Optoelectronic Fuzzy ARTMAP Processor

Matthias Blume and Sadik C. Esener
matthias@ucsd.edu sadik@ece.ucsd.edu
University of California, San Diego
Electrical and Computer Engineering Department
Mail Code 0407
La Jolla, CA 92093

1. Introduction

The realization of practical optical or optoelectronic computers has been hampered by the lack of algorithms suited to optoelectronic implementation. We have chosen an algorithm that is particularly compatible with optoelectronic processors and parallel access optical memory, mapped it onto an architecture which satisfies the constraints of the hardware, and suggest an implementation which is an appropriate combination of optical and electronic technology. The proposed parallel optoelectronic implementation increases throughput by several orders of magnitude over serial implementations, facilitating the real-time solution of large problems.

2. Implementation issues

Even many of the (inherently parallel) neural learning algorithms that have proven useful in practice are difficult to implement in fully parallel hardware. For example, backpropagation requires the multiplication of an input by a weight with numerical precision that is typically at least 13 bits[1] *at each synapse*. This precision requirement is beyond the range achievable with analog information processing, and the corresponding digital circuits are prohibitively large. Also, backpropagation requires weight transport or multiple copies of the synaptic weights[2]. Finally, since update information is stored for every weight after every input presentation, a parallel interface to secondary storage requires a transmitter per synapse.

Fuzzy ARTMAP[3] has received a great deal of attention along with the other ART algorithms, but few implementations have been proposed. We find that it is a practical algorithm for supervised learning that has several important advantages for optoelectronic implementation. In particular, only the weights corresponding to one processing element (PE) are updated after each training sample. This makes it possible to segment a large problem into smaller parts during learning, loading the page of weights corresponding to each subproblem onto the processor from a parallel access optical memory, but downloading the changed weights of only one PE via a low bandwidth electronic link. The resulting system is much more versatile than a system capable of dealing only with problems of a particular size. Furthermore, the algorithm can perform well even with weights truncated to 4 bits during training[4], and requires no multiplications. Finally, it converges rapidly and relatively uniformly with little dependence on the particular choice of adjustable parameter values and initial state.

3. Background: fuzzy ARTMAP algorithm

Fuzzy ARTMAP is essentially a clustering algorithm (vector quantizer), with supervision that redirects training inputs which would be grouped in an incorrect category to a different cluster. As illustrated in figure 1, a fuzzy ARTMAP system consists of two fuzzy ART modules, each of which clusters vectors in an unsupervised fashion, linked by a map field. (Throughout this paper, vectors are denoted by bold letters.) Typically, the cluster first chosen by module *a* is associated with the module *b* cluster containing the desired output vector. However, during training, if the cluster to which input vector $\boldsymbol{I}^a$ is assigned is incorrect, the map field signals module *a* and causes $\boldsymbol{I}^a$ to be assigned to the cluster next most likely to be correct. The process is repeated until $\boldsymbol{I}^a$ is assigned to a correct cluster. New clusters are created as needed.

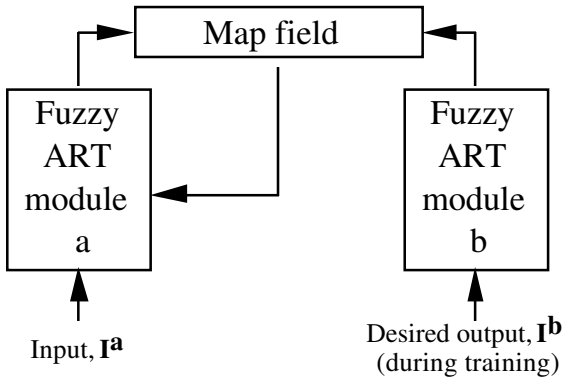

Figure 1: A fuzzy ARTMAP processor consists of two fuzzy ART modules linked by a map field.

3.1 Fuzzy ART algorithm

Fuzzy ART clusters vectors based on two separate distance criteria, *match* and *choice*. The match function is defined by

$$S_j(\mathbf{I}) \equiv \frac{|\mathbf{I} \wedge \mathbf{w}_j|}{|\mathbf{I}|},$$

where $\boldsymbol{w}_j$ is an analog-valued weight vector associated with cluster *j*, $\wedge$ denotes the fuzzy AND operator, $(p \wedge q)_i \equiv min(p_i, q_i)$, and the norm $|\cdots|$ is defined by $|\mathbf{p}| \equiv \sum_i |p_i|$. The choice function is defined by

$$T_j(\mathbf{I}) \equiv \frac{|\mathbf{I} \wedge \mathbf{w}_j|}{\alpha + |\mathbf{w}_j|},$$

where α is a small constant. Increasing α biases the search more towards clusters with large $\boldsymbol{w}_j$.

