Pergamon

Neural Networks, Vol. 8, No. 4, pp. 605–618, 1995

0893-6080/95 $9.50 + .00

0893-6080(94)00106-5

CONTRIBUTED ARTICLE

Adaptive Hamming Net: A Fast-Learning ART 1 Model Without Searching

CHENG-AN HUNG AND SHENG-FUU LIN

National Chiao Tung University

(*Received* 2 *March* 1994; *revised and accepted* 26 *October* 1994)

Abstract—*This paper introduces a neural network architecture called an* adaptive Hamming net *for learning of recognition categories. This model allows new prototypes to be added to an existing set of memorized prototypes without retraining the entire network. Under some model hypotheses, the functional behavior of the adaptive Hamming net is equivalent to that of a fast-learning ART* 1 *network, so some useful properties of ART* 1 *can be applied to the adaptive Hamming net. In addition, the proposed network finds the appropriate category more efficiently than ART* 1: *for the same input sequences, the adaptive Hamming net obtains the same recognition categories as ART* 1 *without any searching. The adaptive Hamming net not only reduces the training time of ART* 1 *but is also easier to implement. The adaptive Hamming net is limited to binary pattern clustering, but it can be extended to the case of analog input vectors by incorporating fuzzy logic techniques.*

Keywords—Neural network architecture, Adaptive Hamming net, Fast-learning ART 1, Fuzzy logic.

1. INTRODUCTION

Neural networks with parallel processing capability and robust performance provide a new approach to adaptive pattern recognition. However, most neural networks suffer from the following disadvantages:

1. the training time is extremely long when solving large-scale problems (this is true in particular for back-propagation algorithms)
2. it is necessary to retrain the entire network when a new pattern is added.

To overcome these difficulties, Carpenter and Grossberg (1991) developed a series of adaptive resonance architectures (ART). The most popular ART models include ART 1 (Carpenter & Grossberg, 1987a), ART 2 (Carpenter & Grossberg, 1987b), fuzzy ART (Carpenter, Grossberg, & Rosen, 1991b), ARTMAP (Carpenter, Grossberg, & Reynolds, 1991), and fuzzy ARTMAP (Carpenter et al., 1992). In this paper, we aim to improve the computational efficiency of ART 1 and apply the results to fuzzy ART. The neurological implications of ART will not be dwelled upon.

ART 1 is a neural network architecture that self-organizes stable recognition categories in real time in response to arbitrary sequences of binary input patterns (Carpenter & Grossberg, 1987a). A category is represented by a node in a specific layer of the ART 1 network, which corresponds to a prototype pattern for the category. It is this prototype that is matched to the input. In operation, the network tentatively selects a winning node in a category layer to represent an input pattern appearing across its input layer. To ensure learning stability, the orienting subsystem (Figure 1) in ART 1 essentially checks that a category prototype pattern corresponding to the winning node in the category layer matches the input pattern as closely as desired. When a pattern mismatch occurs, the winning node is reset and the node that has the next priority in the order of search is selected. The search process continues until the system finds a node to learn the input pattern. Figure 2 illustrates a typical ART search–reset cycle. Although Carpenter and Grossberg (1987a) designed a self-adjusting memory search for the ART network, it is possible that a series of mismatch resets in response to a single input can affect the computational speed and efficiency of the network. To reduce the training time of ART 1, in this paper we propose a neural network architecture called an *adaptive Hamming net,* which can avoid an unnecessary search.

Because many applications of ART 1 use fast-learning conditions, only this case is described in detail here. Fast learning means that the weights in ART 1, which are governed by first-order nonlinear ordinary differ-

Requests for reprints should be sent to Sheng-Fuu Lin, Department of Control Engineering, National Chiao Tung University, 1001 Ta Hsueh Rd, Hsinchu, Taiwan 300, R.O.C.

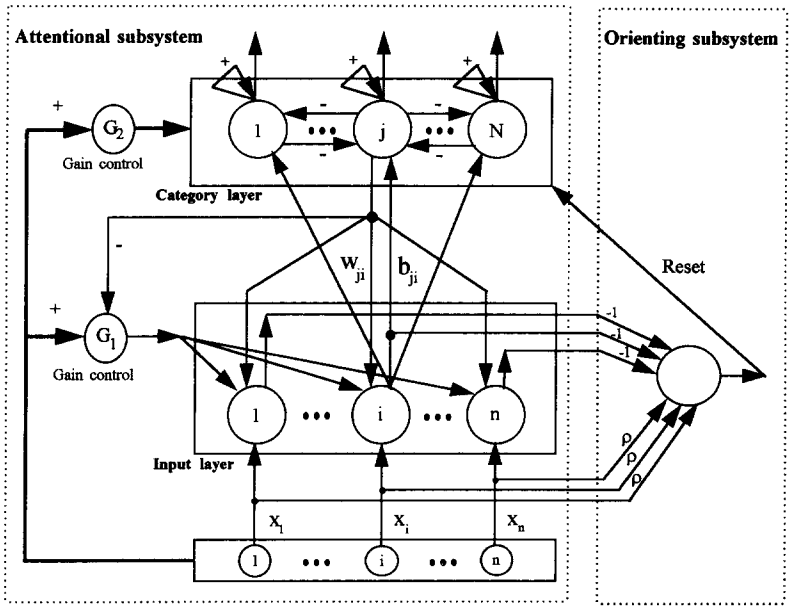

FIGURE 1. The architecture of ART 1. The two major subsystems are the attentional subsystem and the orienting subsystem.

ential equations, are allowed to reach their asymptotic equilibrium values with each pattern presentation. The adaptive Hamming net accurately reproduces the behavior of ART 1 in the fast-learning case and thus may be an efficient method for selecting appropriate recognition codes and reducing the design complexity of ART 1. It processes familiar and unfamiliar events without using a time-consuming search cycle. This feature of the net eliminates consecutive resetting and choosing, and makes the net easier to implement than ART 1.

The key idea behind this approach is to convert the search procedure of ART 1 into an optimization problem (see Section 3). To solve this optimization problem, a set of categories is initially constructed that satisfies the vigilance constraints, and then the best matching solution is found by maximizing a measure of similarity. Such a search process is different from the one used in ART 1. It eliminates all infeasible solutions from existing categories, thereby reducing training time. To realize the above idea a locally defined neural network architecture is utilized. The use of a neural network architecture allows us to exploit the parallelism inherent in the proposed method.

The configuration of an adaptive Hamming net is similar to that of the traditional Hamming net (Lippmann, 1987). The former can be treated as an adaptive version of the latter. Before describing the adaptive Hamming net, we briefly introduce the Hamming net. The Hamming net for binary patterns computes the Hamming distance to each prototype and selects the prototype with the minimum Hamming distance. It comprises a feedforward excitatory network and lateral inhibitory network. The main feature of the Hamming net is that it uses a *MAXNET* to pick the node that has the maximum output value. This action is equivalent to MAXNET picking the prototype that has the least Hamming distance from the input pattern. Another feature of the Hamming net is that its synaptic weights are prestored and are not capable of learning.

Unlike the fixed-weight Hamming net described above, the adaptive Hamming net can learn to adapt based on experience collected from previous training patterns. Furthermore, it is also able to learn new input patterns without forgetting old learned patterns. This property makes the neural model ideal for use as a pattern recognition machine in real-time environments.

Another feature of the adaptive Hamming net is that it picks and chooses adequate prototypes through a similarity checking process. The proposed model uses a nonlinear activation function to select available categories for which prototype patterns are similar to the input pattern and to suppress categories for which prototype patterns are sufficiently different from the input pattern. The nonlinear activation function is similar to the piecewise linear function used in ART 2 (Carpenter & Grossberg, 1987b). The degree of match between the input pattern and stored prototype is defined by a threshold. Note that this is different from Lippmann's Hamming net, which has a predefined threshold level. The threshold level of the adaptive Hamming net is dependent upon the input pattern. This enables the adaptive Hamming net to add new prototypes to an existing set of memorized prototypes without retraining the entire network.

The remainder of this paper is organized as follows. Section 2 introduces both ART 1 and fuzzy ART. Section 3 describes in detail the adaptive Hamming net.

