Biol Res 40: 479-485, 2007

Dynamical Complexity in Cognitive Neural Networks

ERIC GOLES[1*] and ADRIÁN G. PALACIOS[1**]

[1] Instituto de Sistemas Complejos de Valparaíso, Subida Artillería 470, Cerro Artillería, Valparaíso.
[*] Facultad de Ciencias, Universidad Adolfo Ibáñez, Avda. Diagonal Las Torres 2640. Peñalolén, Santiago.
[**] Centro de Neurociencia de Valparaíso, Facultad de Ciencias,
Universidad de Valparaíso. Gran Bretaña 1111, Playa Ancha,
Valparaíso.

ABSTRACT

In the last twenty years an important effort in brain sciences, especially in cognitive science, has been the development of mathematical tool that can deal with the complexity of extensive recordings corresponding to the neuronal activity obtained from hundreds of neurons. We discuss here along with some historical issues, advantages and limitations of Artificial Neural Networks (ANN) that can help to understand how simple brain circuits work and whether ANN can be helpful to understand brain neural complexity.

Key terms: Artificial**,** Neural Net, Brain, Dynamical Complexity, Computational Neurosciences**,** Cellular Automata.

INTRODUCTION

A major challenge in biology today is to understand how "cognitive abilities" emerge from a highly interconnected and complex neural net such as the brain. A heuristic way to approach the problem has been to model the brain on the basis of simple neuronal circuits (interconnected neurons or agents) with a minimum of learning and memory rules. However, to establish a realistic model an enormous challenge still remains, as such model must be embedded with cognitive properties. In this paper we will discuss some approaches that have been used to relate finite discrete networks, in the field of Artificial Neural Networks (ANN), to brain activity.

Artificial Neural Networks (ANN)

An ANN is defined by a -tuple $N = (W, b, \{0,1\}^n)$, where W is a real nxn weight matrix and b an n-dimensional threshold vector. Given a vector $x \in \{0,1\}^n$, the new vector value, y, is given by $y = \mathbf{1}(W x - b)$, or equivalently:

$$y_i = \mathbf{1}\left(\sum_{j=1}^{n} w_{ij}x_j - b_i\right)$$

where $\mathbf{1}(u) = 1$ if and only if $u \geq 0$, and is 0 otherwise.

Such ANN networks were introduced by McCulloch and Pitts (1943) as a model of the nervous system in their seminal article *"A logical calculus of ideas immanent in nervous activity"*. They proved that an ANN was able to simulate a digital computer and to exhibit the most complex behavior associated with a mathematical object in the 40's. Further development of ANN models raises some important considerations. First consider the "net function" which describes how the network input (e.g. a sensory signal) will be combined by each neuron. In mathematical terms net functions can adopt the form of, e.g., linear or higher order equations, where b is a threshold value:

* Corresponding Author: Eric Goles, Universidad Adolfo Ibáñez, Avda. Diagonal Las Torres 2640. Peñalolén, Santiago. E-mail eric.chacc@uai.cl

Received: October 10, 2007. Accepted: March 3, 2008

$$u = \sum_{j=1}^{N} w_j y_j + b,$$

$$u = \sum_{j=1}^{N}\sum_{k=1}^{N} w_{jk} y_j y_k + b,$$

The ANN output follows a "neuron activation function" of the form (e.g.) sigmoid, linear:

$$f(u) = \frac{1}{1+e^{-u/T}},$$

$$f(u) = au + b,$$

The application of an ANN model leads to solutions of problems that tend to be associated with a particular topology, which can be acyclic (with absence of feedback) or cyclic (with recurrence) forms, depending of how neurons or nodes are interconnected.

We don't plan to review extensively the theory and applications of ANN in detail (for a review see Anderson and Rosenfeld, 1988; Arbib et al. 2002), but to present some issues related to their complexity and their possible relation to brain cognitive processes (e.g, learning & memory, decision making).

