Properties of Learning of a Fuzzy ART Variant

Michael Georgiopoulos
Department of Electrical and Computer Engineering
University of Central Florida, Orlando, FL 32816, USA

Issam Dagher
Department of Electrical and Computer Engineering
University of Central Florida, Orlando, FL 32816, USA

Gregory L. Heileman
Department of Electrical and Computer Engineering
University of New Mexico, Albuquerque, NM 87131, USA

George Bebis
Department of Computer Science
University of Nevada Reno, Reno, NV, 89557, USA

Acknowledgments: This research was supported in part by a grant from Boeing Computer Services under contract W-300445.

Reprint Requests:
Michael Georgiopoulos
Department of Electrical and Computer Engineering
University of Central Florida, Orlando, FL 32816, USA
(407) 823-5338

Running Title: Fuzzy ART Variant

Abstract-This paper discusses one variation of the Fuzzy ART algorithm, referred to as *Fuzzy ART Variant.* The Fuzzy ART Variant is a Fuzzy ART algorithm, with a very large choice parameter value. Based on the geometrical interpretation of weights in Fuzzy ART, useful properties of learning pertaining to the Fuzzy ART Variant are presented and proven. One of these properties of learning establishes an upper bound on the number of list presentations required by the Fuzzy ART Variant to learn an arbitrary list of input patterns presented to it. This bound is small enough and consequently it demonstrates the short-training time property of Fuzzy ART Variant. Through simulation, it is shown that the Fuzzy ART Variant is as good a clustering machine as a Fuzzy ART algorithm using more typical values (i.e., small values) of the choice parameter.

Keywords—Neural Network, Unsupervised Learning, Supervised Learning, Clustering, Adaptive Resonance Theory

1 Introduction

Adaptive resonance theory was developed by Grossberg (1976), and a list of the ART architectures introduced in the last ten years are included in the reference list (Carpenter and Grossberg, 1987 a,b, 1990; Carpenter, Grossberg and Reynolds, 1991a; Carpenter, Grossberg and Rosen, 1991b; Carpenter, Grossberg, Markuzon, Reynolds and Rosen, 1992; Carpenter and Ross, 1995, Carpenter and Markuzon, 1996 Healy, Caudell and Smith, 1993; Huang, Georgiopoulos and Heileman, 1995; Marriott and Harrison 1995; Tan, 1995; Williamson, 1996). A major separation among all of these ART architectures is based on whether the learning applied is unsupervised or supervised. Unsupervised learning is implemented when a collection of input patterns needs to be appropriately clustered in categories, while supervised learning is utilized when a mapping needs to be learned between inputs and corresponding output patterns. A prominent member of the class of unsupervised ART architectures is Fuzzy ART (Carpenter, Grossberg and Rosen, 1991b), which is capable of clustering arbitrary collections of arbitrarily complex analog input patterns. Our focus in this paper is Fuzzy ART and its associated properties of learning.

Properties of learning for Fuzzy ART have already been reported in the literature (Carpenter, Grossberg and Rosen, 1991b; Huang, Georgiopoulos and Heileman, 1995) . Most of these properties of learning pertain to a Fuzzy ART network whose choice parameter is small. In particular, one of our favorite properties of learning in Fuzzy ART (i.e., its short training property) has been reported only for small values of the choice parameter. The Fuzzy ART algorithm has been initially introduced (Carpenter, Grossberg and Rosen, 1991b) for values of the choice parameter ranging over the interval $(0, \infty)$. It is therefore an issue of intellectual curiosity, and theoretical importance of how these learning properties change as we move from the domain of small choice parameter values to the domain of large choice parameter values. Some work towards this goal has appeared in the literature (Georgiopoulos, Fernlund, Bebis and Heileman, 1996). In this paper, the "Order of Search" property of learning in Fuzzy ART was examined. The "Order of Search" property identifies the order according to which nodes in the category representation field of Fuzzy ART are chosen. In (Georgiopoulos, Fernlund, Bebis and Heileman, 1996), three distinct orders of search were identified for three different ranges of the choice parameter value: (i) choice parameter small ($\alpha \to 0$), (ii) choice parameter large ($\alpha \to \infty$), and (iii) choice parameter of intermediate value ($0 < \alpha < \infty$). This paper extends the work in (Georgiopoulos, Fernlund, Bebis and Heileman, 1996) to other properties of learning, besides the "Order of Search" property. In particular, the short-training time property of Fuzzy ART, when the choice parameter is very large, is investigated. For simplicity, this Fuzzy ART network is referred to as *Fuzzy ART Variant*, despite the fact that it is simply Fuzzy ART with large values of the choice parameter. In the process of verifying the short-training time property of Fuzzy ART Variant, other useful properties of learning of the Fuzzy ART Variant were discovered. It is worth noting that the aforementioned Fuzzy ART Variant algorithm was discussed in (Carpenter and Gjaja, 1994).

The organization of the paper is as follows: In Chapter 2, the specifics of the Fuzzy ART network, that are pertinent to this paper, are briefly discussed. In Section 3, the Fuzzy ART Variant is introduced, which is the focus of this work, and some of the differences between Fuzzy ART with small choice parameter values and Fuzzy ART Variant are emphasized. In Section 4, three properties of learning of the Fuzzy ART Variant, including the short training time property, are proven and discussed. In Section 5, it is demonstrated, through simulations, that the Fuzzy ART Variant is as good a clustering machine as Fuzzy ART with small values of the choice parameter. Finally, in Section 6, a short review and concluding remarks are provided.

2 Fuzzy ART

2.1 Fuzzy ART Architecture

The Fuzzy ART neural network architecture is shown in Figure 1. It consists of two subsystems, *the attentional subsystem*, and the *orienting subsystem*. The attentional subsystem consists of two fields of nodes denoted F_1 and F_2. The F_1 field is called the *input field* because input patterns are applied to it. The F_2 field is called the *category or class representation field* because it is the field where category representations are formed. These category representations represent the clusters to which the input patterns, presented at the F_1 field, belong. The orienting subsystem consists of a single node (called the *reset node*), which accepts inputs from the F_1 field, the F_2 field (not shown in Figure 1), and the input pattern applied across the F_1 field. The output of the reset node affects the nodes of the F_2 field.

Some preprocessing of the input patterns of the pattern clustering task takes place before they are presented to Fuzzy ART. The first preprocessing stage takes as input an M-dimensional input pattern from the pattern clustering task and transforms it into an output vector $\mathbf{a} = (a_1, \ldots, a_M)$, whose every component lies in the interval [0, 1] (i.e., $0 \leq a_i \leq 1$ for $1 \leq i \leq M$). The second preprocessing stage accepts as an input the output $\mathbf{a}$ of the first preprocessing stage and produces an output vector $\mathbf{I}$, such that

$$\mathbf{I} = (\mathbf{a}, \mathbf{a}^c) = (a_1, \ldots, a_M, a_1^c, \ldots, a_M^c) \tag{1}$$

where

$$a_i^c = 1 - a_i \quad ; \quad 1 \leq i \leq M. \tag{2}$$

The above transformation is called *complement coding.* The complement coding operation is performed in Fuzzy ART at a preprocessor field designated by F_0 (see Figure 1). From now on, the vector $\mathbf{I}$ will be referred to as the *input pattern.* Each category j $(1 \leq j \leq N)$ in the category representation layer corresponds to a vector $\mathbf{w}_j = (w_{j1}, \ldots, w_{j,2M})$ of adaptive weights. The initial values for these weights are chosen to be equal to $w_{j1} = \ldots = w_{j,2M} = 1$, and a category with these weights is said to be *uncommitted.* Initial values for these weights may be taken greater than one. Larger weights bias the system against the selection of uncommitted nodes, leading to deeper searches of previously coded categories. After a category is chosen to represent an input pattern it is referred to as a *committed category* or node. It is worth noting that the Fuzzy ART weight vector $\mathbf{w}_j$ subsumes both the bottom-up and top-down weight vectors of Fuzzy ART.

The training phase of Fuzzy ART works as follows: Given a list of input patterns, designated as $\mathbf{I}^1$, $\mathbf{I}^2$, ..., $\mathbf{I}^P$, we want to train Fuzzy ART to cluster these input patterns into different categories. Obviously, patterns that are similar to each other are expected to be clustered in the same category by Fuzzy ART. In order to achieve the aforementioned goal, the training list is repeatedly presented to the Fuzzy ART architecture. That is, $\mathbf{I}^1$ is presented first, then $\mathbf{I}^2$, and eventually $\mathbf{I}^P$; this corresponds to one *list presentation.* Then, if it is necessary, $\mathbf{I}^1$, $\mathbf{I}^2$, ..., $\mathbf{I}^P$ is presented again. The training list is presented as many times as it is necessary for Fuzzy ART to cluster the input patterns. The clustering task is considered accomplished (i.e., the learning is complete) if the weights in the Fuzzy ART architecture do not change during a list presentation. The above training scenario is called *off-line training*.

Before discussing, in more detail, the training phase of Fuzzy ART let us elaborate on the Fuzzy ART parameters involved in its training phase. The parameter α is called *choice parameter* and it takes values in the interval $(0, \infty)$. Its value affects the bottom-up inputs that are produced at the F_2 nodes due to the presentation of an input pattern at F_1. The parameter ρ is called the *vigilance parameter* and it takes values in the interval $[0, 1]$. Small values of ρ result in coarse clustering of the input patterns, while large values of ρ result in fine clustering of the input patterns. The parameter N corresponds to the number of committed nodes during the training phase of Fuzzy ART. During its training phase Fuzzy ART operates with all the committed nodes and one uncommitted node. A committed node (category) in F_2 is a node that has coded at least one input pattern. An uncommitted node (category) is a node that is not committed.