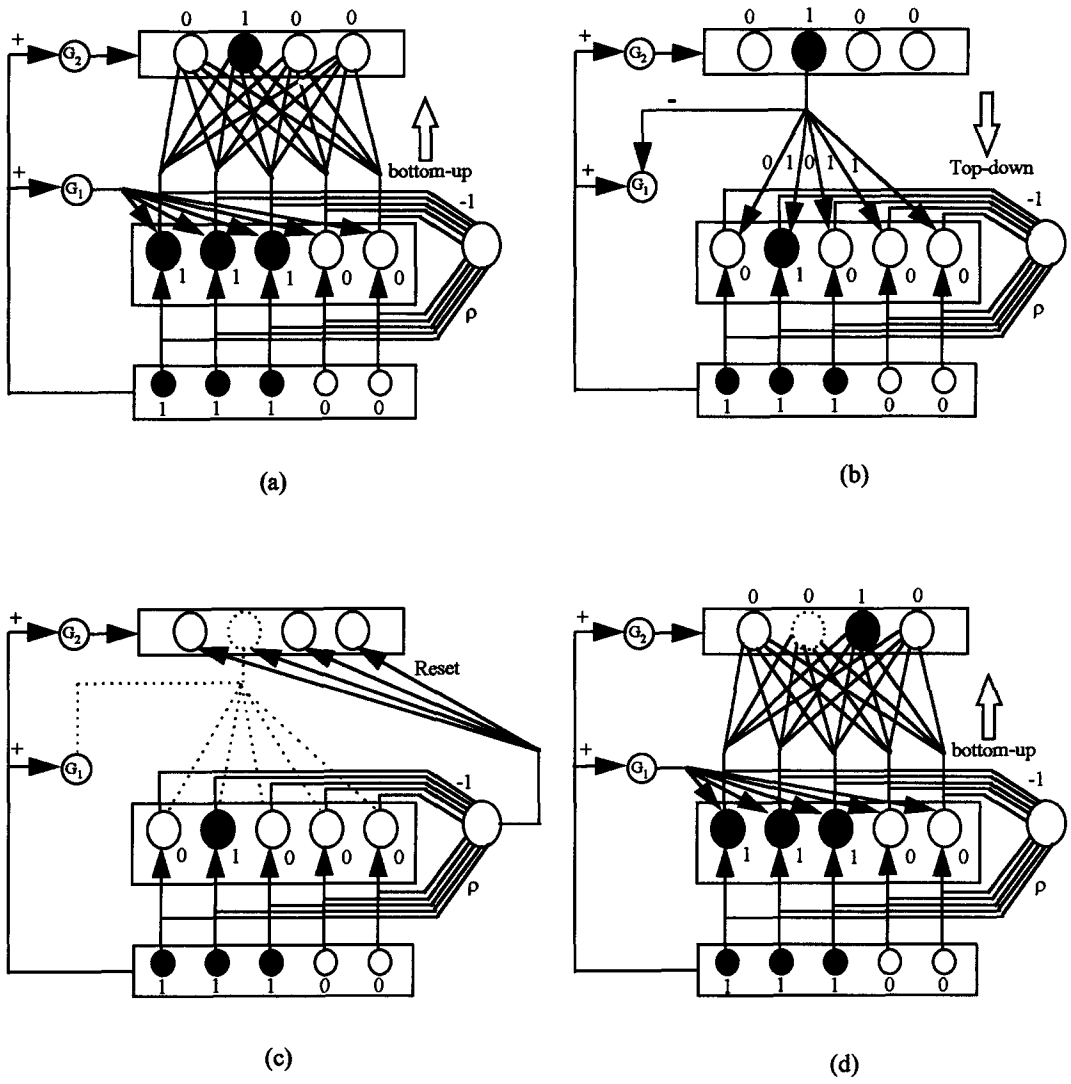

FIGURE 2. A search–reset cycle in an ART 1 network is shown. This search process evolves from a category choice in (a) to vigilance testing in (b), to mismatch reset in (c), to selection of a different candidate category in (d). Searching ensues until either an adequate match is made or a new category is established.

Section 4 demonstrates the functional equivalence of the adaptive Hamming net and fast-learning ART 1. In addition, the learning algorithm of the adaptive Hamming net is modified and applied to fuzzy ART. Section 5 presents simulation results comparing the adaptive Hamming net with fast-learning ART 1. Finally, the discussion and conclusion are presented in Section 6.

2. ADAPTIVE RESONANCE THEORY

2.1. ART 1

The binary-valued adaptive resonance theory (ART 1) neural network, proposed by Carpenter and Grossberg (1987a), is a two-layer classifier that stores an arbitrary number of binary input patterns using competitive learning. The main feature of ART 1 is the ability to self-organize and self-stabilize its recognition codes in real-time nonstationary input environments. To ensure that learning is stable, ART 1 uses a learned top-down expectation to check whether the input pattern and the chosen prototype pattern match. If not, the system will search the established categories until either an adequate match is found or a new category is established. This confidence measure in the ART 1 network is controlled by a parameter ρ, called a *vigilance parameter,* which determines how fine the categories will be.

For illustration, consider the ART 1 architecture shown in Figure 1, which has N output nodes representing N potential categories for learning n-dimensional input patterns. Each category j corresponds to a bottom-up weight vector $\mathbf{B}_j \equiv [b_{j1}, \ldots, b_{jn}]$ and a top-down weight vector $\mathbf{W}_j \equiv [w_{j1}, \ldots, w_{jn}]$. The bottom-up weight b_{ji} is used to multiply a signal that is carried

from the ith node in the input layer to the jth node in the category layer. The top-down weight w_{ji} is used to multiply a signal that is carried from the jth node in the category layer to the ith node in the input layer. These weights, which link the input layer and the category layer, can change as a result of learning. Because the top-down weights encode the salient features of the patterns appearing across the input layer, they are also called *top-down templates.* The top-down templates are the category prototype patterns.

ART 1 is capable of stable learning at either a slow or a fast rate. In slow learning, the first-order nonlinear ordinary differential equations (Carpenter & Grossberg, 1987a) for the bottom-up and top-down weights are not allowed to converge to a steady state for any particular input pattern. In contrast, in fast learning the weights are allowed to converge to new equilibrium values in response to each input pattern. Because many applications of ART 1 use fast learning, only the fast-learning case is described in detail in this paper. An algorithm for the fast-learning ART 1 is presented below.

Step 0. Initialization: Initially, all categories are said to be *uncommitted.* The bottom-up weights b_{ji} satisfy

$$b_{j1}^{(\text{init})} = \ldots = b_{jn}^{(\text{init})} = \lambda_j, \quad j = 1, \ldots, N, \tag{1}$$

where λ_j are ordered according to $\lambda_N < \ldots < \lambda_2 < \lambda_1 < 1/(\alpha + n)$, with $0 < \alpha \ll 1$. Initially, all top-down weights w_{ji} are set equal to 1. That is,

$$w_{j1}^{(\text{init})} = \ldots = w_{jn}^{(\text{init})} = 1, \quad j = 1, \ldots, N. \tag{2}$$

Step 1. Present a binary pattern $\mathbf{X} = [x_1, \ldots, x_n]$ to the input nodes, where each component $x_i \in \{0, 1\}$.

Step 2. Use bottom-up processing to obtain a weighted sum expressing the input to each category node:

$$u_j = \sum_{i=1}^{n} b_{ji} x_i, \quad j = 1, \ldots, N. \tag{3}$$

Step 3. Use a winner-take-all network to select the winner, which yields the maximum weighted sum. The winner is indexed by J, where

$$J = \arg \max_{j=1,\ldots,N} u_j. \tag{4}$$

Step 4. Vigilance test: In a fast-learning ART 1, mismatch reset occurs if

$$\frac{|\mathbf{X} \cap \mathbf{W}_J|}{|\mathbf{X}|} \equiv \frac{\sum_{i=1}^{n} w_{Ji} x_i}{\sum_{i=1}^{n} x_i} < \rho, \tag{5}$$

where $\cap$ is the component-by-component Boolean AND operator and $|\mathbf{X}|$ is the number of ones in binary pattern $\mathbf{X}$. Then the weighted sum u_J of the winner is reset to -1 for the duration of the input presentation to prevent its persistent selection during search and the algorithm returns to Step 3. The search process continues until a winner passes the vigilance test, that is,

$$\frac{\sum_{i=1}^{n} w_{Ji} x_i}{\sum_{i=1}^{n} x_i} \geq \rho, \tag{6}$$

then proceeds to Step 5.

For the vigilance test process, the parameter ρ satisfies $0 \leq \rho \leq 1$.

