IN-61-CR
139907

Fuzzy Set Methods for Object Recognition in Space Applications

First Quarter Report

P-39

James M. Keller

University of Missouri-Columbia

6/1/91 - 9/30/91

445134

Cooperative Agreement NCC 9-16
Research Activity No. SE.42

NASA Johnson Space Center
Information Systems Directorate
Information Technology Division

ricis

Research Institute for Computing and Information Systems
University of Houston-Clear Lake

INTERIM REPORT

RICIS Preface

This research was conducted under auspices of the Research Institute for Computing and Information Systems by James M. Keller of the University of Missouri-Columbia. Dr. Terry Feagin was the initial RICIS research coordinator for this activity. Dr. A. Glen Houston, Director of RICIS and Assistant Professor of Computer Science, later assumed the research coordinator assignment.

Funding was provided by the Information Technology Division, Information Systems Directorate, NASA/JSC through Cooperative Agreement NCC 9-16 between the NASA Johnson Space Center and the University of Houston-Clear Lake. The NASA technical monitor for this activity was Robert N. Lea, of the Software Technology Branch, Information Technology Division, Information Systems Directorate, NASA/JSC.

First Quarter Report: June 1, 1991 to September 30, 1991

Fuzzy Set Methods For Object Recognition In Space Applications

Fixed-Price Subcontract NO. 088

Under Cooperative Agreement No. NCC 9-16

Project No. SE. 42

To:

UNIVERSITY OF HOUSTON-CLEAR LAKE
2700 Bay Area Boulevard
Houston, TX 77058

UHCL Technical Representative:

Dr. Terry Feagin

Principal Investigator:

James M. Keller
Electrical and Computer Engineering Department
University of Missouri-Columbia
Columbia, Mo. 65211
(314) 882-7339
ecekeler@umcvmb.bitnet

Introduction

For the first quarter of this research contract, we are going to report progress on the following four Tasks (as described in the contract):

1. Fuzzy set-based decision making methodologies;

2. Feature Calculation;

4. Clustering for curve and surface fitting;

5. Acquisition of images.

Fuzzy set-based decision making methodologies

In this section, we describe the general structure for networks based on fuzzy set connectives which we are using for information fusion and decision making in Space Applications. We describe the structure and training techniques for such networks consisting of generalized means and γ-operators. We are currently examining the use of other hybrid operators in multicriteria decision making.

In complex computer vision systems, several sources of information (such as multi-spectral color sensors, range sensors, stereo views, different algorithms, multiple expert systems) are commonly employed in order to reduce the uncertainty and to resolve the ambiguity present in the information derived from a single information source (such as an intensity image). The advantages of multi-source fusion lie in redundancy, complementarity, timeliness and cost of the information . Thus, there is a need for methodologies that can aggregate inexact and incomplete information obtained from multiple sources in order to make decisions. The decisions may be of various types. For example, in segmentation based on region growing, one needs to decide if a homogeneity criterion is satisfied; in edge-based segmentation one needs to decide whether an edge is present or not; in object recognition, one needs to assign a class label to each object.

One can also formulate this problem as a multi-criteria decision making problem as follows. The support for a decision may depend on supports for (or degrees of satisfaction of) several different criteria, and the degree of satisfaction of each criterion may in turn depend on degrees of satisfaction of other sub-criteria, and so on. Thus, the decision process can be viewed as a hierarchical network, where each node in the network "aggregates" the degree of satisfaction of a particular criterion from the observed support. The inputs to each node are the degrees of satisfaction of each of the sub-criteria, and the output is the aggregated degree of satisfaction of the criterion. Thus, the decision making problem reduces to i) determining the structure of the network to be used, ii) the nature of the connectives at each node of the network, and iii) computing the input supports (degrees of satisfaction of criteria) based on observed features.

Fuzzy Aggregation Connectives

Fuzzy set theory provides a host of very attractive aggregation connectives for integrating membership functions representing uncertain and subjective information . These connectives can be categorized into the following three classes based on their aggregation behavior: i) union connectives, ii) intersection connectives, and iii) compensative connectives. Compensative connectives can be further classified into mean operators and hybrid operators. In addition to these, there are also other types of operators such as the OWA operators proposed by Yager, which are capable of modeling linguistic quantifiers such as "at least" and "at most". These will not be discussed in this report.

The Union Connective

The union connective has the property that the aggregated value is high whenever any one of the input values representing different features or criteria is high. The most popular union operator is the "max" operator. However, the max operator is the most pessimistic of all union operators. If we want to be more optimistic, we need to consider one of the many generalizations of the max operator. One such operator is the union operator defined by Yager , and is given by

$$u(x_1,x_2,\ldots,x_n) = \min(1,(x_1{}^p+x_2{}^p+\ldots+x_n{}^p)^{1/p}). \qquad (1)$$

It can be shown that the range of this operator is between $\max(x_1,x_2,\ldots,x_n)$ and 1, and by varying the value of p, we can achieve the required degree of optimism.

The Intersection Connective

The intersection connective has the property that the aggregated value is high only when all of the inputs are high. Several fuzzy intersection operators can be defined, depending on the conditions that we would like the intersection to satisfy. The "min" operator is by far the most popular intersection operator. However, the min operator is the most optimistic of all intersection operators. To allow for different degrees of pessimism, one could choose any of the generalizations to the min operator. For example, the intersection operator due to Yager is given by

$$i(x_1,x_2,\ldots,x_n) = 1- \min[1,((1-x_1)^{-p}+(1-x_2)^{-p}+\ldots+(1-x_n)^{-p})^{-1/p})] . \tag{2}$$

Varying the value of p between 0 and $-\infty$, we can achieve various degrees of pessimism.

Connectives with Compensatory Behavior

In many decision-making situations one is likely to take a position between the two extremes of no compensation characterized by the intersection operators and of full compensation characterized by the union operators. In applications such as multifactorial evaluation (decision making based on several criteria), a certain amount of compensation is desirable. In other words, one might be willing to sacrifice a little on one factor, provided the loss is compensated by gain in another factor. For example, intensity and range information may be mutually compensatory in some situations. Several compensative operators have been proposed in the literature. These can be classified into two groups depending to their origins: mean operators and hybrid operators. Mean operators are defined through an axiomatic approach. Hybrid operators are defined as the weighted arithmetic or geometric mean of a pair of conventional union and intersection operators.

Mean Operators and the Generalized Mean

As pointed out in, the mean operators are very effective in decision making when the criteria are mutually compensable in nature. A mean operator m is a mapping m: $[0,1] \times [0,1] \rightarrow [0,1]$ such that

i. $m(a,b) \geq m(c,d)$ if $a \geq c$ and $b \geq d$ {monotonicity}

ii. $\min(a,b) \leq m(a,b) \leq \max(a,b)$.