The step-by-step implementation of the off-line training phase of Fuzzy ART is presented below. The Fuzzy ART network parameters α, ρ, and N are chosen at the beginning of the training phase; α is chosen from $(0, \infty)$ but typically small, ρ is chosen from $[0, 1]$ based on the fineness of the clusters we want to create, and obviously $N = 0$. Furthermore, the initial components of the weight vector corresponding to the first uncommitted node (i.e., $w_{1i}(0)$'s $1 \leq i \leq 2M$) are chosen to be equal to one. The value of the pattern index r is initialized to 1. For compactness of the presentation the definitions of the various functions that appear in the step-by-step implementation of the training phase will be provided after the step-by-step description is completed.

Off-Line Training Phase of Fuzzy ART

1. Choose the r-th input pattern from the training list.

2. Calculate the bottom-up inputs to all the $N + 1$ nodes in F_2 due to the presentation of the r-th input pattern. When calculating bottom-up inputs consider all the committed nodes and the uncommitted node. These

bottom-up inputs are calculated according to the following equation.

$$T_j(\mathbf{I}^r) = \begin{cases} \frac{M}{\alpha+2M} & \text{if } j \text{ is the uncommitted node} \\ \frac{|\mathbf{I}^r \wedge \mathbf{w}_j|}{\alpha+|\mathbf{w}_j|} & \text{if } j \text{ is a committed node} \end{cases} \tag{3}$$

3. Choose the node in F_2 that receives the maximum bottom-up input from F_1. Assume that this node has index j_{max}. Check to see whether this node satisfies the vigilance criterion. Three cases are now distinguished:

   (a) If node j_{max} is the uncommitted node it satisfies the vigilance criterion. Increase the parameter N by one. This way a new uncommitted node in F_2 is introduced, and its initial weight vector is chosen to be the "all-ones" vector. Go to Step 4.

   (b) If node j_{max} is a committed node, and it satisfies the vigilance criterion, go to Step 4. A committed node j_{max} satisfies the vigilance criterion if

$$\frac{|\mathbf{I}^r \wedge \mathbf{w}_{j_{max}}|}{|\mathbf{I}^r|} \geq \rho \tag{4}$$

   (c) If node j_{max} does not satisfy the vigilance criterion, disqualify this node by setting $T_{j_{max}}(\mathbf{I}^r) = -1$, and go to the beginning of Step 3.

4. The weights associated with node j_{max} are modified according to the following equation:

$$\mathbf{w}_{j_{max}} = \mathbf{w}_{j_{max}} \wedge \mathbf{I}^r \tag{5}$$

   If this is the last input pattern in the training list go to Step 5. Otherwise, go to Step 1 to present the next in sequence input pattern by increasing the index r by one.

5. After all the patterns have been presented consider two cases:

   (a) In the previous list presentation at least one component of the weight vectors has been changed. In this case, go to Step 1, and present the first input pattern, by resetting the index r to the value 1.

(b) In the previous list presentation no weight changes occurred. In this case, the learning process is considered complete.

In Step 5(b) above it is implied that there exists a finite-valued list presentation at which no weight changes occur. Unfortunately, no theoretical result exists to justify this claim for all values of the choice parameter α. For very small α parameter values this claim is valid because learning will be over in one list presentation (see Carpenter, Grossberg and Rosen, 1991b). For values of the parameter α that are not small some specialized results are discussed in (Huang, Georgiopoulos, and Heileman, 1995).

In the definition of the bottom-up inputs produced in F_2 due to the presentation of the input pattern $\mathbf{I}^r$ (see equation (3)) the "fuzzy-min" ($\wedge$) operation of two vectors, $\mathbf{I}^r$ and $\mathbf{w}_j$, is introduced. The fuzzy-min operation of two vectors $\mathbf{I}^r$ and $\mathbf{w}_j$ is a vector whose components are equal to the minimum of the corresponding components of $\mathbf{I}^r$ and $\mathbf{w}_j$. This same operation was used in the calculation of the vigilance ratio (see equation (4)) and in the updates of the weights (see equation (5)). Also, in the definition of the bottom-up inputs produced in F_2 due to the presentation of the input pattern $\mathbf{I}^r$ (see equation (3)) the operation $|\cdot|$ (e.g., $|\mathbf{w}_j|$) is introduced, that stands for the size of a vector. The size of a vector is defined to be the sum of its components. This operation was also used in the definition of the vigilance ratio in equation (4).

It has been shown in (Carpenter, Grossberg and Rosen, 1991b) that the weight vectors (i.e., the $\mathbf{w}_j$'s), corresponding to committed nodes in Fuzzy ART, have a geometrical interpretation. That is, $\mathbf{w}_j$ can be expressed as $(\mathbf{u}_j, (\mathbf{v}_j)^c)$, where $\mathbf{u}_j$ is the lower endpoint, and $\mathbf{v}_j$ is the upper endpoint of a hyperrectangle. This hyperrectangle lies in the M-dimensional space and includes all the input patterns that have chosen and were coded by node j. Using this representation the input pattern $\mathbf{I} = (\mathbf{a}, \mathbf{a}^c)$ can also be thought of as a hyperrectangle with lower endpoint $\mathbf{a}$ and upper endpoint $\mathbf{a}$ (that is a hyperrectangle of size 0). To visualize this hyperrectangle notion the hyperrectangle R_j is shown (see Figure 2), with endpoints $\mathbf{u}_j$ and $\mathbf{v}_j$, corresponding to the weight vector $\mathbf{w}_j$, which has coded the input patterns $\mathbf{I}^1 = (\mathbf{a}^1, (\mathbf{a}^1)^c)$, $\mathbf{I}^2 = (\mathbf{a}^2, (\mathbf{a}^2)^c)$, $\mathbf{I}^3 = (\mathbf{a}^3, (\mathbf{a}^3)^c)$, $\mathbf{I}^4 = (\mathbf{a}^4, (\mathbf{a}^4)^c)$, and $\mathbf{I}^5 = (\mathbf{a}^5, (\mathbf{a}^5)^c)$. In Figure 2, the $\mathbf{I}$'s are 4-D and the $\mathbf{a}$'s, $\mathbf{u}_j$ and $\mathbf{v}_j$ are 2-D. Since most of our illustrations from now on will be in the 2-D space hyperrectangles are referred to as rectangles.

3 The Fuzzy ART Variant Algorithm

As it was emphasized in the Introduction, the primary focus in this paper is the Fuzzy ART algorithm with a very large choice parameter value α (i.e., $\alpha \to \infty$). One might question this choice, since when α is large Fuzzy ART has the tendency to choose uncommitted nodes over existing committed nodes (see equation (3)). This way we may end up with a Fuzzy ART algorithm that does not perform useful clustering since every input pattern from the training list forms its own cluster. As it was mentioned in the Fuzzy ART paper though (see Carpenter, Grossberg and Rosen, 1991b) initial values of the weight components corresponding to uncommitted nodes may be taken larger than one. Taking it to the extreme these initial values of the weight components of the uncommitted nodes can be chosen to be very large so that the bottom up inputs to the uncommitted nodes are approximately equal to zero. By choosing, at the same time, a very large value for the choice parameter, the bottom-up inputs to uncommitted nodes will still be approximately zero, and the bottom-up inputs to a committed node will be proportional to the size of the "fuzzy-min" of the input pattern vector and the weight vector corresponding to this node. The Fuzzy ART algorithm with very large values of the initial components for the uncommitted weights, and very large value of the choice parameter is called *Fuzzy ART Variant*.

The step-by-step implementation of the off-line training phase of Fuzzy ART Variant algorithm is presented below. The Fuzzy ART Variant network parameters, α, ρ, and N, are chosen at the beginning of the training phase; α is chosen from $(0, \infty)$ but typically very large, ρ is chosen from $[0, 1]$ based on the fineness of the clusters we want to create, and obviously $N = 0$. Furthermore, the initial weights of the first uncommitted node (i.e., the $w_{1i}(0)$'s) are chosen to be very large too. The value of the pattern index r is initialized to 1.

Off-Line Training Phase of Fuzzy ART Variant

1. Same as in Step 1 of Fuzzy ART.

2. Calculate the bottom-up inputs to all the $N+1$ nodes in F_2 due to the presentation of the r-th input pattern. When calculating bottom-up inputs consider all the committed nodes and the uncommitted node. These bottom-up inputs are calculated according to the following equation.

$$T_j(\mathbf{I}^r) \approx \begin{cases} 0 & \text{if } j \text{ is the uncommitted node} \\ c \cdot |\mathbf{I}^r \wedge \mathbf{w}_j| & \text{if } j \text{ is a committed node} \end{cases} \tag{6}$$

   In the above equation c is a constant. Since it is equal to $1/(\alpha + |\mathbf{w}_j|)$, for very large α it is approximately independent of the index j. As a reminder $|\mathbf{w}_j|$ stands for the size of weight vector $\mathbf{w}_j$, where the size of vector is defined to be equal to the sum of its components.

3. Same as Step 3 of Fuzzy ART.

4. Same as Step 4 of Fuzzy ART.

5. Same as Step 5 of Fuzzy ART.

One way of understanding the differences between Fuzzy ART (small values of the choice parameter) and Fuzzy ART Variant (large values of the choice parameter) is by reporting the order according to which nodes in F_2 are chosen for these two algorithms. This topic has been extensively investigated by Carpenter and Grossberg (1987a) for ART1, and by Georgiopoulos, Fernlund, Bebis and Heileman (1996) for Fuzzy ART. Two of the results discussed in (Georgiopoulos, Fernlund, Bebis and Heileman, 1996) (denoted Results A and B) are reproduced here to illustrate some of the differences of Fuzzy ART and Fuzzy ART Variant.

Result A:

If an input pattern $\mathbf{I}$ *is presented to a Fuzzy ART architecture with small* α *parameter values (i.e.,* α *close to zero), and*

1. $\mathbf{I}$ *is inside rectangles* $R_{j_1}^{old}$ *and* $R_{j_2}^{old}$, *then* $\mathbf{I}$ *will choose first the rectangle of the smallest size.*

2. $\mathbf{I}$ *is outside rectangle* $R_{j_1}^{old}$ *and inside rectangle* $R_{j_2}^{old}$, *then* $\mathbf{I}$ *will choose first rectangle* $R_{j_2}^{old}$.