The McCulloch and Pitts Model

At the cellular level a single artificial neuron should enfold cell excitability, that is the biophysical properties describing the conductivity of the neuron membrane as a function of ion concentrations. The conductivity changes determine neuron inhibition or excitation, what will or will not trigger an action potential (or spike), depending on whether a threshold (*b*) is reached

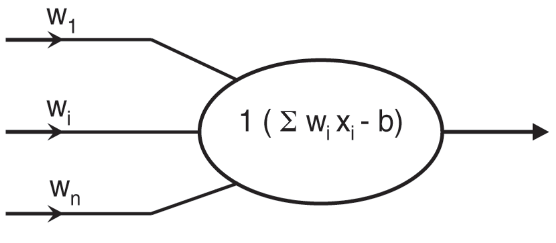

Figure 1: A single artificial neuron.

On the basis of singles rules McCulloch and Pitts proved that by assembling an arbitrary number of such elementary neuron units it was possible to build a computer. Since a computer is nothing else than a set of circuits, they were able to build a universal set of logical gates, say the AND, OR, and NOT gates as follows:

AND $(x_1, x_2) = \mathbf{1}(x_1 + x_1 - 1.5)$,

OR $(x_1, x_2) = \mathbf{1}(x_1 + x_2 - 0.5)$,

NOR $(x) = \mathbf{1}(-x + 0.5)$,

For instance, consider the XOR function, i.e: XOR(1,0) = XOR(0,1) = 1 and XOR(1,1) = XOR(0,0) = 0. This function can be built from the elementary functions as shows in Figure 2.

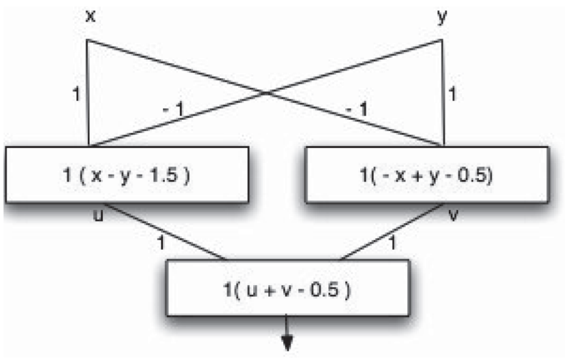

Figure 2: The XOR function.

To recover biological regulatory properties such, e.g. dynamical feedback, recurrence, robustness, learning and memory retrieval in neural cognitive network, a further step was important. The introduction of the Perceptron by Rosenblatt in the latest 50's.

The Perceptron

It is a very simple ANN able to recognize some families of patterns. It consists of a square retina (a bidimensional array of cells) connected by weighted wires ($w_1..w_n$) to a unique artificial neuron, which operates as the output:

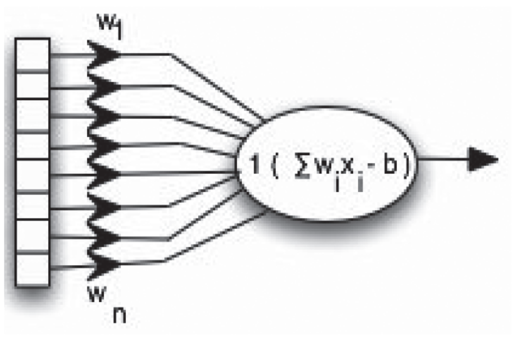

Figure 3: The Perceptron.

