PARTITIONING OF FEATURE SPACE FOR PATTERN CLASSIFICATION

DEBA PRASAD MANDAL*

Department of Industrial Engineering, University of Osaka Prefecture, 1-1 Gakuen-cho, Sakai, Osaka 593, Japan

(*Received 8 December* 1995; *in revised form 23 December* 1996)

Abstract—The article proposes a simple approach for finding a fuzzy partitioning of a feature space for pattern classification problems. A feature space is initially decomposed into some overlapping hyperboxes depending on the relative positions of the pattern classes found in the training samples. A few fuzzy if–then rules reflecting the pattern classes by the generated hyperboxes are then obtained in terms of a relational matrix. The relational matrix is utilized in the modified compositional rule of inference in order to recognize an unknown pattern. The proposed system is capable of handling imprecise information both in the learning and the processing phases. The imprecise information is considered to be either incomplete or mixed or interval or linguistic in form. Ways of handling such imprecise information are also discussed. The effectiveness of the system is demonstrated on some synthetic data sets in two-dimensional feature space. The practical applicability of the system is verified on four real data such as the Iris data set, an appendicitis data set, a speech data set and a hepatic disease data set.

Pattern classification Fuzzy partitioning Fuzzy if–then rules Fuzzy sets
Compositional rule of inference Management of uncertainty

1. INTRODUCTION

In designing information processing systems such as classifiers and controllers, two types of information (features characterizing the patterns) are available. One is numerical information (from measuring instruments) and the other is linguistic (imprecise) information (from human experts). In addition to these, some other types of imprecise data are also frequently observed. For example, instrumental error or noise corruption in the experiment may result in providing partial/unreliable feature information [e.g. *F* is about 10 (mixed form) or *F* is between 10 and 15 or *F* is less than 15 (interval form)]. Again, sometimes the data are found to be incomplete (partial) in the sense that all the feature values may not be known (missing). Most conventional probabilistic and deterministic classifiers[1–3] can utilize only the numerical data. Pattern having imprecise and/or incomplete information are usually ignored or discarded from the design and testing phases. On the other hand, fuzzy control[4,5] is one of the useful approaches in utilizing experts knowledge in the form of fuzzy if–then rules. In the literature, there exist many approaches which utilize fuzzy if–then rules from numerical data [for example, see references (6–12)] but most of such approaches are not applicable for higher dimensional feature space. Moreover, the existing classifier systems (classical and fuzzy) usually provide crisp (two-state) output and are suitable for mechanistic types of problems.

The present article proposes a simple but effective approach for finding a fuzzy partitioning of the feature space for classification problems. The aspect of practical applicability to higher dimensional feature space has been given the foremost importance in the proposed algorithm. The approach initially finds the non-overlapping (unambiguous) and overlapping (ambiguous) regions in the feature spaces based on the relative positions of the pattern classes provided by their training samples, and the training samples of various classes are divided into few subclasses. Accordingly the total feature space is decomposed into some hyperboxes or regions to represent the subclasses by the generated hyperboxes to a better extent. In order to handle the impreciseness of the input feature information and to incorporate the portions possibly uncovered by the training samples, each of the hyperboxes is extended to some extent. The proposed method, therefore, results in decomposing the whole feature range into a few overlapping hyperboxes.

The fuzzy if–then rules reflecting the pattern classes by the hyperboxes are then generated. We consider here a relational matrix to represent the if–then rules more efficiently. To process an unknown pattern, we first find its membership values to various hyperboxes. Then the possibilities of the pattern to different pattern classes are determined by using the modified Zadeh's compositional rule of inference[13] between the membership values of the hyperboxes and the relational matrix. The output of the system is provided in terms of first, second, other and null choices.

.e proposed system is capable of handling various imprecise information both in its learning and recognizing phases. The imprecise information are grouped here into four categories, namely, incomplete, mixed, interval and linguistic forms. We also discuss ways of handling such imprecise information in the present article.

The effectiveness of the system is demonstrated on some artificially generated data sets in two-dimensional feature space. The practical applicability of the system is verified on four real data such as the Iris data set, an appendicitis data set, a speech data set and a hepatic disease data set. In these data sets, the number of features is large and also some of the samples are imprecise. The performance of the proposed system is found to be better than other existing classical and fuzzy approaches.

The outline of this paper is as follows. In Section 2, some preliminaries are stated which include a brief overview of the existing methods with fuzzy if–then rules, a few definitions like pattern class, accuracy factor, coverage factors, membership functions, relational matrix, and a block diagram. The description of our approach in decomposing a feature space and subsequent processing (classification) of an unknown input pattern are provided in Sections 3 and 4, respectively. Ways of handling various imprecise information in the proposed system are furnished in Section 5. Implementations on various artificially generated pattern sets as well as on four real-life data sets are provided in Section 6. Section 7 has the conclusions and discussion. The detailed algorithm for the proposed feature space decomposition procedure is included as Appendix A.

