Distributed Activation, Search, and Learning by ART and ARTMAP Neural Networks

Gail A. Carpenter

Boston University, Center for Adaptive Systems & Department of Cognitive and Neural Systems
677 Beacon Street, Boston, Massachusetts 02215 USA
Phone - (617) 353-9483 Fax - (617) 353-7755 E-mail - gail@cns.bu.edu

ABSTRACT

Adaptive resonance theory (ART) models are being used for learning and prediction in a wide variety of applications. Winner-take-all coding allows these networks to maintain stable memories, but this type of code representation can cause problems such as category proliferation with fast learning and a noisy training set. A new class of ART models overcomes this limitation, permitting code representations to be arbitrarily distributed. With winner-take-all coding, the unsupervised distributed ART model (dART) reduces to fuzzy ART and the supervised distributed ARTMAP model (dARTMAP) reduces to fuzzy ARTMAP. dART automatically apportions learned changes according to the degree of activation of each coding node, for fast as well as slow learning with compressed or distributed codes. Distributed ART models replace the traditional neural network path weight with a dynamic weight equal to the rectified difference between coding node activation and an adaptive threshold. Dynamic weights that project to coding nodes obey a distributed instar leaning law and those that originate from coding nodes obey a distributed outstar learning law. Inputs activate distributed codes through phasic and tonic signal components with dual computational properties, and a parallel distributed match-reset-search process helps stabilize memory.

1. ART, ARTMAP, and Distributed Learning

ART [4,7] and ARTMAP [5,6] neural networks are being used for adaptive recognition and prediction in a variety of applications, including a Boeing parts design retrieval system, satellite remote sensing, medical database prediction, robot sensory-motor control and navigation, machine vision, 3D object recognition, electrocardiogram wave identification, automatic target recognition, air quality monitoring, signature verification, tool failure monitoring, chemical analysis from UV and IR spectra, electromagnetic system device design, and analysis of musical scores. The basic ART and ARTMAP networks feature winner-take-all (WTA) competitive coding, which groups inputs into discrete recognition categories. With fast learning but without WTA coding, certain input sequences may cause catastrophic forgetting of prior memories in these networks. Fast learning is useful for encoding important rare cases, but a combination of WTA coding and fast learning may lead to inefficient category proliferation with noisy training inputs. This problem is partially solved by ART-EMAP [9,10], which uses WTA coding for learning and distributed category representation for test-set prediction. Distributed test-set category representation can significantly improve ARTMAP performance, especially when the size of the training set is small. In medical database prediction problems, which often feature inconsistent training input predictions, the ARTMAP-IC [8] network improves ARTMAP performance with distributed prediction, category instance counting, and a new match tracking search algorithm. Compared to the original match tracking algorithm, the new rule facilitates prediction with sparse or inconsistent data, improves memory compression without loss of accuracy, and is actually a better approximation of the original ARTMAP network differential equations. A voting strategy further improves prediction by training the system several times on different orderings of an input set. Voting, instance counting, and distributed test-set code representations combine to form confidence estimates for competing predictions. However, these and most other ART and ARTMAP variants have used WTA coding during learning, so they do not solve the category proliferation problem of noisy training sets.

A new class of ART models retain stable coding, recognition, and prediction, but allow arbitrarily distributed category representation during learning as well as performance [2]. When the category representation is winner-take-all, the unsupervised distributed ART model (dART) reduces to fuzzy ART [7] and the supervised distributed ARTMAP model (dARTMAP) reduces to fuzzy ARTMAP [5]. Distributed ART and ARTMAP networks automatically apportion learned changes according to the degree of activation of each category node.

This research was supported in part by the National Science Foundation (NSF IRI 94-01659) and the Office of Naval Research (ONR N00014-95-1-0409 and ONR N00014-95-0657).

In: *Proceedings of the International Conference on Neural Networks (ICNN'96)*, Washington DC.