The idea was the following: suppose we want to recognize patterns which belong to a set F^+, and consider also its complement, F^-. So, mathematically we want the Perceptron to answer "1" if and only if the pattern belongs to F^+ (and "0", otherwise). To achieve this initially, the Perceptron is assigned random weights, say in an interval (-a, a). The weights are changed with a procedure that tries to give the correct answer. Roughly speaking, it works as follows: suppose $x \in F^+$ (i.e. it is a good pattern and the Perceptron has to answer the "1"). If P(x) = 1 (the Perceptron output) nothing is done and we pick another pattern in $F^+ \cup F^-$. If P(x) = 0, we proceed to change the weighs in order to improve the recognition (for instance, if $x_i = 1$ weight $a_i = a_i + e_i$, where $e_i = 0$ if $a_i > 0$ and $e_i > 0$ if $a_i < 0$. Similarly, if $x \in$ to F^-: nothing will be changed if P(x) = 0. When P(x) = 1 then we change the weights in a similar way: if $x_i = 1$, $a_i = a_i - e_i$, where $e_i = 0$ if $a_i < 0$ and $e_i > 0$ if $a_i > 0$. Rosenblatt proved that if the class of patterns is *linearly separable* then the Perceptron finds a solution, i.e. a set of weights such that P(x) = 1 if and only if x belongs to F^+.

Unfortunately, later on Minsky and Papert (1988) in their famous book "Perceptrons" proved that the linear separable cases are not the most interesting. There exist a lot of patterns that our brain classifies quickly and yet are not linearly separable. The Minsky and Papert results literally slowed down, for more that ten years the use of ANN *bottom up* approach models to understand brain capabilities. In fact, during this time the same Minsky developed a very powerful school in artificial intelligence which takes a top down approach, *the symbolic approach*, by essentially saying that in order to simulate our mind behavior it is not necessary to imitate neurons.

Some non linear separable classes

Geometrically speaking a *linear separable* problem looks like that in Figure 4a. A well-known case has been presented above. The XOR function cannot be separated by any straight line (see Figure 4b): it is impossible to put in the same side of the plane the states (1,0) and (0,1) without considering at least one of the other two. That it is also true for the general XOR function (also known as the parity function which was studied in detail in Minsky, see 1988 for references).

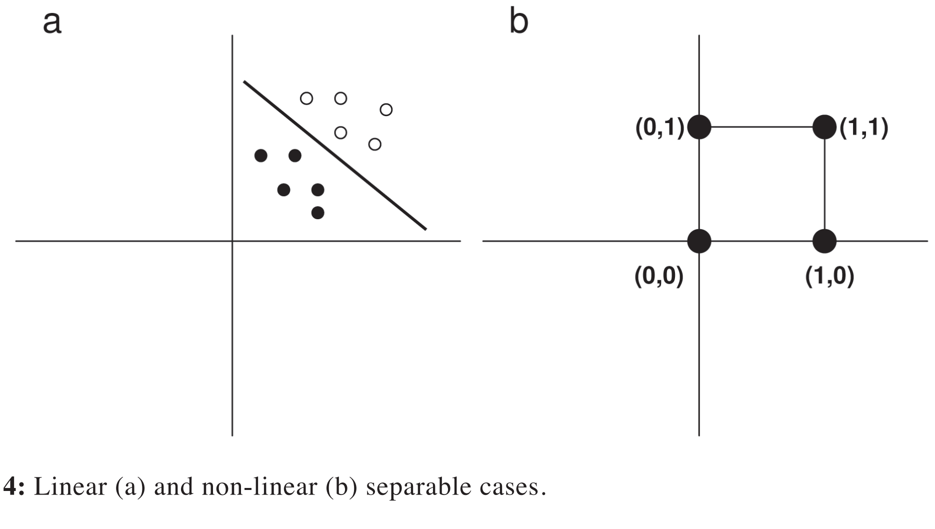

Figure 4: Linear (a) and non-linear (b) separable cases.

Another example more related with our perception is the failure of a *local Perceptron* to analyze the global network register to recognize connected patterns. As we see in Figure 5, the patterns (i) and (ii) are easy to recognize respectively as a connected and non-connected pattern, but the same task is much more difficult when considering (ii) and (iv). In fact it is not possible to recognize such connectivity with a local Perceptron (see Minsky and Papert, 1988).