2. SOME PRELIMINARIES

In this section, a brief overview of existing classification methods with fuzzy if–then rules is given. Then some basic concepts which are useful in developing the proposed classification system are introduced. Finally, a block diagram of the proposed system is presented at the end of this section.

2.1. *Classification methods with fuzzy if–then rules*

For classification problems, many approaches based on fuzzy set theory[14] can be found in the literature (for example, see references (15–18)). The existing fuzzy classification methods may be grouped into the following four categories:[19]

1. methods based on fuzzy relations,
2. methods based on fuzzy pattern matching,
3. methods based on fuzzy clustering, and
4. other methods which are more or less generalization of classical approaches.

We confine our study to approaches based on fuzzy if–then rules which belong to the first category and such approaches usually generate fuzzy if–then rules from the deterministic (numerical) data.

Generation of fuzzy if–then rules from numerical (precise) data for pattern classification problems consists of two phases: (a) fuzzy partitioning of a feature space into fuzzy subspaces and (b) determination of fuzzy if–then rules corresponding to the fuzzy subspaces. For example, Ishibuchi *et al.*[11] generates fuzzy if–then rules from the training samples by employing a fuzzy partitioning approach with fuzzy grids. One shortcoming of such an approach is that the number of fuzzy subspaces increases exponentially with the increase of the number of features. The authors themselves noticed this problem and so they modified it by proposing another approach[12] based on the sequential subdivision of the fuzzy subspaces (of different sizes). It can be observed that the number of subspaces generated by the later approach[12] is also quite large for higher dimensional feature spaces.

One of the inherent assumptions in such existing approaches is that the pattern classes are nonoverlapping, i.e. each feature point of the feature space always corresponds to only one pattern class. But this need not always be true. Because in most real problems, pattern classes are overlapping, i.e. a single feature point may correspond to more than one class. On the other hand, the training sample set cannot represent the pattern classes fully. It may be better to assume that every sample point represents a covered area of a class in the feature space. In this case, finer partitioning of the feature space may not be appropriate.

Other work in this direction includes references (20,21), where Mandal *et al.* decomposes the feature space into some overlapping subspaces using the geometric structure[22,23] of the pattern classes found from the training samples. The relative positions of the sample sets in the feature space are considered to control further partitioning. Pal and Mandal[24] described an approach where a feature space is decomposed into a few (3^N for N features) overlapping regions by considering three primary linguistic properties—small, medium and high—along each of the feature axes. Abe and Lan[25] recently proposed a method which extracts fuzzy rules with variable fuzzy regions by recursively resolving overlap between two classes. When the number of features is large, the number of regions generated by such approaches becomes very high and this leads to serious practical difficulty in implementing/executing such schemes.

2.2. *A few basic concepts*

We propose here a simple approach for partitioning a feature space for classification purposes. To describe the procedure, we consider an M class $(C_1, C_2, \ldots, C_j, \ldots, C_M)$ and N feature $(F_1, F_2, \ldots, F_i, \ldots, F_N)$ problem throughout this article. Depending on the relative positions of the pattern classes (obtained from training samples) along individual feature axes, the training sample sets are decomposed into some subclasses. Accordingly, the whole feature space is decomposed into a few (say, q) hyperboxes or regions to represent the subclasses by the generated regions. To handle the uncertainty of the input information and to incorporate the portions (of the pattern classes) possibly uncovered by the training samples, the hyperboxes are extended to some extent using triangular membership functions. Thus the whole feature space is divided into some (q) overlapping hyperboxes.

The pattern classes contained by the hyperboxes are then noted and a few fuzzy if–then rules are generated. We find it convenient to represent the generated if–then rules in terms of a relational matrix and use then the modified compositional rule of inference[13] for classifying an unknown pattern.

Before going further, some basic concepts are introduced here which are useful in developing (and also in understanding) the proposed classification system. These include the definition of a pattern class (i.e. the collection of subsets of $\mathbb{R}^N$ under consideration in the present investigation), accuracy factor, coverage factors, membership functions and relational matrix.

Pattern class. In most of the real problems, pattern classes are bounded. Thus the pattern classes considered here are all bounded. A formal definition of pattern class in $\mathbb{R}^N$ is given as follows:

Definition 1. A set $\mathscr{A} \subseteq \mathbb{R}^N$ is said to be a pattern class[26] if

1. $\mathscr{A}$ is path connected and compact,
2. $cl(Int(\mathscr{A})) = \mathscr{A}$, [$cl$ means closure, Int means interior]
3. $Int(\mathscr{A})$ is path connected and
4. $\lambda(\delta\mathscr{A}) = 0$, where $\delta\mathscr{A} = \mathscr{A} \cap cl(\mathscr{A}^c)$ and λ is the Lebesgue measure on $\mathbb{R}^N$.

