CONTEXT-SENSITIVE ASSOCIATIVE MEMORY: "RESIDUAL EXCITATION" IN NEURAL NETWORKS AS THE MECHANISM OF STM AND MENTAL SET [*]

Victor Eliashberg [†]

Abstract

Most of the presently popular neural models of associative memory ignore nontrivial temporal processes in neural elements. Such "timeless" models describe interesting effects of spatial cooperation, but do not address the problem of temporal coordination and temporal context. Effects such as temporal associations, short–term memory (STM), mental set, etc. can be represented naturally in neural models by introducing states of "residual excitation" (E–states) in the neural elements (Eliashberg [7][8][9][11]). One such model illustrating the E–state concept is discussed in this paper.

0 INTRODUCTION

Intuitively, a context–sensitive associative memory (CSAM) is a system that stores and executes temporal associations, that is, associations between time sequences (strings) of input and output vectors. If an input string, x, is present in the training sequence, CSAM retrieves an output string, y, associated with x. (In a more complex case it may be desirable to retrieve the whole subset of such strings, or a function, e.g., the weighted sum of such a subset). If x is not present in the training sequence, CSAM retrieves a string y' associated with x', where x' is one of the present input strings most similar to x (an effect of generalization by similarity). A conventional associative memory (CAM) is a special case of CSAM when the lengths of strings are equal to unity.

Different models of associative memory have been studied by Anderson [1], Cooper [6], Eliashberg [7] [8], Grossberg [14], Hopfield [16], Kanerva [18], Keeler [19], Kohonen [20], Kosko [21], Sampolinsky and Kanter [29], and others. Some basic ideas employed in such models were introduced by Rosenblatt [28] and Widrow [32].

Besides temporal associations a CSAM model must be able to explain such effects as "decaying" short–term memory (STM) and mental set.

What is STM?
How does it interact with LTM?
What is the meaning of "memory span"? (The "magical number" and recoding problem of Miller [25]).
Why do we "see more than can be remembered [30]?"

The phenomenon of mental set is even more intriguing. It is well known that the "curse of dimensionality", associated with the problem of a combinatorial number of mental sets (contexts), plagues, in one way or another, most of the existing cognitive theories [33][9].

What is mental set?
How do we dynamically change our interpretation of input information depending on context?
How can a piece of knowledge acquired in one context be efficiently used in a combinatorial number of other possible contexts?

Since 1966 I have been exploring the possibility that the phenomena of STM and mental set can be modelled in a uniform way by CSAM models with

[*] Proceedings of IJCNN-89, June 18-22, Washington., D.C. vol. I, pp. 67-75

[†] Visit the web site www.brain0.com for information

neural elements having states of ”residual excitation” [7][8][9][10][11]. In neurophysiology the intuitive idea of residual excitation as the mechanism of mental set can be traced back to Vvedensky [31]. Somewhat similar general ideas were employed in different forms by Anderson [2], McClelland and Rumelhart [23], and others.

I formalize the intuitive idea of ”residual excitation” in a rather broad sense by introducing the concept of phenomenological E–states (E stands for Excitation). Neural elements have many different ”residual–excitation–like” states of dynamic analog memory (see next section). All such states are referred to as E–states. My goal is to show how such analog memory can be efficiently used to dynamically restructure the sets of associations stored in CSAM.

This paper consists of 9 sections. Section 1 establishes a link between the dynamics of macroscopic E–states and the statistical dynamics of the protein molecules in neural membranes. Section 2 formulates system–theoretical requirements for CSAM models. The intuitive concept of CSAM is connected to the concept of a learning machine universal with respect to the class of finite–memory machines. Section 3 presents a neural network implementing a CSAM model. Sections 4–8 describe some properties of the model from section 3. Section 9 outlines some possibilities for further development of the discussed ideas.

Collective Properties vs. Individual Cells. Since the influential work of Hopfield [16] much of the research in neural modelling has been devoted to the study of collective effects in networks built from simple elements. In contrast, this paper emphasizes the importance of the internal complexity of individual neural elements.

Symbolic vs. Nonsymbolic. There is an ongoing debate about the general style of information processing in the brain: symbolic (or rule–based) models vs. connectionist (or activation–based) models [27]. This paper suggests that the truth lies between these two extremes. The brain’s LTM is essentially symbolic. The brain’s STM and ITM (intermediate–term memory) are essentially nonsymbolic. (See Section 9.)

Acknowledgements. I am grateful to Prof. M. Hirsh and Prof. D. Rumelhart for giving me opportunities to present the results of my work at their seminars. I also thank Prof. B. Widrow for several useful discussions.

1 THE LOGIC OF PROTEIN MOLECULES AND E–STATES

To simplify the analysis of the collective properties of neural networks it is common to treat a synapse as a variable resistor and a neuron as a summing operational amplifier with a nonlinear output. Though valuable as ”zero–approximations”, such theories cannot be taken too serously in light of the data from modern cellular research (see [22, 5] for a comprehensive review of some facts and ideas available in this area).

The cellular data suggests that real neurons and synapses are much more than just operational amplifiers and variable resistors. They are sophisticated information processing elements with complex internal states. One of the goals of this paper is to show that such a complexity is, indeed, needed for the implementation of nontrivial models of CSAM.

In what follows I show how dynamics of macroscopic E–states can be connected to statistical dynamics of the protein molecules in neural membranes. Some other temporal processes in neural elements, e.g., the accumulation and depletion of the releasable synaptic vesicles, etc. [12, 13], are also good candidates for the cellular implementation of E–states. I suggest, however, that protein molecules look particularly attractive from the information processing viewpoint. The theory proposed below can be viewed as an information processing continuation of the classical Hodgkin and Huxley [15] theory.

Let us treat a single protein molecule of a given type (e.g., a sodium or a potassium channel protein molecule) as a probabilistic finite–state machine. The probabilities of transitions between different states (conformations) of such a machine are affected by different external inputs: membrane potential, concentrations of neurotransmitters and other extracellular and intracellular molecular entities, etc. The average numbers of such machines (molecules) in different states affect ionic currents, the rates of the catalytic synthesis of second messengers, etc. Let us identify these average numbers of different machines (molecules) in different states (conformations) with different types of E–states. Though the dynamics of a single machine is discrete, the resulting dynamics of E–states is analog. This analog dynamics is governed by a potentially quite sophisticated logic inside each machine. I suggest that this is exactly the right level of complexity of neural elements needed to implement nontrivial CSAM models.

To give this verbal description a specific formal meaning, in what follows I present a simple mathematical model of a neuron with hypothetical inertial modulating synapse. To avoid misunderstanding I must emphasize that the goal of the model is to demonstrate the possibilities of a mathematical tool rather than to explain any specific cellular data. An effect of inertial neuromodulation qualitatively similar to that described below can be derived from many different hypotheses about cellular mechanisms.