This permits fast as well as slow learning without catastrophic forgetting. In distributed ART models, dynamic weights replace the multiplicative long-term memory weights found in most neural networks. The input signal that activates the distributed code is a function of a phasic component, which depends on the active input, and a tonic component, which depends on prior learning but is independent of the current input. The computational properties of the phasic and tonic components are derived from a formal analysis of distributed pattern learning. However, these components can be interpreted as postsynaptic membrane processes, with phasic terms mediated by ligand-gated receptors and tonic terms mediated by voltage-gated receptors [16]. At each synapse, phasic and tonic terms balance one another and exhibit dual computational properties. During learning with a constant input, phasic terms are constant while tonic terms may grow. Tonic components would then become larger for all inputs, but phasic components become more selective, reducing the total coding signal that would be sent by a significantly different input pattern. A geometric interpretation of distributed ART represents the tonic component as a coding box in input space and the phasic component as the coding box expanded to include the current input.

Although dART with WTA coding is computationally equivalent to fuzzy ART, the dART architecture differs from the standard ART architecture. An ART input from a field F_0 passes through a matching field F_1 before activating a coding field F_2. Activity at F_2 feeds back to F_1, forming a resonant loop. ART networks thus encode matched F_1 patterns rather than the F_0 inputs themselves, a key feature for code stability. With winner-take-all coding, the matched F_1 pattern confirms the original category choice when it feeds back up to F_2. With $F_1 \leftrightarrow F_2$ feedback this essential property may not persist when the F_2 code is distributed. In the distributed ART network, the coding field F_2 receives input directly from F_0, retaining the bottom-up / top-down matching process at F_1 only to determine whether an active code meets the vigilance matching criterion (Figure 1). Nevertheless, dART dynamic weights maintain code stability. When the matching process is disabled by setting the vigilance parameter to 0, dART becomes a type of feedforward ART network.

2. Distributed Activation

A dART network includes a field of nodes F_0 that represents a current input vector; a field F_2 that represents the active code; and a field F_1 that represents a matched pattern determined by bottom-up input from F_0 and top-down input from F_2. Vector $\mathbf{I} \equiv (I_1 \ldots I_i \ldots I_M)$ denotes F_0 activity, $\mathbf{x} \equiv (x_1 \ldots x_i \ldots x_M)$ denotes F_1 activity, and $\mathbf{y} \equiv (y_1 \ldots y_j \ldots y_N)$ denotes F_2 activity. Each component of **I**, **x**, and **y** is contained in the interval

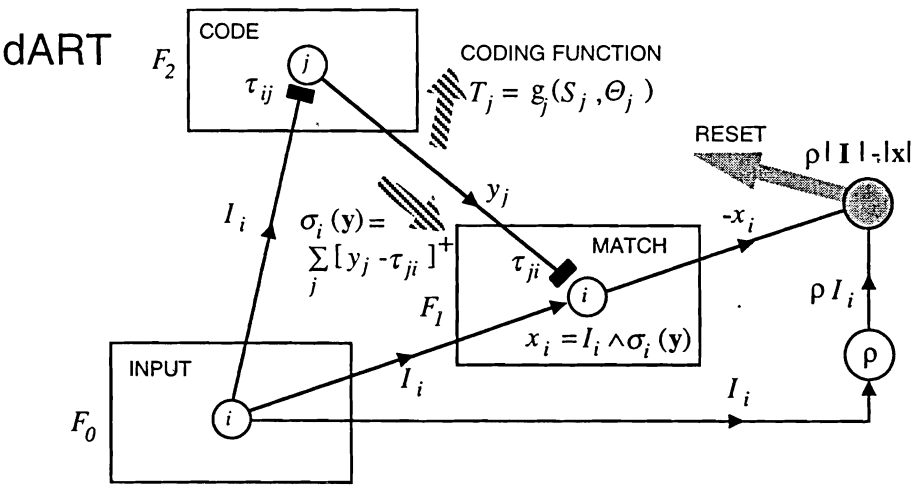

Figure 1: Like fuzzy ART, distributed ART computes a matched pattern **x** at F_1 and resets F_2 if **x** fails to meet the vigilance matching criterion. In dART, however, F_2 receives input directly from F_0. The F_2 code **y**, which is a function of phasic components S_j and tonic components Θ_j, may be arbitrarily distributed. The i^{th} F_1 node receives a positive signal from each F_2 node at which activity y_j exceeds an $F_2 \rightarrow F_1$ adaptive threshold τ_{ji}. With choice at F_2 and fast learning, distributed ART is computationally equivalent to fuzzy ART.

