

## Short Paper

---

# Fuzzy ARTRON: A General-purpose Classifier Empowered by Fuzzy ART and Error Back-propagation Learning

CHENG-SEEN HO\* AND JIA-SHIANG CHOU

*Department of Electronic Engineering  
National Taiwan University of Science and Technology  
Taipei, Taiwan 106, R.O.C.  
\*E-mail: csho@et.ntust.edu.tw*

This paper introduces Fuzzy ARTRON as a general-purpose classifier that can do high-quality classification in continuous, discrete, linear, or nonlinear domains. The topology of Fuzzy ARTRON contains a fuzzy ART network, on which a perceptron layer is superimposed. The learning algorithms involve unsupervised ART learning and supervised error back-propagation learning. The former is used to auto-construct proper clusters through the fuzzy ART self-construction ability. This improves the convergence rate and alleviates the local minima problem usually associated with the error back-propagation learning network. The latter is used to dynamically associate clusters with proper classes via connection weight adjustment. This improves the generalization ability so that Fuzzy ARTRON can successfully handle the linearly nonseparable problems usually associated with fuzzy ART and the weak generalization problem usually associated with fuzzy ARTMAP. Finally, Fuzzy ARTRON employs fuzzy hyperboxes to do clustering, which leads to better generalization performance compared to conventional hyperboxes. Computer simulations were conducted to evaluate the performance and applicability of Fuzzy ARTRON under continuous, discrete, linear, or nonlinear domains.

*Keywords:* back-propagation learning, classifiers, fuzzy ART, neural network, fuzzy theory

## 1. INTRODUCTION

Artificial neural networks (ANNs) simulate the concepts developed through physiological modeling of the human brain in computational mechanics [19]. ANNs contain a set of highly interconnected processing elements that are constructed in a regular architecture and act in parallel. The overall behavior of an ANN exhibits the abilities of learning, recalling, and generalization from training patterns or data by adjusting the

---

Received July 19, 1999; revised May 25, 2000; accepted July 13, 2000.  
Communicated by Chuen-Tsai Sun.

connection weights inside the network. The model of an ANN includes three basic entities: functions of the processing elements, the network topology, and learning rules (methods used to store information in the network). For a multilayer feedforward network topology, the error back-propagation learning algorithm is one of the most widely used supervised learning methods. The algorithm can perform rather *high-quality generalization* by escaping from local minima provided that the number of hidden nodes is properly selected. However, because the error surface may contain lots of areas with shallow slopes in multiple dimensions, the algorithm tends to *converge rather slowly*. ART, a typical example of an unsupervised learning algorithm for ANNs, is capable of rapid unsupervised learning of categories. However, it cannot properly deal with linearly nonseparable problems.

Fuzzy sets were formally introduced by Zadeh in 1965 as a mathematical method to handle uncertain or ambiguous data encountered in real life. A fuzzy set allows the degree of membership to be associated with each element of the universe of discourse. Thus, the membership function of a fuzzy set maps each element to its range space, which, in most cases, is set to the unit interval  $[0, 1]$ . As pointed out by Zadeh, any field can be fuzzified and, thus, enjoy greater generality, higher expressive power, enhanced ability to model real-world problems, and the benefits of a methodology to deal with imprecision. Fuzzy ART [1] is a well known example of fuzzified ART. Fuzzy ART is able to do *rapid*, unsupervised fuzzy classification by *self-constructing* the cluster layer. It, however, still suffers from the *linear nonseparability* problem. The latest version of the fuzzy ART series, called fuzzy ARTMAP, is capable of doing incremental supervised learning of categories or multidimensional maps, given arbitrary sequences of *continuous* or *binary* input feature vectors [15]. Fuzzy ARTMAP is also able to deal with *linearly nonseparable* problems. However, its *generalization ability is limited* given discrete input data because only a simple map field is used to associate hyperboxes with classes.

ANNs and fuzzy theory are complementary technologies in the design of intelligent systems. Each one has its merits and drawbacks. Integration and synthesis approaches have proliferated in the literature [4, 5]. The basic idea is to get the benefits of both ANNs and fuzzy systems. For example, ANNs offer learning abilities [6, 14, 16] while fuzzy systems offer a structural framework to reason with high-level fuzzy rules [7, 17, 18, 22, 23]. Integrating the two technologies in a single system thus allows them to complement each other and achieve better performance. In this paper, we explore whether we can combine fuzzy ART and back-propagation learning networks in a single system so that we can do high-quality fuzzy classification and generalization without such difficulties as slow convergence, local minima, or weak generalization under continuous, discrete, linear, or nonlinear domains.