Among the mean operators that satisfy the above properties are the weighted arithmetic mean and the geometric mean. Another effective mean operator is the generalized mean first proposed by Dujmovic and later by Dyckhoff and Pedrycz. It is defined by

$$g(x_1,x_2,\ldots,x_n;p,w_1,w_2,\ldots,w_n) = \left(\sum_{i=1}^{n} w_i x_i^p \right)^{1/p} . \tag{3}$$

The w_i's can be thought of as the relative importance factors for the different criteria where

$$w_1+w_2+\ldots+w_n = 1. \tag{4}$$

The generalized mean has several attractive properties. For example, the mean value always increases with an increase in p . Thus, by varying the value of p between $-\infty$ and $+\infty$, we can obtain all values between min and max. Therefore, in the extreme cases, this operator can be used as union or intersection. Also, it can be shown that p=-1 gives the harmonic mean, p=0 gives the geometric mean, and p=1 gives the arithmetic mean. We have suggested that one can also use the generalized mean to simulate linguistic concepts such as "at least" and "at most" by choosing appropriate values for the parameters.

Hybrid Connectives and the γ-Model

The γ-model devised by Zimmermann and Zysno is an example of hybrid operators, and it is defined by

$$y = \left(\prod_{i=1}^{n} x_i^{\delta_i}\right)^{1-\gamma} \left(1 - \prod_{i=1}^{n} (1 - x_i)^{\delta_i}\right)^{\gamma}, \quad \text{where} \sum_{i=1}^{n} \delta_i = n \quad \text{and } 0 \le \gamma \le 1 \qquad (5)$$

In (5), the $x_i \in [0,1]$ are the n inputs to be aggregated, δ_i represents the weight associated with x_i, and γ is a parameter that controls the degree of compensation between the union and intersection parts. The dependence of y on the x_i and the δ_i has been omitted for convenience of notation. The γ-model has been observed to provide a close match to human decision makers.

The γ-model has some very attractive properties. It is a monotonically increasing function with respect to x_i and γ, and hence $(x_{\min})^n \le y \le 1 - (1 - x_{\max})^n$, where $x_{\min} = \min(x_1, \ldots, x_n)$ and $x_{\max} = \max(x_1, \ldots, x_n)$. It is to be noted that these limits correspond to the "algebraic product" and the "algebraic sum" respectively. Since $n \ge 2$, this property shows that the γ-model can behave both as a union operator and an intersection operator in addition to being a compensatory operator, and its range will suffice for many applications.

Learning the Structure and Parameters of Networks

Although two-layer networks (with one level aggregation functions) perform well in simple situations, in more complex settings it becomes necessary to use a multi-layer aggregation scheme. The aggregation and propagation of degrees of satisfaction of criteria in hierarchical networks is not a difficult problem if the structure of the hierarchy is known and if the type of connective to be used at each node in the hierarchy is known. Sometimes this is the case. For example, in medical applications, the hierarchy of the symptoms and the diagnoses are fairly well known. However, in most situations we may have only an approximate idea of the structure of the hierarchy, the nature of the connective associated with each node, and the relevant criteria (features) to be used. We show that optimization procedures such as the gradient descent and the backpropagation algorithm can be used to determine the proper type of aggregation connective at each node and its parameters, given only an approximate structure of the network and given a set of training data that describe the desired behavior of the aggregation network in terms of inputs at the bottom-most level and the outputs at the top-most level.

In a particular situation, the type of aggregation function to be used depends on the (conjunctive, disjunctive or compensative) nature of the problem as well as the desired (pessimistic or optimistic) attitude. In a previous paper, we described a method to determine the nature and parameter values of the aggregation functions in a hierarchical network when a mixture of all three classes of connectives are desirable. However, in this report, we confine ourselves to networks that are entirely made up of either the generalized mean or the γ-model. We believe that the flexibility and range of these two connectives suffices for the type of applications we wish to consider. We now briefly describe the learning procedure for these two cases.

Let us assume that there are n inputs to the node, and the training data for this node consists of N sets of inputs $x_{1k}, \ldots, x_{nk}$ with N corresponding desired outputs Y_k (where $k=1, \ldots, N$, and k denotes the set number). Each set represents a known situation (i.e., the degrees of satisfaction of criteria and the corresponding decision made in that case). The problem is to determine the best type of aggregation function and its parameters for this node in such a way that the discrepancy between the desired and actual behavior is minimized. One measure that is commonly used as discrepancy is the sum of squared errors defined by

$$E = \sum_{k=1}^{N} (f_k - Y_k)^2. \tag{6}$$

In the above equation, f_k is the aggregation function evaluated at $(x_{1k}, \ldots, x_{nk})$.

Learning Using the Generalized Mean

In this case, From (3) and (4) we see that the f_k can be written as

$$f_k = \left(\frac{w_1^2}{\sum w_i^2} x_1^p + \ldots . + \frac{w_n^2}{\sum w_i^2} x_n^p \right)^{1/p} . \tag{7}$$

f_k is written in this form so that the w_i can be chosen free of the constraints in (4). We initially set p and w_i to 1. One can also choose them randomly. Then, we update the weights using the following equations based on gradient descent. It is to be noted that the generalized mean is well-behaved everywhere except at $p=0$ where the derivative is infinity.

$$w_i^{new} = w_i^{old} - \eta \frac{\partial E}{\partial w_i} = w_i^{old} - 2\eta \sum_{k=1}^{N} (f_k - Y_k) \frac{\partial f_k}{\partial w_i}, \quad i = \text{i}, \ldots, n, \tag{8}$$

$$p^{new} = p^{old} - \eta' \frac{\partial E}{\partial p} = p^{old} - 2\eta' \sum_{k=1}^{N} (f_k - Y_k) \frac{\partial f_k}{\partial p}, \tag{9}$$

where η and η' are suitable positive constants and

$$\frac{\partial f_k}{\partial w_i} = \frac{2w_i}{p \sum w_i^2} f_k^{1-p} (x_{ik}^p - f_k^p) \tag{10}$$

$$\frac{\partial f_k}{\partial p} = \frac{f_k^{1-p}}{p^2} \left(\sum_{i=1}^{n} \frac{w_i^2}{\sum w_i^2} x_{ik}^p \ln x_{ik}^p - f_k^p \ln f_k^p \right) \tag{11}$$

This process is repeated until there is no change in w_i and p. This happens when $\partial E/\partial w_i = 0$ and $\partial E/\partial p = 0$, i.e., when a minimum of E is reached. We have shown that the solution is unique under practical conditions, and hence the gradient descent procedure should converge to the global minimum. The choice of η and η' is very important and it determines the speed and reliability of convergence. Since we start with a mean aggregation function, if the training data is better described by a union (intersection) operator, then the value of p

will keep increasing (decreasing) and will not converge (i. e., will converge at $\pm\infty$). However, the procedure presented here can be modified to deal with such situations .