In Figure 1a synapse S has M channel protein molecules in the postsynaptic membrane. (For the sake of simplicity the model has only one channel.) Each molecule is a probabilistic machine with the three states (conformations) shown in Figure 1b. State 1 is the rest state. In this state the molecule has zero conductance. State 1 can be transferred into "pre–active" state 2 in the presence of a neurotransmitter released through the presynaptic membrane. State 2 is voltage–sensitive. Increasing membrane potential U increases the probability of transition from state 2 to the conducting state 3.

Let M_1, M_2, and M_3 be the numbers of molecules in state 1, 2, and 3, respectively. Let P_{ij} be the probability of transfer from state j to state i. Let, for the sake of simplicity, P_{21} be proportional to the presynaptic signal J_3 and let P_{32} be proportional to membrane potential U. Let P_{12} and P_{23} be constants, and let the membrane potential be proportional to the sum of currents J_1 and J_2, where J_2 is the current from synapse S, and J_1 is the net current from all other synapses not shown in Figure 1a. Let J_2 be proportional to the number of channels in state 3. (For the sake of simplicity I ignore the dependence of J_2 on potential U.)

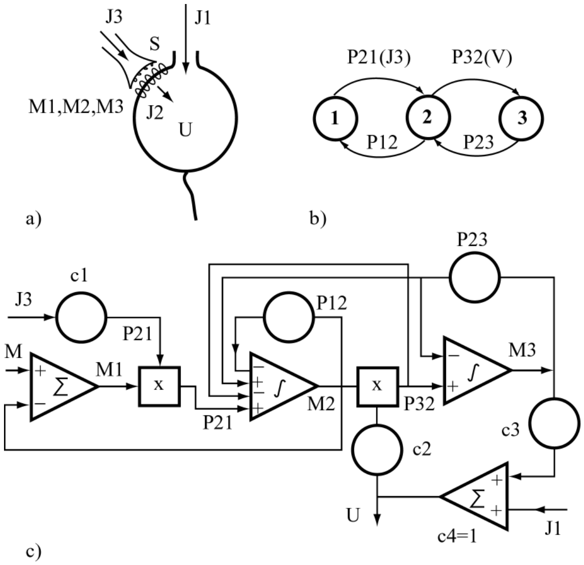


Figure 1: a) A neuron with a modulating synapse. b) A channel protein molecule is treated as a probabilistic finite–state machine with three states (conformations). c) A synapse can be more than just a variable resistor.

It is easy to show that signal J_3 produces an inertial modulation of signal J_1 due to the positive feedback associated with the channels in state 3. M_3 increases with U while there are molecules in state 2. Under the above assumptions, the model is described by the following set of equations (1-7):

$$\frac{dM_2}{dt} + (P_{12} + P_{32}(U)) \cdot M_2 = P_{21}(J_3) \cdot M_1 + P_{23} \cdot M_3 \tag{1}$$

$$M_1 = M - M_2 - M_3 \tag{2}$$

$$\frac{dM_3}{dt} + P_{23} * M_3 = P_{32}(U) * M_2 \tag{3}$$

$$P_{21}(J_3) = c_1 * J_3 \tag{4}$$

$$P_{32}(U) = c_2 * U \tag{5}$$

$$J_2 = c_3 * M_3 \tag{6}$$

$$U = c_4 * (J_1 + J_2) \tag{7}$$

Let $P_{23} \gg P_{12}$. Then $M_3 = \frac{P_{32}(U)}{P_{23}} * M_2$ and

$U = \frac{J_1}{1 - b * M_2}$

if $b * M_2 \ll 1$ then

$$U = J_1 * (1 + b * M_2) \tag{8}$$

I will use this simplified expression in the CSAM model described in section 3. M_2 is an E–state. M_3 is also an E–state, which will be ignored.

Figure 1c shows an implementation of Expressions (1-7) with the use of four operational amplifiers (two of them integrating). The modulating effect is due to the feedback $M_3 \rightarrow U \rightarrow P_{32}$. The temporal effect is due to the small value of P_{12}. If P_{23} is high, one can ignore the additional temporal effect associated with M_3. In Figure 1c the E–states are the outputs of the integrating operational amplifiers. If the temporal properties of M_3 are ignored then only one type of E–states (one integrating amplifier) is needed to describe the corresponding analog memory.

2 SYSTEM–THEORETICAL DEFINITIONS

In what follows I connect the intuitive idea of CSAM with the concept of a learning machine universal with the respect to the class of finite–memory machines.

2.1 Definition

An m-th order probabilistic finite–memory machine (also known as an m-th order Markov machine) is a system M=(**X**,**Y**,**Q**,F,m), where **X** and **Y** are finite sets of input and output symbols, respectively, (input and output alphabets).
$\mathbf{Q} \subset \mathbf{X}^m \times \mathbf{Y}^m$ is a finite set of states, represented by the strings over $\mathbf{X} \times \mathbf{Y}$ with the length not exceeding m.
$\mathbf{F} : \mathbf{X} \times \mathbf{Q} \times \mathbf{Y} \rightarrow [0, 1]$ is the function of output conditional probabilities. The machine works as follows:

$$P(y(\nu) = y'|x(\nu) = x', q(\nu) = q') = F(x', q', y')$$

where,
ν is discrete time,

$q(\nu) = x(\nu - 1), ..x(\nu - m), y(\nu - 1), ..y(\nu - m),$

$P(A|B)$ is the conditional probability of A, given B.

A finite–memory machine is *output–independent* if its state $q(\nu)$ does not depend on its output $y(\nu - 1)$...
A finite–memory machine is *deterministic* if $F(x, q, y) \in [0, 1]$.
A 0–th order finite–memory machine is called a *combinatorial machine*, a machine without memory, or a trivial machine.

2.2 Definition

A learning finite–memory machine is a system ML=(**X**,**Y**,**Q**,**G**,FY,FG), where **X**,**Y**,**Q** are the same as in Definition 2.1.
G is the set of the states of LTM of ML.
$FY : \mathbf{X} \times \mathbf{Q} \times \mathbf{G} \times \mathbf{Y} \rightarrow [0, 1]$ is the function of output conditional probabilities.
$FG : \mathbf{X}^m \times \mathbf{Y}^m \rightarrow \mathbf{G}$ is the data storage procedure (learning procedure).

NOTE. For the sake of simplicity I describe FG as a "batch"–style mapping. In real life such a mapping is always implemented by an "incremental" next G–state procedure.