Step 5. If the winner passes the vigilance test, the bottom-up weights b_{ji} and the top-down weights w_{ji} are updated according to the equations

$$w_{ji}^{(\text{new})} = \begin{cases} w_{ji}^{(\text{old})} x_i & \text{if} \quad j = J \\ w_{ji}^{(\text{old})} & \text{if} \quad j \neq J \end{cases} \tag{7}$$

and

$$b_{ji}^{(\text{new})} = \begin{cases} \dfrac{w_{ji}^{(\text{old})} x_i}{\alpha + \sum_{i=1}^{n} w_{ji}^{(\text{old})} x_i} & \text{if} \quad j = J \\ b_{ji}^{(\text{old})} & \text{if} \quad j \neq J, \end{cases} \tag{8}$$

where $i = 1, \ldots, n$ and $j = 1, \ldots, N$.

Step 6. Go to Step 1 and present a new pattern.

The loop from Step 4 back to Step 3 constitutes a memory search cycle. In ART 1, Carpenter and Grossberg (1987a) designed a self-adjusting search mechanism to maintain efficiency as the prototype patterns become arbitrarily complex due to learning. They also show that the ordering of initial u_j values determines the order of search in each trial. That is, if the initial u_j values are ordered by decreasing size, as in

$$u_{j1} > u_{j2} > u_{j3} > \ldots, \tag{9}$$

then categories are searched in the order $j1, j2, j3, \ldots$ in that trial. All things being equal, a fast-learning ART 1 with a higher level of vigilance will require more accurate pattern matches and hence will search more deeply in response to each input pattern.

From the above discussion, it is evident that the fast-learning ART 1 algorithm has a mixture of both serial and parallel operations. The main operations of ART 1, such as bottom-up input processing and top-down template matching, can be implemented in parallel. However, the search process remains a serial operation if the ART 1 algorithm is implemented as described.

Let us now further analyze the order of search. As in Carpenter and Grossberg (1987a), three types of learned templates are initially defined with respect to an input pattern $\mathbf{X}$: *subset* templates, *superset* templates, and *mixed* templates. The components of a subset template $\mathbf{W}$ satisfy $\mathbf{W} \subseteq \mathbf{X}$. That is,

$$0 < |\mathbf{X} \cap \mathbf{W}| = |\mathbf{W}| \leq |\mathbf{X}|. \tag{10}$$

The components of a superset template $\mathbf{W}$ satisfy $\mathbf{W} \supset \mathbf{X}$. That is,

$$0 < |\mathbf{X} \cap \mathbf{W}| = |\mathbf{X}| < |\mathbf{W}|. \tag{11}$$

The components of a mixed template $\mathbf{W}$ satisfy

$$0 < |\mathbf{X} \cap \mathbf{W}| < |\mathbf{W}| \text{ and } 0 < |\mathbf{X} \cap \mathbf{W}| < |\mathbf{X}|. \quad (12)$$

From the above definitions, an uncommitted node may be treated as a node with an *unlearned superset* template. To illustrate how the search ends, the search order theorem of ART 1 (Carpenter & Grossberg, 1987a) is rewritten below.

THEOREM FOR SEARCH ORDER. *Suppose that*

(A1) $1 \le |\mathbf{X}| \le n - 1$

(A2) $0 < \alpha \le 1/|\mathbf{X}|$

(A3) *N is large enough so that the category layer can represent all $2^n - 1$ possible nonzero input patterns.*

Then the order of search among subset templates, superset templates, and mixed templates is as follows:

1. *Learned subset templates with respect to $\mathbf{X}$ are searched initially, in order of decreasing size. If the largest subset template is reset, then all subset templates are reset.*
2. *If all subset templates are searched (or no subset templates) and if there exist learned superset templates but no mixed templates, then the node with the smallest superset template will be activated next and will code $\mathbf{X}$.*
3. *If all subset templates are searched (or no subset templates) and if both mixed templates and learned superset templates exist, then the node with a template $\mathbf{W}_{J1}$ will be searched before the node with a template $\mathbf{W}_{J2}$ iff*

$$\frac{|\mathbf{X} \cap \mathbf{W}_{J1}|}{|\mathbf{W}_{J1}|} > \frac{|\mathbf{X} \cap \mathbf{W}_{J2}|}{|\mathbf{W}_{J2}|}. \quad (13)$$

4. *If all subset templates are searched (or no subset templates) and if there exist mixed templates but no learned superset templates, then the node with a mixed template $\mathbf{W}_{J1}$ will be searched before an uncommitted node $J2$ iff*

$$\frac{|\mathbf{X} \cap \mathbf{W}_{J1}|}{\alpha + |\mathbf{W}_{J1}|} > \sum_{i=1}^{n} b_{J2,i}^{(\text{init})} x_i. \quad (14)$$

5. *If all subset templates are searched (or no subset templates) and if no other learned templates exist, then the first unlearned superset template (uncommitted node) will be searched and will code $\mathbf{X}$.*

REMARKS. If $\mathbf{W}_{J2}$ is a learned superset template with respect to $\mathbf{X}$, then eqn (13) is equivalent to

$$\frac{|\mathbf{X} \cap \mathbf{W}_{J1}|}{|\mathbf{W}_{J1}|} > \frac{|\mathbf{X}|}{|\mathbf{W}_{J2}|}. \quad (15)$$

If $\mathbf{W}_{J2}$ is also a mixed template with respect to $\mathbf{X}$ and $|\mathbf{X} \cap \mathbf{W}_{J1}|/|\mathbf{W}_{J1}| = |\mathbf{X} \cap \mathbf{W}_{J2}|/|\mathbf{W}_{J2}|$, then node $J1$ will be searched before node $J2$ if

$$|\mathbf{W}_{J1}| < |\mathbf{W}_{J2}|. \quad (16)$$

A proof can be found in Carpenter and Grossberg (1987a).

In simulations, the search process of ART 1 is time consuming when a new input pattern is quite different from all the existing prototype patterns. If logical circuits were added to distinguish subset, superset, and mixed templates from all templates, it would be necessary to search for only the largest subset template, the smallest superset template, and other mixed templates being used. This could save much unnecessary resetting time, but would increase the architectural complexity.

Another method that can reduce the search time of the ART 1 model was proposed by Shih, Moh, and Chang (1992). They used bidirectional testing to determine the similarity between an input pattern and the top-down templates of a chosen category. In addition to eqn (6), they introduced the following inequality to determine how similar a prototype is to the input pattern

$$\frac{\sum_{i=1}^{n} w_{Ji} x_i}{\sum_{i=1}^{n} w_{Ji}} \ge \rho. \quad (17)$$

In eqn (6), for a vigilance test, the ratio represents the percentage of the 1's of the input pattern existing in the top-down templates. In eqn (17), the ratio represents the percentage of the 1's of the top-down templates existing in the input pattern. It is worth noting that the similarity measure used in eqn (17) is just the nonfuzzy form of the similarity measure T_j (denoted by u_j in our notation) used in the conservative limit of fuzzy ART (Carpenter et al., 1991b, p. 765).

When the above modified ART 1 model is used, the search procedure does not need to check the categories with lower matching ratios characterized by eqn (17). This can save a significant amount of time in searching most of the unmatched categories. However, it still does not completely avoid the search cycle.

In this paper, a modification is proposed to prevent this unnecessary searching. More precisely, the architecture proposed here produces the same clustering results as fast-learning ART 1 without any searching. Because fast-learning ART 1 is a special case of fuzzy ART (Carpenter et al., 1991b), all of our analysis can be extended to fuzzy ART. Before describing the adaptive Hamming net, we briefly introduce the fuzzy ART algorithm.

2.2. Fuzzy ART

ART 1 is the earliest ART network designed for binary pattern clustering. To enhance the capabilities of ART 1 for analog inputs, Carpenter et al. (1991b) developed a fuzzy ART algorithm that incorporates a fuzzy logic operator in ART 1. The fuzzy ART system replaces the

dot product used in ART 1 by the min operator. This makes fuzzy ART capable of handling both analog and binary inputs.

Fuzzy ART is an algorithmic clustering technique. It reduces to fast-learning ART 1 in response to binary input patterns. In fuzzy ART, the weight vector $\mathbf{W}_j$ subsumes both the bottom-up and top-down weight vectors of ART 1. The following algorithm implements the basic idea behind fuzzy ART.