Learning Using the γ-Model

The γ-model can behave like a union operator or an intersection operator or a compensation operator, depending on the value of γ. Since the γ-model is continuous and differentiable with respect to γ and δ_i, we can again use gradient descent methods to arrive at the values of γ and δ_i that best match the given inputs and the corresponding desired outputs. To eliminate the constraints on γ and δ_i in (5), we first modify the definition of γ and δ_i as follows.

$$\gamma = \frac{a^2}{a^2 + b^2} \quad \text{and} \quad \delta_i = \frac{n\, d_i^2}{\sum_{k=1}^{m} d_k^2}. \tag{12}$$

In (12), we can choose a, b and d_i without any constraints and still satisfy the constraints on γ and δ_i. It is easily verified that

$$\frac{\partial y}{\partial a} = \frac{2ab^2}{\left(a^2 + b^2\right)^2}\, y \ln\left(\frac{y_1}{y_2}\right); \quad \frac{\partial y}{\partial b} = \frac{2ba^2}{\left(a^2 + b^2\right)^2}\, y \ln\left(\frac{y_2}{y_1}\right), \tag{13}$$

$$\frac{\partial y}{\partial d_j} = y \frac{2md_j}{\left(\sum_{k=1}^{m} d_k^2\right)^2} \left\{ (1-\gamma)\left[\sum_{k=1}^{m} d_k^2 \ln(x_j/x_k)\right] + \gamma \left[\frac{y_2-1}{y_2} \sum_{k=1}^{m} d_k^2 \ln\frac{1-x_j}{1-x_k}\right]\right\}. \tag{14}$$

where

$$y_1 = \prod_{i=1}^{m} x_i^{\delta_i}; \quad y_2 = 1 - \prod_{i=1}^{m} (1 - x_i)^{\delta_i}. \tag{15}$$

Using the above partial derivatives, we can update the values of γ and δ_i to minimize the discrepancy that reflects the error between desired values of aggregation Y_k and computed values $f_k = f(x_{1k}, \ldots, x_{nk}; \gamma, d_1 \ldots, d_m)$. We first update a, b, and d using

$$a^{new} = a^{old} - \eta \frac{\partial E}{\partial a} = a^{old} - 2\eta \left(\sum_{k=1}^{N} (f_k - Y_k) \frac{\partial f_k}{\partial a}\right), \tag{16}$$

$$b^{new} = b^{old} - \eta \frac{\partial E}{\partial b} = b^{old} - 2\eta \left(\sum_{k=1}^{N} (f_k - Y_k) \frac{\partial f_k}{\partial b}\right), \text{ and} \tag{17}$$

$$d_j^{new} = d_j^{old} - \eta' \frac{\partial E}{\partial d_j} = d_j^{old} - 2\eta' \left(\sum_{k=1}^{N} (f_k - Y_k) \frac{\partial f_k}{\partial d_j} \right). \tag{18}$$

The partial derivatives in (16)-(18) are given in (13) and (14). From a^{new}, b^{new} and d_j^{new}, we can update the new values of γ and δ_j using (12).

This training procedure can be extended to a general situation where there are several nodes arranged in a hierarchical network. In this case, the training data normally consists of input values at the bottom-most layer and the desired outputs at the top-most layer. The extension can be done by using the backpropagation algorithm.

The time required for convergence of this algorithm tends to be very large if it is implemented using the simple backpropagation technique. There are several ways to improve the speed of convergence. Also, a common disadvantage of all gradient descent methods is that they may get trapped in a local minimum. However, we would like to note that our training scheme does not necessarily have to use gradient descent methods. We intend to use other optimization techniques such as the random search method or the differential equation method to overcome the problems mentioned above.

Our goal in subsequent quarters is to design appropriate hierarchical decision networks for segmentation, recognition, and pose estimation of Space Objects. These networks will be fed by membership values generated from relevant features and outputs of low level vision algorithms run on the images. They will be trained on sample images, and tested with "unknown" data.

Feature Calculation

Since we are in the early stages of collecting digital images of Space Objects, it is somewhat premature to discuss the features which will be used in the aggregation networks for object recognition and pose estimation. We have implemented numerous classical features on image regions such as:

Gray level statistics;

Edge and curve primitives;

Texture measures from the cooccurance matrix;

Size and Shape parameters.

In addition, we have pioneered the use of several fractal geometric features which may have a considerable impact on characterizing "cluttered" background, such as clouds, dense star patterns, or some planetary surfaces. Should range imagery become available, we have also introduced several features (and algorithms) which utilize differential geometric models.

As a natural result of using fuzzy clustering algorithms, we will be able to derive experimental measures of the "goodness" of a particular feature set toward the problem at hand. These experiments will be described in detail in a future report.

<u>Clustering for Curve and Surface Fitting</u>

The best way to describe the new work in this task is to include a copy of a manuscript recently submitted by Dr. Krishnapuram and two of the graduate students supported by this contract to the *IEEE Transactions on Neural Networks*

The title of the paper is:

"The Fuzzy C-Shells Algorithm: A New Approach".

We are currently extending this work to clustering edge data into general quadratic curves, as well as extending this approach to 3-Dimensional data sets(ie, surfaces).

Acquisition of Images

In June, the PI traveled to Houston (in conjunction with a joint trip with Bob Lea to the MCC Fuzzy Systems Conference in Austin). While at NASA, meetings were held with Drs. Lea, Pal, and Cleghorn about the availability and type of imagery to be used in the project. This group also visited Dr. Richard Juday to discuss possible collaboration or at least a sharing of data.

NASA personnel are in the process of acquiring suitable simulation data and hopefully videotaped actual shuttle imagery. We have the capability in the Computer Vision Lab. at MU to digitize directly from a standard VCR. While we are waiting for this real (or simulated real) imagery, we have been digitizing photographs to use in our algorithms. Also, we have assembled a model of the shuttle, and are constructing a mechanism to orient this model in 3-D to digitize for experiments on pose estimation. Absolute perfection of details for this work is less important than the knowledge of the actual pose parameters to compare with the calculated estimates. As with the section on feature selection, more detailed exposition of this task will be included in subsequent reports.