Let **C** be a class of machines from Definition 2.1. Machine ML is *universal with respect to class* **C** if for each machine M from **C** there exists $g \in \mathbf{G}$ such that FY(g)=F, that is, for all $(x, q, y) \in \mathbf{X} \times \mathbf{Q} \times \mathbf{Y}$ $FY(x, q, g, y) = F(x, q, y)$. Machine ML is a *learning system universal with respect to class* **C** if for each $M \in \mathbf{C}$ there exists a training sequence $s \in \mathbf{X} \times \mathbf{Y}$ such that FG(s)=g and FY(g)=F.

The definition of universality requires that for any machine M from **C** there would be at least one state $g \in \mathbf{G}$ of machine ML such that ML in this state simulates M. The definition of universality as applies to a learning system requires that such states would be attainable in an experiment of supervised learning. It is assumed that in the course of training the teacher can "clamp" the outputs of ML and enable the data storage mechanism. For the sake of simplicity , these additional inputs are not included in the system–theoretical definition of ML.

2.3 Problems

To develop the concept of CSAM it is important to solve the following problems:

a) Describe a neural network universal with respect to the class of finite-memory machines from definition 2.1.

b) Describe a learning procedure FG transforming this network into a learning system universal with respect to this class of machines.

Additional requirement. In the case of incomplete knowledge the network must be able to generalize by similarity.

2.4 Traditional Solution

Conventional associative memories (CAM) solve Problem 2.3 for the case of $m = 0$. A solution for a fixed $m > 0$ is then obtained as shown in Figure 2. This traditional solution, however, has at least three major drawbacks:

1. The amount of LTM required to store associations $x_m^* \to y$, where x_m^* is an input string of length m, increases proportionally to m (as compared with the optimal case of compact storage mentioned later in section 5.4).

2. The order m must be specified in advance.

3. Two input strings such as, say, "abcde" and "bcdef", which differ by a one–step shift, are treated as entirely different input vectors.

Kanerva [18] uses "folds" in his SDM (Sparse Distributed Memory) to solve the problem of the m-th order temporal associations [19]. This approach, however, also encounters problems 1 and 2. In the following section I show how the idea of E–states allows one to overcome these problems. The proposed approach also offers a broad range of other advantages.

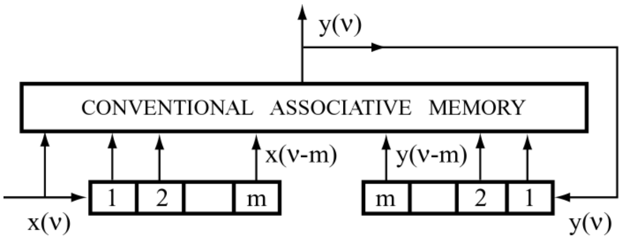


Figure 2: A conventional (context–free) associative memory, with the input delay lines (shift registers) shown, can simulate an arbitrary $m-th$ order finite–memory machine. This approach, however, requires more memory than is needed.

3 A NEURAL MODEL OF CSAM

3.1 Morphological Structure

Consider the four–layer neural network schematically shown in Figure 3. Large circles with incoming and outgoing lines represent neurons with their dendrites and axons, respectively. Small white and black circles represent excitatory and inhibitory synapses, respectively. Figure 3 shows six sets of neurons (N1–N6) and 10 sets of synapses (S21, S22, S25, S52, S32a, S32b, S33, S36, S63, S43).

NOTATION. Because of the considerable complexity of the network in Figure 3, I need sufficiently powerful notation to describe its structure and functions. Therefore, I abandon traditional scientific notation (single letter identifiers with subscripts and superscripts, etc.) in favor of a more powerful computer–like language notation. I do not need, however, the power of a full–blown computer language, so I retain the freedom associated with scientific notation.

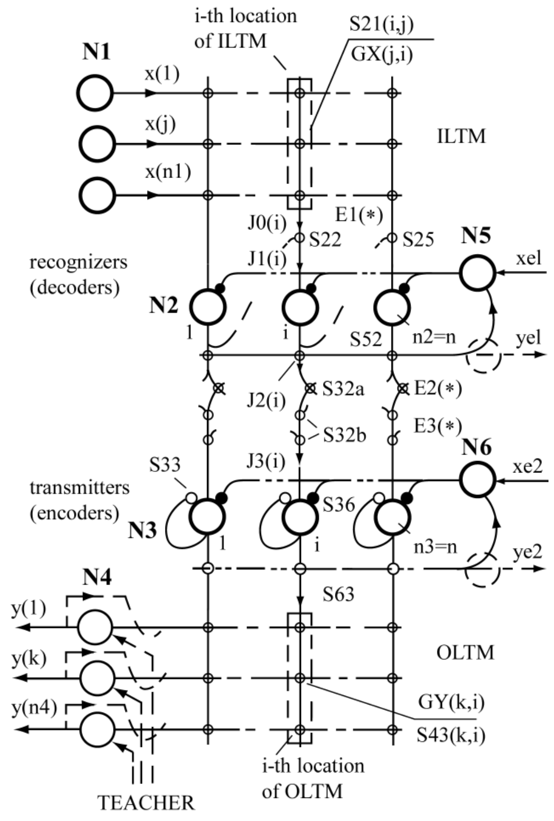


Figure 3: A neural implementation of a CSAM (Model 3.2)

Nj(i) is the i-th neuron from the j-th set. Skja(n,m) is the synapse between neurons Nj(m) and Nk(n), where 'a' is an additional index describing the type of synapse (in case there are several different types of synapses between the sets of neurons Nj and Nk). The '*' substituted for an index indicates the whole subset of elements (variables) corresponding to the entire set of values of this index. S21(i,*)=(S21(i,1),...S21(i,n1)), S21(*,*) is the same as S21, etc.

NOTE. I use notation Skja(n,m) for two purposes:

a) as the name of an element of the network, when discussing the morphological structure of the network,

b) as the characteristic function $Skja(n,m) \in \{0,1\}$ describing a set of available synapses. For example, S21(i,j)=1 (j=1,..n1, i=1,..n2) means that there are n1*n2 synapses in the set S21, etc. It should always be clear from the context what meaning is implied.

I use notation "$\lceil\ \rceil$" borrowed from Minsky and Papert [26] to assign values {0,1} to a predicate: if P=true then $\lceil P \rceil$=1, otherwise $\lceil P \rceil$=0. This allows us to conveniently describe the structure of the sets of available synapses. For example, S22(i,j)=$\lceil i = j+1 \rceil$, S32a(i,j)=$\lceil i = j \rceil$, S32b(i,j)=$\lceil j \in \{i-r, ..i+r\} \rceil$.

The next section presents a simplified "phenomenological" description of the work of the network of Figure 3 in discrete time ν. It can be shown that a similar, but more complex, performance can be derived from the continuous time model of this network [8]. This paper, however, does not have enough space to address this interesting and complex subject.

