Figures 2b-d show that the GNG achieves for the uniform data of Figure 2a a better mapping than the SOM and the GCS. Indeed, all the GNG neurons (represented by small dots and linked to their immediate neighbors with edges) are located in the input space near regions of intense activities, whereas for the SOM and the GCS, their structural constraints have likely had a negative effect over their mapping. In practice, the GNG network has proven to perform better then the SOM and the GCS networks according to different criteria, and thus appears to be a good candidate for the unsupervised part of the SOP network.

[1] A k-dimensional hypertetrahedron contains $k+1$ neurons with $(k+1)k/2$ edges connecting each cell to all others. Note that $k=1$ forms a line and $k=2$ forms a triangle.

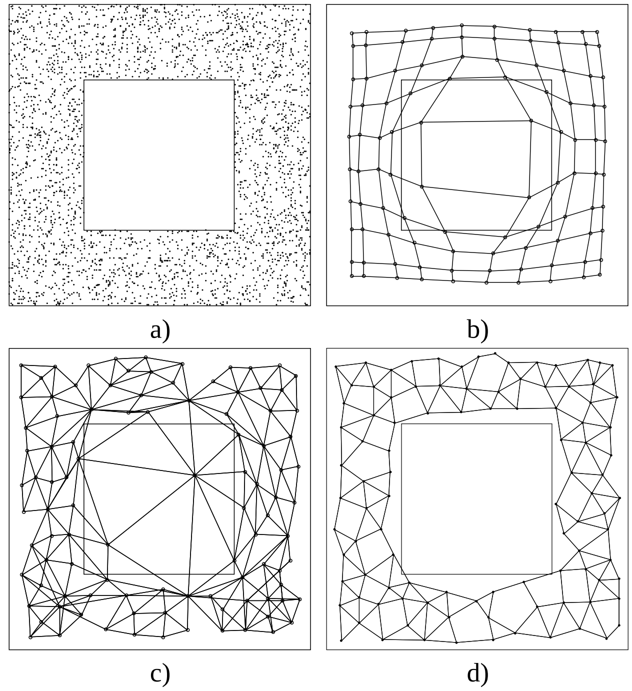

Figure 2: *a) Simulated uniform data and their resulting modeling by b) a SOM , c) a GSC, and d) a GNG. The three final networks are composed of 100 neurons.*

When embedded into a hybrid SOP network, a natural estimation for the sphere of influence parameter σ_c of a neuron c is its mean arc length, that is, the mean distance between its set of neighbors.

4 Incremental learning procedure

Since each neuron of the SOP network is restricted to act only within a localized region of the input space, it is possible to incrementally train the SOP using new data without disturbing previous knowledge. However, the immediate neighborhood of a new datum in the input space must be considered very carefully in order to preserve its acquired knowledge in that region. This is why a small set of local vigilance data is defined for each self-organized neuron of the SOP network.

Let us consider the set $C = \{c_1, c_2, \ldots, c_q\}$ of q self-organized neurons of an already passively trained SOP network, and let set $\Xi = \{\xi_1, \xi_2, \ldots, \xi_n\}$ denote the data used for this initial passive training. Then, set $\Pi = \{\Pi_1, \Pi_2, \ldots, \Pi_q\}$ denotes a partition of Ξ such that:

$$\Pi_j = \{\xi_i \in \Xi \mid p_{c_j}(\xi_i) = \max_{c \in C} p_c(\xi_i)\},\ j = 1, 2, \ldots, q, \tag{5}$$

where the selection factors $p_c(\xi)$ are computed according to Equation (1).

For each neuron $c_j \in C$, a set of local vigilance data $\Pi'_j \subset \Pi_j$ is randomly constructed such that $|\Pi'_j| \leq k$. Usually, k is chosen such that $k << n/q$. Because the GNG network (or any other self-organizing map) preserves the topological structure of the input space, it is assumed that each neuron attracts toward him approximatively the same number of training data, and thus it is justified to use a fixed number of k data into each vigilance set Π'_j.