3. *$\mathbf{I}$ is outside rectangles $R_{j_1}^{old}$ and $R_{j_2}^{old}$, then $\mathbf{I}$ will choose first rectangle $R_{j_1}^{old}$ iff*

$$\mathrm{dis}(\mathbf{I}, R_{j_1}^{old}) < \frac{M - |R_{j_1}^{old}|}{M - |R_{j_1}^{old}|}\, \mathrm{dis}(\mathbf{I}, R_{j_2}^{old}) \tag{7}$$

Result B:

If an input pattern $\mathbf{I}$ is presented to a Fuzzy ART architecture with large α parameter values (i.e., α approaching ∞), and

1. *$\mathbf{I}$ is inside rectangles $R_{j_1}^{old}$ and $R_{j_2}^{old}$, then $\mathbf{I}$ will choose first the rectangle of the smallest size.*

2. *$\mathbf{I}$ is outside rectangle $R_{j_1}^{old}$ and inside rectangle $R_{j_2}^{old}$, then $\mathbf{I}$ will choose first rectangle $R_{j_1}^{old}$ iff*

$$|R_{j_1}^{new}| < |R_{j_2}^{old}| \tag{8}$$

3. *$\mathbf{I}$ is outside rectangles $R_{j_1}^{old}$ and $R_{j_2}^{old}$, then $\mathbf{I}$ will choose first rectangle $R_{j_1}^{old}$ iff*

$$|R_{j_1}^{new}| < |R_{j_2}^{old}| \tag{9}$$

Note that R_j^{old} is the rectangle corresponding to node j of field F_2 prior to the presentation of input pattern $\mathbf{I}$ at the field F_1. Also, R_j^{new} is the new rectangle corresponding to node j that would have been created if pattern $\mathbf{I}$ were to choose and be coded by node j. Note that if the input pattern $\mathbf{I}$ is inside rectangle R_j^{old}, then $R_j^{old} = R_j^{new}$; otherwise $R_j^{old} \neq R_j^{new}$ and, in particular, R_j^{new} includes R_j^{old}. Furthermore, $|R_j|$ stands for the size of rectangle R_j corresponding to node j in the field F_2; the size of a rectangle R_j is defined to be the city block distance between the endpoints $\mathbf{u}_j$ and $\mathbf{v}_j$ of this rectangle. Finally, the distance of an input pattern $\mathbf{I} = (\mathbf{a}, \mathbf{a}^c)$ from a rectangle R_j, which does not contain $\mathbf{I}$, is defined to be the minimum of the city block distances of $\mathbf{a}$ from the points that belong to the boundary of R_j.

To visualize the similarities and differences between the Fuzzy ART choices made when α is small, and when α is large (Fuzzy ART Variant) the consequences of Results A and B are depicted in Figure 3 for three example cases.

4 Properties of Learning of the Fuzzy ART Variant

In this section three properties of learning of the Fuzzy ART Variant are reported. Furthermore, their importance is emphasized, their proofs are provided, and finally an example case is discussed. These properties of learning are referred to as Results 1–3.

4.1 Results

Result 1 tells us that, during the Fuzzy ART Variant training, the identity of the node with the largest size rectangle does not change after the first list presentation, the identity of the node with the second largest rectangle does not change after the second list presentation, and so on. Result 1 also tells us that, during the Fuzzy ART Variant training, the size of the largest rectangle does not change after the first list presentation, the size of the second largest rectangle does not change after the second list presentation, and so on.

Result 1:

Consider the off-line training of a list of P input patterns using the Fuzzy ART Variant algorithm. Assume that after the first list presentation the Fuzzy ART Variant has created N categories in F_2. Designate by $j_n(t)$ $(1 \leq n \leq N; \quad t \geq 1)$ the identity of the node with the n-th largest rectangle immediately after the end of the t-th list presentation. Then,

$$\begin{aligned} j_n(t) &= j_n(n) \\ |R_{j_n(t)}| &= |R_{j_n(n)}| \end{aligned} \qquad \text{for} \quad 1 \leq n \leq N \; ; \; t \geq n+1 \tag{10}$$

Result 2 tells us that, during the Fuzzy ART Variant training, patterns that are coded by the largest rectangle in the second list presentation do not need to be presented to the Fuzzy ART architecture again, patterns that are coded by the second largest rectangle in the third list presentation do not need to be presented to the Fuzzy ART Variant again, and so on. Result 2 is useful because it allows us to eliminate patterns from the training list that do not affect the Fuzzy ART Variant learning process. This way the learning process can be made to be less computationally intensive.

Result 2:

Consider the off-line training of a list of input patterns using the Fuzzy ART Variant algorithm. Assume that after the first list presentation the Fuzzy ART Variant algorithm has created N categories in F_2. Designate by $j_n(t)$ $(1 \leq n \leq N; \quad t \geq 1)$ the identity of the node with n-th largest rectangle immediately after the end of the t-th list presentation. Let S_n $(1 \leq n)$ denote the set of training patterns that choose and are coded by node $j_n(n)$ in the $(n+1)$-th list presentation. Then, the patterns of collection S_n will always be coded by node $j_n(n)$ in list presentations $\geq n+2$.

Result 3 is important because it predicts an upper bound for the number of list presentations required by the Fuzzy ART Variant to learn a list of input patterns. In order to identify this upper bound, it suffices to present **once** the training list through the Fuzzy ART network using the Fuzzy ART Variant training algorithm (this way the value for the parameter N can be found).

Result 3:

Consider the off-line training of a list of input patterns using the Fuzzy ART Variant algorithm. Assume that after the first list presentation the Fuzzy ART Variant algorithm has created N categories in F_2. Then, training will be over in at most N list presentations.

4.2 Proof of the Results

The proofs of the results are based on parts 1 and 2 of Result B. Result B is proven in (Georgiopoulos, Fernlund, Bebis, and Heileman, 1996). Parts 1 and 2 of Result B were pictorially illustrated in Figure 3 (a and b).

Proof of Result 1:

Result 1 will be proven, by induction, in two steps.

- **Step 1:**

  Prove that Result 1 is valid for $n = 1$. That is, prove that

$$\begin{array}{rcl} j_1(t) & = & j_1(1) \\ |R_{j_1(t)}| & = & |R_{j_1(1)}| \end{array} \quad \text{for} \quad t \geq 2 \tag{11}$$

- **Step 2:**

  Assume that Result 1 is valid for all indices $\leq n$ (where $n \geq 1$), and demonstrate its validity for index $n + 1$. Hence, by assuming that

$$\begin{array}{rcll} j_1(t) & = & j_1(1) & t \geq 2 \\ |R_{j_1(t)}| & = & |R_{j_1(1)}| & t \geq 2 \\ \vdots & \vdots & \vdots & \vdots \\ j_n(t) & = & j_n(n) & t \geq n+1 \\ |R_{j_n(t)}| & = & |R_{j_n(n)}| & t \geq n+1 \end{array} \tag{12}$$

  it will be proven that is true for index $n + 1$, that is

$$\begin{array}{rcl} j_{n+1}(t) & = & j_{n+1}(n+1) \\ |R_{j_{n+1}(t)}| & = & |R_{j_{n+1}(n+1)}| \end{array} \quad \text{for} \quad t \geq n+2 \tag{13}$$

To demonstrate the validity of Step 1 two steps are needed:

- **Step 1a:** Verify equation (11) for list presentation $t = 2$.

- **Step 1b:** Assume the validity of (11) for list presentations $\leq t$ (where $t \geq 2$) and then show the validity of equation (11) for list presentation $t + 1$.

Step 1a:

To illustrate Step 1a it will be shown that the identity of the node with the largest rectangle (i.e., $j_1(1)$) and the size of the largest rectangle (i.e., $|R_{j_1(1)}|$) stay intact after the presentation of the first input pattern in list presentation 2 (i.e., pattern $\mathbf{I}^1$). Then, by assuming that the identity of the node with the largest rectangle and the size of the largest rectangle stay intact after the presentation of the first p input patterns in list presentation 2 (i.e., patterns

$\mathbf{I}^1$, $\mathbf{I}^2$, ..., $\mathbf{I}^p$), it will be proven that the identity of the node with the largest rectangle and the size of the largest rectangle stay intact after the presentation of the $(p+1)$-th input pattern in list presentation 2 (i.e., pattern $\mathbf{I}^{p+1}$). Hence, by induction, it can be stated that the identity of the node with the largest rectangle (i.e., $j_1(1)$) and the size of the largest rectangle (i.e., $|R_{j_1(1)}|$) stay intact after the presentation of the P training input patterns in list presentation 2, which is equivalent to saying that equation (11) is true for $t = 2$.

As a result, consider the presentation of the first input pattern $\mathbf{I}^1$, from the training list, during the second list presentation of the training list. We distinguish the following cases:

Case 1:

Rectangle $R_{j_1(1)}$ $(= R^{old}_{j_1(1)})$ is the rectangle of the smallest size that contains $\mathbf{I}^1$, and pattern $\mathbf{I}^1$ chooses and is coded by rectangle $R_{j_1(1)}$ (see Figure 4, case 1). In this case, after $\mathbf{I}^1$'s presentation, node $j_1(1)$ is the node with the largest rectangle and the size of the largest rectangle stays intact and equal to $|R_{j_1(1)}|$.

Case 2:

Rectangle $R_{j_1(1)}$ $(= R^{old}_{j_1(1)})$ is the rectangle of the smallest size that contains $\mathbf{I}^1$, and pattern $\mathbf{I}^1$ chooses and is coded by rectangle R_{j_x} $(= R^{old}_{j_x})$, where $j_x \neq j_1(1)$ (see Figure 4, case 2). Due to Result B, in order for the above case to occur the following inequality must be valid:

$$|R^{new}_{j_x}| < |R_{j_1(1)}| \tag{14}$$

The above inequality guarantees that after $\mathbf{I}^1$'s presentation, node $j_1(1)$ is the node with the largest rectangle and the size of the largest rectangle stays intact and equal to $|R_{j_1(1)}|$.