3.2 Functional Model

Output procedure: (for all i=1,..n)

$$\begin{aligned} J0(i) &= SUM(j=1,n1)(GX(j,i) \cdot x(j) \\ &= FS(GX(*,i), x(*)) \end{aligned} \tag{1}$$

The vector of signals, $x(*)$, from neurons N1 (the input vector of the model) reaches dendrites of all neurons N2. The net postsynaptic current J0(i) of synapses S21(i,*) is the dot product of $x(*)$ and GX(*,i), where GX(j,i) is the gain of S21(i,j). (I intentionally transposed indices i and j.) GX(*,i) will be treated as the vector stored in the i-th location of the Input LTM (ILTM) of the model. SUM(j=1,n1) denotes the sum over j=1,..n1. $FS : \mathbf{X} * \mathbf{X} \to \mathbf{R}$ is called a similarity function. We will require pair (FS,X) to satisfy the following condition:

For all

$$x, x' \in \mathbf{X}\ (x \neq x' \to FS(x,x) > FS(x,x')) \tag{2}$$

Each vector from **X** is closer to itself than to any other vector from **X**. Expression (2) is referred to as the *correct decoding* condition.

$$J1(i) = J0(i) * (1 + b1 * E1(i)) \tag{3}$$

In Figure 3 neuron N2(j) temporarily modulates neuron N2(i) via modulating synapse S22(i,j). Expression (3) describes the multiplicative effect of $E1(i)$ on $J0(i)$.
$b1$ is a constant. Expressions (8) and (11), presented later, describe the process of temporal lateral modulation.

I use word *"pre–tuning"* to denote such a process. This word is a translation of the Russian word *"prednastroika"* used by Bernstein [4].

if $J1(i) - xe1 > 0$ then

$$J2(i) = J1(i) - xe1 \tag{4}$$

else J2(i)=0

Neuron N2(i) transforms J1(i) into J2(i). The threshold of N2(i) is determined by the global inhibitory input $xe1$. For the sake of simplicity we do not take into account inhibitory feedback N2-S52-N5-S25-N2. (The only reason this feedback and output $ye1$ are shown in Figure 3 is to suggest that they can be useful in more complex models.)

$$J3(i) = G32a(i,i) * J2(i) * (1 + b2 \cdot E2(i) + b3 \cdot E3(i)) \tag{5}$$

Output J2(i) of neuron N2(i) is transformed by synapse S32a(i,i). The synapse has gain G32a(i,i) describing the state of its LTM. It also has an E–state, E2(i), describing an effect of temporary facilitation. The effect of J2(i) on neuron N3(i) is further influenced by modulating synapses S32b. For the sake of simplicity the net effect of such temporal lateral modulation is described by a single state E3(i).

The following expressions describe the winner–take–all choice of a neuron from layer N3.

$i0 :\in MAXSET =$
$\{i : J3(i) = max(J3(i), ..J3(n)) > xe2\}$

if $i = i0$ then

$$J4(i0) = 1 \tag{6}$$

else $J4(i) = 0$

I use the Pascal–like notation $i :\in M$ to designate a random equally probable choice of an element i from the set M. Expressions (6) describe the winner–take–all choice performed by neuron layer N3. The dynamics of such a layer was studied in detail in [7, 10].

In some cases it is interesting to replace Expressions (6) with the following Expression (6a):

$i0 \in MAXSET$ then $J4(i) = 1$ else $J4(i) = 0$ (6a)

The output is described as follows:

$$y(k) = SUM(i-1, n)(GY(k,i) * J4(i)) \tag{7}$$

where $GY(k,i)$ is the gain of S43(k,i). Since in the case (6) only a single neuron N3(i0) is firing, $y(*) = \ GY(*, i0)$. Vector $GY(*, i)$ is treated as the vector stored in the $i - th$ location of the Output LTM (OLTM) of the model. In the case of (6a), a superposition of vectors stored in all locations from MAXSET is retrieved.

Next E–state procedure (pre–tuning):

$$Q1(i) = SUM(j = 1, n)(G22(i,j) * J2(j)) \tag{8}$$

$$Q2(i) = G32a(i,i) * J2(i)) \tag{9}$$

$$Q3(i) = SUM(j = 1, n)(G32b(i,j) * J2(j)) \tag{10}$$

where Q1(i), Q2(i) and Q3(i) are net modulating inputs affecting states E1(i), E2(i) and E3(i), respectively. G22, G32a, G32b are the gains of S22, S32a and S32b, respectively.

if $Qj(i) >= Ej(i)$ then $Ej(i, \nu + 1) = Qj(i)$
else

$$Ej(i, \nu + 1) = cj * Ej(i) \tag{11}$$

where $j = 1, 2, 3$

Expression (11) represents three statements for $j = \ 1, 2, 3$.

c1,c2,c3 are constants that determine the rate of decay of E1,E2,E3.

ν is discrete time. For the sake of simplicity I explicitly write index ν only in the variables depending on

$\nu + 1$. That is, $E1(i)$ is the same as $E1(1,\nu)$, etc. Note that index ν is used in the same way as it is used in scientific notation. To simulate Expression (11) on a computer one needs only one–dimensional arrays $E1(*)$, $E2(*)$, $E3(*)$.

Data storage procedure (learning):

To solve Problem 2.3b it is sufficient to assume that the training sequence is simply "tape–recorded" into ILTM and OLTM (rote learning).

Such a learning procedure ("next impression next cell") was advocated by Meynert [24]. It could be implemented, in principle, by a genetic mechanism. Importantly, more complex but less universal learning procedures (e.g., backpropagation) are unable to solve problem 2.3b. Such procedures lose information about the order of vectors in the training sequence, and, therefore, cannot be used in CSAM models.

In this paper I treat synaptic gains G22, G32a and G32b as parameters. In a general case it is useful to view the sets of such gains as the states of a "Structural" LTM (SLTM). The concept of SLTM is intuitively similar to the concept of a traditional connectionist LTM (links between nodes). In contrast, ILTM and OLTM can be better characterized as the memories for storing encoded (symbolic) knowledge.

To demonstrate some nontrivial functional properties of Model 3.2, in the following sections 4–8 I discuss four of its special cases.

4 INSTANT ASSOCIATIONS: SIMULATION OF COMBINATORIAL LOGIC AND FINITE–STATE MACHINES

4.1. MODEL

This model is defined as follows:

$b1 = b2 = b3 = 0$
$G32a(i,j) = \lceil i = j \rceil$ (see notation in 3.1)
$xe1 = xe2 = 0$

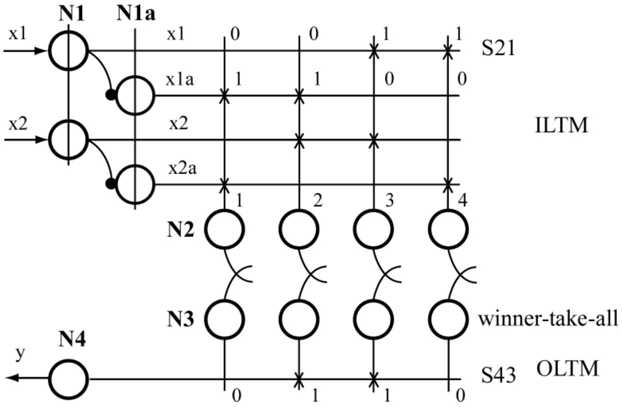


Figure 4: A special case of Model 3.2 (Model 4.1): simulation of the XOR function.