Basically, we propose to superimpose a simple perceptron layer on a fuzzy ART architecture, thus called Fuzzy ARTRON, to work as a general purpose classifier. Fuzzy ARTRON employs both unsupervised fuzzy ART learning and supervised error back-propagation learning to train the classifier. The former's self-construction ability can auto-construct proper clusters, which speeds up the convergence rate and alleviates the local minima problem usually associated with an error back-propagation learning network. The latter's connection weight adjustment ability can dynamically associate clusters with proper classes, which can successfully deal with linearly nonseparable problems usually associated with fuzzy ART as well as weak generalization ability usu-

ally associated with the fuzzy ARTMAP. In addition, Fuzzy ARTRON employs fuzzy hyperboxes to do clustering, which leads to better generalization performance compared to conventional crisp hyperboxes. We have conducted a series of experiments to evaluate the generalization and classification capabilities of Fuzzy ARTRON under continuous, discrete, linear, or nonlinear domains.

The rest of the paper is organized as follows. Section 2 describes the topology of Fuzzy ARTRON. Section 3 details the learning algorithms of Fuzzy ARTRON. Section 4 reports the results of three experiments conducted to evaluate Fuzzy ARTRON's performance and concludes the work.

### 2. SYSTEM ARCHITECTURE

Fuzzy ARTRON has four layers as illustrated in Fig. 1. Layer 1 contains *input nodes* responsible for receiving input feature vectors. They are used to hold the input feature values and distribute them to their respective fuzzy nodes. Each node in layer 2 is a *fuzzy node* that works as a membership function to measure the membership degree of an input feature. Note that a feature's value may be located in a dimension of one or more fuzzy hyperboxes [14] because fuzzy hyperboxes may overlap. Each node in layer 3 is a *cluster node* representing a fuzzy hyperbox defined by the connections of the layer 2 nodes. Layer 4 contains the conventional output nodes of a multilayer feedforward neural network. Here, they are used as *class nodes* representing the target classes of the input feature vectors. We want each input feature vector to produce an output value from its desired class node that is higher than those produced by the others.

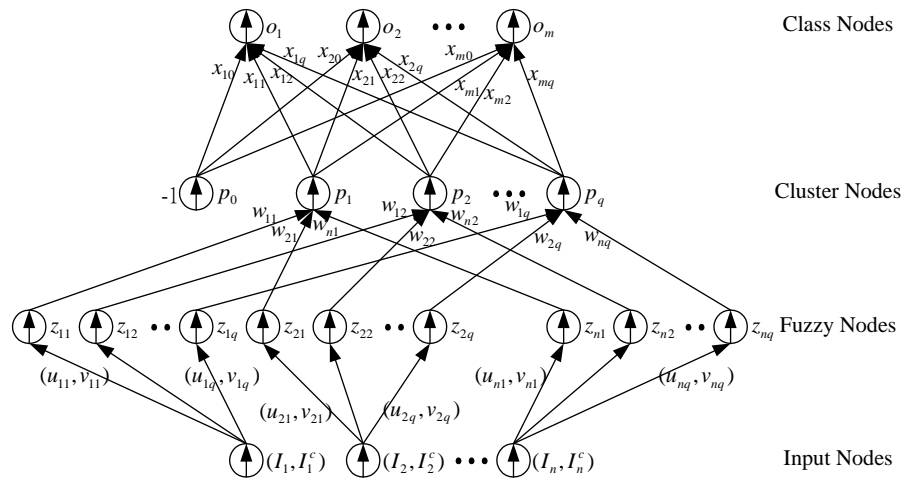


Fig. 1. Fuzzy ARTRON.