Input vector $\boldsymbol{I}$ is assigned to the category which maximizes $T_j(\boldsymbol{I})$ while satisfying $S_j(\boldsymbol{I}) \geq \rho$, where the *vigilance*, ρ, is a constant, $0 \leq \rho \leq 1$.

The fuzzy ART learning rule is given by

$$w_{Ji}^{new} = \begin{cases} w_{Ji}^{old} & w_{Ji} \leq I_i \\ w_{Ji}^{old} - \beta\left(w_{Ji}^{old} - I_i\right) & w_{Ji} > I_i \end{cases},$$

where $0 < \beta \le 1$. Only the weights of the cluster to which $\boldsymbol{I}$ has been assigned are updated. All w_{ji} are initially set to 1.

Carpenter et al.[3] propose searching for the category J which maximizes T_j and then checking whether the chosen category satisfies $S_j(\boldsymbol{I}) \ge \rho$. If not, category J is marked as ineligible and the search is repeated until a satisfactory category is found. The length of time between input presentation and selection of the corresponding cluster is variable, depending on how many search cycles are required. Furthermore, the associated three-layer architecture is not well suited to parallel implementation because it requires weight transport or multiple, independently updated copies of the weights. Section 4.1 describes how all of these undesirable properties may be eliminated.

3.2 The map field

The map field is essentially a look-up table, retrieving an analog-valued weight w_{JL}^{ab} when module a node J and module b node L are active. Note that only one node of each module is active at a given time. If $w_{JL}^{ab} < \rho^{ab}$ the vigilance of module a, ρ^a, is raised until node J becomes inactive (and some other node becomes active). This process is repeated until $w_{JL}^{ab} \ge \rho^{ab}$. When the next input is presented, ρ^a is returned to its baseline value. All w_{JL}^{ab} are initially set to 1. During learning, when nodes J and L become active and $w_{JL}^{ab} \ge \rho^{ab}$, all w_{Jl}^{ab}, $l \ne L$, are reduced in value (typically set to 0).

4. Optoelectronic implementation

4.1 A novel mapping of fuzzy ART onto a suitable architecture

Fuzzy ARTMAP specifies a precoding scheme, referred to as *complement coding*. Given M-dimensional feature vectors $\boldsymbol{a}$, $2M$-dimensional input vectors $\boldsymbol{I} = (\boldsymbol{a}, \boldsymbol{a}^c)$ are generated, where $a_i^c \equiv (1-a_i)$. The norm of every input vector, $|\boldsymbol{I}|$, then equals M, the dimension of $\boldsymbol{a}$. The match function becomes $|\boldsymbol{I} \wedge \boldsymbol{w}_j| / M$, and the match criterion becomes $|\boldsymbol{I} \wedge \boldsymbol{w}_j| \ge \rho M$. Fuzzy ART with complement coded inputs may be mapped onto a neural network consisting of only two layers, as shown in figure 2. $|\boldsymbol{I} \wedge \boldsymbol{w}_j|$ is determined at each synapse and the norm (summation of the synaptic outputs) is performed during fan-in along the dendritic tree. If the match criterion is not met, the output of that node is disabled. Thus the match criterion is computed for all nodes in parallel, and the search procedure is carried out only once per input vector, eliminating the variable delay described above. Weight updates are carried out at each synapse using only locally available information, and no weight transport is required.

4.2 Implementation using D-STOP

The most computationally intensive step in the fuzzy ARTMAP algorithm is the computation of the $T_j(\boldsymbol{I})$ values in each fuzzy ART module. Whereas previous optical ART processors[5] have been limited to the multiplications required by the earlier ART algorithms, the Dual-Scale Topology Optoelectronic Processor[6] (D-STOP) is ideally suited to implement this operation. D-STOP utilizes optical interconnections and electronic computations, enabling it to perform the necessary *generalized* matrix-vector multiplication. The optical system is space invariant, consisting of a 4-f imaging system that forms a reduced image of the input array, followed by a single lens and a computer-generated hologram (CGH). The single lens images the intermediate plane onto the output plane, and the CGH replicates the de-magnified image. Each D-STOP PE has n_s optical inputs, where n_s is the number of synapses, and only one output. One copy of the input array falls onto the detectors of each PE in the output plane. For simplicity, the output signals in the implementation described below are electrical. However, D-STOP is fully compatible with optical outputs as well.