X is a set of real vectors satisfying correct decoding condition 3.2(2) with similarity function 3.2(1). Expression 3.2(6) is used (not expression 3.2(6a)).

The following general result can be proved.

4.2. THEOREM

For any probabilistic combinatorial machine M=(**X**,**Y**,F) with rational output probabilities ($F(x) = m/n$, where m is a non–negative integer and n is a positive integer), there exists a training sequence such that Model 4.1 learns to simulate M, provided the size of ILTM and OLTM (parameter n) is big enough to store this sequence.

4.3. Normalization

There are many ways to encode input symbols of machine M from 4.2 to satisfy the correct decoding condition 3.2(2). One way is to normalize vectors from **X**. There are several ways to obtain such a normalization. In what follows I discuss the case of $x, y \in \{0,1\}$. In this case (Boolean functions) one can use the idea of the ON and OFF neurons shown in Figure 4.

Model 4.1 with the contents of ILTM and OLTM shown in this figure simulates the XOR function. N1a are OFF neurons. They fire if not inhibited. If xj=0 then xja=1, otherwise xja=0. The crossed synapses

S21 and S43 have gains equal to unity. The rest of these synapses have gains equal to zero.

4.4. Introducing delayed feedback

Let in Model 4.1 **X**=**X1**× **X2**, **Y**=**Y1**× **Y2**. Let **X2**=**Y2** and let both **X1** and **X2** satisfy correct decoding condition 3.2(2). Let us introduce delayed feedback $x2(\nu+1) = y2(\nu)$ and view $y1$ as the output of the modified Model 4.1. It follows from 4.2 that the resulting model is a learning system universal with respect to the class of probabilistic finite–state machines with rational output probabilities.

5 TEMPORAL ASSOCIATIONS: SIMULATION OF FINITE–MEMORY MACHINES

5.1. MODEL

The same as Model 4.1, except:
$b1 > 0, 1 > c1 > 0$
$G22(i,j) = \lceil i = j+1 \rceil$. That is, there is a left–to–right–next–neighbor pre–tuning (temporal neuromodulation) in layer N2 (see Figure 5). The following general result can be proved.

5.2. THEOREM

For any output–independent finite–memory machine M=(**X**,**Y**,**Q**,F,m) with rational output probabilities there exists a training sequence such that Model 5.1 learns to simulate M, provided n is big enough to store this sequence.

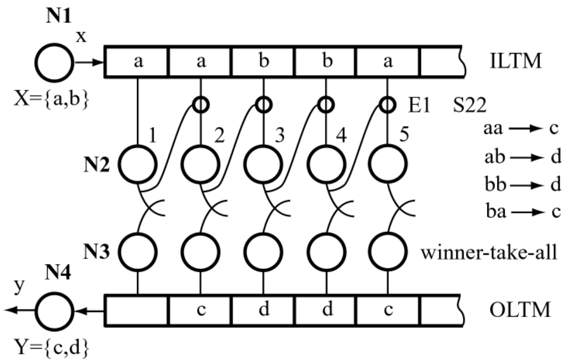


Figure 5: A special case of Model 3.2 (Model 5.1): simulation of output–independent finite memory machine.

NOTE. There exists a broad range of parameters of Model 5.1 for which theorem 5.2 holds. For example, let **X** be the set of vectors with length equal to unity. Then the theorem holds for $b1 < 1$ and $c1 < 1 - b1$.

5.3. Calculating activation level

Let for $\nu = 0$ $E1(i,\nu) = 0$, and let the input sequence $x(*,0),..x(*,\nu) = GX(*, i-\nu)...GX(*,i)$ for some $i \in \{1,..n\}$. That is, the input string is equal to a string written in locations $(i-\nu)...i$ of ILTM. Let **X** be the set of vectors with length equal to unity. It can be shown that in this case:

$$J2(i,\nu) = 1 + b1 + (b1)^2 + ..(b1)^\nu =$$
$$(1-(b1)^{\nu+1})/(1-b1)$$

5.4. Compact learning sequence

Model 5.1 with the training sequence (stored in ILTM and OLTM) shown in Figure 5 simulates the first order deterministic finite–memory machine with the following set of commands: $aa \to c$, $ab \to d$, $bb \to d$, $ba \to c$. It can be shown that for an $m-th$ order machine the shortest training sequence (a comact sequence) has length $L = N^{m+1} + m$, where $N = |\mathbf{X}|$ is the number of symbols in **X**.

Since Model 5.1 can learn using a compact training sequence, no learning system universal with respect to the class of output independent finite–memory machines can learn faster than Model 5.1. The model does not have the drawbacks 1-3 mentioned in section 2.4.

5.5. General case

Let in Model 5.1 **X**=**X1**×**X2**, **X2**=**Y** and, **X1** and **X2** both satisfy 3.2(2). Let $x2(*,\nu+1) = y(*,\nu)$. It follows from 5.2 that the resulting model is a learning system universal with respect to the general class of finite–memory machines with rational output probabilities.

6 MENTAL SET AND STM: CONTEXT–SWITCHABLE COMBINATORIAL LOGIC

6.1. MODEL

The same as Model 4.1, except: $b2 > 0$, $1 > c2 > 0$. Synapse $S32a(i,i)$ becomes temporarily more efficient after firing of neuron N2(i) due to the fast "charging" of E2(i) (Expressions 3.2.9 and 3.2.11). If after such firing N2(i) is inactive, E2(i) gradually "discharges" (with the time constant 1/(1-c2)). This effect of temporary facilitation in synapses S32a, added to the properties of Model 4.1 described in section 3, creates the interesting effect of "mental set" discussed below.

6.2. Pre–activating logic functions

Let us extend the network of Figure 4 as shown in Figure 6, and let us ignore for a while input $x3$ and feedback $y \rightarrow x3$. As Figure 6 shows, the network was trained to perform two functions: AND and NAND. As a result of that, it has a set of ambiguous associations $(x1, x2) \rightarrow y$ stored in its LTM (ILTM and OLTM): $(0,0) \rightarrow 0$ at $i = 1$, $(0,0) \rightarrow 1$ at $i = 5$, etc. Let us now take into account input $x3$. The training sequence in this input was exactly the same as the y–sequence. Associations $(x1, x2, x3) \rightarrow y$ are unambiguous. During the examination stage let the experimenter send the following sequence of input signals: $(0,0,0), (0,1,1), (1,0,1)$ and $(1,1,0)$. The resulting profile of $E2(*)$ pre–tunes the subset of locations MXOR={1,6,7,8} corresponding to the XOR function (associations from MXOR are more active than competing associations). Depending on the rate of decay, $c2$, this situation holds for a certain period of time, T. During this time Model 6.1 reacts to inputs $(x1, x2)$ like the XOR function.