Step 0. Initialization: Initially, all categories are said to be *uncommitted.* The weights w_{ji} are set equal to 1. That is,

$$w_{j1}^{(\text{init})} = \ldots = w_{jn}^{(\text{init})} = 1, \quad j = 1, \ldots, N. \tag{18}$$

Step 1. Present a binary or analog pattern $\mathbf{X} = [x_1, \ldots, x_n]$ to the input nodes. All input values x_i must be within the range [0, 1].

Step 2. The input to the jth node in the category layer is given by

$$u_j = \frac{\sum_{i=1}^{n} \min(w_{ji}, x_i)}{\alpha + \sum_{i=1}^{n} w_{ji}}, \quad j = 1, \ldots, N. \tag{19}$$

Step 3. Use a winner-take-all network to select the winner, which yields the maximum weighted sum. The winner is indexed by J, where

$$J = \arg \max_{j \; j=1,\ldots,N} u_j. \tag{20}$$

If more than one u_j is maximal, the output node with the smallest index is chosen to break the tie.

Step 4. Vigilance test: In fuzzy ART, mismatch reset occurs if

$$\frac{\sum_{i=1}^{n} \min(w_{Ji}, x_i)}{\sum_{i=1}^{n} x_i} < \rho. \tag{21}$$

Then the weighted sum u_J of the winner is reset to -1 for the duration of the input presentation and the algorithm returns to Step 3. The search process continues until a winner passes the vigilance test, that is,

$$\frac{\sum_{i=1}^{n} \min(w_{Ji}, x_i)}{\sum_{i=1}^{n} x_i} \geq \rho, \tag{22}$$

and then proceeds to Step 5.

For the vigilance test process, the parameter ρ satisfies $0 \leq \rho \leq 1$. After a category is selected for learning it becomes *committed.*

Step 5. If the winner J passes the vigilance test, the weights w_{Ji} can be updated according to the equations

$$w_{Ji}^{(\text{new})} = \begin{cases} w_{Ji}^{(\text{old})} + \bar{\beta}(\min(x_i, w_{Ji}^{(\text{old})}) - w_{Ji}^{(\text{old})}) \\ \quad \text{if} \quad J \text{ is committed} \\ x_i \quad \text{if} \quad J \text{ is uncommitted,} \end{cases} \tag{23}$$

where the learning rate $\bar{\beta}$ is set in [0, 1] and $i = 1, \ldots, n$. In the fast-learning case, $\bar{\beta}$ is set to 1.

Step 6. Go to Step 1 and present a new pattern.

Fuzzy ART, as well as other ART models, implements a best-first search that is performed in the order of descending value of u_j. As mentioned before, the memory search cycle from Step 4 back to Step 3 can affect the computational speed and efficiency of fuzzy ART. To reduce the training time of fuzzy ART, we propose an efficient algorithm to allow analog patterns to be encoded without performing an extensive serial search through all stored prototypes. This algorithm, which is derived from fuzzy ART, is called a *fuzzy adaptive Hamming net* (see Section 4). For the special case of binary inputs and fast learning, the computations of the fuzzy adaptive Hamming net are identical to those of the adaptive Hamming net. The adaptive Hamming net and its learning algorithm are described in the next section.

3. ADAPTIVE HAMMING NET

An *adaptive Hamming net* is a two-layer neural network that consists of a *matching score net* with a threshold θ and a MAXNET. The matching score net is designed to map an input pattern into N matching scores, appearing respectively at the N output nodes of that net. For a binary-valued input, the matching scores are defined in terms of the inner product between the synaptic weight matrix and the input pattern. After multiplying the N matching scores by a set of weights, one can use a MAXNET to select an adequate category for which the weighted matching score is the highest. If an adequate match is found, the weights connected to that net are updated. Learning within an adaptive Hamming net either refines the prototype of a previously established category or assigns the input pattern to a new category. When the input pattern is familiar, the system will access a previously learned category. When the input pattern is novel, the system will memorize it.

The architecture of an adaptive Hamming net is shown schematically in Figure 3. The net contains n input layer neurons, N hidden layer neurons, and N output layer neurons. To carry out *fast learning* of stable recognition categories in response to arbitrary sequences of binary input patterns, the matching score between an input pattern and an existing prototype must be larger than a certain threshold. This means that learning occurs only when the input pattern and stored prototype are sufficiently similar. Note that the threshold level of the matching score net is autotuned by a linear combiner, as shown in Figure 3.

The adaptive Hamming net and the fixed-weight Hamming net (Lippmann, 1987) have a similar architecture, but use different weight learning and thresholding techniques. In a traditional Hamming net, the prototype patterns are directly assigned to the weights and the threshold value is fixed. In contrast, in an adap-

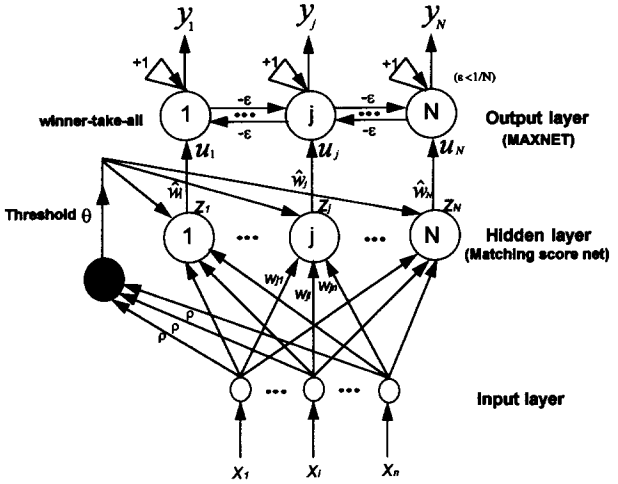

FIGURE 3. Adaptive Hamming net architecture. The large solid circle is a linear combiner that controls the threshold value of a matching score net.

tive Hamming net, the weights can be refined by a similar input pattern or directly assigned by a novel pattern and the threshold value is controlled by the input pattern and *vigilance parameter* ρ. Unlike the fixed-weight Hamming net, the matching scores in the adaptive Hamming net are also weighted by a set of parameters.

To clarify the approach used here, the recognition category J of ART 1 may be treated as a solution to the maximization problem (Healy, Caudell, & Smith, 1993)

$$J := \arg \max_{j \; j=1,\ldots,N} \frac{\sum_{i=1}^{n} w_{ji} x_i}{\alpha + \sum_{i=1}^{n} w_{ji}} \tag{24}$$

$$\text{subject to } \frac{\sum_{i=1}^{n} w_{ji} x_i}{\sum_{i=1}^{n} x_i} \geq \rho \tag{25}$$

where $0 \leq \rho \leq 1$, $0 < \alpha \ll 1$, and $x_i = 0$ or 1. This shows clearly that the selection of category J is a global decision over the network: it depends upon the current state of all templates stored in the synaptic weights.

In the ART 1 model, initially a category J is sought to maximize the similarity measure $(\sum_{i=1}^{n} w_{ji} x_i)/(\alpha + \sum_{i=1}^{n} w_{ji})$. The chosen category is then checked to determine whether it satisfies the constraint in eqn (25). If this constraint is not satisfied for the category J, then a reset occurs and the search continues. The above procedure is different from the approach used in this study.

The key idea underlying the adaptive Hamming net is to utilize a piecewise linear function to select a set of prototypes that satisfy the constraints in eqn (25). After all surviving candidate categories are formed, the function of the MAXNET is to determine the prototype that is most like the input pattern. That is, the arg max function in eqn (24) returns the index of the maximum element in the sequence of surviving nodes. This approach is more efficient than a fast-learning ART 1.

In the remainder of this section, the main operations of the adaptive Hamming net (Figure 3) will be described. The fast-learning algorithm of the adaptive Hamming net is summarized below.

Input pattern: The input pattern $\mathbf{X} = [x_1, \ldots, x_n]$ is a binary vector, where each component $x_i \in \{0, 1\}$.