[0,1]. The number of input components (M) and the number of coding nodes (N) may be arbitrarily large. The input **I** and the matched pattern **x** may be continuously varying functions of time t, but the code **y** acts as a content-addressable memory that is held constant between resets by strong competition at F_2.

2.1. Dynamic Weights

In fuzzy ART the path from the i^{th} F_1 node to the j^{th} F_2 node contains an adaptive weight w_{ij}, and the path from the j^{th} F_2 node to the i^{th} F_1 node contains a weight w_{ji}. With fast learning, $w_{ij} \equiv w_{ji}$. In contrast, in the distributed outstar network [1] the unit of long-term memory (LTM) is an adaptive threshold τ_{ji}. Formally,

$$\tau_{ji} \equiv 1 - w_{ji}. \tag{1}$$

The distributed outstar signal from the j^{th} F_2 node to the i^{th} F_1 node is $\left[y_j - \tau_{ji}\right]^+$, where $[\ldots]^+$ denotes the rectification operator: $[\xi]^+ \equiv \max\{\xi, 0\}$. This path signal helps avoid catastrophic forgetting because $\left[y_j - \tau_{ji}\right]^+ = \left[w_{ji} - \left(1 - y_j\right)\right]^+ = 0$ when w_{ji} is small, unless $y_j = 1$. Other types of signals such as the product $y_j w_{ji}$ remain positive, and the weights subject to erosion, whenever y_j is positive, no matter how small w_{ji} has become. When the j^{th} F_2 node is chosen $\left(y_j = 1\right)$, the dynamic weight equals the traditional weight since then $w_{ji} = \left(1 - \tau_{ji}\right) = \left[y_j - \tau_{ji}\right]^+$.

Distributed ART takes this idea one step further, substituting a dynamic weight for each fuzzy ART weight. The formal substitutions:

$$w_{ji} \rightarrow \left[y_j - \tau_{ji}\right]^+ \tag{2}$$

and

$$w_{ij} \rightarrow \left[y_j - \tau_{ij}\right]^+ \tag{3}$$

convert fuzzy ART to distributed ART. Thresholds τ_{ji} in paths from the j^{th} F_2 node to the i^{th} F_1 node adapt according to a distributed outstar learning law, while thresholds τ_{ij} in paths from the i^{th} F_0 node to the j^{th} F_2 node obey a distributed instar learning law (Section 4). Adaptive thresholds remain in the range [0,1], starting at or near 0 and increasing monotonically during learning.

2.2. Signal Functions

For each input **I** and $j = 1 \ldots N$, the total signal T_j from the dART input field F_0 to the j^{th} F_2 node is a function of the form:

$$T_j = T_j\left(y_j\right) = g_j\left(S_j\left(y_j\right), \Theta_j\left(y_j\right)\right) \tag{4}$$

where $g_j(0,0) = 0$ and $\dfrac{\partial g_j}{\partial S_j} > \dfrac{\partial g_j}{\partial \Theta_j} > 0$ for $S_j > 0$ and $\Theta_j > 0$.

In (4) the *phasic* component S_j, which depends on the input **I**, is a sum:

$$S_j = S_j\left(y_j\right) = \sum_{i=1}^{M} S_{ij}\left(y_j\right). \tag{5}$$

A term in the sum (5) may be visualized as a certain fraction of the membrane sites at the i^{th} synapse of the j^{th} F_2 node. Sites primed, or gated, by the dynamic weight $\left[y_j - \tau_{ij}\right]^+$ can be activated by an input I_i, but a number of these sites $\left(\Delta_{ij}\right)$ may be refractory, or depleted, due to their recent activation. Formally,

$$S_{ij}\left(y_j\right) = \left[I_i \wedge \left[y_j - \tau_{ij}\right]^+ - \Delta_{ij}\right]^+, \tag{6}$$

where $\wedge$ represents the fuzzy intersection, or component-wise minimum: $(\mathbf{a} \wedge \mathbf{b})_i \equiv (a_i \wedge b_i) \equiv \min(a_i, b_i)$; the dual operator $\vee$ represents the fuzzy union, or component-wise maximum: $(\mathbf{a} \vee \mathbf{b})_i \equiv (a_i \vee b_i) \equiv \max(a_i, b_i)$ [17]. For $y_j \in [0,1]$,

$$0 \le S_j(y_j) \le \sum_{i=1}^{M} [y_j - \tau_{ij}]^+ \le \sum_{i=1}^{M} y_j = M y_j . \tag{7}$$