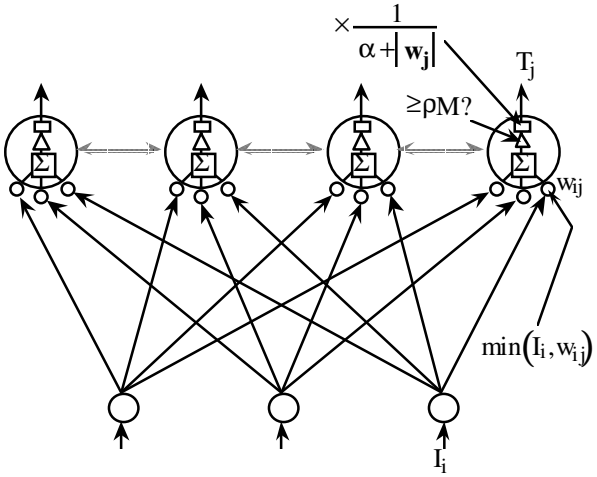

Figure 2: Neural architecture for fuzzy ART. In practice, the inhibitory interconnections (shaded) within layer 2 are replaced by one additional PE that determines which layer 2 PE has the maximal output value.

A complete system for implementing fuzzy ARTMAP, consisting of one D-STOP per fuzzy ART module, is illustrated in figure 3. The module a processor plane is also interfaced to a parallel access optical memory. The pixelated output of the memory is imaged onto the detector array of the processor plane. The imaging system must be tailored to the particular type of parallel access memory which is to be used. The map field may be implemented using standard random access memory (RAM) chips and minimal additional logic. The total size of the RAM (typically a few kilobytes) is $\sigma N_a N_b$, where σ is the number of bits of precision per weight and N_a and N_b are the number of PEs in modules a and b, respectively.

Figure 4 is a schematic diagram of one processor plane. If N_a, the number of PEs required for module a, is greater than the number of PEs present in hardware, the input vector may be presented once and stored. Pages of weights corresponding to sub-arrays of PEs are subsequently loaded onto the processor plane. Once the maximal T_j of one set of PEs has been determined, that page of weights is no longer needed and is overwritten by the next page. The single page containing the weights, $\boldsymbol{w}_J$, corresponding to the cluster to which $\boldsymbol{I}$ is finally assigned must be loaded again before presentation of the next input vector in order update $\boldsymbol{w}_J$.

Fan-in along the H-tree of each processing element requires $O(\log n_s)$ time, where n_s is the number of

synapses in the PE. Determination of the maximal T_j along the larger H-tree of the processor array requires $O(log\ N)$ time. The throughput is increased by $O(n_S / logn_S)$ over a serial implementation, since the time required by the latter is $O(n_S)$, where the total number of synapses is $n_S = n_s N$.

Other operations, such as distribution of global clock and control signals or fan-in of the T_j values might also benefit from optical interconnections. However, we have concentrated on that aspect of the implementation which results in the greatest increase in performance and the greatest reduction of circuit area, yielding a simple, conservative, realizable scheme which relies only on hardware which has been demonstrated in the lab[7,8].

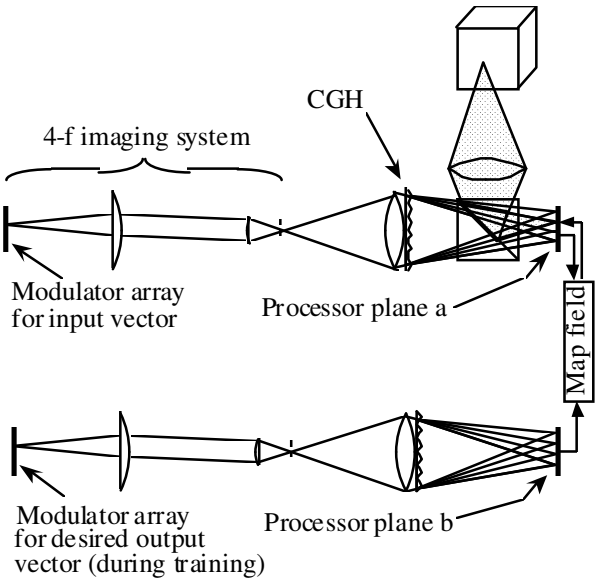

Figure 3: Optoelectronic fuzzy ARTMAP processor. Optical connections are represented by light cones. All inputs may be active simultaneously, but the connections of only one input per module are shown. Module *a* is shown interfaced to a parallel access optical memory, drawn schematically as a box. (The actual medium used might, for example, be an optical disk) The map field consists of electronic logic and RAM, and connections to the map field are electrical lines.