An incremental learning step for the SOP network is defined as follows:

1. Choose a new datum $\xi \in I\!R^n$ and find its best matching neuron $b \in C$.
2. Locate the set C' of neighboring neurons for ξ such that:

$$C' = \{c_i \in C \mid ||\xi - \omega_{c_i}|| \leq \delta \times \sigma_i\}, \qquad (6)$$

   where parameter δ balances the sphere of influence of each neuron[2].
3. Try to train the SOP using $\xi \cup \left(\bigcup_{c_j \in C'} \Pi'_j \right)$.
4. If training is successful, go back to step 1, else modify the GNG structure:
   (a) If $\exists\ c_j \in C'$ such that c_j is a newly inserted neuron[3] and $||\xi - \omega_{c_j}|| \leq \sigma_{c_j}$, then neuron c_j is adapted and $\omega_j^{\text{new}} = \omega_j^{\text{old}} + \epsilon(\xi - \omega_j^{\text{old}})$, where ϵ is an adaptation strength parameter (see Figure 3a).
   (b) Else, insert either one or two new neurons in the existing SOP structure[4]. If $||\xi - \omega_b|| \leq \sigma_b$, create neuron c_{q+1} with $\omega_{q+1} = \xi$ and link it to neuron b (see Figure 3b). Else, create neurons c_{q+1} and c_{q+2} with $\omega_{q+1} = \xi$ and $\omega_{q+2} = \omega_b + \epsilon(\xi - \omega_b)$ and link them together (see Figure 3c).
5. Retrain the SOP using $\xi \cup \left(\bigcup_{c_j \in C'} \Pi'_j \right)$.
6. Go back to step 1.

During an incremental learning step, the local vigilance data sets mainly serve to ensure that the training of new data does not alter inadvertency the existing knowledge. Indeed, for step 3 and 5, errors for the vigilance data are backpropagated only if necessary, that is, the vigilance data are trained only if the new datum interferes with them. In step 4(b), neurons are added in such a way that they do not disturb the existing structure.

[2]For example, if $\delta = 1.5$, only those neurons having a selection factor $p_c(\xi) \geq e^{-(1.5)^2} \approx 0.1$ are considered (see Equation (1)).

[3]New neurons are those inserted during an incremental learning step.

[4]Note that any new neuron added to the GNG network must be linked to a new hidden neuron in the Perceptron network.

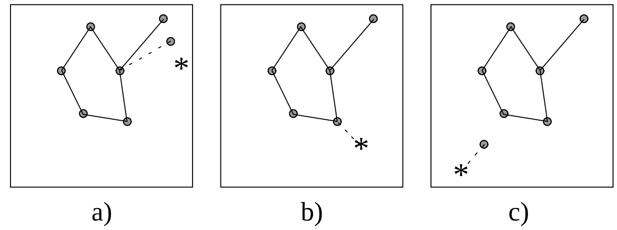

Figure 3: *Three possible cases for adapting the SOP structure. a) An existing neuron is moved toward the new datum. b) A new neuron is inserted and linked to the previous structure. c) Two new disjoint neurons are added.*

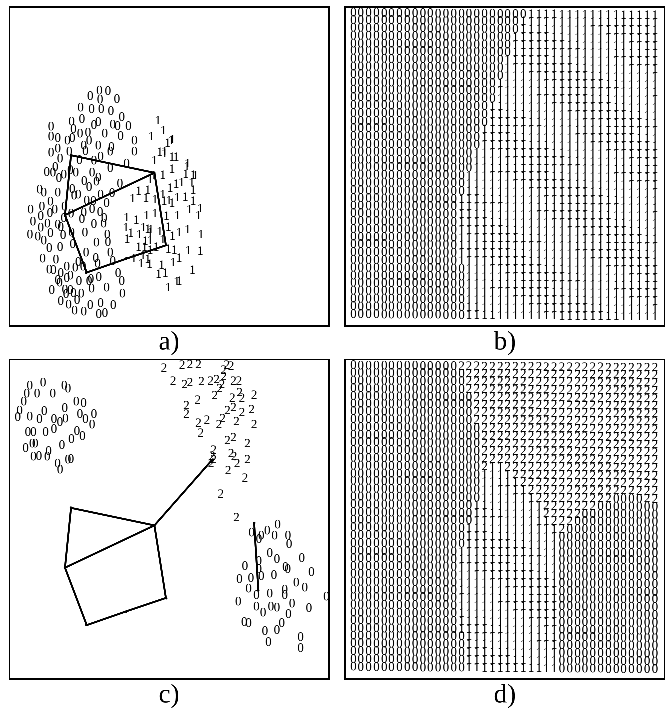

Figure 4: *Three classes problem. In a), the first group of data and the resulting GNG network after passive training are shown b) the resulting decision frontiers. In c), the second group of data and the modified GNG network after incremental learning are shown with d) the updated decision frontiers.*