In (4), the *tonic* component Θ_j is a sum:

$$\Theta_j = \Theta_j(y_j) = \sum_{i=1}^{M} \Theta_{ij}(y_j) \tag{8}$$

where:

$$\Theta_{ij}(y_j) = [y_j \wedge \tau_{ij} - \delta_{ij}]^+ . \tag{9}$$

The sum $\Theta_j(y_j)$, which is independent of the input $\mathbf{I}$, plays the role of a nodal bias term that increases during learning. A fraction τ_{ij} of membrane sites are primed by the node's activity (y_j), but recently active sites may be refractory (δ_{ij}). Like $S_j(y_j)$, $\Theta_j(y_j)$ lies in the interval $[0, My_j]$.

A distributed version of the fuzzy ART choice-by-difference (CBD) function [3] defines one signal rule (4) by $T_j = S_j + (1-\alpha)\Theta_j$, with $0 < \alpha < 1$. Like S_j and Θ_j, the CBD signal function $T_j \in [0, My_j]$. A distributed version of the Weber law signal function [4] defines a different signal rule by $T_j = S_j / (\alpha + My_j - \Theta_j)$, with $\alpha > 0$. For the Weber law coding function, $T_j \in [0,1)$.

2.3. Code Representation

In distributed ART networks, activity $\mathbf{y} = (y_1 \ldots y_j \ldots y_N)$ at a competitive coding field F_2 is stored as a content-addressable memory. An algorithm that approximates the dynamics of strong competition postulates that external inputs initially determine $\mathbf{y}$, but then internal feedback holds $\mathbf{y}$ constant until F_2 is actively reset. Except during reset, $\mathbf{y}$ is normalized:

$$|\mathbf{y}| \equiv \sum_{j=1}^{N} y_j = 1, \tag{10}$$

where $|\ldots|$ represents the city-block norm.

In ART models, F_2 reset occurs when the bottom-up / top-down matched pattern $\mathbf{x}$ at F_1 fails to meet a matching criterion defined by a vigilance parameter ρ (Section 3). Reset is effected by a large nonspecific arousal signal. In the dART model, reset momentarily sends all y_j to 1 at a time $t = r$. This allows the values $T_1(1)|_{t=r} \ldots T_N(1)|_{t=r}$ to determine which $\mathbf{y}$ will be established next. Until the next reset,

$$y_j = f_j(T_1(1) \ldots T_N(1))|_{t=r} \tag{11}$$

where $\partial f_j / \partial T_j \ge 0$.

3. Distributed Search

The distributed ART match-reset-search process is similar to that of other ART networks. When an F_2 code $\mathbf{y}$ becomes active, the activity pattern $\mathbf{x}$ at F_1 represents a match between the current bottom-up input $\mathbf{I}$ and a top-down input $\sigma(\mathbf{y})$, where:

$$\sigma_i = \sigma_i(\mathbf{y}) = \sum_{j=1}^{N} [y_j - \tau_{ji}]^+ \tag{12}$$

for $i = 1 \ldots M$. Since $\sum_j y_j = 1$, $\sigma_i \in [0,1]$. Activity $\mathbf{x}$ at F_1 then equals the fuzzy intersection of $\mathbf{I}$ and $\sigma(\mathbf{y})$:

$$x_i = I_i \wedge \sigma_i(\mathbf{y}). \tag{13}$$

Signals from F_2 thereby prime F_1 in the sense that $\sigma_i(\mathbf{y})$ imposes an upper bound on inputs I_i that can be fully represented at the i^{th} F_1 node. If an input fails to meet the vigilance matching criterion, that is, if:

$$|\mathbf{x}| = |\mathbf{I} \wedge \sigma(\mathbf{y})| < \rho|\mathbf{I}|, \tag{14}$$

then a nonspecific reset signal shuts off the code $\mathbf{y}$. Reset also leaves an enduring trace of $\mathbf{y}$ via the refractory terms Δ_{ij} (6) and δ_{ij} (9), so that the network does not simply reactivate the same code.