The Fuzzy C-Shells Algorithm: A new Approach

Raghu Krishnapuram, Olfa Nasraoui, and Hichem Frigui
Department of Electrical and Computer Engineering
University of Missouri, Columbia, MO 65211

Abstract

The fuzzy C-Shells (FCS) algorithm is specially designed to search for clusters that can be described by circular arcs, or more generally by shells of hyperspheres. In this paper, a new approach to the FCS algorithm is presented. This algorithm is computationally and implementationally simpler than other clustering algorithms that have been suggested for this purpose. An unsupervised algorithm which automatically finds the optimum number of clusters is also proposed. This algorithm can be used when the number of clusters is not known, and uses a new cluster validity measure. Experimental results on several data sets are presented.

1. Introduction

Many fuzzy (and hard) clustering algorithms have been suggested and used in the literature to partition data into clusters. There is a whole class of clustering algorithms in which an objective function based on a distance measure is iteratively minimized to obtain the final partition. The distance measure chosen and the objective function being optimized depend on the geometric structure of the clusters. Different distances have been invented to search for clusters of specific shapes in the feature space. For example, the K means algorithm, using the Euclidian distance, looks for clusters that are hyperspherical in shape. Until recently it has been difficult to detect clusters that can be described by circular arcs, or more generally by shells of hyperspheres. Dave's [1,2] Fuzzy C-Shells (FCS) algorithm has proved to be successful in detecting such clusters, and several impressive examples involving two-dimensional data sets are given in [1,2]. This algorithm has also been generalized to the case of elliptical shells [3,4]. However the FCS algorithm is somewhat implementationally complex since it requires the use of Newton's method to solve two coupled nonlinear equations for the center and radius of each cluster in each iteration. Bezdek et al have suggested a modification to this algorithm to reduce the computational burden due to the use of Newton's method [5]. In this paper, we propose a new FCS algorithm to overcome this problem. Unlike Dave's method, our method does not involve nonlinear equations. This makes our algorithm straightforward, and more importantly, computationally more attractive. In addition, we also propose an unsupervised algorithm to determine the optimum number of clusters C, when this is not known. This unsupervised algorithm involves minimizing a new validity (performance) measure called the total average shell thickness.

In section 2, we present the hard and fuzzy versions of our C-Shells algorithm. In section 3, we introduce our new cluster validity measure and describe an unsupervised algorithm which can be used to determine the optimum number of clusters when this is not known a priori. In section 4, several examples of clustering using the proposed unsupervised algorithm are shown. Finally, section 5 gives the summary and conclusions.

2. The C-Shells Algorithms

Let x_j be a point in the feature space. We assume that each cluster resembles a hyperspherical shell. Therefore, the prototypes λ_i consist of two parameters (c_i, r_i), where c_i is the center of the hypersphere and r_i is the radius. We define the distance from x_j to a prototype $\lambda_i = (c_i, r_i)$ as

$$d_{ij}^2 = d^2(x_j,\lambda_i) = (\| x_j - c_i \|^2 - r_i^2)^2. \tag{1}$$

Note that the right hand side of (1), when equated to zero, also gives the equation of the hypersphere. In general, the closer x_j is to the specific hypersphere, the smaller the distance will be. Based on this distance measure, we now define the hard and fuzzy C-Shells algorithms.

2.1 The C-Shells Algorithm : The Hard Case

We define the objective function to be minimized in this case, as

$$J(L) = \sum_{i=1}^{K} \sum_{x_j \in \lambda_i} d_{ij}^2 , \tag{2}$$

where $L = (\lambda_1,...,\lambda_K)$, and K is the number of clusters. In order to minimize the objective function in (2), we rewrite the distance in (1) as

$$d_{ij}^2 = p_i^T M_j p_i + v_j^T p_i + b_j ,$$

where

$$b_j = (x_j^T x_j)^2, \qquad v_j = 2(x_j^T x_j) y_j . \qquad y_j = \begin{bmatrix} x_j \\ 1 \end{bmatrix},$$

$$M_j = y_j y_j^T, \quad \text{and} \quad p_i = \begin{bmatrix} -2c_i \\ c_i^T c_i - r_i^2 \end{bmatrix}. \tag{3}$$

Therefore,

$$J(L) = \sum_{i=1}^{K} \sum_{x_j \in \lambda_i} (p_i^T M_j p_i + v_j^T p_i + b_j). \tag{4}$$

We may assume that the vectors p_i are independent of each other. Hence, the vectors p_i that minimize (4) must satisfy

$$\sum_{x_j \in \lambda_i} (2M_j p_i + v_j) = 0. \tag{5}$$

If we define

$$H_i = \sum_{x_j \in \lambda_i} M_j, \text{ and } w_i = \sum_{x_j \in \lambda_i} v_j, \tag{6}$$

from (5) we obtain

$$p_i = -\frac{1}{2} (H_i)^{-1} w_i \tag{7}$$

The resulting Hard C-Shells (HCS) algorithm is summarized below.

THE HARD C-SHELLS (HCS) ALGORITHM:

```
Fix the number of clusters $K$;
Set iteration counter $l = 1$ and initialize the hard $K$-partition;
Repeat
    Calculate $H_i^{(l)}$ and $w_i^{(l)}$ for each cluster using (6);
    Compute $p_i^{(l)}$ for each cluster using (7);
    Classify $x_j$ into cluster $\lambda_i$ if $d_{ij}^2 \le d_{kj}^2$, for all $k \neq i$;
    Increment $l$;
Until ( $\| p^{(l-1)} - p^{(l)} \| < \varepsilon$ );
```

2.2 The C-Shells Algorithm : The Fuzzy case

For the fuzzy case, we minimize the following objective function:

$$J(L,U) = \sum_{i=1}^{K} \sum_{j=1}^{N} (\mu_{ij})^m d_{ij}^2. \tag{8}$$

In (8) N is the total number of feature vectors, and $U = [\mu_{ij}]$ is a $K \times N$ matrix called the fuzzy K-partition matrix [6] satisfying the following conditions:

$$\mu_{ij} \in [0,1] \text{ for all } i \text{ and } j, \sum_{i=1}^{K} \mu_{ij} = 1 \text{ for all } j, \text{ and } 0 < \sum_{j=1}^{N} \mu_{ij} < N \text{ for all } i.$$