Let us take into account feedback $y \rightarrow x3$. This feedback provides a refresh of the pre–activated profile of $E2(*)$. Suppose that during each time interval of length T, any input (x1,x2) from $(0,0), (0,1), (1,0), (1,1)$ appears at least once. Then it can be shown that MXOR set remains in the dominant position arbitrarily long. The mental set supports itself.

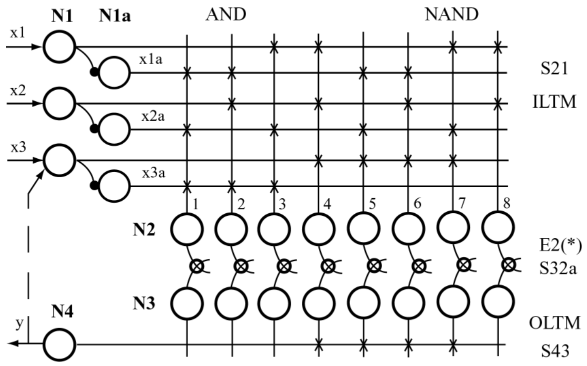


Figure 6: Context–switchable combinatorial logic. A CSAM that was taught to simulate the AND and NAND functions can simulate all 16 2–input 1–output Boolean functions. This effect of mental set is dramatic in the case of more inputs. A CSAM with $n = 2^{11} = 2048$ learns to simulate 2^{1024} 10–input 1–output Boolean functions.

The described effect is particularly dramatic in the case of more than two inputs. Let us consider the class of Boolean functions with p inputs and r outputs. The number of such functions is:

$N(p,r) = Ny^{Nx}$,
where $Ny = |\mathbf{Y}| = 2^r$ and $Nx = |\mathbf{X}| = 2^p$

It can be shown that Model 6.1 with $n = Nx * Ny$, $n1 = 2 * (Nx + Ny)$ and $n4 = Ny$ can, theoretically, have mental set corresponding to all these functions. Since E2 is a continuous variable, there are no theoretical limitations on the time interval T. In reality, of course, one can get only a limited memory span using this technique.

Let $p = 10$ $r = 1$. The number of such Boolean functions is 2^{1024}. Model 6.1 with $n1 = 22$, $n4 = 1$ and $n = 2048$ has mental sets corresponding to all these functions. This shows that the "curse of dimensionality" mentioned in the Introduction is a penalty for the attempt to describe the phenomenon of mental set in the state space of insufficiently large dimensionality. That is, the penalty for ignoring E–states [9].

7 SIMULATION OF RANDOM ACCESS MEMORY. MENTAL IMAGERY

7.1 Simulation of RAM

Let $\mathbf{A} = \{a1, ..ak\}$ be the set of addresses of a RAM, and let $\mathbf{S} = \{s1, ..sp\}$ be the set of symbols to be stored in the RAM. It can be shown that Model 6.1 with $n = k * p$ can simulate such a RAM. The basic idea is similar to that of 6.2. Let $\mathbf{X} = \mathbf{A} * \mathbf{S}$, $\mathbf{Y} = \mathbf{S}$, and let all $k * p$ associations $(a1, s1) \rightarrow s1$, $(a1, s2) \rightarrow s2, ..(ak, s1) \rightarrow s1, ..(ak, sp) \rightarrow sp$ be already recorded in ILTM and OLTM. Let us present an input string $(a1, z1), ..(ak, zk)$ to the model, where $z1, ..zk \in \mathbf{S}$. From this moment on, the model retrieves symbol zj in response to the address aj, as long as the associations pre–tuned by the input string, have a higher level of E2 than competing associations. Presenting a pair (aj, s') always retrieves s'. It also produces a writing effect. Association $(aj, s') \rightarrow s'$ gets a higher level of E2 than any competing association with address part equal to aj (the recency effect). From this moment on, presenting address aj alone retrieves s' [8].

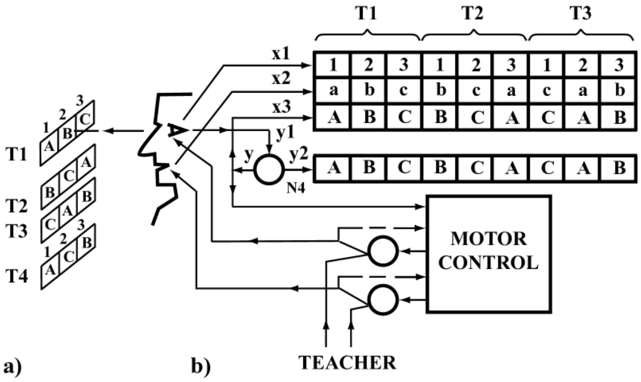
The described effect sheds light on the following problems:

- The "magical number" problem of Miller [25].
- The "more–is–seen–than–can–be–remembered" problem of Sperling [30].
- The ability to retain information in STM increases if similar information is present in LTM.
- One can lose the ability to memorize the new data but still have STM, and vice versa.

7.2 Mental imagery

How can one imagine writing symbols on a sheet of paper, moving pieces on a chess board, etc? Such an ability is essential for mental calculations. Figure 7 offers a simplified but nontrivial explanation of this phenomenon. The idea was used in the model of a universal learning system described in [8].

At the stage of training the teacher forces the robot to scan three tables $T1, T2$ and $T3$, shown in Figure 7a. Each time the robot sees a symbol A, B, or C it utters the name of this symbol a, b, or c, respectively. The motor sequences representing the position of the eye $\{1, 2, 3\}$ and the names of the symbols $\{a, b, c\}$ are sent to inputs $x1$ and $x2$, respectively, and are recorded in ILTM. The visual (sensory) sequence representing the seen symbols $\{A, B, C\}$ is recorded in ILTM (input $x3$) and OLTM. The resulting contents of ILTM and OLTM, the table of Motor,Sensory→ Sensory (MS→S) associations is shown in Figure 7b. It can be proved that Model 6.1 with this experience allows the robot to imagine the process of writing symbols into the table of Figure 7a. For example, to imagine table T4 it is sufficient for the robot to move the closed eye to positions 1, 2, 3 (in any order) and utter the names of the symbols to be imagined in these positions. From this moment on just moving the eye will retrieve the symbols from T4. The output of neurons N4 with the closed eye will be the same as if the eye were open.