4. Distributed Learning

Catastrophic forgetting is a problem faced by all neural networks with distributed activation, especially in the fast-learn limit where system variables go to asymptote with each input presentation. The outstar [11,12] and instar [13-15] learning laws used in previous ART networks would cause catastrophic forgetting if transferred directly to a network with a distributed code $\mathbf{y}$. Stable distributed coding with fast learning would require internal or external control of the learned changes that one input can induce.

The distributed outstar [1] solves the catastrophic forgetting problem for learning in paths that originate from the coding field F_2. During distributed outstar learning, thresholds τ_{ji} in paths from F_2 to F_1 adapt according to the equation:

$$\frac{d}{dt}\tau_{ji} = \left[y_j - \tau_{ji}\right]^+ \left(\sigma_i(\mathbf{y}) - x_i\right) = \left[y_j - \tau_{ji}\right]^+ \left[\sigma_i(\mathbf{y}) - I_i\right]^+. \tag{15}$$

During distributed instar [2] learning, thresholds τ_{ij} in paths from F_0 to F_2 adapt according to the equation:

$$\frac{d}{dt}\tau_{ij} = \left[y_j - \tau_{ij} - I_i\right]^+ = \left[\left[y_j - \tau_{ji}\right]^+ - I_i\right]^+. \tag{16}$$

With distributed outstar learning, the total signal $\sigma_i(\mathbf{y})$ from the coding field to a target F_1 node can only decrease, by a principle of atrophy due to disuse. During distributed instar learning, the total signal $T_j(y_j)$ from the input field to a target F_2 node can only increase, as the tonic component of the signal increases while the phasic component remains constant for a constant input. In both cases distributed instar and outstar learning laws bound the total change any one input can impose upon the system.

5. Distributed ARTMAP

ARTMAP networks for supervised learning self-organize mappings from input vectors, representing features

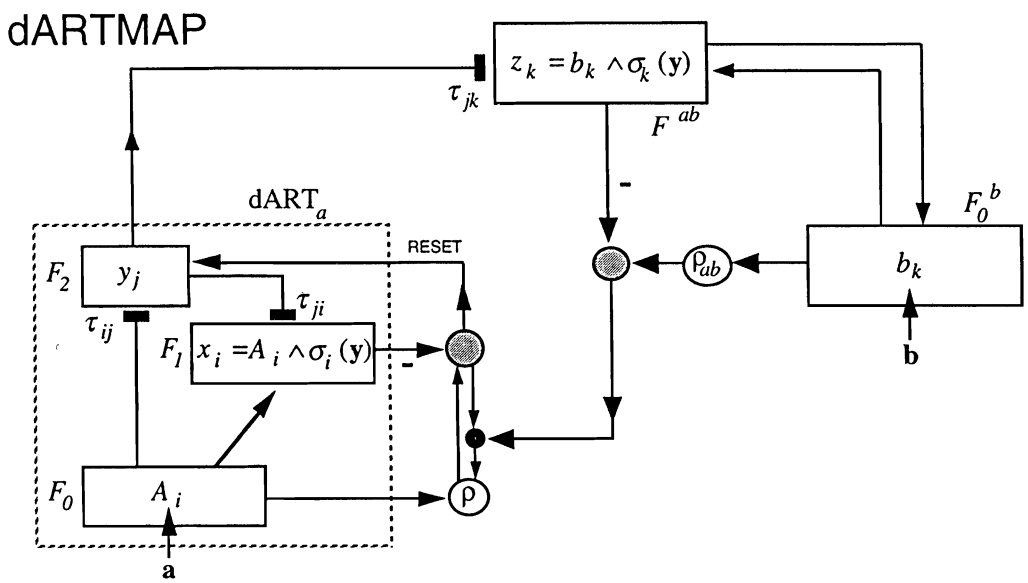

Figure 2: A simplified dARTMAP network computes classification probabilities, with $|\mathbf{b}| = 1$ at an output field F_0^b.

such as patient history and test results, to output vectors, representing predictions such as the likelihood of an adverse outcome following an operation. The original binary ARTMAP network [6] incorporates two ART 1 modules, ART_a and ART_b, that are linked by a *map field* F^{ab}. Inputs $\mathbf{a}$ are *complement coded*, so that the ART_a input is $\mathbf{I} = \mathbf{A} \equiv \left(\mathbf{a}, \mathbf{a}^c\right)$. At the map field the network forms associations between categories via outstar learning and triggers search, via the ARTMAP match tracking rule, when a training set input fails to make a correct prediction. Match tracking increases the ART_a vigilance parameter ρ_a in response to a predictive error at ART_b. Fuzzy ARTMAP [5] substitutes fuzzy ART for ART 1. Distributed ARTMAP (dARTMAP) substitutes dART for fuzzy ART and distributed outstar learning for outstar learning at the map field.

