A RELATUONSHIP BETWEEN NEURAL NETWORKS AND PROGRAMMABLE LOGIC ARRAYS *

Victor Eliashberg †

Abstract

A useful relationship between some associative neural networks and programmable logic arrays (PLA) is discussed. The shown analogy helps to understand the properties of this class of neural networks as extensions of the properties of PLA's.

1 Introduction

Different versions of the following basic problem are often discussed in neural network literature:

PROBLEM 1. Given a set of input/output pairs (associations) $P = \{(x^1, y^1), ...(x^n, y^n)\}$, where $x^i \in \mathbf{X}$, $y^i \in \mathbf{Y}$, and $\mathbf{X}$ and $\mathbf{Y}$ are the sets of binary or real input and output vectors, respectively, design a system implementing a function $f : \mathbf{X} \rightarrow \mathbf{Y}$ such that $y^i = f(x^i)$ for $i = 1, ...n$.

Let $\mathbf{X}_p$ be the set of input vectors that are included in the pairs (x^i, y^i) from $\mathbf{P}$. In the special case when $\mathbf{X}_p = \mathbf{X}$ and both $\mathbf{X}$ and $\mathbf{Y}$ are sets of binary vectors, Problem 1 can be solved with the use of a programmable logic device (PLD) known as a programmable logic array (PLA). In the more general case, when $\mathbf{X}$ and $\mathbf{Y}$ are sets of real vectors, Problem 1 can be solved with the use of associative neural networks. Some such networks, having a general architecture similar to that of PLA's, are discussed in this paper.

*International Conf. on Neural Networks, San Francisco, March 28 -Apr.1 1993, vol III.

† visit the web site www.brain0.com for information

2 Solution of Problem 1 with the use of PLA

The basic architecture of a PLA is shown in Figure 1. The diagram uses the following notation:

- $x_i (i = 1, ...m)$ is the i-th component of the input vector, where $x_i \in \{0, 1\}$; $x_i^+ = x_i$ and $x_i^- = \bar{x}_i$.
- g_{ij}^{x+} and g_{ij}^{x-} are, respectively, the conductivities of fuses between lines x_i^+ and x_i^- and the inputs of the i-th AND gate.
- s_i is an auxiliary variable describing a similarity between input vector $x = (x_1, ...x_m)$ and the vector represented by the conductivities of fuses.

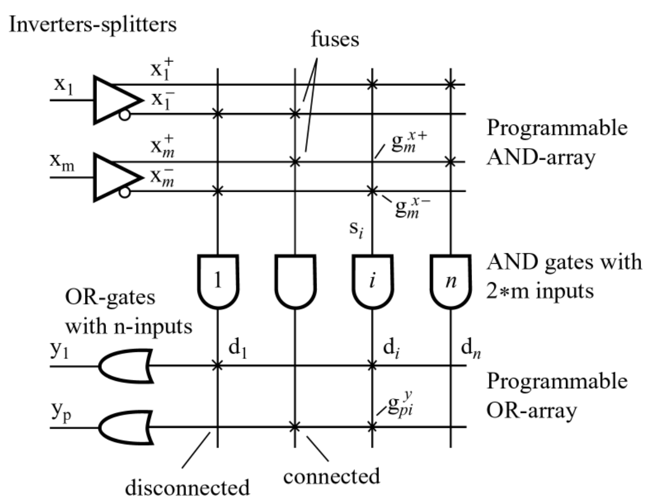


Figure 1: Basic architecture of a PLA

- d_i is the output of the i-th AND gate.

- g^y_{ji} is the conductivity of the fuse between the output of the i-th AND gate and the input of the j-th OR gate.