Generalized Perceptrons

Only in the 80's do more appear powerful models of ANN, which learn to recognize non linear separable patterns. Essentially it considers a multilayer Perceptron[1], and more importantly the introduction of algorithms in order to change the weights of the network: *the back-propagation algorithm* (Rumelhart et al. 1986), which in fact tries to minimize the quadratic error of the network recognition by approximated gradient methods which changes the weights from the last layer (the output) to the first one (the input). This kind of procedure has been very well studied and applied to many problems related to engineering, pattern recognition, brain biology, etc.

Symmetric Neural Networks (SNN)

SNNs appear in the 80's related to some automata dynamics and the Ising model in statistical physics. We say that $N = (W, b, \{0,1\}^n)$ is a SNN if W is a nxn symmetric matrix. In a SNN to observe temporal state changes, we need to introduce discrete time steps and to manually synchronize the update of nodes, i.e., every site changes at the same time; and the input sites are changed one by one in a prescribed order. In Goles et al. (1985) it was proved that the

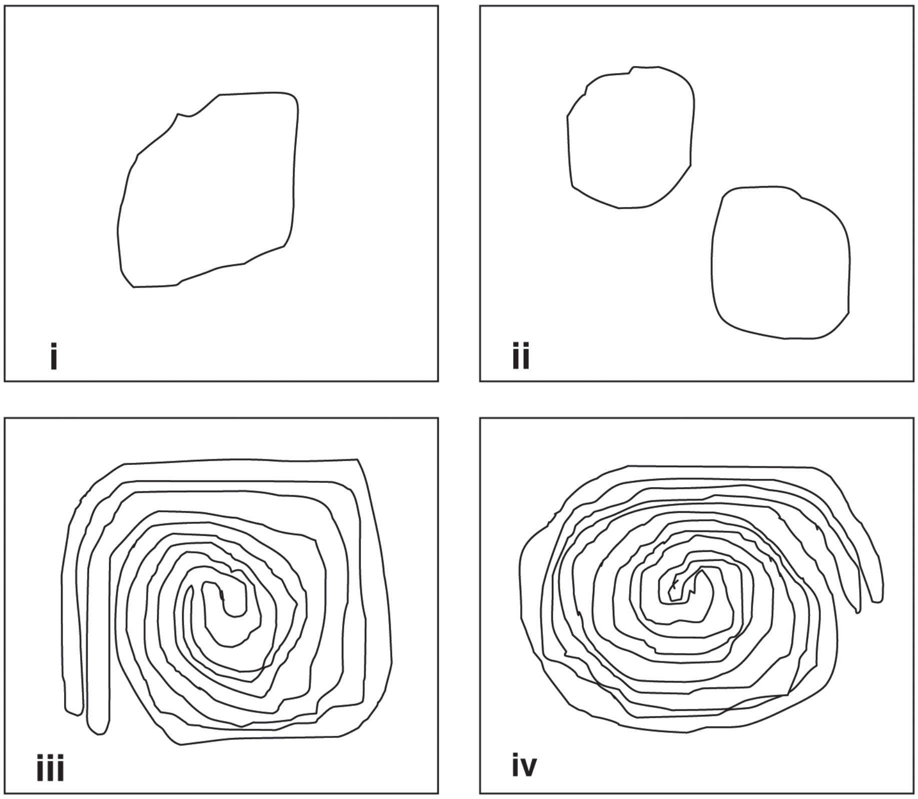

Figure 5: (i) connected pattern, (ii) non-connected pattern, (iii-iv) complex patterns.

[1] The perceptron does not have an internal or hidden layer: the output is directly link to the bidimensional retina.

synchronous iteration on SNN converges in a finite number of steps to fixed points (configurations such that F(x) = x) or two periodic configurations (Configurations x, y such that F(x) = y and F(y) = x) , and when the diagonal of the matrix has non-negative entries diag(W) >= 0 the sequential iteration converges to fixed points. Hopfield (1982) elaborates a model using this class of networks as associative memories: i.e. the fixed points are the retrieval patterns. When a pattern "is near" in terms of Hamming distance (number of different entries in digital patterns) to an attractor, it's taken that the iteration of the network converges to it. Consider S_1, ... , S_p vectors in $\{-1,1\}^n$ to be memorized (i.e to be fixed points of the network).