Let $A = \{\mathscr{A} : \mathscr{A}$ satisfies Definition 1$\}$. A is the collection of all classes in $\mathbb{R}^N$. Any $\mathscr{A} \in A$ is referred to as a pattern class in $\mathbb{R}^N$.

Accuracy factor. An accuracy factor (δ, $0 < \delta < 1$) is considered in the proposed approach to manage the uncertainty of the training samples to some extent. It is well known that with the increase of the size of training sample set, the description of actual classes by the training set improves (i.e. the accuracy of the training set to represent actual classes increases). Therefore, the value of δ may be decided based on the size (say, t) of training set.

In the proposed procedure of decomposing a feature space, each feature is considered separately. In this view, the value of δ satisfies the following inequality:[22]

$$\frac{1}{t} < \delta < \frac{1}{\sqrt{t}}, \tag{1}$$

so that as $t \to \infty$, $\delta \to 0$ and $t\delta \to \infty$. Since the value of δ decreases with the increase of t, the accuracy of the proposed algorithm also increases with the increase of t. The inequality (1) is due to Grenander[27] who used it for estimation of a set or class.

The inequality (1) provides a set (interval) of values for δ. A smaller value of δ finds a finer partition of the feature space. On the other hand, as the partitions are carried out based on the information provided by the training samples, the value of δ should not be too small. Taking these points into consideration, the value of δ used in the present work is

$$\delta = \frac{1 + \sqrt{t}}{2t}. \tag{2}$$

Coverage factors. We have assumed that every sample point represents a covered area in the feature space. To delineate the coverage, coverage factors (ε_i, $i = 1, 2, \ldots, N$) corresponding to each feature axis are found using the training sample information and the accuracy factor (δ). Let $\min_i$ and $\max_i$ be the lower most and the upper most values of the ith feature in the training samples respectively. Then the value of ε_i is defined as

$$\varepsilon_i = (\max_i - \min_i) \times \delta, \quad i = 1, 2, \ldots, N.$$

The ε_i's are utilized in the proposed approach of partitioning the feature space and also to extend the hyperboxes for incorporating portions possibly uncovered by the training samples.

Membership functions. In the proposed method, a feature space is decomposed into q overlapping hyperboxes $S_1, S_2, \ldots, S_h, \ldots, S_q$. Each of these hyperboxes consists of feature ranges in individual feature axes, i.e. S_h ($h = 1, 2, \ldots, q$) can be represented as

$$S_h = \mathop{\times}_{i=1}^{N} \text{ Support } (s_h^i), \tag{4}$$

where s_h^i is a fuzzy set whose support is a subinterval of the range of feature axis F_i ($i = 1, 2, \ldots, N$). For a given pattern point on an individual feature axis, the possibility of its being a member of a feature range is maximum if it lies in the center of the feature range. As the distance of the point from the central point increases, the possibility decreases and ultimately go to zero. As all triangular functions have the previous property, any triangular function may be considered as the membership function of a feature range. We consider here a symmetric piecewise linear triangular function to serve the purpose.

The functional form of the symmetric piecewise linear triangular function, we use, is

$$T_h^i(x_i; \alpha_h^i, \beta_h^i, \gamma_h^i) = \begin{cases} \frac{1}{2} + \frac{(x_i - \alpha_h^i + \beta_h^i)}{2\gamma_h^i} & \text{for } \alpha_h^i - \beta_h^i - \gamma_h^i < x_i < \alpha_h^i - \beta_h^i, \\ 1 - \frac{|x_i - \alpha_h^i|}{2\beta_h^i} & \text{for } \alpha_h^i - \beta_h^i \le x_i \le \alpha_h^i + \beta_h^i, \\ \frac{1}{2} - \frac{(x_i - \alpha_h^i - \beta_h^i)}{2\gamma_h^i} & \text{for } \alpha_h^i + \beta_h^i \le x_i \le \alpha_h^i + \beta_h^i + \gamma_h^i, \\ 0 & \text{otherwise.} \end{cases}$$

The structure of such a symmetric piecewise linear triangular function is shown in Fig. 1. The fuzzy subset s_h^i is characterized by this function [equation (5)], where the central value is α_h^i and the support is the open interval $(\alpha_h^i - \beta_h^i - \gamma_h^i,\ \alpha_h^i + \beta_h^i + \gamma_h^i)$ (in the present work, $\gamma_h^i = \varepsilon_i$, $\forall h, i$). Here the portion $[\alpha_h^i - \beta_h^i,\ \alpha_h^i + \beta_h^i]$ is assumed to be reflected by the training samples, and $(\alpha_h^i - \beta_h^i - \gamma_h^i, \alpha_h^i - \beta_h^i)$ and $(\alpha_h^i + \beta_h^i, \alpha_h^i + \beta_h^i + \gamma_h^i)$ are the extended portions. The extended portions reflect the possible uncovered regions of the pattern classes by the training samples and the overlapping between the pattern classes.