Similarity evaluation: To test whether the input pattern is sufficiently similar to the stored prototype of a category, a piecewise linear function is used to suppress summed input signals that fall below a certain threshold. That is,

$$z_j = f_\theta \left(\sum_{i=1}^{n} w_{ji} x_i \right), \tag{26}$$

where w_{ji} denotes the weight value linked between the ith neuron at the input layer and the jth neuron at the hidden layer and $f_\theta(\cdot)$ is the following piecewise linear function

$$f_\theta(x) = \begin{cases} 0 & \text{if } x < \theta \\ x & \text{if } x \geq \theta. \end{cases} \tag{27}$$

Equation (27) is similar to a function used in ART 2 (Carpenter & Grossberg, 1987b) and ART 2-A (Carpenter et al., 1991a). The only difference in the function $f_\theta(\cdot)$ used in ART 2 and in an adaptive Hamming net is that the threshold θ is assumed to be constant in the former and variable in the latter. The adaptive threshold θ in eqn (27) satisfies the equation

$$\theta = \sum_{i=1}^{n} \rho x_i, \tag{28}$$

where ρ is the *vigilance parameter,* with $0 \leq \rho \leq 1$. Equation (28) can be implemented by using a single element neural network with n constant weight values ρ for which output is the product of ρ and the sum of input components.

Category choice: The output layer uses a winner-take-all network to find a winner from among all output units. The winner J is defined as the unit for which the weight-gated signal u_j is the largest. That is,

$$J = \arg \max_{j \; j=1,\ldots,N} u_j \tag{29}$$

where $u_j = \hat{w}_j z_j$, $j = 1, \ldots, N$, and the weight value $\hat{w}_j$ links the jth neuron in the hidden layer and the jth neuron in the output layer.

In a real network, one can use *lateral interaction* to implement a set of winner-take-all neurons. A well-known network for winner selection is MAXNET, which was introduced by Lippmann (1987) and can be regarded as a subnet of the Hamming net. A MAXNET (shown schematically in Figure 3) is a fully interconnected network with lateral inhibitions between its N nodes and excitatory connections from each node's out-

put back to its input. In the subnet, all excitatory weights are set to 1 and all inhibitory weights are set to a value of $-\varepsilon$. To ensure convergence, the inhibition constant ε must satisfy the constraint $0 < \varepsilon < 1/N$ (Lippmann, Gold, & Malpass, 1987).

The MAXNET usually needs several iterations to choose the winner. Once the hidden layer outputs u_j have been received at the output layer, MAXNET iterates until the output of only one node is positive. The one surviving positive output is the winner. Of course, in simulations, one need not use a MAXNET, but can instead find the winner algorithmically.

Fast-learning: When the output node J is chosen, the weight values are updated by the equations

$$w_{ji}^{(\text{new})} = \begin{cases} w_{ji}^{(\text{old})} x_i & \text{if} \quad j = J \\ w_{ji}^{(\text{old})} & \text{if} \quad j \neq J \end{cases} \tag{30}$$

and

$$\hat{w}_j^{(\text{new})} = \begin{cases} \dfrac{1}{\alpha + z_j} & \text{if} \quad j = J \\ \hat{w}_j^{(\text{old})} & \text{if} \quad j \neq J, \end{cases} \tag{31}$$

where $i = 1, \ldots, n$; $j = 1, \ldots, N$, and α is taken to be a small positive constant value (i.e., $0 < \alpha \ll 1$).

Equation (30) is the same as the *fast-learning* algorithm used in ART 1 (Carpenter & Grossberg, 1987a). In this algorithm, because $x_i \in \{0, 1\}$, the weights of the winner J are updated only by removing 1's without adding them. By eqn (31), both subset and superset input patterns can directly access distinct categories. This feature plays an important role in ART 1 (Carpenter & Grossberg, 1987a).

Initial weights: For the network to be able to encode each input $\mathbf{X}$ given an arbitrary value of the vigilance parameter ρ, it is necessary to take all initial weights w_{ji} greater than or equal to 1. To understand this, suppose that for all weights $w_{ji} < 1$ and $\rho = 1$. Then, because $\Sigma_{i=1}^n w_{ji} x_i < \Sigma_{i=1}^n x_i = \theta$, all hidden layer outputs z_j will become zero. Thus, no category could learn any input pattern. To avoid this problem, the initial weights w_{ji} in this study are given by

$$w_{j1}^{(\text{init})} = \ldots = w_{jn}^{(\text{init})} = 1, \quad j = 1, \ldots, N. \tag{32}$$

Condition (32) ensures that the network can classify any input pattern into a category, because

$$\sum_{i=1}^{n} w_{ji}^{(\text{init})} x_i = \sum_{i=1}^{n} x_i \geq \sum_{i=1}^{n} \rho x_i, \quad \text{for} \quad 0 \leq \rho \leq 1. \tag{33}$$

This implies that after thresholding at least one node in the hidden layer is active; that is, there exists a positive z_j. The only exception is that the network has used up its full memory capacity and the input pattern is sufficiently different from all stored prototype templates. Naturally, if the network is at full capacity, adapting should stop.

Another set of weights $\hat{w}_j$ is initially chosen to satisfy a fundamental ART design constraint; namely, an input pattern $\mathbf{X}$ must be able to directly access a learned category j for which weight vector $\mathbf{W}_j$ equals $\mathbf{X}$. This can be accomplished by letting

$$\hat{w}_j^{(\text{init})} = \lambda_j, \quad j = 1, \ldots, N, \tag{34}$$

where

$$\lambda_N < \ldots < \lambda_j < \ldots < \lambda_1 < \frac{1}{\alpha + n}. \tag{35}$$

If weight values $\hat{w}_j$ are ordered according to eqn (35), then a new category is chosen in the order $j = 1, 2, \ldots, N$.

4. FUNCTIONAL EQUIVALENCE AND ITS IMPLICATIONS

From the previous two sections, it is clear that the functional equivalence of an adaptive Hamming net and a fast-learning ART 1 model can be established if the following are true:

1. The constant α and the vigilance parameter ρ in these two networks are assigned as the same values.
2. The initial weights of ART 1 satisfy eqns (1) and (2). On the other hand, the initial weights of the adaptive Hamming net satisfy eqns (32) and (34).

Under these conditions, these two networks can produce the same clustering results for the same input sequences. Hence, the weights w_{ji} that associate the input layer with the hidden layer in the adaptive Hamming net are the same as the top-down templates of the fast learning ART 1.

Because of the functional equivalence shown above, some useful properties of ART 1 may be applied to the adaptive Hamming net. One of these properties of ART 1 is its learning convergence. Georgiopoulos, Heileman, and Huang (1991) have provided a theoretical upper bound for the number of list presentations necessary to stabilize the learning of ART 1. This bound depends only on the number of distinct sizes of binary patterns in the input list. As is evident from the above discussion, the adaptive Hamming net has the same convergence property as ART 1. Further, after learning has stabilized, no new category will be formed and no existing prototype will be changed. In other words, each input pattern in the training set will directly find its cluster.

It should be noted that the adaptive Hamming net can be functionally equivalent to a fast-learning ART 1 while using fewer weights. For example, with n input nodes and N output nodes the adaptive Hamming net requires $n \times N + N$ weights ($n \times N$ weights w_{ji} plus

N weights $\hat{w}_j$), whereas ART 1 requires $2(n \times N)$ weights ($n \times N$ weights w_{ji} plus $n \times N$ weights b_{ji}). In this case, the adaptive Hamming net saves $(n - 1) \times N$ weights. With the reduced number of weights, one looses the ability to read out modified templates from the input layer of ART 1.

A primary limitation of the adaptive Hamming net is its restriction to binary pattern clustering. Utilizing the same idea as the fuzzy ART, the adaptive Hamming net can be extended to the case of analog input vectors. The new fuzzy adaptive Hamming net, as well as the ART models and adaptive Hamming net, can overcome the stability–plasticity problem. In the fuzzy adaptive Hamming net, plasticity is maintained because the net can directly memorize novel input patterns at any time. To obtain stability without sacrificing plasticity, the fuzzy adaptive Hamming net allows the weight vectors to move only in one direction by taking the min operator. However, the unidirectional weight updating rule will introduce the category proliferation problem described by Moore (1989). The fuzzy ART solution to the category proliferation problem is to use a preprocessing step called *complement coding* (Carpenter et al., 1991b). The same idea can be applied to the algorithm proposed here.

Complement coding is an input normalization process that represents the presence or absence of a particular feature in the input pattern. If $\mathbf{X}$ is the given input pattern vector of n features, the complement of $\mathbf{X}$, denoted by $\mathbf{X}^c$, represents the absence of each feature where

$$x_i^c = 1 - x_i, \quad i = 1, \ldots, n. \tag{36}$$

Because complement coding normalizes the input patterns, the sum in eqn (28) reduces to a constant

$$\begin{aligned} \theta &= \sum_{i=1}^{n} \rho x_i + \sum_{i=1}^{n} \rho x_i^c \\ &= n\rho. \end{aligned}$$

With complement coding, the initial condition (32) is replaced by

$$w_{j1}^{(\text{init})} = \ldots = w_{j,2n}^{(\text{init})} = 1, \quad j = 1, \ldots, N. \tag{37}$$

Note that this additional preprocessing stage will double the number of input nodes.