To illustrate the associative memory model we will consider the weight matrix defined by the Hebb-rule:

$$w_{ii} = 0 \text{ and } w_{ij} = \sum_{e-1}^{p} s_i^e s_j^e$$

Hopfield established the conditions under which such vectors are attractors and also proved experimentally the retrieval qualities of that network: vectors at small Hamming distance of an attractor converge to it by using a sequential dynamic procedure. In fact, researchers in statistical physics show that the retrieval capacity of this model is high when the number of patterns to be memorized is small, i.e about a 0.14 n, where n is the number of artificial neurons (Kamp and Hasler, 1990). An interesting topic relate to Hopfield analysis is that each network is driven by an energy operator similar to the Ising model. In this context, each step of the network reduces its energy, so fixed points (i.e. memorized vectors) are local minima (see also Goles et al, 1985). A similar result was obtained for the synchronous update concerning also the two periodic configurations (Goles and Olivos, 1980).

Apparently one may believe that SNN are not as powerful compared to a non-symmetric one but their computing performances are equivalent. In fact, to convey information in a SNN it is enough to establish a line of neurons with a decreasing ratio between thresholds and weights. Furthermore, after building symmetric gates it is possible to simulate any circuit (Figure 6).

Dynamical Neural Network Complexities

In order to illustrate the complexity of ANN dynamics lets consider the following: given a symmetric nxn neural net with states -1 and 1. Is it possible to know in a reasonable amount of time if the problem (H) admits fixed points? Apparently, it seems to be a very simple problem if there exists at least a fixed point, but is actually very hard. In fact the majority of problems related to the dynamical neural net complexities are hard to solve. To illustrate this, consider a very simple Hopfield network and the following problem:

(H) Let $N = (A,\vec{0},\{-1,1\}^n)$ be a neural net such that A is an n×n symmetric integral matrix and $\vec{0}$ is the threshold vector. Does *N* admit fixed points?

Following (Floreen, 1992) we will prove that (H) is NP - Complete[2]. For this consider the following problem:

(P) Given a positive set of s integers: a_1, a_2, …, a_s, does there exist a vector $y \in \{-1,1\}^n$ such that $\sum_{i=1}^{s} a_i y_i = 0$

To prove that (H) is NP-Complete we will reduce (P) to it. Let us consider

$c = \sum_{i=1}^{s} a_i$ and the matrix:

$$A = \begin{bmatrix} -c & c & a_1 & . & . & a_s \\ c & -c & -a_1 & . & . & a_s \\ a_1 & -a_1 & 1 & . & . & 0 \\ a_2 & -a_2 & . & . & . & . \\ a_i & -a_i & . & . & . & . \\ a_s & -a_s & 0 & . & . & 1 \end{bmatrix}$$

2 This means that it is a problem which is in class NP (i.e., any candidate solution can be checked in polynomial time), and such that any other problem in NP can be polynomial reduced to it. NP-complete problems are believed to be essentially beyond the reach of polynomial algorithms.