The significance of this design can be explained from two viewpoints. On the one hand, the architecture can be thought of as a fuzzy ART network as described in [1], on

which a perceptron layer is superimposed. On the other hand, it looks like a multi-layer feed-forward neural network in which the hidden layer is replaced with a fuzzy ART network. Now, from the first viewpoint, the perceptron's weight adjustment capability allows dynamic association of relevant clusters with proper classes. This, in fact, can support the creation of hyper-spaces of any shape to approximate the given training set by flexibly associating relevant hyperboxes with the same hyper-space. This design, thus, solves the problem of inability to deal with linear non-separability problems associated with fuzzy ART networks. It, at the same time, improves the generalization ability of fuzzy ARTMAP. From the second viewpoint, the fast learning capability of fuzzy ART allows rapid self-construction of clusters. As a result, hidden nodes for the multi-layer feedforward neural network can be quickly generated, thus improving the convergence rate. Moreover, the adjustability of the vigilance parameter leads to a best decision on the number of the hidden nodes, which helps the multi-layer feedforward neural network escape from the local optima problem.

## 2.1 Input Feature Vectors

First, each component of an input feature vector is coded in the unit interval  $[0, 1]$ . Fuzzy ARTRON adopts the technique of complement coding from fuzzy ART [1] to achieve normalization of each input vector. Assuming the input vector is an  $n$ -dimensional vector,  $\mathbf{I} = (I_1, I_2, \dots, I_n)$ , the complement coding method will preprocess it into a  $2n$ -dimensional complement coding form,  $\mathbf{I}'$ . Formally,

$$\begin{aligned}\mathbf{I}' &= (I_1, I_1^c, I_2, I_2^c, \dots, I_n, I_n^c) \\ &= (I_1, 1-I_1, I_2, 1-I_2, \dots, I_n, 1-I_n),\end{aligned}$$

where  $I_i^c$  is the complement of  $I_i$ , that is,  $I_i^c = 1 - I_i$ . Thus, in Fuzzy ARTRON, all input feature vectors are converted to complement-coded vectors, which are then used for training. As mentioned in [1], the good thing about using complement coding is that category proliferation can be avoided during fuzzy clustering by fuzzy ART. Complement coding also preserves the amplitude information of the input vectors, thus leading to a symmetric theory in which the AND operator ( $\wedge$ ) and the OR operator ( $\vee$ ) of fuzzy theory play complementary roles.

## 2.2 Fuzzy Nodes

Each of the fuzzy nodes acts as a one-dimensional membership function to measure the degree of membership of its corresponding input feature. Fuzzy ARTRON adopts the trapezoidal membership function described in [21]. Eq. (1) gives the  $i$ th dimensional membership degree of the  $j$ th fuzzy hyperbox or, equivalently, the output of the  $ij$ th fuzzy node, i.e.,  $z_{ij}$ :

$$z_{ij} = f[I_i] = \{1 - g[I_i - v_{ij}, \lambda] - g[u_{ij} - I_i, \lambda]\}, \quad (1)$$

where  $(u_{ij}, v_{ij})$  is the pair of connection weights between the  $ij$ th fuzzy node and the  $i$ th

input node;  $I_i$  is the input feature value distributed from the  $i$ th input node; and

$$g(s, \gamma) = \begin{cases} 1, & \text{if } s\gamma > 1, \\ s\gamma, & \text{if } 0 \leq s\gamma \leq 1, \\ 0, & \text{if } s\gamma < 0, \end{cases}$$

where  $\gamma$  is the sensitivity parameter controlling the fuzziness of  $f[\cdot]$ . It regulates the fuzziness of one dimension of a fuzzy hyperbox. As  $\gamma$  becomes smaller, the fuzzy set becomes fuzzier.

**2.3 Cluster Nodes**

A set of  $n$  fuzzy nodes (each connected to an input node) is connected to a node in layer 3, forming an  $n$ -dimensional membership function, which in turn defines a fuzzy hyperbox or a fuzzy cluster in the input feature space. Each node in layer 3 is, thus, called a cluster node and represents a fuzzy hyperbox.

A cluster node is designed to be a neuron that realizes the standard fuzzy set operation of AND [17, 18] as shown in Fig. 2. It performs a min-max operation over its inputs,  $z_{ij}$ , and the connection weights between the cluster node and the fuzzy nodes, i.e.,  $w_{ij}$ 's. Eq. (2) gives the operation:

$$p_j = \min_i \max [w_{ij}, z_{ij}] \tag{2}$$

where  $p_j$  is the output of the  $j$ th cluster node. The weights,  $w_{ij}$ 's, reflect the relative importance of the inputs to a cluster node [8]. They may take on values in  $[0, 1]$  with the following two extreme cases:  $w_{ij} = 1$ , meaning the connection is broken, or the input is not important to the cluster node;  $w_{ij} = 0$ , meaning that the connection is the strongest one. The output value of a cluster node,  $p_j$ , thus represents the degree to which an input feature vector belongs to the fuzzy hyperbox  $j$ .