The fuzzy adaptive Hamming net is an algorithmic clustering technique. The learning algorithm for this net is as follows:

Step 0. Initialization: Initially, all weights w_{ji} are set to 1. That is,

$$w_{j1}^{(\text{init})} = \ldots = w_{j,2n}^{(\text{init})} = 1, \quad j = 1, \ldots, N. \tag{38}$$

Step 1. Input: Present a binary or analog pattern $\mathbf{X} = [x_1, \ldots, x_n]$ to the input nodes. All input values x_i must be within the range $[0, 1]$.

Step 2. Complement coding: The complement of the original input vector $\mathbf{X}$ is denoted by $\mathbf{X}^c$, where $\mathbf{X}^c \equiv [x_1^c, \ldots, x_n^c]$ and $x_i^c = 1 - x_i$. Therefore, the complement-coded representation $\mathbf{I}$, internal to the fuzzy adaptive Hamming net, is given by the $2n$-dimensional vector

$$\begin{aligned} \mathbf{I} &= [\mathbf{X}, \mathbf{X}^c] \\ &= [x_1, \ldots, x_n, x_1^c, \ldots, x_n^c] \\ &\equiv [I_1, \ldots, I_{2n}], \end{aligned} \tag{39}$$

where $\sum_{i=1}^{2n} I_i = n$.

Step 3. Pattern matching: The matching score between the input $\mathbf{I}$ and the weight vector $\mathbf{W}_j$ is computed as

$$s_j = \sum_{i=1}^{2n} \min(I_i, w_{ji}). \tag{40}$$

To test whether the input pattern matches the stored prototype of a category closely enough or not, a piecewise linear function is used to suppress matching scores below a threshold. That is,

$$z_j = f_\theta(s_j) \equiv \begin{cases} 0 & \text{if } s_j < \theta \\ s_j & \text{if } s_j \geq \theta. \end{cases} \tag{41}$$

The threshold value is set at $\theta = n\rho$ and the vigilance parameter ρ is set in $[0, 1]$.

Step 4. Category choice: Choose the pattern category according to the choice function

$$u_j = \frac{z_j}{\alpha + \sum_{i=1}^{2n} w_{ji}}, \tag{42}$$

where $\alpha > 0$ and the category choice J is defined as

$$J = \arg \max_{j=1,\ldots,N} u_j. \tag{43}$$

If more than one u_j is maximal, the output node with the smallest index is chosen to break the tie.

Step 5. Learning: When the Jth category is chosen, the weight values are updated by the equation

$$w_{Ji}^{(\text{new})} = w_{Ji}^{(\text{old})} + \beta(z_J)(\min(I_i, w_{Ji}^{(\text{old})}) - w_{Ji}^{(\text{old})}), \tag{44}$$

where $i = 1, \ldots, 2n$ and the learning rates $\beta(z_J)$ satisfy

$$\beta(z_J) = \begin{cases} 1 & \text{if } z_J = n \\ \bar{\beta} & \text{if } z_J < n. \end{cases} \tag{45}$$

The parameter $\bar{\beta}$ is set in $[0, 1]$.
Equations (44) and (45) combine fast initial learning with a slower rate of forgetting (Carpenter et al., 1991b). Fast learning corresponds to setting $\beta(z_J) = 1$ (or $\bar{\beta} = 1$). If the category J is chosen for the first time, then $z_J = n$ and $\beta(z_J) = 1$. In this case, the com-

plement-coded input I_i is simply incorporated directly into the weight w_{Ji}. That is,

$$w_{Ji}^{(\text{new})} = I_i, \quad i = 1, \ldots, 2n. \tag{46}$$

Step 6. Backing Step 1: A new pattern enters the fuzzy adaptive Hamming net.

The fuzzy adaptive Hamming net is designed as a generalization of the adaptive Hamming net. It can learn stable categories in response to both analog and binary input patterns. For the special case of binary inputs and fast learning, the computations of the fuzzy adaptive Hamming net are identical to those of the adaptive Hamming net. Because the fuzzy adaptive Hamming net is functionally equivalent to fuzzy ART, the geometric interpretation of fuzzy ART dynamics (Carpenter et al., 1991b) can be applied to this algorithm.

As previously stated, the proposed neural networks eliminate the entire search process characteristic of ART 1 and fuzzy ART. Therefore, the complexity of the ART networks and the cycling time of the search process can be reduced. With some modification, it is possible to apply the basic concepts of the adaptive Hamming net to other ART-type neural networks, such as the fuzzy min–max clustering neural network (Simpson, 1993), adaptive fuzzy leader clustering (Netwon, Pemmaraju, & Mitra, 1992), ARTMAP, and fuzzy ARTMAP. We have recently submitted a comparative study of these neural networks for publication.

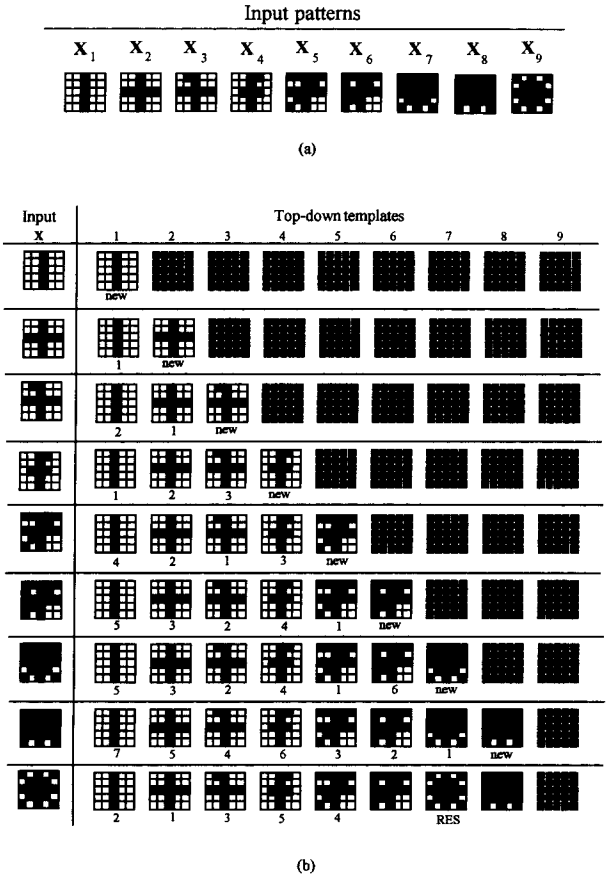

FIGURE 4. (a) The set of binary training patterns. (b) Template formation in ART 1 when the nine patterns are presented in the list order $X_1, X_2, \ldots, X_9$. The order of search is indicated by the numbers beneath the templates.

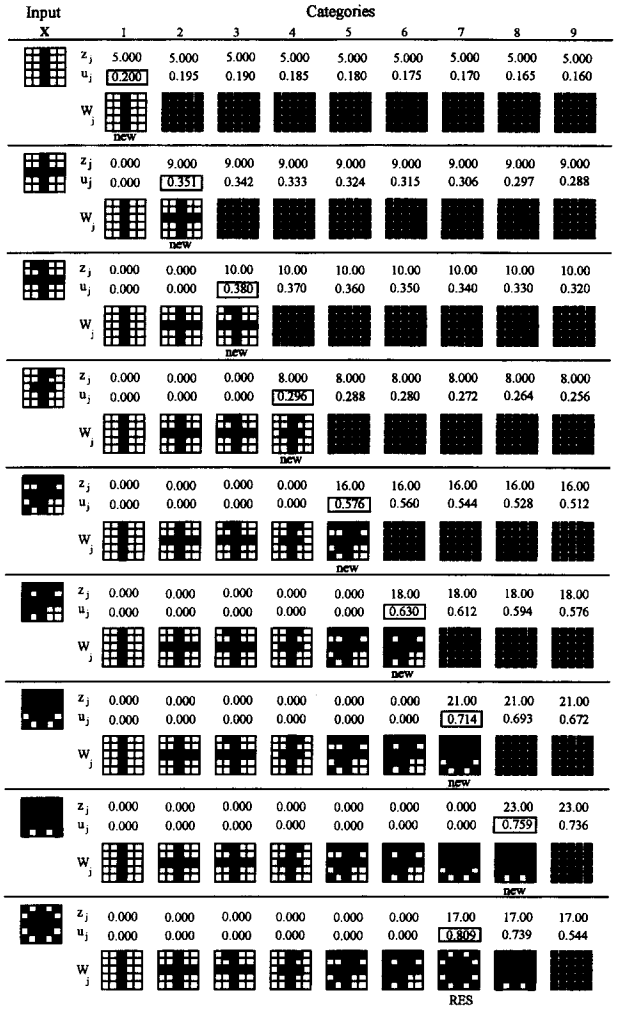

FIGURE 5. Fast-learning clustering results for the adaptive Hamming net. The first column indicates the bit-map input patterns and the rectangular boxes mark the maximum u_j value on each trial. For a training pattern, the weight vectors W_j are represented by 5 × 5 bit-map patterns.