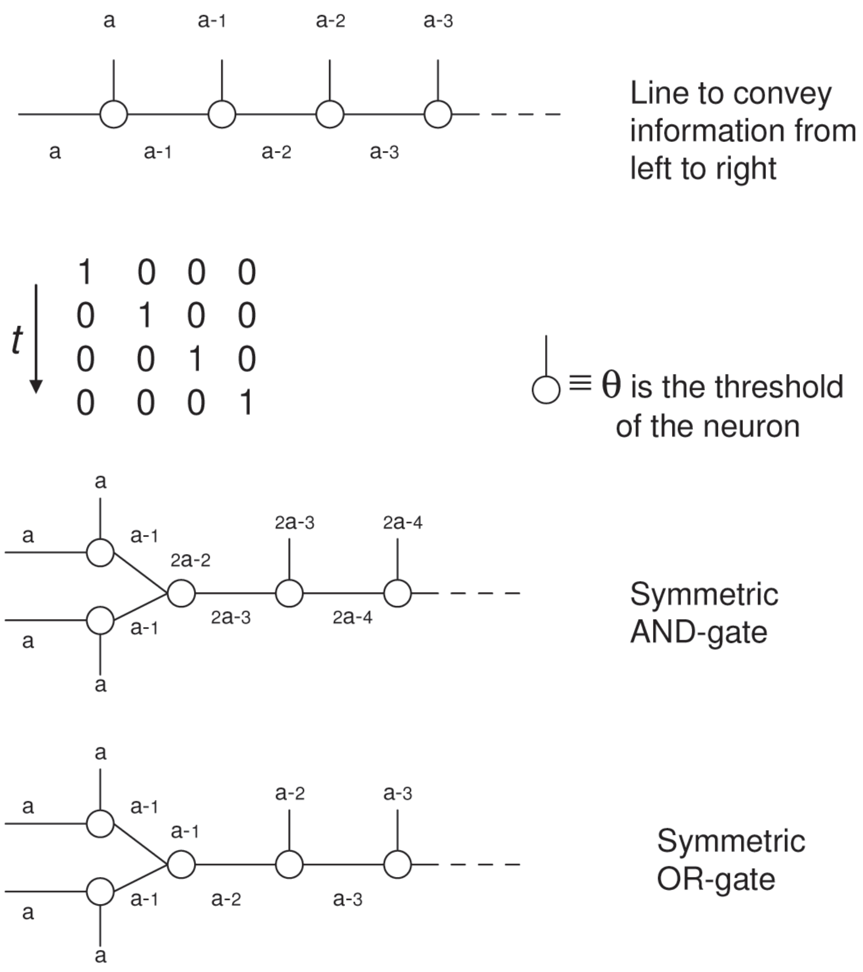

Figure 6: Symmetric neural logical gates.

Now we will prove that there exists a solution of (P) if and only if there is a fixed point of $N = (A,\vec{0},\{-1,1\}^n)$. Suppose first that there exists a fixed point z = (u,v, y_1, .., y_s); i.e:

$$y_1) = \mathrm{sgn}(-cu + cv + \sum_{i=1}^{s} a_i y_i) = \mathrm{sgn}(\sum_{i=1}^{s} a_i y_i) \quad (1)$$

$$y_2) = \mathrm{sgn}(cu - cv + \sum_{i=1}^{s} a_i y_i) = \mathrm{sgn}(\sum_{i=1}^{s} a_i y_i) \quad (2)$$

$$y_3) = \mathrm{sgn}(c_1 u - a_1 v + y_3) \quad (3)$$

....

$$y_s = \mathrm{sgn}(a_1 u - a_s v + y_s) \quad (s)$$

where $sgn(u) = 1$ iff $u \geq 0$ (-1 otherwise)

1. Let us first consider the case (u, v) = (1,1) from equations (1) we get:

$\mathrm{sgn}(\sum_{i=1}^{s} a_i y_i) = 1$, which is equivalent to $\sum_{i=1}^{s} a_i y_i \geq 0$.

Now, in a similar way to equation (2) we have $\sum_{i=1}^{s} a_i y_i \leq 0$. From the two inequalities we conclude $\sum_{i=1}^{s} a_i y_i = 0$. So the fixed point of the network is a solution of the problem (P).

So it's direct. That is to say if y is a solution of (P) then $\sum_{i=1}^{s} a_i y_i = 0$, so the vector y satisfies the set of equations (1),(2), …, (s) hence it is a fixed point of the network N.