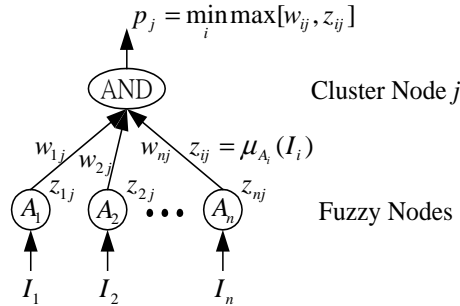


Fig. 2. A single cluster node and connections.

**2.4 Class Nodes**

The nodes in layer 4 are called class nodes, each representing a target class. Each class node associates those fuzzy clusters (fuzzy hyperboxes) that belong to the same

class together. Fuzzy ARTRON employs a conventional simple perceptron with linear graded units as the neuron structure in this layer. (A node with a linear integration function and a graded activation function is called a linear graded unit.) The following linear integration function  $f$  and unipolar sigmoid function  $a$  are used, respectively, to combine the outputs from the cluster nodes and to compute the output of a class node:

$$f_k = net_k = \sum_{j=1}^q x_{kj} p_j - \theta_k, \quad (3)$$

and

$$a(f) = \frac{1}{1 + e^{-\lambda f}}, \quad (4)$$

where  $\theta_k$  is the threshold of the  $k$ th class node,  $p_j$  is the output of the  $j$ th cluster node, and  $\lambda > 0$  determines the steepness of the activation function  $a(f)$  near  $f = 0$ . As usual, the threshold of each class node is treated as a weight connected to an input terminal permanently clamped at  $-1$  (Fig. 1). By applying the delta learning rule, this layer allows the application of a variation of the conventional error back-propagation learning algorithm to flexibly associate fuzzy hyperboxes with a desired class.

### 3. LEARNING ALGORITHMS

Fuzzy ARTRON employs a two-phase learning algorithm. First, an unsupervised structure learning phase is employed to learn the network topology. This includes the decision on the initial connection weights for all  $(u_{ij}, v_{ij})$ 's and the number of fuzzy and cluster nodes in the network by means of the fast-learning fuzzy ART mechanism. This will create proper fuzzy hyperboxes in the input pattern space. Second, a supervised parameter learning phase is then applied to optimally regulate all the connection weights for the desired outputs using the error back-propagation mechanism. This will associate relevant hyperboxes with proper classes. The salient feature of this learning method is that the cluster nodes, fuzzy nodes, and their connections all can be dynamically created and optimized during structure and parameter learning.

#### 3.1 Structure Learning Phase

We use the fast-learning fuzzy ART algorithm [1, 2] to initialize the parameters  $u_{ij}$  and  $v_{ij}$  of the trapezoidal membership function for each fuzzy node. This is identical to finding proper input-space fuzzy clusters or, more exactly, to making proper fuzzy hyperboxes in the input pattern space. Given a complement-coded input feature vector  $\mathbf{I}'$ , the structure learning phase starts by applying a cluster choice function  $T_j$  given below to each cluster node:

$$T_j(\mathbf{I}') = \frac{|\mathbf{I}' \wedge \mathbf{Y}_j|}{\varepsilon + |\mathbf{Y}_j|}, \quad j = 1, 2, \dots, q, \quad (5)$$

where  $\varepsilon > 0$  is a constant, the fuzzy AND operator  $\wedge$  is defined by  $(\mathbf{A} \wedge \mathbf{B})_i \equiv \min(a_i, b_i)$ , and the norm  $|\cdot|$  is defined by  $|\mathbf{A}| = \sum_{i=1}^{2n} |a_i|$ .  $q$  is the number of cluster nodes, and