5. SIMULATION RESULTS

In this section the superior efficiency of an adaptive Hamming net is demonstrated using two bit-map pattern clustering examples from Carpenter and Grossberg (1987a). The purpose of these examples is to show how the search process of ART 1 can be eliminated by the adaptive Hamming net. In these simulations, only the fast-learning case is considered.

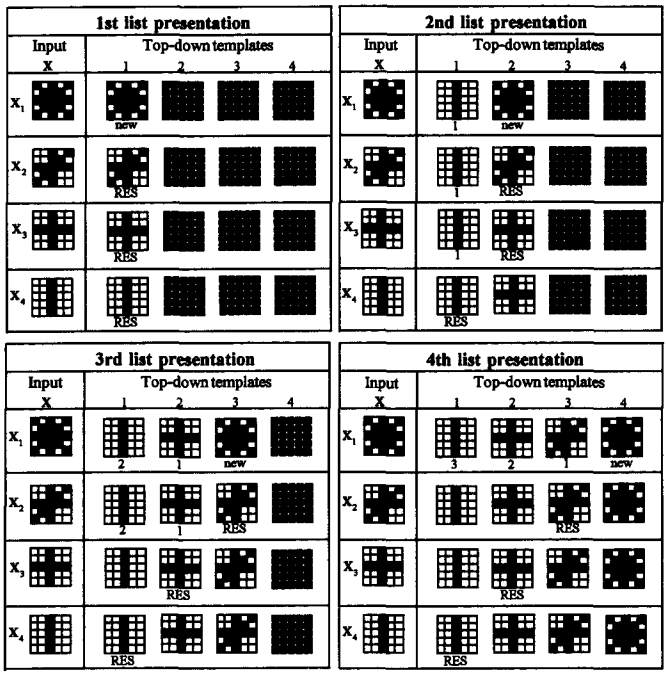

FIGURE 6. Template formation in ART 1 for a set of four nested binary input patterns. The order of search is indicated by the numbers beneath the templates. After four list presentations, no new templates are created and no existing learned templates are modified.

5.1. Bit-Map Pattern Clustering

As mentioned before, the ART 1 model and an adaptive Hamming net are limited to storing binary images. Therefore, in this example, these two neural networks are used to cluster 25-bit binary-valued patterns.

Consider the nine binary patterns shown in Figure 4a, where a black square indicates a binary 1 and a blank square indicates a zero. In this simulation, each of nine input patterns was presented once. The vigilance parameter ρ was set to be close to 1, the parameter α was chosen to be 0.01, and the number of uncommitted nodes was set to 9. Let us now try to compare the computational efficiency of ART 1 with that of an adaptive Hamming net.

Figure 4b illustrates how the ART 1 network modifies its search order on each trial to reflect the cumulative effects of prior learning. The first row shows the first pattern presented and the top-down templates formed in response to this pattern. The second row shows the second pattern presented and the top-down templates formed after search and learning, and so on. The symbol "new" represents a new category that codes the input pattern on that trial. The numbers 1, 2, 3, ... listed under the top-down templates itemize the search order. The symbol "RES" represents a resonant state that refines the corresponding template on that trial. In Figure 4b, the order of search is specified by the *Theorem for Search Order* (see Section 2). To complete the whole learning process, the system had to be reset 33 times. Note the order of search that occurred in response to the ninth input pattern. The sixth template was not searched.

For the same input patterns and presentation order, Figure 5 shows the learning results of the adaptive Hamming net with $\rho = 1$ and $\alpha = 0.01$. The outputs z_j of the hidden nodes, the weight-gated signals u_j, and the weight vectors $\mathbf{W}_j$ are calculated according to the algorithm described in Section 3. Note that the outputs z_j of all inadequate recognition categories are suppressed to zero.

To guarantee that the functional behavior of the adaptive Hamming net is equivalent to ART 1, it is sufficient to initialize all weights with the values that satisfy eqns (32) and (34). More precisely, all w_{ji} values are initially set to 1 and the weights $\hat{w}_j$ are initially arranged in the following order:

$$[0.04, 0.039, 0.038, 0.037, 0.036, 0.035, 0.034, 0.033, 0.032].$$

With the above initial conditions, the adaptive Hamming net and the previous fast-learning ART 1 model produce identical clustering results, as expected. In fact, the same weight vectors $\mathbf{W}_j$ are obtained as the top-down templates of ART 1. In Figure 5, the maximum u_j value is marked by a rectangular box, which represents the winning node on that trial. Clearly the recognition categories are formed without any searching.

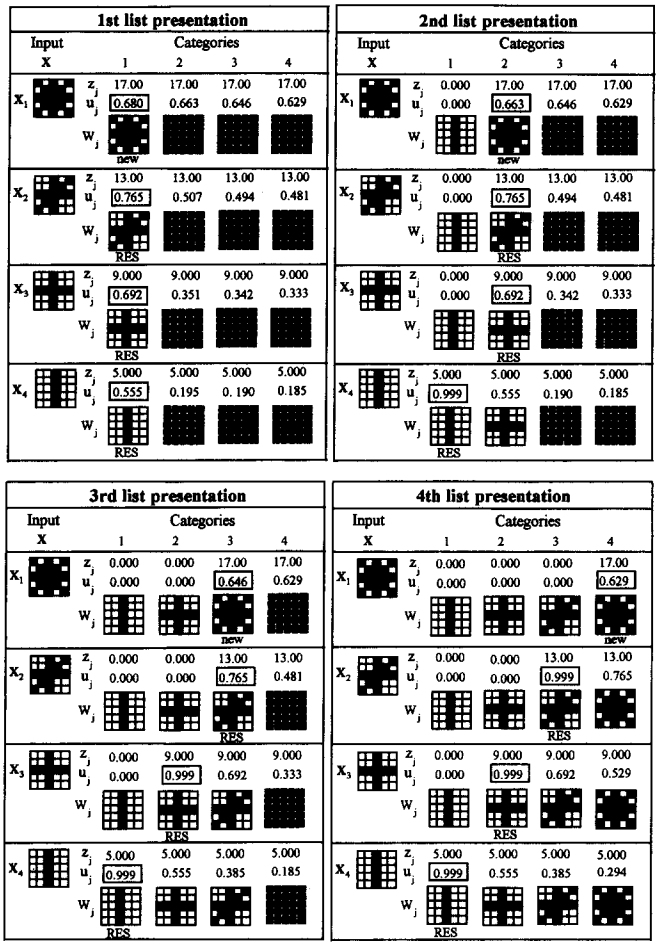

FIGURE 7. Behavior of the adaptive Hamming net in the second example. The network is repeatedly presented with a list of binary input patterns. After four list presentations, all patterns directly access distinct categories.

5.2. Extended Bit-Map Pattern Clustering

In the previous subsection each input pattern is presented only once. To demonstrate that the functional behavior of the adaptive Hamming net is equivalent to ART 1 in more complex cases, we consider a list of binary input patterns that is repeatedly presented to ART 1 and the adaptive Hamming net.