μ_{ij} is the grade of membership of the feature point $\mathbf{x}_j$ in cluster λ_i, and $m \in [1,\infty)$ is a weighting exponent called the fuzzifier. As in the hard case, it is easy to show that the vectors $\boldsymbol{p}_i$ that minimize (8) are given by (7), where

$$\boldsymbol{H}_i = \sum_{j=1}^{N} (\mu_{ij})^m \boldsymbol{M}_j, \qquad \boldsymbol{w}_i = \sum_{j=1}^{N} (\mu_{ij})^m \boldsymbol{v}_j, \tag{9}$$

and $\boldsymbol{v}_j$ and $\boldsymbol{M}_j$ are given by (3). Following Bezdek's theorem for the fuzzy C means [6], it can be shown that the memberships will be updated according to

$$\mu_{ik} = \begin{cases} \dfrac{1}{\sum_{j=1}^{K} \left(\dfrac{d_{ik}}{d_{jk}}\right)^{\frac{2}{m-1}}} & \text{if } I_k = \Phi \\ 0 \quad i \notin I_k & \text{if } I_k \neq \Phi \\ 1 \quad i \in I_k & \text{if } I_k \neq \Phi \end{cases} \tag{10}$$

where $I_k = \{ i \mid 1 \le i \le K, d_{ik}^2 = 0 \}$. The resulting Fuzzy C-Shells (FCS) algorithm is summarized below.

THE FUZZY C-SHELLS (FCS) ALGORITHM:

```
Fix the number of clusters $K$;  fix $m$ , $1 < m < \infty$;
Set iteration counter $l = 1$;
Initialize the fuzzy $K$-partition $U^{(0)}$;
Repeat
    Calculate $H_i^{(l)}$ and $w_i^{(l)}$ for each cluster $\lambda_i$ using (9);
    Compute $p_i^{(l)}$ for each cluster $\lambda_i$ using (7);
    Update $U^{(l)}$ using (10);
    Increment $l$;
Until ( $\| U^{(l-1)} - U^{(l)} \| < \varepsilon$ );
```

Both the hard and fuzzy C-shells algorithms require the inversion of the matrix $\boldsymbol{H}_i$. This is quite trivial when the feature space is two-dimensional or three-dimensional. In the hard case, the inverse will exist if there are at least n+1 non-collinear points in each cluster, where n is the dimensionality of the feature space. In the fuzzy case, theoretically the inverse will always exist as long as $N > n+1$ and the feature vectors are not colliner.

3. Determination of the Optimal Number of Clusters

The algorithm discussed in Section 2 assumes that the number of clusters K is known, when this is not the case, one method to determine the optimal number of clusters is to perform clustering for a range of K values, and pick the K value for which a suitable performance measure is minimized (or maximized). We define a new performance (or cluster validity) measure called the total average shell thickness as follows.

In the hard case, the total average shell thickness is defined as

$$T_h(K) = \sum_{i=1}^{K} \frac{1}{N_i} \sum_{x_j \in \lambda_i} (\|x_j - c_i\| - r_i)^2 \tag{11}$$

where N_i is the number of points in cluster λ_i . In the fuzzy case, the total fuzzy average shell thickness is defined to be

$$T_f(K) = \sum_{i=1}^{K} \frac{\sum_{j=1}^{N} \mu_{ij}^m (\|x_j - c_i\| - r_i)^2}{\sum_{j=1}^{N} \mu_{ij}^m} \tag{12}$$

Thus, to find the optimum number of clusters, one can start with $K = 1$, and keep incrementing K while calculating $T(K)$ after each run of the FCS algorithm, and stop as soon as a local minimum of $T(K)$ is found (or K reaches K_{max}). However, this simple method sometimes finds a solution in which some of the circular or hyperspherical shells are split into two or more subclusters (usually when K is larger than the actual number of clusters). Therefore, merging back all compatible clusters into one cluster is necessary. Two clusters λ_i and λ_j are considered compatible if

$$\|c_i - c_j\| < \varepsilon_1 \quad \text{and} \quad \|r_i - r_j\| < \varepsilon_2 \tag{13}$$

When minimizing the validity measure for a range of K values, sometimes the algorithm finds a few small spurious clusters. These spurious clusters frequently arise due to noise points.

Such tiny clusters are not compatible with any of the rest of the clusters, and hence merging cannot correct this problem. To eliminate such spurious clusters which contain too few points, we just discard the prototypes for the small clusters, and rerun the algorithm (after decrementing K by the number of tiny clusters), using the remaining prototypes as the initial guesses (i. e., skip the first two steps inside the **Repeat** loop of the C-shells algorithms). This forces the points belonging to the spurious clusters to be reassigned to the best-fitting clusters. This procedure is repeated until no more elimination takes place. The unsupervised algorithm that finds the optimum number of clusters taking into account the problems mentioned above, is summarized below.

```
THE UNSUPERVISED C-SHELLS ALGORITHM:
    Set $K = 1$; fix $m$ , $1 < m < \infty$;
    local_min = false;
    While $K < Kmax$ and local_min = false do
        Initialize the fuzzy $K$-partition $U^{(0)}$;
        Perform the C-Shells algorithm with the number of clusters = $K$;
        Store the final $K$ prototypes;
        Calculate $T(K)$ as given by (11) or (12);
        If $T(K-1)$ is a significant local minimum Then
            local_min = true;
            K_optimal = $K$-1;
        Else
            $K = K + 1$;
        End If
    End While
    Merge compatible prototypes among the K_optimal prototypes and update K_optimal;
    Update $U$ using new prototypes and (10)
    Do
        Eliminate tiny clusters and decrement K_optimal accordingly;
        Perform the C-Shells algorithm with the new K_optimal;
    Until No More Elimination Takes Place
```

4. Experimental Results

Although the algorithms presented in the previous sections are applicable to feature spaces of any dimension, we present only results of two-dimensional data sets here. We found that the HCS algorithm is much faster than the FCS algorithm, but performs well only when the data set is "clean". This is because the HCS algorithm has a higher tendency to get stuck in local minima, and sometimes it terminates abruptly due to the occurrence of singular matrices. Therefore, the HCS algorithm is not very robust, and we do not present the results of the HCS algorithm in this paper.

In all the examples shown in this paper, the FCS algorithm was applied with the fuzzifier $m = 5$. Smaller values did not yield good results. This may be because we initialize the fuzzy partition matrix U with the fuzzy C means algorithm [6] which does not yield a good partition of the clusters, particularly in the case of overlapping or concentric circles. By making the partitioning as fuzzy as possible, it is possible to disentangle the overlapping clusters from each other using the FCS algorithm. The value of ε_1 and ε_2 used was 2 (see Eq.(13)).

The data sets were artificially generated, and had between 50 and 200 feature points. Uniformly distributed noise with an interval of 3 was added to the feature point locations so that they do not always lie exactly on the ideal circles. In addition, noise points were added at random locations to some of the data sets. Any cluster with less than 5 points was considered a spurious cluster.