Many applications of supervised learning systems such as ARTMAP are classification problems, where the trained system tries to predict a correct category given a test set input vector. A prediction might be a single category or distributed as a set of scores or probabilities. A dARTMAP algorithm [2] outlines a procedure for applying distributed ART learning and prediction to this problem, which does not require the full dART_b architecture. Even for this special case, however, the large number of design choices for dARTMAP, compared to the basic fuzzy ARTMAP network, imply that research would be required to understand where and how distributed coding improves performance, generalization, and code compression. In a simplified dARTMAP network an input $\mathbf{a} = (a_1 \ldots a_i \ldots a_M)$ learns to predict an outcome $\mathbf{b} = (b_1 \ldots b_k \ldots b_L)$ (Figure 2). A classification problem would set one component $b_K = 1$ during training, placing an input $\mathbf{a}$ in class K. With choice at F_2, the dARTMAP algorithm reduces to a fuzzy ARTMAP algorithm.

References

[1] Carpenter, G.A. (1994). A distributed outstar network for spatial pattern learning. *Neural Networks*, **7**, 159-168.

[2] Carpenter, G.A. (1996). Distributed learning, recognition, and prediction by ART and ARTMAP neural networks. Submitted to *Neural Networks*. Technical Report CAS/CNS TR-96-004, Boston, MA: Boston University.

[3] Carpenter, G.A., & Gjaja, M.N. (1994). Fuzzy ART choice functions. In *Proceedings of the World Congress on Neural Networks (WCNN'94)* (pp. **I**-713-722). Hillsdale, NJ: Lawrence Erlbaum Associates.

[4] Carpenter, G.A., & Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing*, **37**, 54-115.

[5] Carpenter, G.A., Grossberg, S., Markuzon, N., Reynolds, J.H., & Rosen, D.B. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, **3**, 698-713.

[6] Carpenter, G.A., Grossberg, S., & Reynolds, J.H. (1991). ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks*, **4**, 565-588.

[7] Carpenter, G.A., Grossberg, S., & Rosen, D.B. (1991). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, **4**, 759-771.

[8] Carpenter, G.A., & Markuzon, N. (1996). ARTMAP-IC and medical diagnosis: Instance counting and inconsistent cases. CAS/CNS Technical Report, Boston, MA: Boston University.

[9] Carpenter, G.A., & Ross, W.D. (1993). ART-EMAP: A neural network architecture for learning and prediction by evidence accumulation. In *Proceedings of the World Congress on Neural Networks (WCNN'94)* (pp. **III** - 649-656). Hillsdale, NJ: Lawrence Erlbaum Associates.

[10] Carpenter, G.A., & Ross, W.D. (1995). ART-EMAP: A neural network architecture for object recognition by evidence accumulation. *IEEE Transactions on Neural Networks*, **6**, 805-818.

[11] Grossberg, S. (1968). A prediction theory for some nonlinear functional-differential equations, I: Learning of lists. *Journal of Mathematical Analysis and Applications*, **21**, 643-694.

[12] Grossberg, S. (1970). Some networks that can learn, remember, and reproduce any number of complicated space-time patterns, II. *Studies in Applied Mathematics*, **49**, 135-166.

[13] Grossberg, S. (1972). Neural expectation: Cerebellar and retinal analogs of cells fired by learnable or unlearned pattern classes. *Kybernetik*, **10**, 49-57.

[14] Grossberg, S. (1976). Adaptive pattern classification and universal recoding, I: Parallel development and coding of neural feature detectors. *Biological Cybernetics*, **23**, 121-134.

[15] Malsburg, C. von der (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, **14**, 85-100.

[16] Nicholls, D. G. (1994). *Proteins, Transmitters and Synapses*. Oxford: Blackwell Science Ltd.

[17] Zadeh, L. (1965). Fuzzy sets. *Information and Control*, **8**, 338-353.