Using this notation the work of the PLA shown in Figure 1 can be described as follows:

$$s_i = \sum_{j=1}^{m} u_{ij} \qquad (i = 1, ...n) \tag{1}$$

where u_{ij} is found using Table 1.

$$d_i = \begin{cases} 1 & \text{if } s_i = m \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$y_j = \sum_{i=1}^{n} g^y_{ij} \qquad (j = 1, ...p) \tag{3}$$

It is easy to verify that expressions (1)–(3) produce the same result as traditional logic expressions. That is, if input vector x matches the input fuse–vector of the i-th AND gate (vector g^x_i) then d_i=1 and the output fuse–vector of this gate (vector g^y_i) is retrieved as the output vector of the network (vector y).

NOTE. The first on–chip PLA was developed at IBM in 1969 and was referred to as a read–only associative memory (ROAM). The term PLA was introduced in 1970 by Texas Instruments [13].

x_j	g^{x+}_{ij}	g^{x-}_{ij}	u_{ij}
×	0	0	1
×	1	1	0
0	0	1	1
0	1	0	0
1	0	1	0
1	1	0	1

× is ether 0 or 1

Table 1: Table for calculating u_{ij} in Expression 1

3 Neural Network Solution of Problem 1

A simple neural network providing a solution of Problem 1 is shown schematically in Figure 2. 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. The network has three layers of neurons: input neurons N1, intermediate neurons N2, and output neurons N3. Neurons N2 have a global inhibitory feedback via neuron N4 and local excitatory feedbacks. it can be shown that in such a network neurons N2 can compete via reciprocal inhibition in the *winner–take–all* fashion. A similar effect can be obtained in a network with lateral inhibitory feedbacks. Figure 2 uses the following notation:

- $Nk[j]$ is the j-th neuron from set Nk.

- $Smk[i, j]$ is the synapse between neuron $Nk[j]$ and neuron $Nm[i]$.

- x_j is the output of neuron $N1[j]$.

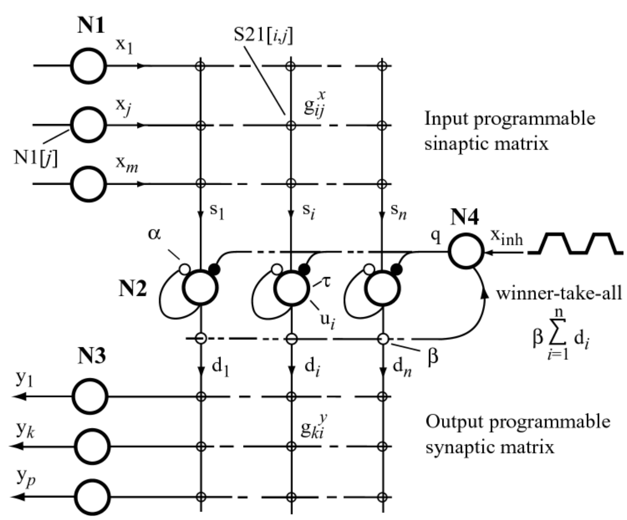


Figure 2: A neural network solution of Problem 1

- g_{ij}^x is the gain of synapse $S21[ij]$.
- s_i is the net synaptic current of synapses $S21[i,1],...S21[i,m]$. s_i represents a similarity between input vector x and vector g_i^x (expression (4)).
- d_i is the output of neuron $N2[i]$.
- q is the output of neuron N4. This output is the sum of the feedback signal $\beta \sum d_i$ and an external signal x_{inh}.
- β is the gain of synapse between any neuron from N2 and neuron N4.
- τ is the time constant of any neuron from N2.
- α is the gain of synapse providing local excitatory feedback for a neuron from N2.
- g_{ki}^y is the gain of synapse between neuron $N2[i]$ and neuron $N3[k]$.