The first example consists of two concentric circles contaminated by a few noise points. This is an example where conventional clustering methods fail miserably. The unsupervised FCS algorithm stops at $K = 3$, after finding a local minimum in the total fuzzy average shell thickness performance measure $T_f(K)$ at $K = 2$. The plot of $T_f(K)$ versus K is shown in Fig. 1(a). In this case, no merging or small-cluster elimination was required, and the final result is shown in Fig. 1(b). The values of $T_f(K)$ beyond $K = 3$ were obtained by expressly letting the algorithm run, even though it actually stops at $K = 3$. The bold line in Fig. 1(a) depicts the actual running path of the algorithm. It can be seen that there are other local minima at $K = 5, 7$, and 9. At these values, the partition is still acceptable and a final value of $K = 2$ would have been obtained after merging compatible clusters and eliminating tiny clusters. However, the algorithm is designed to stop as soon as the first local minimum is detected to eliminate unnecessary running time. As seen in Fig. 1(b), the two concentric circles are correctly classified, and the noise points are assigned to the closest cluster.

Fig. 1(c) shows $T_f(K)$ versus K for the data set in Fig. 1(d) ("the crying baby"). This is a very difficult example because the clusters have wide-ranging radii, and the outer cluster completely encloses all the remaining clusters. Thus, a truly global search for clusters is required, which can be achieved only by a relatively large m (=5). Again, the bold line in Fig. 1(c), depicts the actual running path of the algorithm which stops at $K = 8$ (as soon as it detects a local minimum at $K = 7$). At this point, the algorithm merges two compatible clusters into one cluster, and reclusters the data set using the prototypes obtained after merging as the initial guesses. In this run, one tiny cluster is eliminated, and the remaining 5 prototypes are used as the initial values. This forces the few points belonging to the tiny cluster to be assigned to the remaining clusters. The final result is $K = 5$, as shown in Fig. 1(d).

Four more examples are shown in Fig 2. Fig 2(a) shows the result of clustering two semicircles contaminated by noise. This example shows that the algorithm is successful even when only parts of circles are present. Fig. 2(b) shows the results of clustering three overlapping circles contaminated by noise, and Fig. 2(c) shows the clustering of five sparsely sampled overlapping circles. These are both very difficult cases, because the circles are truly entangled, and the initial partition is quite wrong. Fig. 2(d) shows the result of clustering the face of "Smiley". The CPU time required on a Sun 4 workstation to run the unsupervised algorithm ranged from 8 s to over 100 s, depending on the complexity of the data set. The plain FCS algorithm typically takes only a few seconds.

5. Conclusions

In this paper, we introduced a new approach to the Fuzzy C-Shells algorithm, which seeks clusters in hyperspherical shells. This algorithm does not involve solving coupled nonlinear equations, and hence is implementationally more attractive than other clustering algorithms that have been suggested in the literature for this purpose. We also presented an unsupervised C-shells algorithm which automatically finds the optimum number of hyperspherical clusters when this information is not known. The unsupervised algorithm is based on minimizing a new validity measure called total average shell thickness. Experimental results on a variety of data sets demonstrate that the algorithms are effective.

6. Acknowledgment

Partial support of this research by a NASA grant through University of Houston, Clear Lake, RICIS Project No. SE 42, is gratefully acknowledged.

7. References

1. R. N. Dave and S. K. Bhamidipati, "Application of the fuzzy-shell clustering algorithm to recognize circular shapes in digital images", *Proceedings of the International Fuzzy Systems Association Congress*, Seattle, 1989, pp. 238-241.

2. R. N. Dave, "Fuzzy-shell clustering and applications to circle detection in digital images", *International Journal of General Systems*, vol. 16, 1990, pp. 343-355.

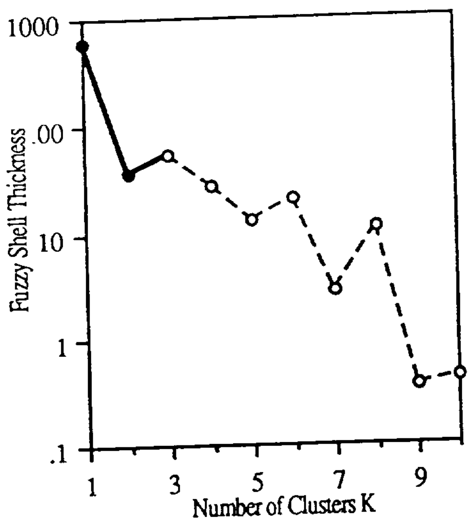
3. R. N. Dave and K. J. Patel, "Fuzzy ellipsoidal-shell clustering algorithm and detection of ellipsoidal shapes", *Proceedings of the SPIE Conference on Intelligent Robots and Computer Vision IX: Algorithms and Techniques*, Boston, Nov. 1990, pp. 320-333.

4. R. N. Dave, "Adaptive C-shells clustering", *Proceedings of the North American Fuzzy Information Processing Society Workshop*, Columbia, Missouri, 1991, pp. 195-199.

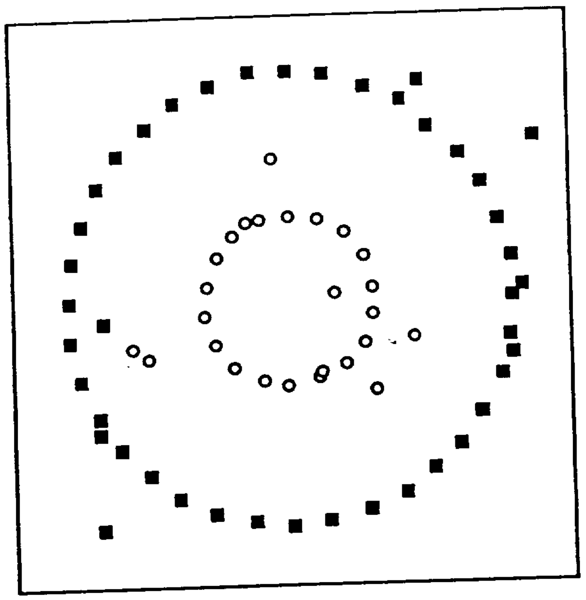
5. J. C. Bezdek and R. J. Hathaway, "Accelerating convergence of the Fuzzy C-Shells clustering algorithms", *Proceedings of the International Fuzzy Systems Association Congress*, Brussels, July 1991, Volume on *Mathematics*, pp. 12-15.