T1			T2			T3		
1	2	3	1	2	3	1	2	3
a	b	c	b	c	a	c	a	b
A	B	C	B	C	A	C	A	B

A	B	C	B	C	A	C	A	B

Figure 7: An effect of mental imagery. The robot learns tables T1, T2 and T3. It then can imagine any of the 27 possible tables

7.3 Mental imagery and universality

Block Motor Control in Figure 7 can also be implemented as an associative memory (e.g., Model 6.1) that stores Sensory,Motor→Motor ($SM \to M$) associations. By introducing some additional motor signals it is possible to extend the basic idea of Figure 7 to get a universal learning system, a system that can be trained in an experiment of supervised learning to simulate an arbitrary Turing machine with finite tape [8]. By performing computations with the use of the external memory device the model learns to perform similar mental calculations. This sheds light on a number of interesting psychological phenomena. For example, Baddely [3] mentions that an extremely skilled abacus user can "dispense with the real thing and operate on an imagery abacus instead."

One can suggest, therefore, that the brain has at least two types of associative memories: $SM \to M$ type controlling motor reactions and $MS \to S$ type responsible for the phenomena of sensory memory and mental imagery (simulating the environment as it is "seen" by the motor control system via sensory and motor devices). Therefore, the outlined theory explains how the interaction of these two types of CSAM can provide the universality of the human brain as a learning information processing system.

It is natural to assume that, anatomically, $SM \to M$ and $MS \to S$ systems are located in the frontal and the rear parts of the brain. In fact, the brain also needs a CSAM of the $SE \to E$ type (where "E" stands for "Emotion"). This CSAM, storing the brain's "motivational software", could be located in the frontal part of the brain.

8 CONTEXT–SWITCHABLE FINITE–MEMORY MACHINE

8.1 MODEL. The same as Model 5.1 except: $c2, b3, c2, c3 > 0$; $G32b(i,j) = d * exp(-.5 * ((i - j)/r)^2)$, where $d > 0$ and $r > 0$ are constants (the Gaussian function).

8.2. Model 8.1 combines the effects of Model 5.1 and Model 6.1. Distributed connections $S32b$ provide an additional mechanism of pre–tuning, which results in an effect of mental set more complex than (but qualitatively similar to) that described in section 6.

8.3. The situation becomes too complex for a precise analytical study. Interesting results of computer simulation were described in [8]. Some simplified counterparts of the following psychological effects were produced:

- Switching the motor programs due to a speech pre–tuning. The experimenter utters a name associated with a trajectory and the robot's arm starts moving along this trajectory.
- Producing a trajectory after seeing its sensory pattern (the reversed kinematics problem). The experimenter moves the robot's arm along a cyclic trajectory and the arm continues to move along this trajectory.
- Seeing different sub-pictures in a mixed picture (the Necker cube effect). The experimenter utters the name of a sub-picture and the robot's eye starts scanning this sub-picture.
- Pre–tuning different branches in a syntax tree. The system generates different sentences with a correct grammatical structure. The selection of nonterminals is determined by an additional sensory input associated with these nonterminals.

9 THE CONCEPT OF E– MACHINE

How can one build a complex associative learning system from building blocks similar to Model 3.2?

An attempt to advance in this direction leads one to the concept of E–machine [8]. The building blocks are called primitive E–machines or associative fields. An E–machine built from two primitive E–machines can be taught to simulate an arbitrary Turing machine with a finite tape (the result mentioned in section 7.3). A single primitive E–machine can simulate an arbitrary context–free grammar with a limited memory (a push–down automation with a limited stack). This gives E–machines the powerful capability to ”call subroutines”. A subset of associations formed in one context can be used in many other contexts. An E–machine with a two–level hierarchical structure of associative fields, employing the idea of a sparse quasi–random (or hash) recoding, can dynamically select statistically important associations. This statistical filtering is context–sensitive, as are all the other effects in E–machines.

All the basic mechanisms discussed in this paper are scalable. Using additional layers of intermediate neurons it is possible to implement ILTM and OLTM with the number of locations of the order of 10^9. The mechanism of choice performed be layer N3 (Figure 3) can be expanded by introducing several levels of competition: competition among individual neurons N3, competition among subsets of such neurons, etc.

The problem of learning is particularly important. A universal learning system (such as the human brain) must use a universal learning procedure. The rote learning employed in Model 3.2 is universal, whereas more sophisticated learning algorithms (e.g., backpropagation, simulated annealing, etc.) are not. As it was mentioned in section 3.2, such algorithms are incompatible with the idea of CSAM because they lose information about the order of vectors in the training sequence.

In general, there is a subtle pitfall in an attempt to simplify the decision making process by doing a lot of pre–processing of the ”raw” knowledge before putting this knowledge in memory. A system which has only a generalized representation of its experience cannot, in principal, change its attitude toward such an experience depending on context. Such a system also cannot produce the effects of STM and mental imagery described in section 7.

The mechanism of E–states allows an E–machine to perform a dynamic context–sensitive processing of its knowledge. Accordingly, there is no need to do much pre–processing of this knowledge before putting it in LTM. Therefore, rote learning becomes quite attractive as a ”zero–approximation” learning algorithm. This algorithm can be considerably enhanced by using the above mentioned ideas of hash–recoding and hierarchical learning.

An additional possibility is to use the concepts of novelty and attention. An efficient learner must be able to learn to ”pay attention” to important information. The concept of ”important information”, however, must be learned. It cannot be specified in the description of the learning algorithm, because what is important and what is not depends on context. This is another reason against an attempt to develop an ”overwise” learning procedure. No matter how smart, such a procedure can never be smart enough, because the skill of learning is acquired in the course of learning. Therefore, the data storage procedure should be dumb. However, the decision–making procedure should be considerably smarter than those presently in use in the learning neural models.

System–Theoretical Description

To express the general idea of E–Machine at the system theoretical level it is useful to define the concept of a context–switchable learning machine as an extension of the definition 2.2.

DEFINITION. A *context–switchable learning machine* is a system MCL=(**X,Y,Q,E,G**,FY,FE,FG), where **X,Y,Q,G**,FY,FG are the same as in section 2.2, **E** is the set of states representing mental sets of MCL, called the E–states, $FE : \mathbf{X} * \mathbf{Q} * \mathbf{E} * \mathbf{G} \rightarrow \mathbf{E}$ is the next E–state procedure. MCL is *universal and context–switchable with respect to a class of machines* **C**, if for any $M \in \mathbf{C}$ there exists a pair $(E, G) \in \mathbf{E}*\mathbf{G}$ such that (MCL,E,G) simulates M.

Intuitively, the difference between *universality* and *context–switchability* is that the former implies that the system can be reprogrammed into different machines from **C**, whereas the latter implies that the system with a given knowledge can dynamically reconfigured into different machines from **C** by changing its mental set (E–state).