Case 3:

Rectangle R_{j_x} $(= R^{old}_{j_x})$, where $j_x \neq j_1(1)$, is the rectangle of the smallest size that contains $\mathbf{I}^1$, and pattern $\mathbf{I}^1$ chooses and is coded by rectangle R_{j_x} (see Figure 4, case 3). In this case, after $\mathbf{I}^1$'s presentation, node $j_1(1)$ is the node with the largest rectangle and the size of the largest rectangle stays intact and equal to $|R_{j_1(1)}|$.

Case 4:

Rectangle R_{j_x} $(= R_{j_x}^{old})$, where $j_x \neq j_1(1)$, is the rectangle of the smallest size that contains $\mathbf{I}^1$, and pattern $\mathbf{I}^1$ chooses and is coded by rectangle R_{j_y} $(= R_{j_y}^{old})$, where $j_y \neq j_x$ (see Figure 4, case 4). Obviously, due to Result B, j_y cannot be node $j_1(1)$. Also, due to Result B, in order for the above case to occur the following inequality must be valid:

$$|R_{j_y}^{new}| < |R_{j_x}| \tag{15}$$

But,

$$|R_{j_x}| < |R_{j_1(1)}| \tag{16}$$

The above two inequalities guarantee that after $\mathbf{I}^1$'s presentation, node $j_1(1)$ is the node with the largest rectangle, and the size of the largest rectangle stays intact and equal to $|R_{j_1(1)}|$.

Cases 1-4 cover all the possible scenarios, and they illustrate that during the presentation of the first pattern $\mathbf{I}^1$ in list presentation 2, the identity of the node with the largest rectangle and the size of this rectangle remain intact.

If it is now assumed that after $\mathbf{I}^1$'s, $\mathbf{I}^2$'s, ..., $\mathbf{I}^p$'s presentations, node $j_1(1)$ is the node with the largest rectangle, and the size of the largest rectangle stays intact and equal to $|R_{j_1(1)}|$, it is easy to duplicate the above arguments (Cases 1-4) to illustrate that after pattern's $\mathbf{I}^{p+1}$ presentation, node $j_1(1)$ is the node with the largest rectangle and the size of the largest rectangle stays intact and equal to $|R_{j_1(1)}|$. Hence, by induction, equation (11) has been proven for $t = 2$.

Step 1b:

If it is now assumed that equation (11) is valid for all list presentations $\leq t$ (where $t \geq 2$), that is

$$\begin{array}{rcl} j_1(2) & = & j_1(1) \\ |R_{j_1(2)}| & = & |R_{j_1(1)}| \\ \vdots & \vdots & \vdots \\ j_1(t) & = & j_1(1) \\ |R_{j_1(t)}| & = & |R_{j_1(1)}| \end{array} \tag{17}$$

then by duplicating the procedure, discussed in Step 1a, it can be demonstrated that

$$\begin{array}{rcl} j_1(t+1) & = & j_1(1) \\ |R_{j_1(t+1)}| & = & |R_{j_1(1)}| \end{array} \tag{18}$$

The details are omitted due to their similarity with Step 1a. Hence, by induction, the validity of equation (11) has been proven.

One important byproduct of the proof of equation (11) is that the input patterns from the training list that chose and were coded by node $j_1(1)$ in the second list presentation will always choose and be coded by node $j_1(1)$ in subsequent list presentations (i.e., list presentations ≥ 3). This is true because if pattern **I**, from the training list, chose node $j_1(1)$ in the second list presentation it implies that

$$|R_{j_1(1)}| < |R_{j_x}^{new}| \tag{19}$$

for any $j_x \neq j_1(1)$. Since the size of $R_{j_1(1)}$ remains intact after the first list presentation, while the sizes of other rectangles can increase, it is obvious that the above inequality stays valid in presentations of the input pattern **I** at subsequent lists. Thus, input pattern **I** will always choose and be coded by node $j_1(1)$ in list presentations ≥ 3, and rectangle $R_{j_1(1)}$ will be the rectangle of the smallest size that contains **I**.

Step 2:

Assuming now the validity of equation (12), the truth of equation (13) can be demonstrated. As was the case with the proof of equation (11) an important byproduct of the assumption of the validity of equation (12) is that input patterns that chose and were coded by node $j_k(k)$ in list presentation $k+1$, will choose and be coded by node $j_k(k)$

in list presentations $\geq k+2$, where $1 \leq k \leq n$. To demonstrate the validity of Step 2 two steps are needed:

- **Step 2a:** Verify the validity of equation (13) for list presentation $t = n+2$.
- **Step 2b:** Assume the validity of equation (13) for all list presentations $\leq t$ (where $t \geq n+2$) and show the validity of equation (13) for list presentation $t+1$.

Step 2a:

To illustrate Step 2a it will first be shown that the identity of the node with the $(n+1)$-th largest rectangle (i.e., $j_{n+1}(n+1)$) and the size of the largest rectangle (i.e., $|R_{j_{n+1}(n+1)}|$) stay intact after the presentation of the first input pattern in list presentation $n+2$ (i.e., pattern $\mathbf{I}^1$). Then, by assuming that the identity of the node with the $(n+1)$-th largest rectangle and the size of the $(n+1)$-th largest rectangle stay intact after the presentation of the first p input patterns (i.e., patterns $\mathbf{I}^1$, $\mathbf{I}^2$, ..., $\mathbf{I}^p$) in list presentation $n+2$, it will be proven that the identity of the node with the $(n+1)$-th largest rectangle and the size of the $(n+1)$-th largest rectangle stay intact after the presentation of the $(p+1)$-th input pattern (i.e., pattern $\mathbf{I}^{p+1}$) in list presentation $n+2$. Hence, by induction, it can then be stated that the identity of the node with the $(n+1)$-th largest rectangle (i.e., $j_{n+1}(n+1)$) and the size of the $(n+1)$-th largest rectangle (i.e., $|R_{j_{n+1}(n+1)}|$) stay intact after the presentation of the P training input patterns in list presentation $n+2$, which is equivalent to saying that equation (13) is true for list presentation $t = n+2$.

As a result, consider the presentation of the first input pattern $\mathbf{I}^1$, from the training list, during the $(n+2)$ list presentation of the training list. The following cases are distinguished.

Case 0:

Rectangle R_{j_x} $(= R_{j_x}^{old})$, where $j_x \in \{j_1(1), j_2(2), \ldots, j_n(n)\}$, is the smallest rectangle that contains $\mathbf{I}^1$. For example, if $j_x = j_2(2)$, this means that $R_{j_2(2)}$ was still the smallest rectangle that contained $\mathbf{I}^1$ in list presentation 3, since $R_{j_2(2)}$ did not change its size, while other rectangles have either increased their sizes or kept their sizes intact since the beginning of list presentation 3; it also means that $\mathbf{I}^1$ chose and was coded by rectangle $R_{j_2(2)}$ in list presentation 3, otherwise $R_{j_2(2)}$ would not have been the smallest rectangle that contains $\mathbf{I}^1$ in list presentation $n+2$. Based on our previous statements (see comments immediately after Step 2), $\mathbf{I}^1$ will always choose and be coded by $R_{j(2)}$ in list presentations ≥ 3. Similar reasoning holds if j_x is equal to $j_1(1)$ or $j_3(3)$, or $\ldots, j_n(n)$. Hence, in this case, pattern $\mathbf{I}^1$ chooses and is coded by rectangle R_{j_x}. Consequently, after $\mathbf{I}^1$'s presentation, node $j_{n+1}(n+1)$ is the node with the $(n+1)$-th largest rectangle and the size of the $(n+1)$-th largest rectangle stays intact and equal to $|R_{j_{n+1}(n+1)}|$.

Case 1:

Rectangle $R_{j_{n+1}(n+1)}$ $(= R^{old}_{j_{n+1}(n+1)})$ is the rectangle of the smallest size that contains $\mathbf{I}^1$, and pattern $\mathbf{I}^1$ chooses and is coded by rectangle $R_{j_{n+1}(n+1)}$. In this case, after $\mathbf{I}^1$'s presentation, node $j_{n+1}(n+1)$ is the node with the $(n+1)$-th largest rectangle and the size of the $(n+1)$-th largest rectangle stays intact and equal to $|R_{j_{n+1}(n+1)}|$.

Case 2:

Rectangle $R_{j_{n+1}(n+1)}$ $(= R^{old}_{j_{n+1}(n+1)})$ is the rectangle of the smallest size that contains $\mathbf{I}^1$, and pattern $\mathbf{I}^1$ chooses and is coded by rectangle R_{j_x} $(= R^{old}_{j_x})$, where $j_x \neq j_{n+1}(n+1)$. Due to Result B, in order for the above case to occur the following inequality must be valid:

$$|R^{new}_{j_x}| < |R_{j_{n+1}(n+1)}| \tag{20}$$

The above inequality guarantees that after $\mathbf{I}^1$'s presentation, node $j_{n+1}(n+1)$ is the node with the $(n+1)$-th largest rectangle and the size of the $(n+1)$-th largest rectangle stays intact and equal to $|R_{j_{n+1}(n+1)}|$.

Case 3:

Rectangle R_{j_x} $(= R^{old}_{j_x})$, where $j_x \neq j_k(k)$ $;1 \leq k \leq n+1$, is the rectangle of the smallest size that contains $\mathbf{I}^1$, and pattern $\mathbf{I}^1$ chooses and is coded by rectangle R_{j_x}. In this case, after $\mathbf{I}^1$'s presentation, node $j_{n+1}(n+1)$ is the node with the $(n+1)$-th largest rectangle and the size of the $(n+1)$-th largest rectangle stays intact and equal to $|R_{j_{n+1}(n+1)}|$.