$\mathbf{Y}_j$  is the complement weight vector of the  $j$ th cluster node defined by  $\mathbf{Y}_j = \{[u_{1j}, 1 - v_{1j}], \dots, [u_{ij}, 1 - v_{ij}], \dots, [u_{nj}, 1 - v_{nj}]\}$ .  $\{[u_{1j}, v_{1j}], \dots, [u_{ij}, v_{ij}], \dots, [u_{nj}, v_{nj}]\}$  is the weight vector of layer 2 related to the cluster node  $j$ .  $T_j(\cdot)$  calculates the similarity between the input feature vector  $\mathbf{I}$  and the complement weight vector  $\mathbf{Y}_j$ . We are looking for a cluster node that has the complement weight vector closest to  $\mathbf{I}$ . This is identical to looking for a fuzzy hyperbox to which  $\mathbf{I} = (i_1, i_2, \dots, i_n)$  could belong. Let the chosen cluster node be indexed by  $J$ ; i.e., the cluster node  $J$  has the largest choice value among all the cluster nodes, or

$$T_J = \max\{T_j : j = 1, \dots, q\}, \quad (6)$$

Now, this chosen cluster node is checked against the vigilance criterion using (7):

$$\frac{|\mathbf{I} \wedge \mathbf{Y}_J|}{|\mathbf{I}|} \geq \rho. \quad (7)$$

If it satisfies (7), resonance occurs and learning ensues. In this case, the complement weight vector  $\mathbf{Y}_J$  of the fuzzy hyperbox  $J$  is updated according to the fast-learning rule [1], i.e.,  $\mathbf{Y}_J^{(new)} = \mathbf{I}' \wedge \mathbf{Y}_J^{(old)}$ . If the vigilance criterion is not met, mismatch reset occurs. In this case, the largest choice value  $T_J$  is set to  $-1$  for the remaining duration to prevent persistent selection of the same cluster node during the subsequent search. Another cluster node of index  $J$  is then chosen using (6). This search process continues until we find a cluster node  $J$  that satisfies (7).

### 3.2 Parameter Learning Phase

With the basic structure learned in the structure learning phase, we are ready to do parameter learning to adjust the weight of each connection by the error back-propagation algorithm. The goal, as in traditional neural nets, is to minimize the following performance function:

$$E = \frac{1}{2} \sum_{k=1}^m (o_k - d_k)^2,$$

where  $o_k$  is the actual output of the  $k$ th class node,  $d_k$  is the desired output of the  $k$ th class node, and  $m$  is the number of class nodes. Let  $w$  be the adjustable weight on a connection. The general learning rule would be

$$\begin{aligned} w(t+1) &= w(t) + \Delta w(t) \\ &= w(t) + \eta \left( -\frac{\partial E}{\partial w} \right) + \alpha \Delta w(t-1), \end{aligned}$$

where  $\eta$  is the learning rate and  $\alpha$  is the momentum parameter. In our experiments,  $\alpha$  was only used to update the connection weights in the class layer; that is, we set  $\alpha$  to 0 for learning in the other layers. The following summarize the respective  $\Delta w(t)$  in each layer. The reader can find detailed derivations from [3]. Note that for the sake of clarity, an architecture that involves only a single cluster node like Fig. 3 is used for explanation.

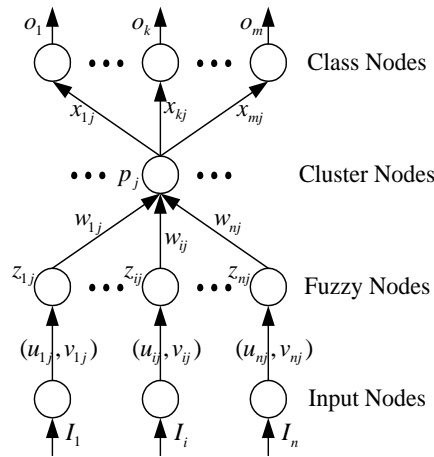


Fig. 3. Fuzzy ARTON with a single cluster node.

First, for the class layer,  $\Delta w(t)$  becomes  $\Delta x(t)$  (Fig. 3) and

$$\Delta x_{kj} = \eta \delta_k p_j, \text{ where } \delta_k = (d_k - o_k)[a'(net_k)].$$

$$\text{For the cluster layer, } \Delta w_r = \eta \sum_{k=1}^m (\delta_k x_{kj}) \frac{\partial \min[M_r, M_r^*]}{\partial M_r} \frac{\partial M_r}{\partial w_r},$$

where  $M_r^* = \min_i(M_i)$  for  $i \neq r$ , and  $M_i = \max(w_i, z_i)$ . Note that the derivatives in the above equation can be linearized using Lukasiewicz's formula [10, 17, 18]:

$$\frac{\partial M_r}{\partial w_r} = \begin{cases} 1, & \text{if } w_r \geq z_r \\ 1 + w_r - z_r, & \text{if } w_r < z_r, \end{cases}$$

and

$$\frac{\partial \min[M_r, M_r^*]}{\partial M_r} = \begin{cases} 1, & \text{if } M_r \leq M_r^* \\ 1 - M_r + M_r^*, & \text{if } M_r > M_r^*. \end{cases}$$