Symbolic vs. Nonsymbolic

An E–machine is neither a purely symbolic system like, say, a Turing machine, nor a purely nonsymbolic system like, say, an analog computer for solving differential equations. Input vector $x(*)$ of Model 3.2 can be treated as an encoded symbol. This symbol is compared with the contents of the essentially symbolic ILTM. The result of this comparison is the nonsymbolic similarity vector $J0(*)$. This vector participates in nonsymbolic computations involving nontrivial transformations of the E–states $E1(*)$, $E2(*)$ and $E3(*)$. The result of these computations is vector $J4(*)$ describing the levels of activation of different locations of the essentially symbolic OLTM. The vector $y(*)$ retrieved from OLTM can again be treated as an encoded symbol.

This chain *symbolic* → *nonsymbolic* → *symbolic* creates quite nontrivial situation because of the presence of powerful intermediate nonsymbolic E–states. There exists a broad set of "nonclassical" symbolic functions, which can be naturally expressed in terms of E–machines, but are very unnatural for Turing machines (and even von Neumann computers). Accordingly, it is very difficult to find adequate representations of such functions in terms of traditional (classical) symbol manipulation procedures (unless, of course, one knows what one is looking for).

Imagine a cognitive modeler trying to find a phenomenological description of the symbolic behavior of Model 3.2 (or even Model 6.1) observed as a black box. This modeler is unaware of the E–states inside this model, or believes these implementation–level states are not important at the level of cognition. Suppose he has succeeded in deciphering one of the 10–input–1–output Boolean functions simulated by Model 6.1 with given mental set. Such a phenomenological cognitive theory covers only a 2^{-1024}–th part of the whole symbolic behavior of the studied system [9].

The Keyword is Integration

Concluding this paper I want to emphasize that many partial ideas underlying the concept of E–machine were described in different forms elsewhere. Therefore, the main challenge of this concept is to show that a specific formalization, extrapolation and integration of such mainly known partial ideas can produce a broad range of unexpected nontrivial implications (a complex system is more than just the sum of its parts).

References

[1] Anderson, J.A. 1968. A memory model utilizing spatial correlation functions. *Kybernetik, 5*, 113-119.

[2] Anderson, J.R. 1976. Language, memory, and thought. *Hillsdale, N.J.: Lawrence Erlbaum.*

[3] Baddeley, A.D. 1982. Your memory: A user's guide. *MacMillan Publishing Co., Inc.*

[4] Bernstein, N.A. 1966. Assays on the physiology of movements and the physiology of activity. *Moscow: "Medicina".*

[5] Byrne, J.H. 1987. Cellular analysis of associative learning. *Physiological Reviews, 67. No 2*, 329-439.

[6] Cooper, L.N. 1973. A possible organization of animal memory and learning. *Proc. of the Nobel Symposium on Collective Properties of Physical Systems, Aspenasgarden, Sweden.*

[7] Eliashberg, V. 1967. On a class of learning machines. *Moscow: Proceedings of VNIIB, #54*, 350-398.

[8] Eliashberg, V. 1979. The concept of E-machine and the problem of context-dependent behavior. *TXU 40-320, US Copyright Office.*

[9] Eliashberg, V. 1981. The concept of E-machine: On brain hardware and algorithms of thinking. *Proceedings of of the Third Annual Meeting of Cognitive Science Soc.*, 289-291.

[10] Eliashberg, V. 1988a. Neuron layer with reciprocal inhibition as a mechanism of random choice. *Proceedings of the IEEE ICNN 88.*

[11] Eliashberg, V. 1988b. The E-machines: Associative neural networks as nonclassical symbolic processors. *Boston: Abstract. The First Annual Meeting of INNS.*

[12] Gingrich, K.J., & Byrne, J.H. 1985. Simulation of synaptic depression, posttetanic potentiation, and presynaptic facilitation of synaptic potentials from sensory neurons mediating gill–withdrawal reflex in Aplisia. *Journal of Neurophysiology, 53, No.3*, 652-669.

[13] Gingrich, K.J., & Byrne, J.H. 1987. Single–cell neuronal model for associative learning. *Journal of Neurophysiology, 57, No.6*, 1705-1715.

[14] Grossberg, S. 1982. Studies of mind and brain. *Boston: Reidel Press.*

[15] Hodgkin, A.L., & Huxley, A.F. 1952. A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology, 117*, 500-544.

[16] Hopfield, J.J. 1982. Neural networks and physical systems with emergent collective computational abilitis. *Proceedings of the National Academy of Science USA, 79*, 2554-2558.

[17] Johnson, R.C. 1989. 'E–states' mediate cognition. *Electric Engineering Times, January 2*, 67-68, 90.

[18] Kanerva, P. 1988. Sparse distributed memory. *Cambridge MA: MIT Press.*

[19] Keeler, J.D. 1988. Comparison between Kanerva's SDM and Hopfield–type neural networks. *Cognitive Sci., 12*, 299-329.

[20] Kohonen, T. 1988. Self-Organization and Associative Memory. *Second edition, Berlin: Springer-Verlag Press.*

[21] Kosko, B. 1987. Adaptive biderectional associative memories.*Applied Optics, Voi 26, No. 23*, 4947-4960.

[22] Kuffler, S.W., Nicholls, J.G., Martin, A.R. 1984. From neuron to brain. *Second edition. Sinauer Associates Inc. Sunderland, Massachusetts.*

[23] McClelland, J.L., Rumelhart, D.E. 1980.An interactive activation model of the effect of context in perceptron. *Tech. Report No.91. center for Human Information Processing, University of California at San Diego.*

[24] Meynert, T. 1884. Psychiatrie. Wien.

[25] Miller, G.A. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review, 63.* 81-97.

[26] Minsky, M. & Papert, S. 1969. Perceptrons. *Cambridge MA: MIT Press.*

[27] Pinker, S. & Mehler, J. (Eds.) 1988. Connections and symbols. *Cambridge MA: MIT Press.*

[28] Rosenblatt, F. 1962. Principles of neurodynamica. *Washington DC: Spartan Books.*

[29] Sampolinsky, H. & Kanter, I. 1986. Temporal associations in asymmetric neural network. *Physical Review Letters, 57*, 2861-2865.

[30] Sperling, G.A. 1960. The information avalable in brief presentations. *Psychological Monographs, 74, No. 498.*

[31] Vvedensky, N.E. 1901. Excitation, inhibition and narcosis. In "Complete collection of works". *USSR, 1953.*

[32] Widrow, B. 1962. Generalisation and information storage in networks of Adaline neurons. In Self–organizing systems. *Washington DC: Spartan Books.*

[33] Zopf, G.W. 1961. Attitude and Context. In "Principles of Self–organization". *Pergamon Press*, 325-346.