Case 4:

Rectangle R_{j_x} $(= R^{old}_{j_x})$, where $j_x \neq j_k(k)$ $;1 \leq k \leq n+1$, is the rectangle of the smallest size that contains $\mathbf{I}^1$, and pattern $\mathbf{I}^1$ chooses and is coded by rectangle R_{j_y} $(= R_{j_y}^{old})$, where $j_y \neq j_x$. Obviously, due to Result B, j_y cannot be node $j_k(k)$, where $1 \leq k \leq n+1$. Also, due to Result B, in order for the above case to occur the following inequality must be valid:

$$|R_{j_y}^{new}| < |R_{j_x}| \tag{21}$$

But,

$$|R_{j_x}| < |R_{j_{n+1}(n+1)}| \tag{22}$$

The above two inequalities guarantee that after $\mathbf{I}^1$'s presentation, node $j_{n+1}(n+1)$ is the node with the $(n+1)$-th largest rectangle and the size of the $(n+1)$-th largest rectangle stays intact and equal to $|R_{j_{n+1}(n+1)}|$.

Cases 0-4 cover all the possible scenarios, and they illustrate that during the presentation of the first pattern $\mathbf{I}^1$ in list presentation $n+2$, the identity of the node with the $(n+1)$-th largest rectangle and the size of this rectangle remain intact.

If it is now assumed that after $\mathbf{I}^1$'s, $\mathbf{I}^2$'s, $\ldots$, $\mathbf{I}^p$'s presentations, node $j_{n+1}(n+1)$ is the node with the $(n+1)$-th largest rectangle and the size of the $n+1$-th largest rectangle stays intact and equal to $|R_{j_{n+1}(n+1)}|$, it is easy to duplicate the above arguments to illustrate that after pattern's $\mathbf{I}^{p+1}$ presentation node $j_{n+1}(n+1)$ is the node with the $(n+1)$-th largest rectangle and the size of the $(n+1)$-th largest rectangle stays intact and equal to $|R_{j_{n+1}(n+1)}|$. Hence, by induction, equation (13) has been proven for list presentation $t = n+2$.

Step 2b:

If it is now assumed that

$$\begin{array}{rcl} j_{n+1}(n+2) & = & j_{n+1}(n+1) \\ |R_{j_{n+1}(n+2)}| & = & |R_{j_{n+1}(n+1)}| \\ \vdots & \vdots & \vdots \\ j_{n+1}(t) & = & j_{n+1}(n+1) \\ |R_{j_{n+1}(t)}| & = & |R_{j_{n+1}(n+1)}| \end{array} \tag{23}$$

then by duplicating the procedure discussed in Step 2a, it can be demonstrated that

$$\begin{array}{rcl} j_{n+1}(t+1) & = & j_{n+1}(n+1) \\ |R_{j_{n+1}(t+1)}| & = & |R_{j_{n+1}(n+1)}| \end{array} \tag{24}$$

The details are omitted due to their similarity with Step 2a. Hence, by induction, the validity of equation (13) has been proven.

Proof of Result 2:

The above result was actually proved during the proof of Result 1 (see comments in the proof of Result 1 immediately prior, and immediately after the beginning of the proof of Step 2).

Proof of Result 3:

The proof of this result is an immediate consequence of Result 1. This is true because the application of Result 1, at the end of list presentations $1,\ 2,\ 3, \ldots, N$ implies that the sizes of the N largest rectangles do not change after list presentation N. If none of the N rectangles change after list presentation N, then none of the N weight vectors changes after list presentation N, which is equivalent to saying that *learning in the Fuzzy ART Variant* is over in at most N list presentations.

4.3 Example

One of the interesting questions that arises, pertinent to Results 1 and 3, is the tightness of the results. This is especially important for Result 3, which gives us an upper bound on the number of list presentations required by the Fuzzy ART Variant to cluster a list of input patterns. Below, we present a simple example that illustrates the tightness of Results 1 and 3.

In this example we have eight input patterns, listed below, that are presented to the Fuzzy ART Variant in the order $\mathbf{I}^1, \mathbf{I}^2, \ldots, \mathbf{I}^7, \mathbf{I}^8$ in the first list presentation, and in the order $\mathbf{I}^8, \mathbf{I}^7, \ldots, \mathbf{I}^2, \mathbf{I}^1$ in list presentations 2 and 3. The vigilance parameter ρ is chosen equal to 0.55.

$$\begin{array}{lll} \mathbf{I}^1 & = (\mathbf{a}(1), \mathbf{a}^c(1)) & = (1,\ 1,\ 0,\ 0) \\ \mathbf{I}^2 & = (\mathbf{a}(2), \mathbf{a}^c(2)) & = (0.55,\ 0.55,\ 0.45,\ 0.45) \\ \mathbf{I}^3 & = (\mathbf{a}(3), \mathbf{a}^c(3)) & = (0.7,\ 0.7,\ 0.3,\ 0.3) \\ \mathbf{I}^4 & = (\mathbf{a}(4), \mathbf{a}^c(4)) & = (0.3,\ 0.3,\ 0.7,\ 0.7) \\ \mathbf{I}^5 & = (\mathbf{a}(5), \mathbf{a}^c(5)) & = (0.5,\ 0.5,\ 0.5,\ 0.5) \\ \mathbf{I}^6 & = (\mathbf{a}(6), \mathbf{a}^c(6)) & = (0,\ 0,\ 1,\ 1) \\ \mathbf{I}^7 & = (\mathbf{a}(7), \mathbf{a}^c(7)) & = (0.35,\ 0.35,\ 0.65,\ 0.65) \\ \mathbf{I}^8 & = (\mathbf{a}(8), \mathbf{a}^c(8)) & = (0.24,\ 0.24,\ 0.76,\ 0.76) \end{array} \tag{25}$$

It is not difficult to show that

1. After the first list presentation we create three categories (i.e., $N = 3$), where $j_1(1) = 1$, $j_2(1) = 3$, $j_3(1) = 2$, and $|R_{j_1(1)}| = 0.9$, $|R_{j_2(1)}| = 0.48$, $|R_{j_3(1)}| = 0.4$.

2. After the second list presentation $j_1(2) = 1$, $j_2(2) = 2$, $j_3(2) = 3$, and $|R_{j_1(2)}| = 0.9$, $|R_{j_2(2)}| = 0.8$, $|R_{j_3(2)}| = 0.48$ Hence, in the second list presentation the identity and the size of the second and third in size rectangles change (as predicted by Result 1).

3. After the third list presentation $j_1(3) = 1$, $j_2(3) = 2$, $j_3(3) = 3$, and $|R_{j_1(3)}| = 0.9$, $|R_{j_2(3)}| = 0.8$, $|R_{j_3(3)}| = 0.7$ Hence, in the third list presentation the size of the third in size rectangle changes (as predicted by Result 1).

4. In subsequent list presentations (i.e., list presentations ≥ 4) the identity and size of all the rectangles remain intact, which implies that learning in the Fuzzy ART Variant is over in three list presentations (as predicted by Result 3).

The rectangles created in this example after list presentations 1, 2, and 3 are shown in Figure 5. In Figure 5, we see in a pictorial fashion the tightness of Results 1 and 3.

5 Simulations

In order to assess the practicality of the Fuzzy ART Variant (Fuzzy ART with large values of the choice parameter) compared to typical Fuzzy ART (Fuzzy ART with small values of the choice parameter) we evaluated the performance of these algorithms on two databases. These databases were chosen from the collection of databases found at the UCI repository Murphy and Aha (1994). The databases chosen were: Iris and Glass. The Iris database is perhaps the best known database to be found in the pattern recognition literature. The data set consists of three classes of 50 instances each, where each class refers to an iris plant (Iris Setosa, Iris Versicolour and Iris Virginica). The number of features for each instance are the sepal length, the sepal width, the petal length, and the petal width, all in cm. For the training of Fuzzy ART (Fuzzy ART Variant) the $P = 150$ input patterns are used, where each input pattern has $M = 4$ components. The glass database is used for classification of some type of glass. This database was motivated by criminological investigation, where the glass left at the crime scene, can be used as evidence. Each instance has 9 features and it can be classified as one of 6 classes. There are 214 instances of input/output pairs. For the training of Fuzzy ART (Fuzzy ART Variant) the $P = 214$ input patterns are used, where each input pattern has $M = 9$ components.

To evaluate the clustering performance of Fuzzy ART (Fuzzy ART Variant) we trained it with the training list of input patterns until it learned the list completely. After training was over we assigned a label to each category formed in the F_2 field. A category formed in the F_2 field takes the label of the output pattern to which most of the input patterns that chose this category belong. The input patterns from the training list that choose a category and their output pattern (label) matches the output pattern (label) of this category, belong to the group of correctly clustered input patterns. The input patterns from the training list that choose a category and their output pattern (label) does not match the output pattern (label) of this category, belong to the group of not correctly clustered input patterns. The clustering performance of Fuzzy ART (Fuzzy ART Variant) is defined to be the percentage of training input patterns that are correctly clustered, according to the above described procedure. For example, suppose that 10 patterns belong to the training list and patterns $\mathbf{I}^1$ through $\mathbf{I}^6$ are to be mapped to output pattern $\mathbf{O}^1$, while input patterns $\mathbf{I}^7$ through $\mathbf{I}^{10}$ are to be mapped to output pattern $\mathbf{O}^2$. Furthermore, assume that after training of Fuzzy ART is completed two categories are formed in the field F_2. Finally, assume that patterns $\mathbf{I}^1$ through $\mathbf{I}^4$ and pattern $\mathbf{I}^7$ choose category 1, while patterns $\mathbf{I}^5$ and $\mathbf{I}^6$, and patterns $\mathbf{I}^8$ through $\mathbf{I}^{10}$ choose category 2. Hence the label of category 1 is output pattern $\mathbf{O}^1$, and the label of category 2 is output pattern $\mathbf{O}^2$. As a result, patterns $\mathbf{I}^1$ through $\mathbf{I}^4$ and patterns $\mathbf{I}^8$ through $\mathbf{I}^{10}$ are correctly clustered (they choose categories with labels that are equal to the output patterns that they need to be mapped to); the rest of the input patterns are not correctly clustered. The aforementioned procedure to evaluate the clustering performance of clustering algorithms was initially introduced by Dubes and Jain (1976).