Finally for the fuzzy layer, we need to take care of both  $\Delta u$  and  $\Delta v$ . We have

$$\Delta v_r = \eta \sum_{k=1}^m (\delta_k x_{kj}) \frac{\partial \min[M_r, M_r^*]}{\partial M_r} \frac{\partial M_r}{\partial z_r} \frac{\partial z_r}{\partial v_r},$$

and

$$\Delta u_r = \eta \sum_{k=1}^m (\delta_k x_{kj}) \frac{\partial \min[M_r, M_r^*]}{\partial M_r} \frac{\partial M_r}{\partial z_r} \frac{\partial z_r}{\partial u_r}.$$

## 4. EXPERIMENTS AND CONCLUSIONS

Three experiments were reported here to demonstrate the performance from various viewpoints of Fuzzy ARTRON. They included the XOR problem, the IRIS plant classification [24], and the Heart Disease Database [24]. The simulation program was coded in C and run on a PC. Each experiment was executed ten times, each time using a different random data partition. Table 1 summarizes the parameter settings for the experiments.

**Table 1. Suggested parameter settings for the experiments.**

	$\varepsilon$	$\rho$	$\gamma$	$\alpha$	$\eta_1$	$\eta_2$	$\eta_3$
XOR	0.5	0.5	4	0.95	4.5	0.04	0.04
IRIS	0.5	0.946	37	0.95	0.95	0.04	$10^{-5}$
Heart Disease	0.5	0.991	222	0.95	0.3	0.04	$3 \cdot 10^{-7}$

### 4.1 Experiment Results

#### 4.1.1 XOR problem

The first example is the XOR problem. This example is the most famous one due to its *linear nonseparability*. We used it here as a benchmark test to compare the performance of our model with the results of conventional back-propagation networks. Table 2 presents the results reported in [9] with respect to the number of training epochs and TSSE (Total Sum of Squared Error). Clearly, Fuzzy ARTRON outperforms conventional back-propagation networks.

**Table 2. The XOR problem experimental results.**

XOR	Epochs	TSSE
Conventional Back-propagation	320	0.063900
<b>Fuzzy ARTRON</b>	<b>2-37</b>	<b>0.002254-0.045260</b>
<b>Fuzzy ARTRON (average)</b>	<b>18</b>	<b>0.015828</b>

#### 4.1.2 IRIS plants classification

The second example is the IRIS plant classification, created by R. A. Fisher. This is one of the best known databases in the pattern recognition literature. The goal is to classify a set of IRIS flowers into 3 classes, Virginica, Setosa, and Versicolour, using 4 *continuous* features: the sepal length, sepal width, petal length, and petal width. This data set contains 150 instances, 50 for each class. One class is *linearly separable* from the other two, which are, however, *linearly nonseparable* from each other. The components of each input vector were normalized within the interval [0, 1].

The purpose of this experiment is to evaluate the generalization ability of Fuzzy ARTRON. We randomly picked 25 instances in each class for training and used the remaining instances for testing. Fuzzy ARTRON was constructed by the training set and

then tested on the testing set. The instances were randomly permuted for generality. Table 3 compares the performance of Fuzzy ARTRON with that of several other neural, fuzzy, or traditional classifiers collected in [11, 13].

**Table 3. Generalization capability experiment on the IRIS data set.**

Technique	No. of misclassifications
Bayes classifier	2
k-nearest neighbor	4
Fisher ratios	3
Ho-Kashyap	2
Fuzzy min-max network	2
SEART	0-6
ID3	6-16
Quick propagation	2-35
ssPPc	0-5
<b>Fuzzy ARTRON</b>	<b>1-5</b>
<b>Fuzzy ARTRON(average)</b>	<b>2</b>

#### 4.1.3 Heart disease database

The Heart Disease Database was created by R. Detrano at the Cleveland Clinic Foundation. It contains 76 attributes, but only 14 of them have been used in published experiments. They contain 6 *continuous* and 8 *discrete* attributes. There are two classes: healthy heart and diseased heart. The full data set includes 303 instances, which are nearly equally divided between the two classes. We skipped the six instances that contain missing feature values. The value of each feature was normalized within [0, 1]. Two-thirds of the instances were randomly selected as the training set with the rest used for testing. Without losing generality, the instances were randomly permuted. Table 4 shows the results of classification compared with those obtained using some other techniques [12, 13, 20]. Fuzzy ARTRON has superior success rates in both training and testing sets using reasonable training epochs.