5 Experiments and results

The general behavior of the proposed incremental strategy is now illustrated with simulation results on 2D data. In Figure 4a), a passive training of a SOP network has reached stability for a first group of data formed by two classes represented by "0" and "1". The position of each neuron of the SOP in the input space is indicated by a small dot linked by edges to its immediate neighbors. The resulting decision

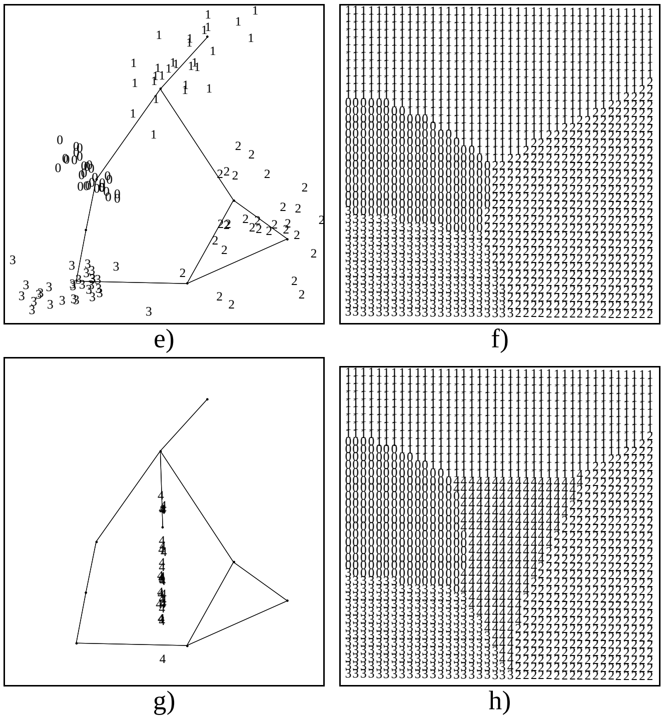

Figure 5: *Five classes problem. In a), the first group of data and the resulting GNG network after passive training are shown with in b) the resulting decision frontiers. In c), the second group of data and the modified GNG network after incremental learning are shown with in d) the updated decision frontiers.*

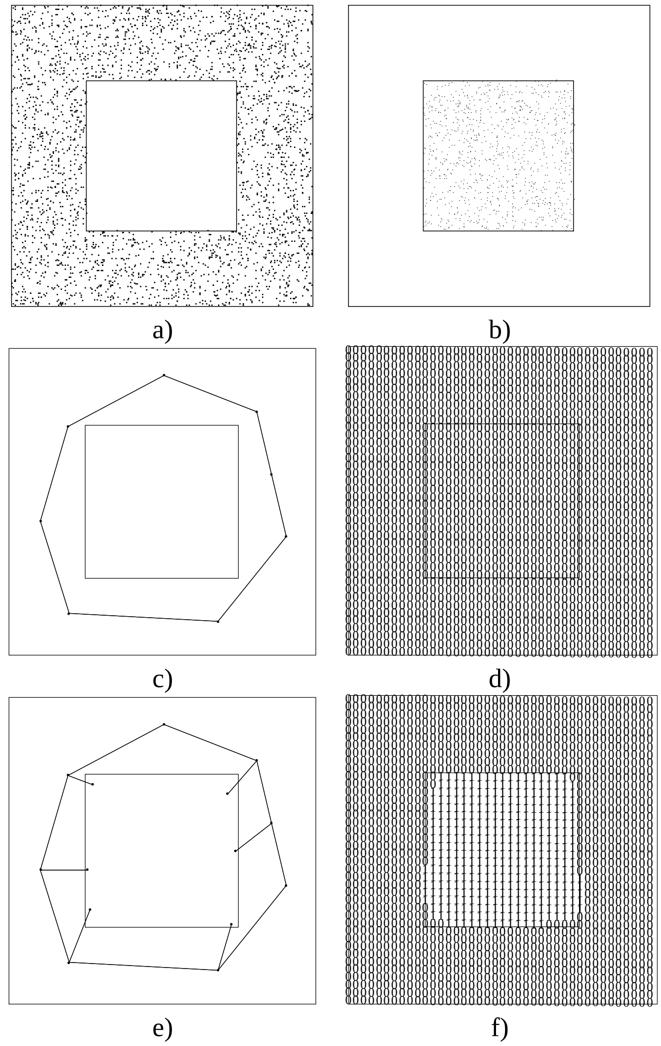

Figure 6: *a) Uniform data from class "0" for passive training and b) from class "1" for incremental training. In c), the resulting GNG network after passive training for class "0" is shown with d) the resulting decision frontiers, and e) the modified GNG network after incremental learning for class "1" is shown with f) the updated decision frontiers.*

frontiers is shown in Figure 4b). In Figure 4c), new data from class "0" are presented together with data from a new class "2" for incremental learning. As expected, the SOP network has correctly learned these new data by adapting its structure, and the new decision frontiers are updated as shown in Figure 4d). Figure 5 and Figure 6 give other results for various sets of 2D data that demonstrate the robustness of the incremental learning strategy.