For the other pairs (u,v) $\neq$ (1,1) the vector z is not a fixed point. Suppose for instance the case (u,v) = (-1,-1). Clearly from equations (1) and (2): $\sum_{i=1}^{s} a_i y_i < 0$ and $\sum_{i=1}^{s} a_i y_i > 0$, which is a contradiction. From previous analysis we have established that fixed points of (H) and solutions of (P) are *exactly* the same, so, since (P) is NP-Complete also (H).

CONCLUSIONS AND PERSPECTIVE

As we had said in the beginning the majority of interesting dynamical problems for neural nets are hard to solve, so one has to develop approximations or local strategies to build neural nets to solve particular problems. From this point of view one may say that after McCulloch work we continue, at least from the theoretical point of view, to see that the use of ANN models to understand real neurons is still a challenge at the edge of our mathematical and algorithmical knowledge.

An important issue that can help to develop further application of ANN to brain sciences is to expand, mathematically, their internal adaptive properties. For instance, the θ threshold function should be set to depend not only on a short term plasticity process but also on a long term plasticity process that would involve, e.g., synthesis of new protein, similar to the long term plasticity process that follows learning (Whitlock et al. 2006).

Recently, the introduction of new methodologies, such as multielectrode arrays for simultaneous neural brain recording opens new issues for ANN, since we are able now to explore the way that brain neural networks develop associative or cooperative neural behavior (i.e. synchronization, oscillation), which is critical to understand the mechanisms involved in the brain communication necessary to bring perceptual coherence to brain activity.

ACKNOWLEDGMENTS

Partially supported by Fondecyt 1070022, Centro de Modelamiento Matemático (CMM-UCH) (E.G.), PBCT-CONICYT ACT45 (AGP). We are grateful to John Ewer for editorial suggestions.

REFERENCES

ANDERSON JA, ROSENFIELD E (1988) Neurocomputing. Foundations of Research, MIT Press

ARBIB MA (2002) The Handbook of Brain Theory and Neural Networks. The MIT Press

FLOREEN P (1992) Computational Complexity Problems in Neural Associative Memories. Report A-1992-5 Dept- of Computer Science, Univ of Helsinki, Finland

GOLES E (1982) Fixed point of threshold functions on a finite set. in SIAM Journal on Algebraic and Discrete Methods Vol. 3. Nº 4

GOLES E, FOGEMAN F, PELLEGRIN D (1985) The energy as a tool for the study of threshold networks. in Discrete Applied Maths 12: 261-277

GOLES E, OLIVOS J (1980) Periodic Behavior of Generalized Threshold Functions. Discrete Math 30: 187-89

HOPFIELD JJ (1982) Neural networks and physical systems with emergent collective computational abilities. Proc Natl Acad Sci USA 79: 2554-58

KAMP Y, HASLER M (1990) Réseaux de neurones récursifs pour mémoires associatives, Presses Polytechniques Romandes, Lausanne, Switzerland. English edition *Recursive neural networks for associative memory* by Wiley and Sons, London, 1990

MCCULLOCH W., PITTS W. (1943). A logical calculus of the ideas immanent in nervous activity, Bull of Mathematical Biophysics 5: 521-531

MINSKY M, PAPERT S., (1988). Perceptrons. The MIT Press, expanded edition, (first edition 1969)

ROSENBLATT F. (1958). The Perceptron: a probabilistic model for information storage and organization in brain. Psychological Review 65: 386-408

RUMELHART D.E., HINTON G.E., WILLIAMS R.J. (1986). Learning internal representation by error propagation, in Parallel distributed processing: exploration in the microstructure of cognition. In: Rumelhart and McClelland eds, Cambridge, MIT Press, pp 318-362

WHITLOCK JR, HEYNEN AJ, SHULER MG, BEAR MF. (2006). Learning induces long-term potentiation in the hippocampus. Science. 13: 1093-7