The following functional model of the network of Figure 2 was studied in [2, 3].

$$s_i = \sum_{j=1}^{m} g_{ij}^x \cdot x_j \tag{4}$$

$$\tau \frac{du_i}{dt} + u_i = s_i + \alpha \cdot d_i - q \tag{5}$$

$$d_i = \begin{cases} u_i & \text{if } u_i > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$q = \beta \sum_{i=1}^{n} d_i + x_{inh} \tag{7}$$

$$y_k = \sum_{i=1}^{n} g_{ki}^y \cdot d_i \tag{8}$$

Let all x_j and x_{inh} (and, therefore, all s_i) be step functions of time. Then, for all active neurons from layer N2 (the neurons for which $u_i > 0$, the solution of equations (5)–(7) can be represented in the following explicit form:

$$\begin{aligned} u_i = {} & \frac{(s_i - s_{av})}{\alpha - 1}(e^{at} - 1) + (u_i^0 - u_{av}^0)e^{at} \\ & + \frac{(s_{av} - x_{inh})}{1 + \beta \cdot n_1 - \alpha}(1 - e^{-bt}) + u_{av}^0 \cdot e^{-bt} \end{aligned} \tag{9}$$

where

- n_1 is the number of active neurones from N2.
- u_i^0 $(i = 1, ...n)$ are the values of u_i at t=0.
- s_{av} and u_{av}^0 are the average values of s_i and u_i^0 for all active neurons from N2. That is,

$$s_{av} = \frac{1}{n_1} \sum_{i=1}^{n_1} s_i \tag{10}$$

$$u_{av}^0 = \frac{1}{n_1} \sum_{i=1}^{n_1} u_i^0 \tag{11}$$

Parameters a and b in e^{at} and e^{-bt} are as follows:

$$a = (\alpha - 1)/\tau \tag{12}$$

$$b = (1 + \beta \cdot n_1 - \alpha)/\tau \tag{13}$$

Let $1 < \alpha < 1 + \beta$. Then $a > 0$ and $b > 0$. According to expression (9), neurons $N2[i]$ with $s_i > s_{av}$ increase their potentials u_i. Neuron $N2[i]$ with $s_i < s_{av}$ decrease their potentials and switch off once $u_i < 0$. This reduces n_1 and increases s_{av} making $s_i < s_{av}$ for some additional neurons from N2. Eventually, only neurons with $s_i = max(s_1, ...s_n)$ will have $u_i > 0$. it can be shown that this equilibrium is unstable if $n_i > 1$. Therefore, in the presence of noise, at the end of the transient response there will be only one winner randomly selected from the set of neurons with the maximum level of s_i.

It was shown in [2, 3] that the described neural model provides a solution of Problem 1.

4 PLA–like vs. Connectionist Notation

The drawing shown in Figure 2 uses a graphical notation similar to that employed in the area of programmable logic devices. It is my belief that this

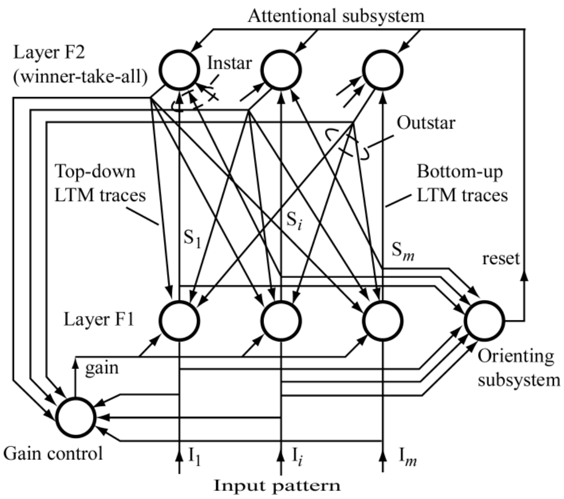


Figure 3: A connectionist representation of the ART1 model

notation is more efficient than the connectionist notation employed in the majority of neural network publications. Figures 3–6 illustrate this point. Figure 3 displays the neural network corresponding to the ART1 [5] using connectionist notation. In this notation each connection is represented by a separate line. One can hardly notice a similarity between this network and the network shown in Figure 2. This similarity is revealed in Figure 4. Another illustration is provided by Figures 5 and 6. Figure 5a shows the architecture of the counterpro-pagation network (CPN) as it was presented in [7]. A different connectionist representation of the CPN architecture borrowed from [4] is shown in Figure 5b. Finally, a PLA–like diagram corresponding to the same network is presented in Figure 6.

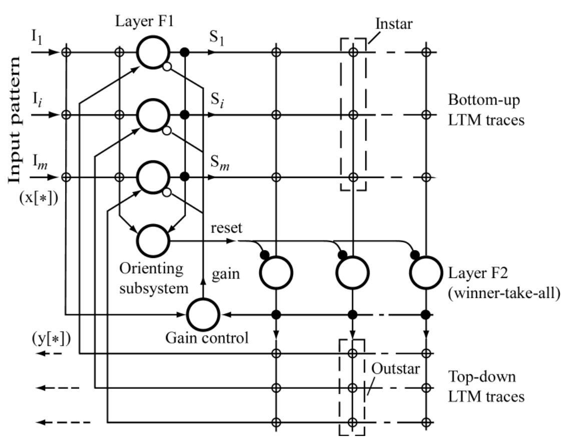
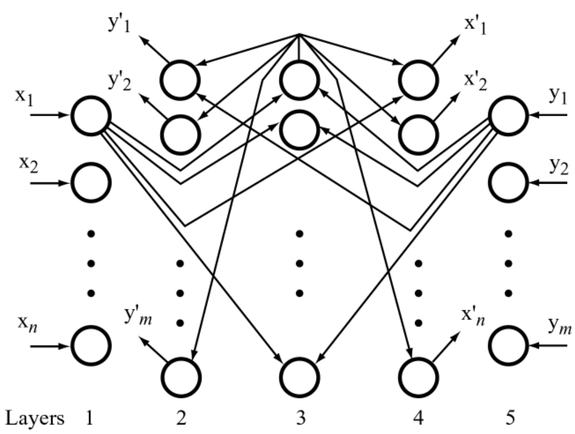
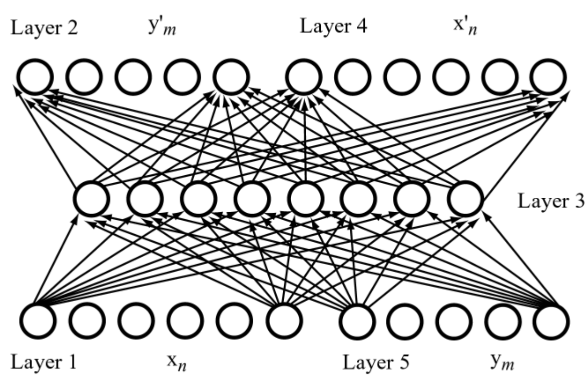


Figure 4: A PLA–like representation of the ART1 model



a)



b)

Figure 5: Two connectionist representation of the CPN model

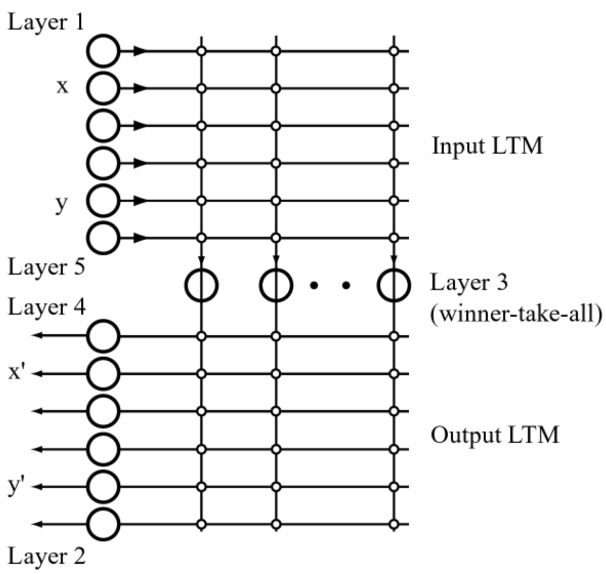


Figure 6: A PLA–like representation of the CPN model

5 History

As is known, the study of artificial neural networks (ANN) began with the classical work of McCulloch and Pitts [9]. Many interesting engineering and mathematical ideas inspired by this pioneering paper were introduced in the late forties, fifties and the sixties [21, 6, 10, 8, 12, 15, 19, 18, 16, 1] and others.

In the late sixties the notion of a "simple" model of the brain hardware (such as the perceptron) became the target of sharp criticism on the part of the proponents of software–oriented AI [11]. As a result of this criticism, the ANN research lost much of its popularity and financing. Nowadays, the seventies are often referred to as the Dark Ages of neural networks.

The eighties and the early nineties witnessed a resurgence of interest in neurocomputing, so the nineties promise to bring many new and exciting results in this challenging area of research.

Unfortunately, the Dark Ages and the new renaissance have erased the memory about many good works that were done in the early days of ANN. Some old ideas have been rediscovered and given new names. Because of this situation it is appropriate to make the following historical references concerning the basic neurocomputing ideas mentioned in this paper.

1. The concept of a neuron as a similarity (match) detector appears in different forms in [15, 20, 18] and some other works of this period.
2. The concept of a programmable synaptic matrix ("Die Lernmatrix") has been promoted by[16].
3. The concept of the neuron layer with lateral (reciprocal) inhibition as the mechanism of the winner–take–all choice can be found in [14]. The dynamics of the layer described by expression (5)–(7) was studied in [2].
4. The basic PLA–like neural architecture shown in Figure 2 (two programmable synaptic matrices combined with the winner–take–all layer) can be found in [2, 3].

ACKNOWLEDGEMENT I express my gratitude to Dr. L. Stark for inspiring discussions.

References

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

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

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

[4] Freeman, J.A, Skapura, D.M. (1991). Neural networks. Algorithms, applications and programming techniques. *Addison–Wesley Publishing Company*

[5] Grossberg, S. (1987). neural networks and natural intelligence. *Cambridge, MA: MIT Press*

[6] Hebb, D.O. (1949). The organization of behavior. *New York: Wiley*

[7] Hecht-Nielsen, R. (1987). Counterpropagation networks. *proceedings of the IEEE First International Conference on Neural Networks*

[8] Kleene, S.(1956). Representation of events in nerve nets and finite automata. *Automata Studies. Prinston: Annals of Mathematics Studies, No.34*.

[9] McCulloch, W.S.,& Pitts, W. (1943). A logic calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics, 5*, 115-133.

[10] Minsky, M. (1954). Neural nets and the brain–model problem. *Unpublished doctoral dissertation, Princeton University*

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

[12] Von Neumann, J. (1958). The computer and the brain. *New Haven: Yale University Press*, pp.66-82.

[13] Pellerin, D., Holley, M. (1991). Practical design using programmable logic. *Englewood Cliffs, New Jersey: Prentice Hall*

[14] Reichardt, W., MacGinitie, G. (1962). Zur Theory der Lateralen Inhibition*Kybernetik, Vol. 1, Nr. 4*.

[15] Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review 65* pp, 386–408.

[16] Steinbuch, K. (1961). "Die Lernmatrix". *Kybernetik, Vol. 1*, 36-45.

[17] Steibuch, K. & Schmit, E. (1967). Adaptive system using learning matrices. *Biocybernetics in Avionics, New York: Gordon and Breach*, pp. 751–768.

[18] Stark, L. et al. (1960). Computer pattern recognition techniques: Electrocardiographic diagnosis. *Communications of the Associations for Computing Machinery, 5*, pp. 527-532.

[19] Widrow, B. and Hoff, M.E. (1960). Adaptive switching circuits. *IRE WESCON Convention Record, New York*, pp. 96-104.

[20] Widrow, B. (1962). Generalisation and information storage in networks of Adaline neurons. *In Self–organizing systems. Washington DC: Spartan Books, Cambridge, MA: MIT Press*

[21] Wiener, N. (1948). Computing machines and the nervous system. *Cybernetics, Cambridge, MA MIT Press*, pp. 116-132.