**Table 4. The heart disease database experimental results.**

Technique	Training Epochs	Training Set Correctness	Test Set Correctness
ID3		100%	71.2%
Perceptron	769	63.1%	60.5%
Back-propagation	5000	96.0%	80.6%
FVGE	9.5	99.5%	81.2%
SEART	1	100%	73.1%
<b>Fuzzy ARTRON</b>	<b>118-250</b>	<b>97.0%-98.5%</b>	<b>76.0%-87.0%</b>
<b>Fuzzy ARTRON (average)</b>	<b>178</b>	<b>97.0%</b>	<b>83.0%</b>

## 5. CONCLUSIONS

We have described the general purpose classifier Fuzzy ARTRON, which employs a learning method that combines unsupervised learning, i.e., the fast-learning fuzzy ART procedure, and supervised learning, i.e., the error back-propagation learning scheme. This design has many features. First, the fuzzy ART's self-construction ability is able to partition in a flexible manner the input feature space and then find a proper number of hidden nodes. Specifically, fuzzy ART can self-generate the hidden nodes (cluster nodes) and auto-initialize the connection weights of the fuzzy layer so as to reduce the possibility of falling into a local minimum. It can, thus, overcome the slow convergence rate problem, which usually appears in conventional error back-propagation learning networks. In addition, the back-propagation mechanism's generalization ability can help the conventional fuzzy ART efficiently deal with *linearly nonseparable* problems. It also enhances the weak generalization ability associated with fuzzy ARTMAP. Second, we adopt fuzzy hyperboxes as in fuzzy ART to obtain better generalization ability compared with the crisp hyperbox mechanism. Third, expert knowledge can be directly used in forming the fuzzy ART part of Fuzzy ARTRON before learning, which reduces the amount of learning effort required. After learning, the fuzzy ART part becomes an optimized fuzzy rule base through pruning of the connections that have weak or insignificant weights. According to the results of our experiments, Fuzzy ARTRON not only improves the learning speed, alleviates the local minima problem and guarantees convergence, but also exhibits high-quality classification and generalization capabilities in continuous, discrete, linear, or nonlinear domains.

## REFERENCES

1. G. A. Carpenter, S. Grossberg, and D. B. Rosen, "Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system," *Neural Networks*, Vol. 4, No. 6, 1991, pp. 759-771.
2. G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynolds, and D. B. Rosen, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps," *IEEE Transactions on Neural Networks*, Vol. 3, No. 5, 1992, pp. 698-713.
3. J. S. Chou, *A Neural Classifier Based on Fuzzy ART and Back-Propagation Learning Capability*, Master thesis, Dept. of Electronic Engineering, National Taiwan University of Science and Technology, 1999.
4. E. Cox, "Integrating fuzzy logic into neural nets," *AI Expert*, Vol. 7, No. 6, 1992, pp. 43-47.
5. M. M. Gupta and J. Qi, "On fuzzy neuron models," in *Proceedings of IEEE International Joint Conference on Neural Networks*, 1991, pp. 431-436.
6. C. F. Juang and C. T. Lin, "An on-line self-constructing neural fuzzy inference network and its applications," *IEEE Transactions on Fuzzy Systems*, Vol. 6, No. 1, 1998, pp. 12-32.
7. N. B. Karayiannis and P. I. Pai, "A fuzzy algorithm for learning vector quantization," in *Proceedings of IEEE International Conference on Systems, Man, and Cy-*