In the example shown in Figure 6, an ART 1 network with a vigilance parameter of $\rho = 0.8$ is presented with a list of four binary input patterns in order of decreasing size. (The size of a binary input pattern is equal to the number of its components that have a value of 1.) In particular, the patterns in the input list $\{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_4\}$ are such that $\mathbf{X}_4 \subset \mathbf{X}_3 \subset \mathbf{X}_2 \subset \mathbf{X}_1$, and this list is presented in the order $\mathbf{X}_1\mathbf{X}_2\mathbf{X}_3\mathbf{X}_4$. The order of pattern presentation is kept fixed from list presentation to list presentation. This example is the same as that used in Carpenter and Grossberg (1987a, Figure 8), except that the order of pattern presentation is reversed.

In this simulation, the choice parameter α was set to 0.01, the number of uncommitted nodes was set to 4, the initial values for the bottom-up weights were chosen to satisfy eqn (1), and all top-down weights were set to 1. In Figure 6, the input patterns and the templates formed after the first four list presentations are depicted as two-dimensional images of black and blank squares; a black square stands for a value of 1, and a blank square stands for a value of zero. As we can see from Figure 6, in the first list presentation no templates are searched; in the second list presentation the templates are searched three times; in the third list presentation the templates are searched four times; and in the fourth list presentation the templates are also searched three times. Because the four binary patterns $\mathbf{X}_1$, $\mathbf{X}_2$, $\mathbf{X}_3$, and $\mathbf{X}_4$ are presented in order of decreasing size, each pattern has direct access to a category node after exactly four list presentations (Carpenter & Grossberg, 1987a). In other words, no search occurs on subsequent presentations of the input list. In Figure 6, the order of search is indicated by the numbers beneath the templates.

The next simulation, shown in Figure 7, is the same as the one shown in Figure 6, except that the ART 1 network is replaced by the adaptive Hamming net. In

this simulation, the vigilance parameter ρ was chosen to be 0.8, the choice parameter α was chosen to be 0.01, the initial values for the weights w_{ji} were set to 1, and the weights $\hat{w}_j$ were initially arranged in the following order:

$$[0.04, 0.039, 0.038, 0.037].$$

In Figure 7, during each list presentation the outputs z_j of the hidden nodes, the weight-gated signals u_j, and the weight vectors $\mathbf{W}_j$ are shown. We observe from the simulations illustrated in Figures 6 and 7 that the weight vectors $\mathbf{W}_j$ formed in the adaptive Hamming net are the same as the top-down templates formed in ART 1. The adaptive Hamming net and the ART 1 network produce identical clustering results in each list presentation.

6. DISCUSSION AND CONCLUSION

One potential problem with ART models is the serial search time, which in the worst case is proportional to the number of learned templates. When a large number of patterns must be stored, this problem is a serious concern. This is particularly true when an ART 1 model is implemented in software simulations on conventional computer systems. To overcome this difficulty, in this paper the search process of ART 1 was first converted into an optimization problem, and then an effective algorithm was developed to find the best match.

A typical ART 1 search cycle can be characterized as maximizing eqn (24) under the constraints in eqn (25). When an ART 1 model is run, the objective function is evaluated and checked to see whether the constraint is violated. If the constraint is violated, the solution is infeasible and a parallel memory search is triggered. Because of its self-adjusting search mechanism, ART 1 is capable of discovering and learning appropriate categories without getting trapped in local maxima. However, this search process is not the most efficient method for finding the optimal solution. If biological nets are not concentrated upon, there is greater freedom in solving this problem.

It is clear that the search process will become more efficient if all infeasible solutions are eliminated from the solution set of eqn (24). In other words, a set of categories is initially selected that satisfy the constraints in eqn (25), and then the best of these feasible solutions is found. In this paper a neural network architecture called the adaptive Hamming net was proposed to perform the above process. The adaptive Hamming net can reproduce the behavior of ART 1 in the fast-learning limit without performing an extensive serial search through all the stored prototypes. It not only saves much unnecessary search time but also simplifies the ART 1 architecture. A primary limitation of the adaptive Hamming net is that it is restricted to binary pattern clustering. For analog inputs we have proposed a neural computing-based clustering algorithm called the fuzzy adaptive Hamming net.

This approach can also be applied to alternative match-based learning neural networks that utilize an ART-like search process. These neural networks include the fuzzy min–max clustering neural network, adaptive fuzzy leader clustering, ARTMAP, and fuzzy ARTMAP. In a more complex ART architecture, such as a hierarchy of ART 1 modules (Caudell, 1992), the approach in this study should also be quite efficient.

REFERENCES

Carpenter, G. A., & Grossberg, S. (1987a). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing,* **37,** 54–115.

Carpenter, G. A., & Grossberg, S. (1987b). ART2: Stable self-organization of category recognition codes for analog input patterns. *Applied Optics,* **26**(23), 4919–4930.

Carpenter, G. A., & Grossberg, S. (1991). *Pattern recognition by self-organizing neural networks.* Cambridge, MA: MIT Press.

Carpenter, G. A., Grossberg, S., & Reynolds, J. H. (1991). ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks,* **4,** 565–588.

Carpenter, G. A., Grossberg, S., & Rosen, D. B. (1991a). ART 2-A: An adaptive resonance algorithm for rapid category learning and recognition. *Neural Networks,* **4,** 493–504.

Carpenter, G. A., Grossberg, S., & Rosen, D. B. (1991b). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks,* **4,** 759–771.

Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., & Rosen, D. B. (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks,* **3**(5), 698–713.

Caudell, T. P. (1992). Hybrid optoelectronic adaptive resonance theory neural processor, ART 1. *Applied Optics,* **31**(29), 6220–6229.

Georgiopoulos, M., Heileman, G. L., & Huang, J. (1991). Properties of learning related to pattern diversity in ART 1. *Neural Networks,* **4,** 751–757.

Healy, M. J., Caudell, T. P., & Smith, D. G. (1993). A neural architecture for pattern sequence verification through inferencing. *IEEE Transactions on Neural Networks,* **4**(1), 9–20.

Lippmann, R. P. (1987). An introduction to computing with neural nets. *IEEE ASSP Magazine,* **4,** 4–22.

Lippmann, R. P., Gold, B., & Malpass, M. L. (1987). *A comparison of Hamming and Hopfield neural nets for pattern classification (Tech. Rep.* TR-769). MIT Lincoln Laboratory. Cambridge, MA: MIT Press.

Moore, B. (1989). ART 1 and pattern clustering. In D. Touretzky, G. Hinton, & T. Sejnowski (Eds), *Proceedings of the 1988 Connectionist Models Summer School* (pp. 174–185). San Mateo, CA: Morgan Kaufmann Publishers.

Netwon, S. C., Pemmaraju, S., & Mitra, S. (1992). Adaptive fuzzy leader clustering of complex data sets in pattern recognition. *IEEE Transactions on Neural Networks,* **3**(5), 794–800.

Shih, F. Y., Moh, J., & Chang, F. C. (1992). A new ART-based neural architecture for pattern classification and image enhancement without prior knowledge. *Pattern Recognition,* **25**(5), 533–542.

Simpson, P. (1993). Fuzzy min-max neural networks—Part 2: Clustering. *IEEE Transactions on Fuzzy Systems*, **1**(1), 32–45.

NOMENCLATURE

α choice parameter ($0 < \alpha \ll 1$)
$\bar{\beta}$ learning rate parameter ($0 \leq \bar{\beta} \leq 1$)
ε inhibitory weights of MAXNET ($\varepsilon < 1/N$)
ρ vigilance parameter ($0 \leq \rho \leq 1$)
θ threshold value of the matching score net
λ_j initial value of the bottom-up weight b_{ji} or initial value of the weight $\hat{w}_j$
$\mathbf{B}_j$ bottom-up weight vector of ART 1
b_{ji} the ith component of $\mathbf{B}_j$
$\mathbf{I}$ complement-coded input to the fuzzy adaptive Hamming net
I_i the ith element value of $\mathbf{I}$
J index of the chosen category
N number of pattern categories
n number of input nodes
s_j matching score between the input $\mathbf{I}$ and the weight vector $\mathbf{W}_j$
u_j weight-gated signal from the jth hidden node to the jth output node
$\mathbf{W}_j$ top-down weight vector of ART 1 or connection weights from the input layer to the jth hidden node
w_{ji} the ith component of $\mathbf{W}_j$
$\hat{w}_j$ connection weight from the jth hidden node to the jth output node
$\mathbf{X}$ input pattern
x_i the ith component of $\mathbf{X}$
$\mathbf{X}^c$ the complement of $\mathbf{X}$
x_i^c the ith component of $\mathbf{X}^c$
z_j the jth output value of the matching score net
$\cap$ Boolean AND operator