The average clustering performance (Avg_{cl}) of Fuzzy ART (Fuzzy ART Variant) is depicted in Table 1. The average clustering performance of Fuzzy ART (Fuzzy ART Variant) is calculated by evaluating the average of the clustering performances of ten networks trained by Fuzzy ART (Fuzzy ART Variant) for different orders of training pattern presentations. In Table 1, other measures of performance are depicted such as standard deviation of the clustering performances (Std_{cl}), as well as average number of categories formed in F_2 (Avg_{cat}), and standard deviation of the number of categories formed in F_2 (Std_{cat}). Our conclusion by comparing the results in Table 1 (i.e., Fuzzy ART performance for small α values, and Fuzzy ART performance for large α values (Fuzzy ART Variant)) is that Fuzzy ART Variant performance is close to that of Fuzzy ART that uses small α values. Similar conclusions were drawn in (Georgiopoulos, Fernlund, Bebis, and Heileman, 1996) where the Fuzzy ART and the Fuzzy ART Variant clustering performances were compared on additional databases (e.g., Heart, Diabetes, Wine, Ionosphere, Sonar) from the UCI repository (Murphy and Aha, 1994).

6 Summary–Discussion

In this paper attention was focused on a Fuzzy ART Variant that is obtained if in Fuzzy ART we use very large values for the choice parameter α, and the initial components of the weights. Useful properties of learning were developed and proved for this Fuzzy ART Variant. One of these properties of learning provided an upper bound on the number of list presentations required by this Fuzzy ART Variant to learn an arbitrary list of analog input patterns. This upper bound verified one of the important properties of this Fuzzy ART Variant, the short-training time property. Knowing that properties of learning for Fuzzy ART with small values for the choice parameters (including the short-training time property) have already been demonstrated in the literature, this paper fills an important theoretical gap. What remains to be seen is whether the short training time property is valid in Fuzzy ART for intermediate values of the choice parameter (i.e., not too small and not too large values). We have also illustrated through simulations that this Fuzzy ART Variant exhibits a clustering performance that is comparable to the Fuzzy ART (with small choice parameter values) clustering performance. Hence, it can be used in practice instead of, or in conjunction with Fuzzy ART.

References

Carpenter, G. A. and Gjaja, M. N. (1994). Fuzzy Art choice functions. In *Proceedings of the World Congress on Neural Networks*, pages I–713–722, San Diego, CA.

Carpenter, G. A. and Grossberg, S. (1987a). ART 2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, 26(23):4919–4930.

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

Carpenter, G. A. and Grossberg, S. (1990). ART 3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks*, 3(2):129–152.

Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., and 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.

Carpenter, G. A., Grossberg, S., and Reynolds, J. H. (1991a). ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks*, 4(5):565–588.

Carpenter, G. A., Grossberg, S., and Rosen, D. B. (1991b). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, 4(6):759–771.

Carpenter, G. A. and Markuzon, N. (1998). ARTMAP-IC and medical diagnosis: Instance counting and incosistent cases. *Neural Networks*, 11:323–326.

Carpenter, G. A. and Ross, W. D. (1995). ART-EMAP: A neural architecture for object recognition by evidence accumulation. *IEEE Transactions on Neural Networks*, 6:805–818.

Dubes, R. and Jain, A. (1976). Clustering techniques: The user's dilemma. *Pattern Recognition*, 8:247–260.

Georgiopoulos, M., Fernlund, H., Bebis, G., and Heileman, G. L. (1996). Order of search in Fuzzy ART and Fuzzy ARTMAP: Effect of the choice parameter. *Neural Networks*, 9(5):1541–1559.

Grossberg, S. (1976). Adaptive pattern recognition and universal recoding II: Feedback, expectation, olfaction, and illusions. *Biological Cybernetics*, 23:187–202.

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

Huang, J., Georgiopoulos, M., and Heileman, G. L. (1995). Fuzzy ART properties. *Neural Networks*, 8(2):203–213.

Marriott, S. and Harrison, R. (1995). A modified Fuzzy ARTMAP architecture for the approximation of noisy mappings. *Neural Networks*, 8(4):619–641.

Murphy, P. and Aha, D. (1994). UCI Repository of Machine Learning Databases. Technical report, Department of Computer Science, http://www.ics.edu/mlearn/MLRepository.html, University of California, Irvine, CA.

Tan, A. H. (1995). Adaptive resonance associative map. *Neural Networks*, 8(3):437–446.

Williamson, J. R. (1996). Gaussian ARTMAP: A neural network for fast incremental learning of noisy multi-dimensional maps. *Neural Networks*, 9(5):881–897.

Table 1: Comparisons of the clustering performances of Fuzzy ART (small choice parameter values α) and Fuzzy ART Variant ($\alpha \to \infty$)

Network Parameters		**Iris Database**				**Glass Database**			
α	ρ	Avg_{cl}	Std_{cl}	Avg_{cat}	Std_{cat}	Avg_{cl}	Std_{cl}	Avg_{cat}	Std_{cat}
0.01	0.4	74.40	7.27	5	0	44.11	4.20	6	0
0.1	0.4	76.33	8.92	5	0	43.73	3.51	6	0
1.0	0.4	77.06	5.16	4	1	43.41	3.81	6	0
∞	0.4	74.00	4.30	6	1	43.69	3.77	7	1
0.01	0.6	87.00	3.14	7	0	54.90	4.31	14	1
0.1	0.6	86.93	4.04	7	0	55.41	4.43	14	1
1.0	0.6	87.19	4.88	8	0	54.15	2.52	14	0
∞	0.6	86.13	3.57	8	1	51.35	1.98	16	1
0.01	0.8	93.40	2.03	22	1	63.22	2.26	31	1
0.1	0.8	93.46	2.06	22	1	63.17	2.36	31	1
1.0	0.8	94.13	2.10	22	1	61.49	3.31	31	2
∞	0.8	93.66	1.20	24	1	64.43	3.17	34	2

Figure Captions

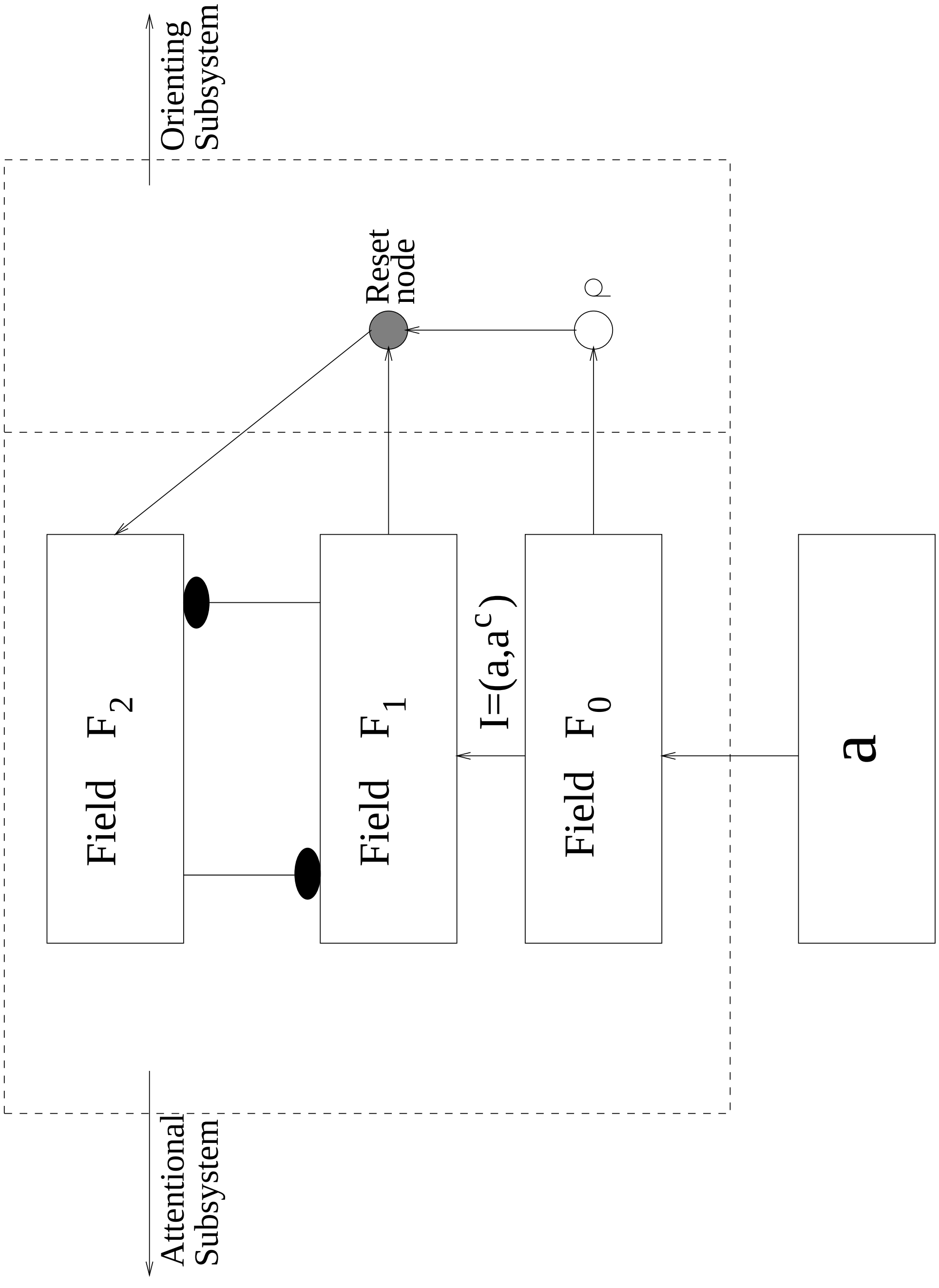
Figure 1: A block diagram of the Fuzzy ART architecture.