- bernetics*, 1994, pp. 126-131.
8. J. M. Keller, R. Krishnapuram, and F. C. Rhee, "Evidence aggregation networks for fuzzy logic inference," *IEEE Transactions on Neural Networks*, Vol. 3, No. 5, 1992, pp.761-769.
  9. K. B. Kim, M. H. Kang, and E. Y. Cha, "A fuzzy self-organized backpropagation using nervous system," in *Proceedings of IEEE International Conference on Systems, Man, and Cybernetics*, 1997, pp. 1457-1462.
  10. G. J. Klir, *Fuzzy Sets, Uncertainty, and Information*, Prentice Hall, New York, 1988.
  11. N. T. Labzour and A. Bensaid, "Improved semi-supervised point-prototype clustering algorithms," in *Proceedings of IEEE International Conference on Fuzzy Systems*, 1998, pp. 1383-1387.
  12. H. M. Lee and W. T. Wang, "A neural network architecture for classification of fuzzy inputs," *Fuzzy Sets and Systems*, Vol. 63, No. 2, 1994, pp. 159-173.
  13. H. M. Lee and C. S. Lai, "Supervised extended ART: A fast neural network classifier trained by combining supervised and unsupervised learning," *Applied Intelligence*, Vol. 6, No. 2, 1996, pp. 117-128.
  14. C. J. Lin and C. T. Lin, "An ART-based fuzzy adaptive learning control network," *IEEE Transactions on Fuzzy Systems*, Vol. 5, No. 4, 1997, pp. 477-496.
  15. C. T. Lin and C. S. G. Lee, *Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems*, Upper Saddle River, Prentice Hall, NJ, 1996.
  16. Y. Lin, G. A. Cunningham III, and S. V. Coggeshall, "Using fuzzy partitions to create fuzzy systems from input-output data and set the initial weights in a fuzzy neural network," *IEEE Transactions on Fuzzy Systems*, Vol. 5, No. 4, 1997, pp. 614-621.
  17. Z. Q. Liu and F. Yan, "Fuzzy neural network in case-based diagnostic system," *IEEE Transactions on Fuzzy Systems*, Vol. 5, No. 2, 1997, pp. 209-222.
  18. W. Pedrycz and A. F. Rocha, "Fuzzy-set based models of neurons and knowledge-based networks," *IEEE Transactions on Fuzzy Systems*, Vol. 1, No. 4, 1993, pp. 254-266.
  19. F. Rosenblatt, *Principles of Neurodynamics, Perceptrons and Theory of Brain Mechanisms*, Spartan, Washington, 1961.
  20. J. W. Shavlik, R. J. Mooney, and G. G. Towell, "Symbolic and neural learning algorithms: An experimental comparison," *Machine Learning*, Vol. 6, No. 2, 1991, pp. 111-143.
  21. P. K. Simpson, "Fuzzy min-max neural networks-Part 2: Clustering," *IEEE Transactions on Fuzzy Systems*, Vol. 1, No. 1, 1993, pp. 32-45.
  22. P. A. Stadter and A. K. Garga, "A neural architecture for fuzzy classification with application to complex system tracking," in *Proceedings of IEEE International Conference on Neural Networks*, 1997, pp. 2439-2444.
  23. Y. Q. Zhang and A. Kandel, "Compensatory neurofuzzy systems with fast learning algorithms," *IEEE Transactions on Neural Networks*, Vol. 9, No. 1, 1998, pp. 83-105.
  24. The UCI repository of Machine Learning Database, <http://www.ics.uci.edu/pub/machine-learning-databases/>.

**Cheng-Seen Ho (何正信)** received the B.S., M.S., and Ph.D. degrees from the Department of Electrical Engineering, National Taiwan University in 1976, 1982, and 1987, respectively. He was a visiting scholar at the Department of Electrical and Computer Engineering, Illinois Institute of Technology, IL, USA, in 1986. He is currently a professor at the Department of Electronic Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan. He chaired the department from 1994 to 1997. He worked with the Taiwan Power Company, Long Distance Telecommunication Administration, and Telecommunication Laboratories before joining the department. His current research interests include intelligent agents, knowledge-based techniques, hybrid systems, and applied artificial intelligence. Dr. Ho is a member of AAAI, ACM, IEEE, and the Taiwanese Association for Artificial Intelligence (TAAI). He served as the first-term president of TAAI from 1995 to 1996.

**Jia-Shiang Chou (周嘉祥)** received the B.S. and M.S. degrees in Electronics Engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan, in 1997 and 1999, respectively. He is currently an associate researcher of the Telecommunication Laboratories, Chunghwa Telecom, Taoyuan, Taiwan. His research interests include neural networks, fuzzy systems, and pattern recognition.