For these experiments, the simulation parameters for the incremental learning strategy were $k = 0.1 \times n/q$, $\delta = 1.5$ and $\epsilon = 0.5$. Other parameter values[5] for the initial unsupervised learning of the GNG were $\lambda = 100$, $\epsilon_b = 0.2$ and $\epsilon_n = 0.006$ (see Section 3.1).

6 Conclusion

This paper has presented an hybrid architecture combining a self-organizing map with a multilayer Perceptron to form the Self-Organized Perceptron (SOP) network. The unsupervised map is used to model the input data space and to partition the supervised Perceptron into clusters of neurons. The structure and learning algorithm of the proposed architecture has been shown to allow incremental learning while preserving previous knowledge.

This approach is currently being integrated into an interactive cursive handwriting recognition system where basic knowledge stems from large isolated character data sets that enable initial multi-writer capabilities. In this system, interaction with the user, who becomes a teacher, enables both knowledge evolution and reinforcement.

[5] Additional parameters not described in that paper are defined for the unsupervised GNG learning algorithm. For those extra parameters, we used the same values used by B. Fritzke for his experiments on 2D data [14].

References

[1] S.E. Fahlman, C. Lebiere, "The Cascade-Correlation Learning Architecture", in D.S. Touretzky (ed.), *Advances in neural information processing systems*, 2 (pp. 524-532). San Mateo, CA: Morgan Kaufmann Publishers, 1990.

[2] B. Fritzke, "Growing Cell Structures - A Self-Organizing Network for Unsupervised and Supervised Learning", *Neural Networks*, Vol. 7, no. 9, pp. 1441-1460, 1994.

[3] N. Dunkin, J. Shawe-Taylor, P. Koiran, "A New Incremental Learning Technique", NeuroCOLT Technical Report Series, NC-TR-97-014, http://www.neurocolt.com, 1997.

[4] J. Ratsaby, "Incremental Learning with Sample Queries", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-20, No. 8, pp. 883-888, 1998.

[5] G.A. Carpenter, S. Grossberg, N. Markuzon, J.H. Reynolds, and D.B. Rosen, "Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps", *IEEE Transactions on Neural Networks*, Vol. NN-3, no. 5, pp. 698-713, 1992.

[6] M. Gori, F. Scarselli, "Are Multilayer Perceptrons Adequate for Pattern Recognition and Verification?", *IEEE Transactions on Pattern and Machine Intelligence*, Vol. PAMI-20, no.11 pp. 1121-1132, 1998.

[7] J. Moody, C. Darken, "Learning with Localized Receptive Fields", *Proc, of the 1988 Connectionist Models Summer School, pp. 133-143, Morgan Kaufmann Publishers, San Mateo, CA, 1989.*

[8] T. Poggio and F. Girosi, "Networks for Approximation and Learning", Proc. of the IEEE, *Vol. 78, no. 9, pp. 1481-1497, 1990.*

[9] S. Haykin, Neural Networks — A Comprehensive Foundation, *section 7.9, IEEE Press, 1995.*

[10] M.H. Hassoun, Fundamentals of Artificial Neural Networks, *section 6.1, MIT Press, 1995.*

[11] J.C. Platt, "A Resource-Allocation Network for function interpolation", Neural Computation, *Vol. NC-3, pp. 213–225, 1991.*

[12] E. Blanzieri, P. Katenkamp, A. Giordana, "Growing Radial Basis Function Networks", Proc. of the 4th Workshop on Learning Robots, Karlsruhe, 1995.

[13] T. Kohonen, "The Self-Organizing Map", Proceedings of the IEEE, *Vol. 78, no. 9, pp. 1464-1480, 1990.*

[14] B. Fritzke, "A Growing Neural Gas Network Learns Topologies", In D.S. Touretzky, and T.K. Leen, editors, Advances in Neural Information Processing Systems 7, *MIT Press, Cambridge MA, 1995.*