5. Conclusion

To our knowledge, this is the first design for a parallel implementation of the fuzzy ARTMAP algorithm. The proposed mapping of the algorithm onto a neural architecture is efficient, requiring only an input layer and one processing layer per fuzzy ART module, and requiring neither weight transport nor multiple copies of weights. The proposed optoelectronic system is simple, yet versatile, and relies on proven components. Operations which may be carried out using standard electronic components without loss of performance are carried out electronically. Computing the generalized matrix-vector multiplication in parallel results in an $O(n_S / logn_S)$ speed-up over a serial computation, where n_S is the number of synapses implemented in hardware.

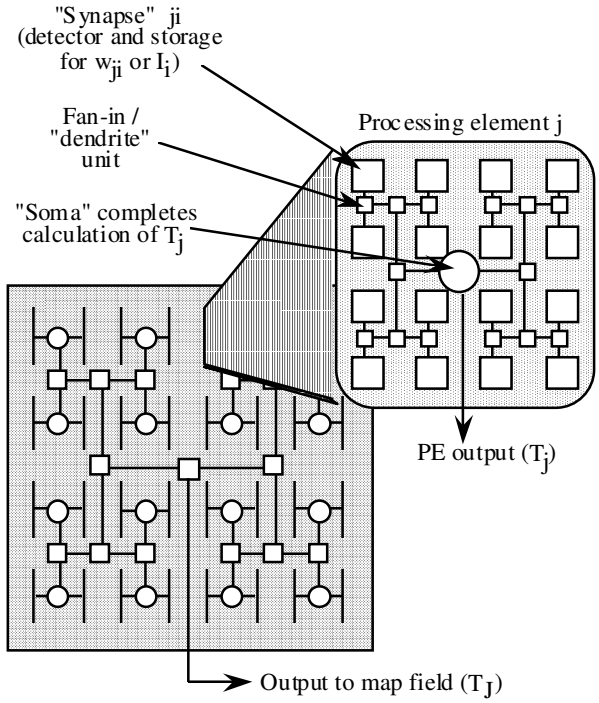

Figure 4: Fuzzy ART processor plane layout. Calculation of $|\boldsymbol{I} \wedge \boldsymbol{w}_j|$ (summation of the synaptic outputs) is performed during fan-in along the H-tree of PE *j*. Which T_j is largest is determined during fan-in along the H-tree of the PE array. When interfaced with a parallel access memory, the same detectors are used to receive the input values I_i and the weight values w_{ji} in subsequent time steps. Since the weights of only one PE are modified for every input vector, the changed weights may be off-loaded via a low bandwidth electronic link.

[1]P. W. Hollis, J. S. Harper, and J. J. Paulos, "The effects of precision constraints in a backpropagation learning network", *Neural Computation,* **2**:3, pp. 363-373, Fall 1990.

[2]G. Marsden, A. Krishnamoorthy, J. Mercklé, and S. Esener, "Tandem D-STOP architecture for backpropagation networks", *Proc. OSA Topical Meeting on Optical Computing,* Paper OWE4, March 1993.

[3]G. A. Carpenter; et al., "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps", *IEEE Transactions on Neural Networks*, **3**:5, p. 698-713, Sept. 1992.

[4]M. A. Rubin, "Issues in automatic target recognition from radar range profiles using fuzzy ARTMAP", Naval Air Warfare Center preprint, 1994.

[5]D. C. Wunsch II, D. J. Morris, R. L. McGann, and T. P. Caudell, "Photorefractive adaptive resonance neural network", *Applied Optics*, **32**:8, p. 1399-1406, 10 March 1993.

[6]G. C. Marsden, et al., "Dual-scale topology optoelectronic processor", *Optics Letters*, **16**:24, p. 1970-2, 15 Dec. 1991.

[7]G. Yayla, et al., "Prototype 3-D optoelectronic neural system", *SPIE Proceedings,* vol.1773, p. 4-13, 1993.

[8]P. Marchand, et al., "Motionless-head parallel readout optical-disk system", *Applied Optics*, **32**:2, p. 190-203, 10 Jan. 1993.