6. J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York, 1981.

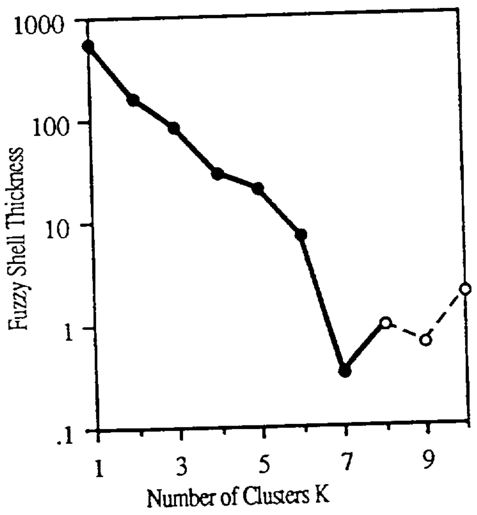
List of Figures



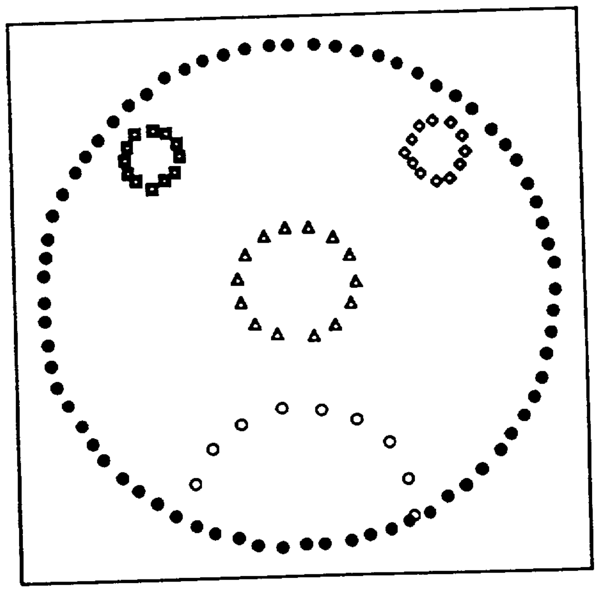
(a)



(b)



(c)



(d)

Figure 1

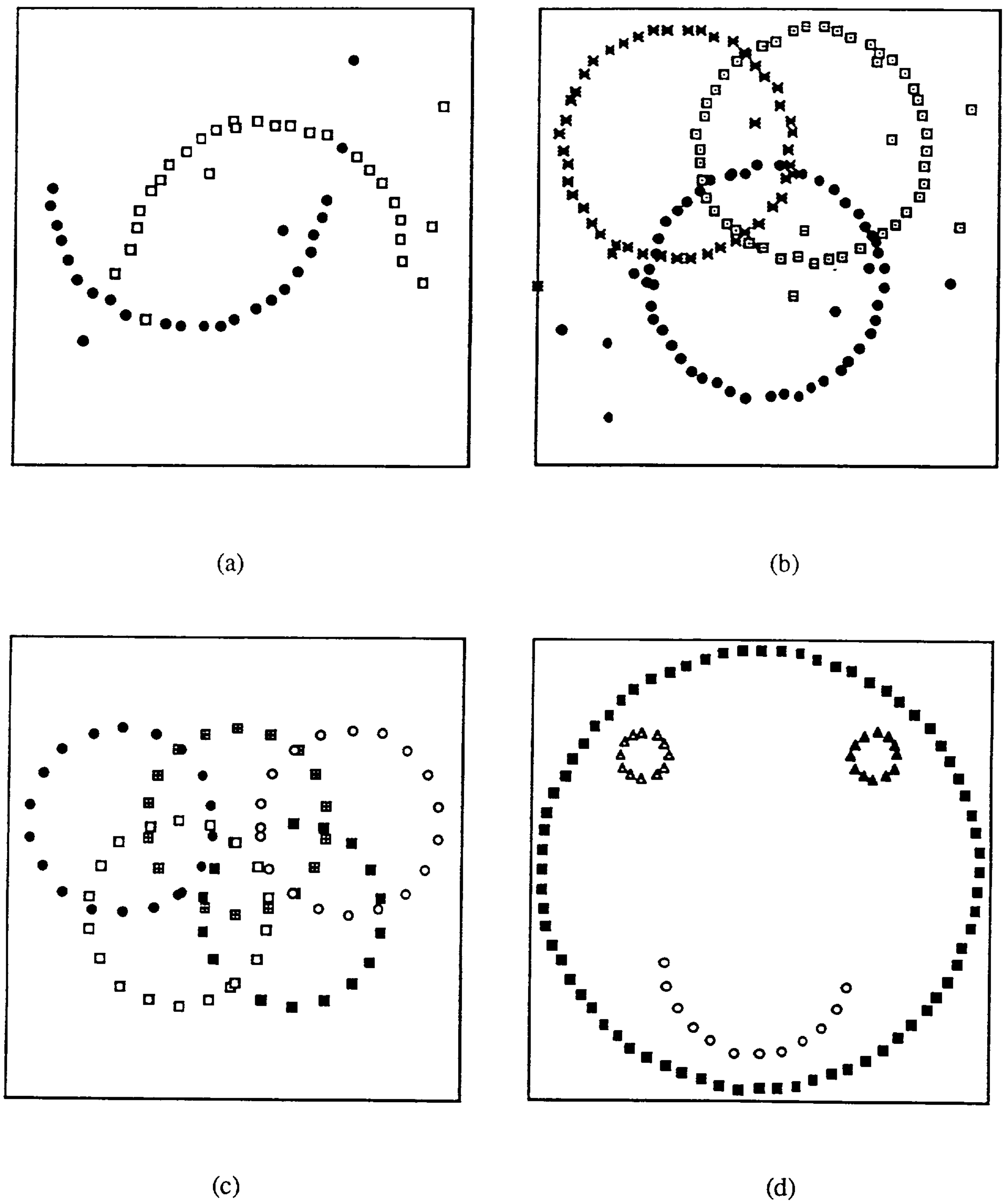


Figure 2

Bibliography

Andress, K. M., and Kak, A. C. (1988). Evidence Accumulation and Flow of Control in a Hierarchical Spatial Reasoning System. *Artificial Intelligence Magazine*, Summer, 75-94.

Baba, N. (1989). A New Approach for Finding the Global Minimum of Error Function of Neural Networks. *Neural Networks*, **2**, 367 - 373.

Carter, J. P. (1987). Successfully Using Peak Learning Rates of 10 (and Greater) in Backpropagation Networks with Heuristic Learning. *Proceedings of the International Conference on Neural Networks*, **II**, 645-652.

Dahl, E. D. (1987). Accelerated Learning Using the Generalized Delta Rule, *Proceedings of the International Conference on Neural Networks*, **II**, 523-530.