Figure 2: The hyperrectangle (rectangle) which has coded the input patterns $\mathbf{I}^1 = (\mathbf{a}^1, (\mathbf{a}^1)^c)$, $\mathbf{I}^2 = (\mathbf{a}^2, (\mathbf{a}^2)^c)$, $\mathbf{I}^3 = (\mathbf{a}^3, (\mathbf{a}^3)^c)$, $\mathbf{I}^4 = (\mathbf{a}^4, (\mathbf{a}^4)^c)$, and $\mathbf{I}^5 = (\mathbf{a}^5, (\mathbf{a}^5)^c)$.

Figure 3: Illustration of the Order-of-Search properties (Result A and B) for Fuzzy ART (small values of the choice parameter) and Fuzzy ART Variant. The left hand side plots correspond to Fuzzy ART and the right-hand side plots correspond to Fuzzy ART Variant. **(a):** Pattern **I** is inside rectangles $R_{j_1}^{old}$, and $R_{j_2}^{old}$, where $|R_{j_1}^{old}| < |R_{j_2}^{old}|$; the "+" signs indicate patterns **I** that choose first rectangle $R_{j_1}^{old}$ and the "-" signs indicate patterns **I** that choose first rectangle $R_{j_2}^{old}$. **(b):** Pattern **I** is outside rectangle $R_{j_1}^{old}$, and inside rectangle $R_{j_2}^{old}$; the "+" signs indicate patterns **I** that choose first rectangle $R_{j_1}^{old}$ and the "-" signs indicate patterns **I** that choose first rectangle $R_{j_2}^{old}$. **(c):** Pattern **I** is outside rectangles $R_{j_1}^{old}$ and $R_{j_2}^{old}$; the "+" signs indicate patterns **I** that choose first rectangle $R_{j_1}^{old}$ and the "-" signs indicate patterns **I** that choose first rectangle $R_{j_2}^{old}$.

Figure 4: Illustration that the presentation of pattern $\mathbf{I}^1$ in the second list presentation does not change the identity or size of rectangle $R_{j_1(1)}$. *Case 1:* $\mathbf{I}^1$ is inside $R_{j_1(1)}$ and is coded by $R_{j_1(1)}$. *Case 2:* $\mathbf{I}^1$ is inside $R_{j_1(1)}$ and is coded by R_{j_x} ($|R_{j_x}^{new}| < |R_{j_1(1)}|$). *Case 3:* $\mathbf{I}^1$ is inside R_{j_x} and is coded by R_{j_x} ($|R_{j_x}| < |R_{j_1(1)}|$). *Case 4:* $\mathbf{I}^1$ is inside R_{j_x} and is coded by R_{j_y} ($|R_{j_y}^{new}| < |R_{j_x}| < |R_{j_1(1)}|$).

Figure 5: Illustration of the tightness of Results 1 and 3. Rectangles created in the first three list presentations of the data $\mathbf{I}^1 \ldots \mathbf{I}^8$ to Fuzzy ART Variant. Data were presented from $\mathbf{I}^1$ to $\mathbf{I}^8$ in the first list presentation and from $\mathbf{I}^8$ to $\mathbf{I}^1$ in the second and third list presentation. The identity and the size of the second and the third in size rectangles change in the second list presentation (tightness of Result 1). Also, the size of the third in size rectangle changes in the third list presentation (tightness of Result 1). Finally, it takes three list presentations to learn the data (tightness of Result 3).



Orienting Subsystem
Reset node
ρ
Field F_2
Field F_1
I=(a,a^c)
Field F_0
a
Attentional Subsystem

Figure 1:

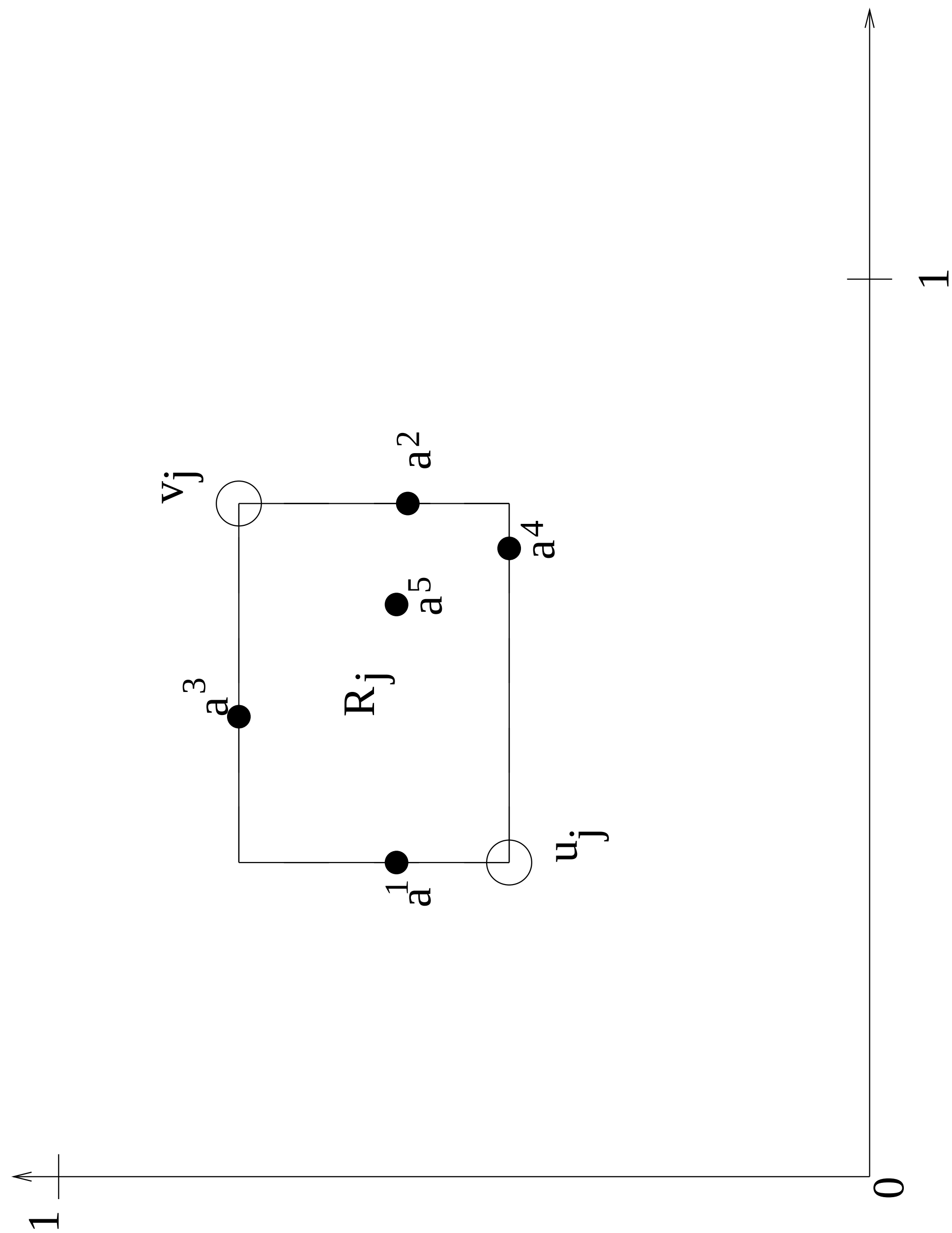


Figure 2:

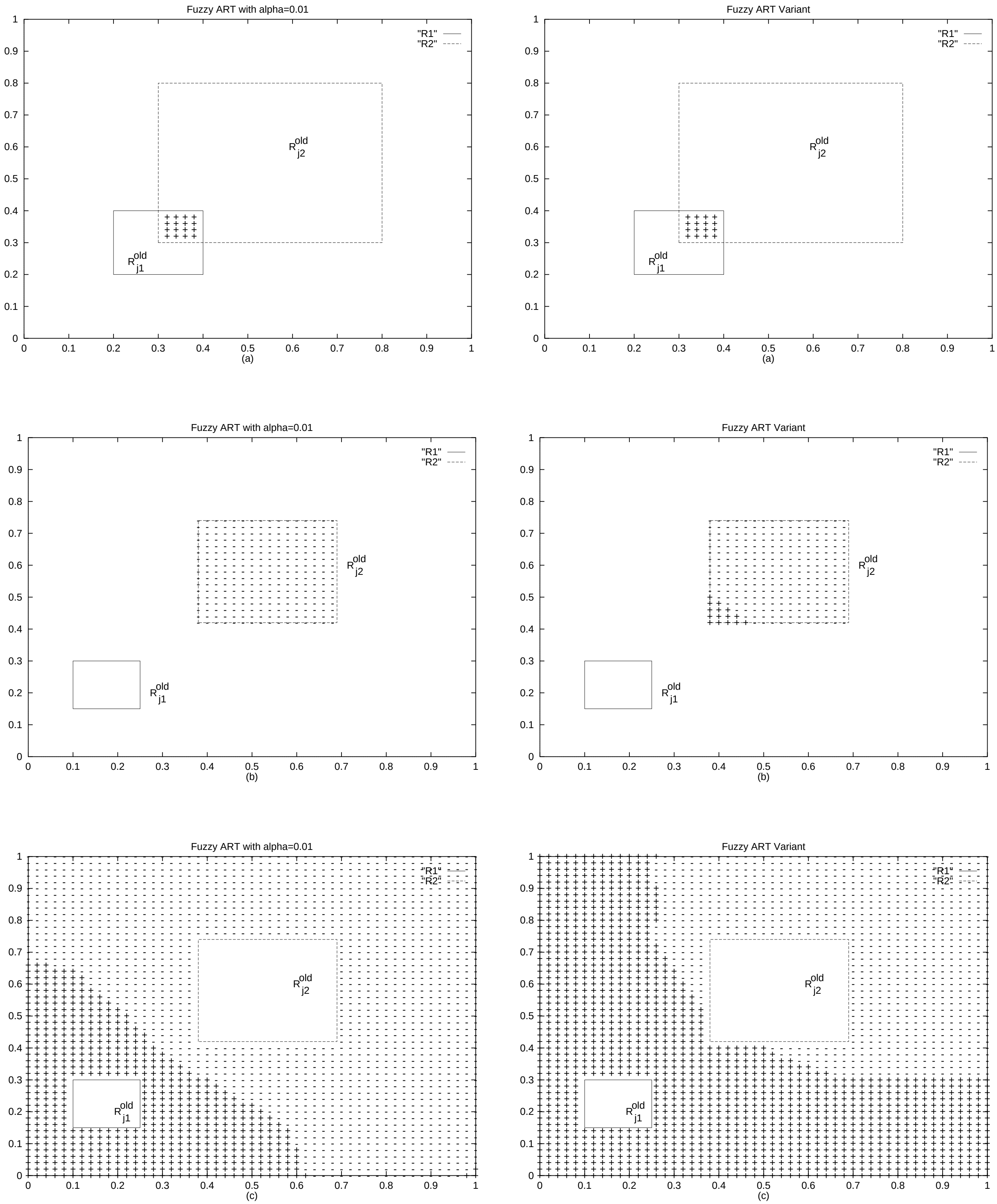

Fuzzy ART with alpha=0.01
"R1"
"R2"
R$^{old}_{j2}$
R$^{old}_{j1}$
(a)
Fuzzy ART Variant
"R1"
"R2"
R$^{old}_{j2}$
R$^{old}_{j1}$
(a)
Fuzzy ART with alpha=0.01
"R1"
"R2"
R$^{old}_{j2}$
R$^{old}_{j1}$
(b)
Fuzzy ART Variant
"R1"
"R2"
R$^{old}_{j2}$
R$^{old}_{j1}$
(b)
Fuzzy ART with alpha=0.01
"R1"
"R2"
R$^{old}_{j2}$
R$^{old}_{j1}$
(c)
Fuzzy ART Variant
"R1"
"R2"
R$^{old}_{j2}$
R$^{old}_{j1}$
(c)

Figure 3:

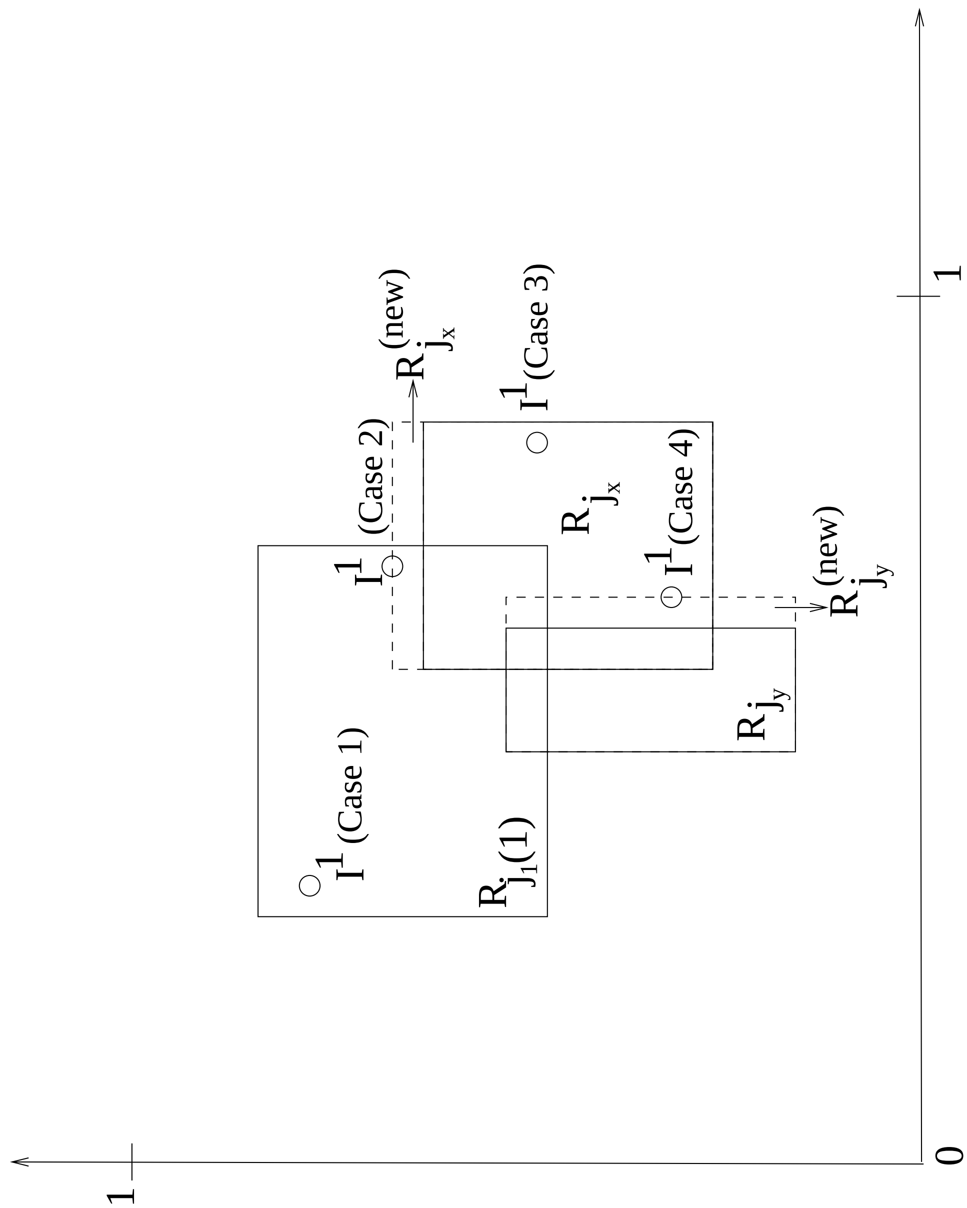

0
1
1
$R_{j_1}(1)$
I^1 (Case 1)
I^1 (Case 2)
I^1 (Case 3)
I^1 (Case 4)
R_{j_x}
R_{j_y}
$R_{j_x}^{(new)}$
$R_{j_y}^{(new)}$

Figure 4:

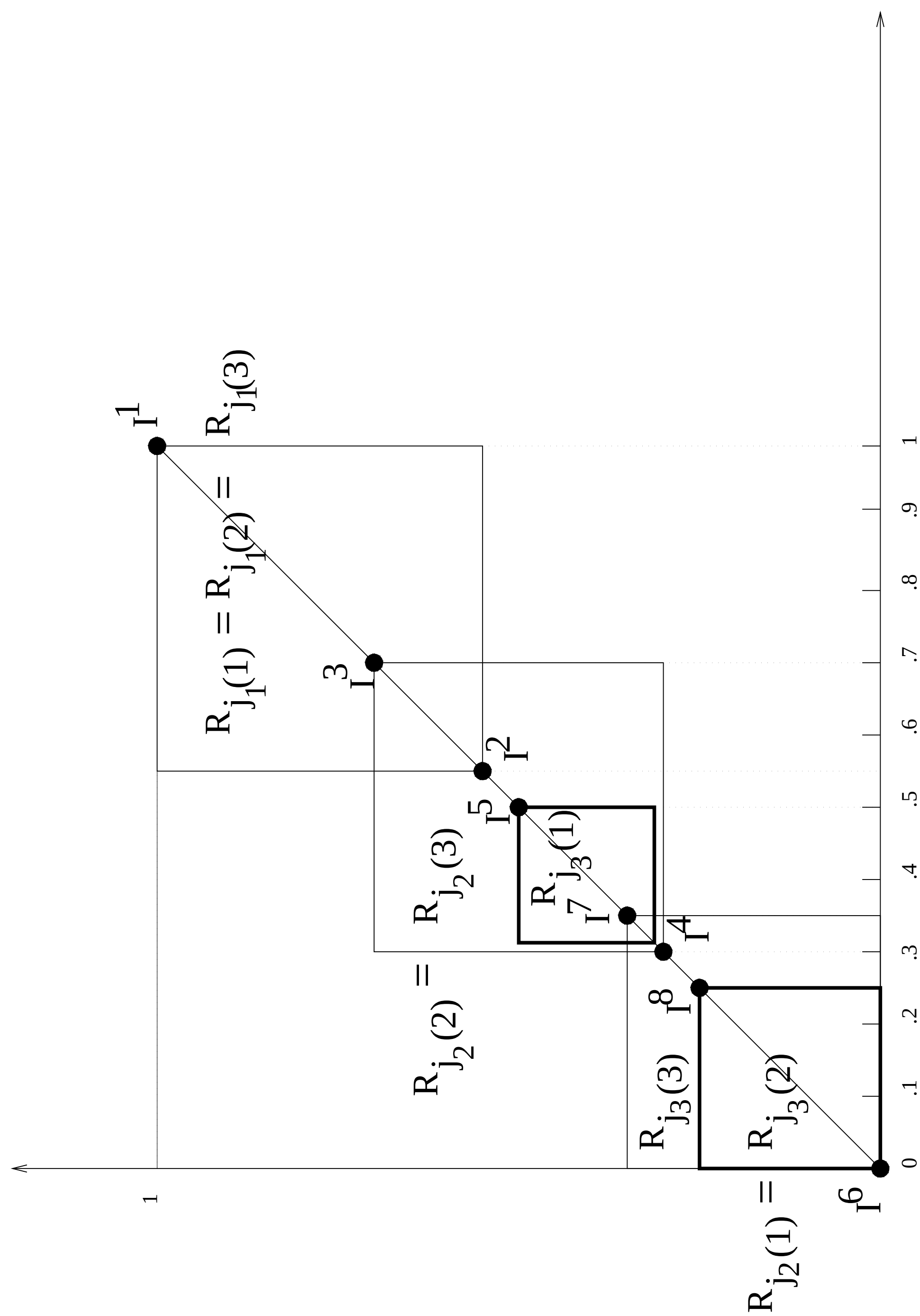

I1
I2
I3
I4
I5
I6
I7
I8
Rj1(1) = Rj1(2) = Rj1(3)
Rj2(2) = Rj2(3)
Rj2(1) =
Rj3(1)
Rj3(2)
Rj3(3)
0
.1
.2
.3
.4
.5
.6
.7
.8
.9
1
1

Figure 5: