

Brief Papers

A Biologically-Inspired Improved MAXNET

Orly Yadid-Pecht and Moshe Gur

Abstract—A biologically-inspired modification to MAXNET is proposed. Unlike the original net where the weights are constant, the weights in the new net are dynamically changed. Consequently, the modified net achieves a drastic improvement in convergence rate. A simple hardware implementation for the modified net is presented.

I. INTRODUCTION

SEVERAL neural nets have been proposed to select a maximum value from the input. The MAXNET introduced by Lippman [1] achieves this aim by implementing the “winner takes all” strategy in a way that mimics the heavy use of lateral inhibition evident in biological neural nets, the one found in the Limulus eye is the best known example [2]. The MAXNET iterates by repeatedly decreasing the values of all units, until only one—that which had the maximal initial value, remains active (has a positive value).

Lippman *et al.* [3] describe other techniques of finding the maximum value. One is based on hard limit nodes which perform binary comparisons. The weights and internal thresholds of the output nodes are set such that an output node is positive only when the results of all binary comparisons with the associated inputs indicate that a particular input is greatest. A limitation of this net is that it becomes very large for a large number of inputs (M) because it requires $O(M^2)$ nodes to pick the maximum of M inputs. For example, 5050 nodes are required to pick the maximum of 100 inputs. Another network described by Lippman *et al.* [3] is one that picks the maximum of M inputs by using $M - 1$ comparator subnets arranged in a layered binary tree, i.e., in roughly $\log_2 M$ layers. In such a case it requires, for example, only 594 nodes to pick the maximum of 100 inputs.

The above two nets use strictly feedforward connections and are relatively large. MAXNET is a less complex net that uses feedback connections. It uses only M threshold logic nodes, the same as the inputs number (as explained in the following section) and iterates to find the maximum. The net is motivated by the large number of connections in biological neural nets and by the laterally interconnected networks described by Kohonen [4].

Manuscript received February 4, 1993; revised July 16, 1993 and March 15, 1994.

The authors are with the Department of Biomedical Engineering, The Julius Silver Institute of Biomedical Sciences, Technion—Israel Institute of Technology, Haifa 32000, Israel.
IEEE Log Number 9404859.

If the number of nodes required is a key issue in deciding which net to choose, the MAXNET should be preferred. The downside is the delay added by the need to iterate the net.

As documented in Lippman *et al.* [3], if the MAXNET is to classify classes in a noisy environment (noise $>30\%$), the average number of iterations rises to a value that is roughly 10% above M . The number of iterations needed is linear with the number of elements in the net. In this paper we show that the delay limitation of the MAXNET can be overcome by dynamically changing the net's weights and thus achieving a drastic decrease in the number of iterations needed for convergence. Recently, the same convergence results were achieved by Suter and Kabrisky [5] using another modified MAXNET. Suter and Kabrisky's approach, however, is different from ours, and their suggested implementation is more demanding in hardware and time. Winters and Rose [6] have also suggested an improved net that converges in $\lg(M)$ iterations, but requires special switching node elements. Our modification uses the standard hardware elements.

Section II describes the MAXNET, Section III describes the modified MAXNET, Section IV gives a proof of convergence, Section V describes a hardware implementation, and in Section VI our results are discussed.

II. THE MAXNET

MAXNET, as described in [3], is similar in its structure to the Hopfield net [7] but uses threshold logic instead of hard-limit nodes and feeds the output of each node back to its input. The MAXNET is a fully connected net made up of M nodes with internal thresholds set to zero. A schematic description of the net is given in Fig. 1. Input values are applied at time zero through the input nodes. This initializes the node's outputs, for each node at time zero ($\mu_j(0)$), to the input values

$$\mu_j(0) = y_j, \quad j = 0, 1, \dots, M-2, M-1. \quad (1)$$

The network then iterates to find the maximum input value via the following equation

$$\mu_j(t+1) = f \left[\mu_j(t) - \sum_{i \neq j} w_{ij} \mu_i(t) \right] \quad (2)$$

where f is the threshold logic function described in (3) and w_{ij} is the inhibitory weight between nodes

$$f(\alpha) = \begin{cases} \alpha & \text{if } \alpha \geq 0 \\ 0 & \text{if } \alpha \leq 0. \end{cases} \quad (3)$$

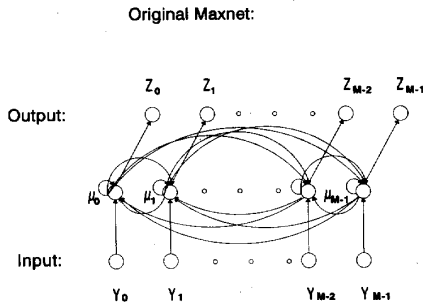


Fig. 1. Schematic description of MAXNET—an iterative neural net that determines which of M inputs is the maximum. The inputs are applied prior to time zero and then removed, and the outputs are valid after the network converges. All nodes are analog-threshold logic nodes with internal thresholds equal to zero. Each node connects to itself and to all other nodes (after [3]).

Each node inhibits all other nodes by an amount equal to the node's output multiplied by a small negative weight. Each node also feeds back on itself with a unity gain.

After convergence, only that output node corresponding to the maximum input will have a nonzero value. This value will, in general, be less than the original time-zero value of that node. The output values of the net are thus simply the nodes output values after convergence

$$z_j = \mu_j(\infty), \quad j = 0, 1, \dots, M-2, M-1. \quad (4)$$

The MAXNET converges and finds the maximum input when

$$w_{ij} = w < \frac{1}{M-1}. \quad (5)$$

III. THE MODIFIED MAXNET

In our modified MAXNET every unit is connected to every other, with a dynamic weight $w_{ij}(t)$. The connections between the output of each unit and its input are of value one and are kept constant through the iterations. Every iteration, each active unit i (whose output is $\mu_i > 0$) is updated by

$$\mu_i(t+1) = f\left[\mu_i(t) - \sum_{j=1}^M w_{ij}(t)\mu_j(t)\right]. \quad (6)$$

After updating all units, $M(t+1)$, the number of currently active units is calculated, and the weights of all connections between different units are changed by

$$w_{ij}(t+1) = \frac{1}{M(t+1)}. \quad (7)$$

The weights are initialized according to $M(0) = M$, where M is the number of all units in the network.

The above algorithm differs from the original one suggested by Lippman *et al.* [3] in the dynamic change of the weights. This change allows a drastic decrease in the number of iterations needed for the network to converge. This difference becomes very significant when large nets are used. The original MAXNET requires an enormous number of iterations which are directly proportional to the number of units, while in the

modified algorithm the number of iterations is proportional to the logarithm of the number of units.

IV. PROOF OF CONVERGENCE

Theorem: The modified MAXNET, when initialized so that no two or more units have the same maximal value, converges until only one unit is active (has a nonzero output), and that unit is the one which had the maximal initial value.

Convergence proof for the improved MAXNET is simple. Let us denote the input nodes by their values, i.e., the largest one is designated I_1 , the second largest I_2 , and the smallest one I_M . Thus, I_1 will always stay positive after the iteration, and I_M will always be zero at the end of each iteration. This is true since if $I_1 > I_M$ then $I_1 > \langle \text{avg } I_1, I_M \rangle > I_M$. By induction, at each iteration, at least one output will be zeroed. Therefore, after M iterations, only the largest output will remain active.

This is true if the inputs do not contain two or more units with the same maximal value. In practice, a net having more than one maximal unit still converges and finds a maximum if a small amount of randomness is introduced into the net.

For commonly found statistical distributions of input values, the rate of convergence depends on the logarithm of the number of inputs. This is evident from the fact that at each iteration about half of the active units become inactive, since approximately half will have values below the average. Suter and Kabrisky [5] give a rigorous proof for a uniform distribution of inputs and an experimental proof for a normalized distribution case.

V. HARDWARE IMPLEMENTATION

The same neurons found in MAXNET are used, while adding a sign function, which could be implemented as a limiter, at each node. The sign output is one if input ≥ 0 and zero otherwise.

To calculate the new number of "positive" neurons $M(t)$, positive outputs are summed. This can be done by an extra neuron which can be implemented by an M -bit input summer (Fig. 2). This node's output is then forwarded to a divider to calculate the new weights, $\frac{1}{M(t)}$. Since all weights are the same, all nodes in the net can be summers, and the output value of each node multiplies the sum by $\frac{1}{M(t)}$ (Fig. 3). This is a simplification of the regular neuron design, where each input has to be multiplied by a different weight. This can be expressed as

$$\sum w_{ij}(t)\mu_i(t) = \sum \frac{1}{M(t)}\mu_i(t) = \frac{1}{M(t)} \sum \mu_i(t). \quad (8)$$

Consequently, the hardware implementation is very simple. Each node receives $1/M(t)$ from the extra node and then multiplies it by the node's sum of values. This way, the whole operation is accelerated, and each node computation is limited to one multiplication instead of M ones.

This type of net can be used to process time-varying input signals. The worst-case delay required for producing an output at each stage is M times the delay of a simple M input bit adder, a divider, and the nodes computation.

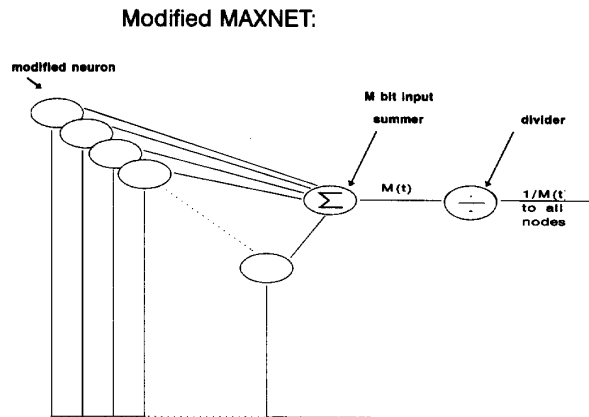


Fig. 2. General description of the modified MAXNET. Modified neurons, to which a sign output is added, are used. An extra neuron (M -bit input summer) sums the positive outputs. New weights are calculated using a divider and the extra neuron output. These weights are forwarded to all neurons in the modified net.

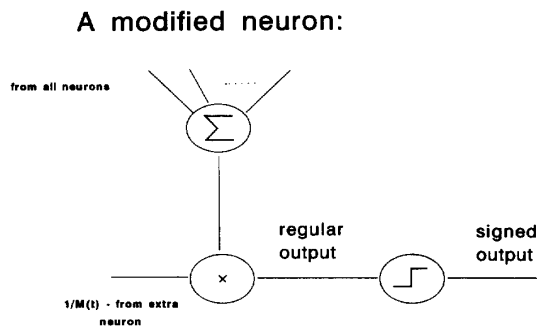


Fig. 3. An implementation of the modified neuron. Each node is a summer, whose output is multiplied by the current weight value to achieve the regular output. The sign function is then implemented.

Recent work by Suter and Kabrisky [5] also describes a modified MAXNET, yielding similar convergence results. They use threshold logic units which give an output of one if the input passes some threshold value and zero otherwise. In their design, each input is passed to two sets of threshold logic units. The first set calculates the positive outputs, by summing the threshold units outputs, and the second set calculates the sum of inputs. Magnitude is preserved by multiplying the output of the threshold unit by the threshold value. Afterwards, the sum of inputs is divided by the number of positive outputs to get the average signal.

Note, however, that to calculate the average signal, a full M input adder is required. This implementation is more demanding in hardware and time than our implementation. Note also that, as the input passes a threshold unit and a multiplier, errors might change the original magnitude, such that the preservation of the magnitude might not be accurate. Our design is simpler and consequently, faster. In our implementation, the same expanded node performs

both functions, i.e., indicating the magnitude and the above mentioned average "positiveness." Before the sign function, there is the regular output, and after the sign function, output is passed to a single extra M -bit summer for producing the net weights. As a byproduct, the maximum nodes magnitude (regular output) and position (sign output) are obtained. To expand our net to a K winner-takes-all circuit, i.e., to find the K units with the highest values, an addition of a comparator is sufficient.

VI. DISCUSSION

It is known that nerve cells change their attributes dynamically. Visual cells, for example, may change their orientation preference or their receptive field size [8]. In conventional neural nets, however, the weight changes are limited to the "learning stage," and no further changes are made after this stage is complete. What is shown here is that it is advantageous to change the weights after the initial learning stage. Weights need not remain constant while the net is working—they can change so the net can continue to learn, similar to living creatures who learn all the time. This approach is reinforced by engineering and theoretical considerations which show that such changes drastically improve performance.

The modified algorithm can be easily implemented in hardware. Complexity of $O(n^2)$ in the number of connections still remains while a straightforward determination of the number of active units ($M(t+1)$) requires one extra M -bit summer. The implementation also requires a sign function for each node, but, on the other hand, each node can be a summer, and only one multiplication is required. Consequently, average time for convergence is reduced.

ACKNOWLEDGMENT

The authors wish to thank Z. Kirshenbaum and R. Rotenberg for the software development and M. Azaria for helpful comments.

REFERENCES

- [1] R. P. Lippman, "An introduction to computing with neural nets," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 35, pp. 2-44, Apr. 1987.
- [2] H. K. Hartline and F. Ratliff, "Inhibitory interaction of receptor units in the eye of the limulus," *J. General Physiology*, vol. 40, pp. 357-376, 1957.
- [3] R. P. Lippman, B. Bold, and M. L. Malpass, "A comparison of Hamming and Hopfield neural nets for pattern classification," Techn. Rep. 769, Lincoln Laboratory, MIT, May 1987.
- [4] T. Kohonen, *Self-Organization and Associative Memory*. Berlin: Springer-Verlag, 1984.
- [5] B. W. Suter and M. Kabrisky, "On a magnitude preserving iterative Maxnet algorithm," *Neural Computation*, vol. 4, pp. 224-233, 1992.
- [6] J. H. Winters and C. Rose, "Minimum distance automata in parallel networks for optimum classification," *Neural Networks*, vol. 2, pp. 127-132, 1989.
- [7] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. National Academy of Sciences*, vol. 79, pp. 2554-2558, 1982.
- [8] C. D. Gilbert and T. N. Wiesel, "The influence of contextual stimuli on the orientation selectivity of cells in primary visual cortex of the cat," *Vision Research*, vol. 30, pp. 1689-1701, 1990.