Dubois, D., and Prade, H. (1982) A Class of Fuzzy Measures Based on Triangular Norms. *International Journal of General Systems*, **8**(1), 43-61.

Dubois, D., and Prade, H. (1985). A Review of Fuzzy Set Aggregation Connectives. *Information Sciences*, **36**(1&2), 85-121.

Dujmovic, A. (1974), Weighted Conjunctive and Disjunctive Means and Their Application in System Evaluation, *Publikacije Elektrotehnickog Faculteta Beograd, Serija Matematika i Fizika*, No. 483, pp. 147-158.

Dyckhoff, H., and Pedrycz, W. (1984). Generalized Means as a Model of Compensation Connectives. *Fuzzy Sets and Systems*, **14**(2),143-154.

Erman, L.D. and Lesser, V. R. (1975). A Multi-level Organization for Problem Solving Using Many Diverse, Cooperating Sources of Knowledge. *Proceedings of the 4th International Joint Conferences on Artificial Intelligence*, Cambridge MA, 483-490.

Feldman, J. A., and Ballard, D. H. (1982). Connectionist Models and Their Properties. *Cognitive Science*, **6**, 205-256.

Gottfried, B. S., and Weisman, J. (1973). *Introduction to Optimization Theory*, Prentice Hall, NJ.

Hanson, A., and Riseman, E., The VISIONS Image Understanding System 1986 (1987). *COINS Technical Report 86-62*, University of Massachusetts.

Huntsberger, T. L., Rangarajan, C., and Jayaramamurthy, S. N. (1986). Representation of Uncertainty in Computer Vision Using Fuzzy Sets. *IEEE Transactions on Computers*, **35**(2), 145-156.

Hush, D. R., and Saks, J. M. (1988). Improving the Learning Rate of Backpropagation with the Gradient Reuse Algorithm. *Proceedings of the International Conference on Neural Networks*, **I**, 441-448.

Klir, G. J., and Folger, T. A. (1988). *Fuzzy Sets, Uncertainty and Information*, Prentice Hall, Englewood Cliffs, NJ.

Kollias, S., and Anastassiu, D. (1989). An Adaptive Least Squares Algorithm for the Efficient Training of Artificial Neural Network. *IEEE Transactions on Circuits and Systems*, **36**(8).

Krishnapuram, R., and Lee, J., Fuzzy-Connective-Based Hierarchical Aggregation Networks for Decision Making, to appear in *Fuzzy Sets and Systems*.

Krishnapuram, R., and Lee, J. (1988). Propagation of Uncertainty in Neural Networks, *Proceedings of the SPIE Conference on Intelligent Robots and Computer Vision*, Cambridge MA, November, 377-383.

Krishnapuram, R., and Lee, J. (1989). Determining the Structure of Uncertainty Management Networks, *Proceedings of the SPIE Conference on Robotics and Computer Vision*, Philadelphia.

Lee, J. (1990). Fuzzy-Connective-Based Information Fusion Networks and Their Application to Computer Vision. Ph. D. Dissertation, Dept. of Electrical and Computer Engineering.

Luo, R. C., and Kay, M. G. (1989). Multisensor Integration and Fusion in Intelligent Systems. *IEEE Transactions on Systems Man and Cybernetics*, **19**(5), 901-931.

Mitiche, A., and Aggarwal, J. K. (1986). Multiple Sensor Integration/Fusion through Image Processing: A Review. *Optical Engineering*, **25**(3), 380-386.

Mizumoto, M. (1989). Pictorial Representations of Fuzzy Connectives, Part I: Cases of t-norms, t-conorms, and Averaging Operators. *Fuzzy Sets and Systems*, **31**, 217-242.

Mizumoto, M. (1989). Pictorial Representations of Fuzzy Connectives, Part II: Cases of Compensatory Operators, and Self-Dual Operators. *Fuzzy Sets and Systems*, **32**, 45-79.

Owens, A. J., and Filkin, D. L. (1989). Efficient Training of the Backpropagation Network by Solving a System of Stiff Ordinary Differential Equations. *Proceedings of the International Joint Conference on Neural Networks*, Washington D. C., **II**, 381-386.

Parker, D. B. (1987). Optimal Algorithms for Adaptive Networks: Second Order Back Propagation, Second Order Direct Propagation, and Second Order Hebbian Learning. *Proceedings of the International Conference on Neural Networks*, II, 593-599.

Perugini, N. K., and Engeler, W. E. (1989). Neural Network Learning Time: Effects of Network and Training Size. *Proceedings of the International Conference on Neural Networks*, **2**, 395- 401.

Richardson, J. M., and Marsh, K. A. (1988). Fusion of Multisensor Data. *International Journal of Robotics Research*, **7**(6), 78-96.

Rumelhart, D. E., McClelland, J. M., and the PDP Research Group (1986). *Parallel Distributed Processing*, Chapter 8, 1, MIT Press.

Tucker, L. W., Feynman, C. R., and Fritsche, D. M. (1988). Object Recognition Using the Connection Machine. *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Ann Arbor MI, 871-878.

Thole, U., and Zimmermann, H.-J. (1979). On the Suitability of Minimum and Product Operations for the Intersection of Fuzzy Sets. *Fuzzy Sets and Systems*, 2, 167-180.

Wang, J., and Malakooti, B. (1989). On Training of Artificial Neural Networks. *Proceedings of the International Joint Conference on Neural Networks*, Washington D. C., **II**, 387-393.

Yager, R. P.. (1978). Fuzzy Decision Making Including Unequal Objectives. *Fuzzy Sets and Systems*, **1**(2), 87-95.

Yager, R. P. (1980). On a General Class of Fuzzy Connectives. *Fuzzy Sets and Systems*, **4**, 235-242.

Zadeh, L. A. (1987). Fuzzy Sets as a Basis for a Theory of Possibility. In Yager, R. P., Ovchinnikov, S., Tong, R. M., and Nguen, N. T. (Eds.) (193-218), *Fuzzy Sets and Applications: Selected Papers by L. A. Zadeh*, John Wiley, New York, 1987.

Yager, R. P. (1988). On Ordered Weighted Averaging Aggregation Operations in Multicriteria Decision making, IEEE Transactions on Systems, Man and Cybernetics, **18**(1), 183-190.

Zimmermann, H.-J., and Zysno, P. (1980). Latent Connectives in Human Decision Making. *Fuzzy Sets and Systems*, **4**(1), 37-51.

Zimmermann, H.-J., and Zysno, P. (1983). Decisions and Evaluations by Hierarchical Aggregation of Information. *Fuzzy Sets and Systems*, **10**(3), 243-260.