Relational matrix. The proposed decomposition procedure generates q hyperboxes to represent various classes. Suppose r_{hj} ($h = 1, 2, \ldots, q$ and $j = 1, 2, \ldots, M$) denotes

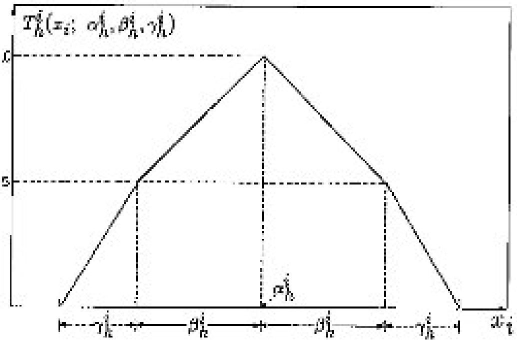


Fig. 1. Symmetric piecewise linear triangular function.

the possibility value that S_h represents the class C_j. In terms of fuzzy if–then rules, r_{hj} can be expressed as

if $X = (x_1, x_2, \ldots, x_N)$ belongs to hyperbox S_h,
then X belongs to class C_j with possibility value r_{hj},
i.e.
if $x_1 \in s_h^1, x_2 \in s_h^2, \ldots, x_N \in s_h^N$,
then $X \in C_j$ with possibility r_{hj}.

We find it convenient to represent such fuzzy if–then rules by a relational matrix,[28] denoted by $\mathscr{R}$, as

$$\mathscr{R} = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1j} & \cdots & r_{1M} \\ r_{21} & r_{22} & \cdots & r_{2j} & \cdots & r_{2M} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ r_{h1} & r_{h2} & \cdots & r_{hj} & \cdots & r_{hM} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ r_{q1} & r_{q2} & \cdots & r_{qj} & \cdots & r_{qM} \end{pmatrix}, \qquad (6)$$

where each row (h) corresponds to a particular hyperbox (S_h) and each column (j) corresponds to a particular class (C_j). The way of determining the elements of $\mathscr{R}$ is explained in the next section.

2.3. Block diagram

The block diagram of the proposed classification system is shown in Fig. 2. It consists of two parts, namely, *fuzzy learning* and *fuzzy processor*. Based on only the training sample information, the learning phase generates representative hyperboxes and estimates the relational matrix $\mathscr{R}$. The *feature space decomposer* block decomposes a feature space into some overlapping hyperboxes. The *if–then rule generator* block finds the fuzzy if–then rules in terms of a relational matrix that represents the pattern classes by the hyperboxes.

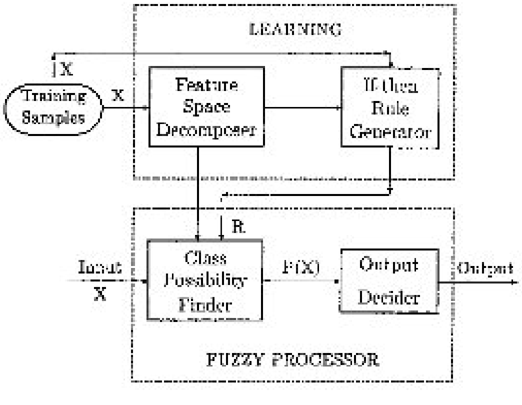


Fig. 2. Block diagram.

The fuzzy processor basically finds the output decision regarding the class or classes to which an unknown pattern may belong. The *class possibility finder* block uses the relational matrix in the modified compositional rule of inference[13] to determine the possibility (similarity) values of an unknown pattern to belong to various pattern classes. These possibility values are analyzed in the *output decider* block to find the final class assignment of an unknown pattern. Note that the proposed system provides multiple class choices (in order of preference) in case the unknown pattern lies in an ambiguous (overlapping) region. The system may be viewed as a generalized classifier as it can handle both precise (deterministic) and imprecise information in the learning as well as in the processing phases.

We have introduced the proposed classification system in this section. The operations of various blocks of Fig. 2 are discussed in the following sections.

3. FUZZY LEARNING

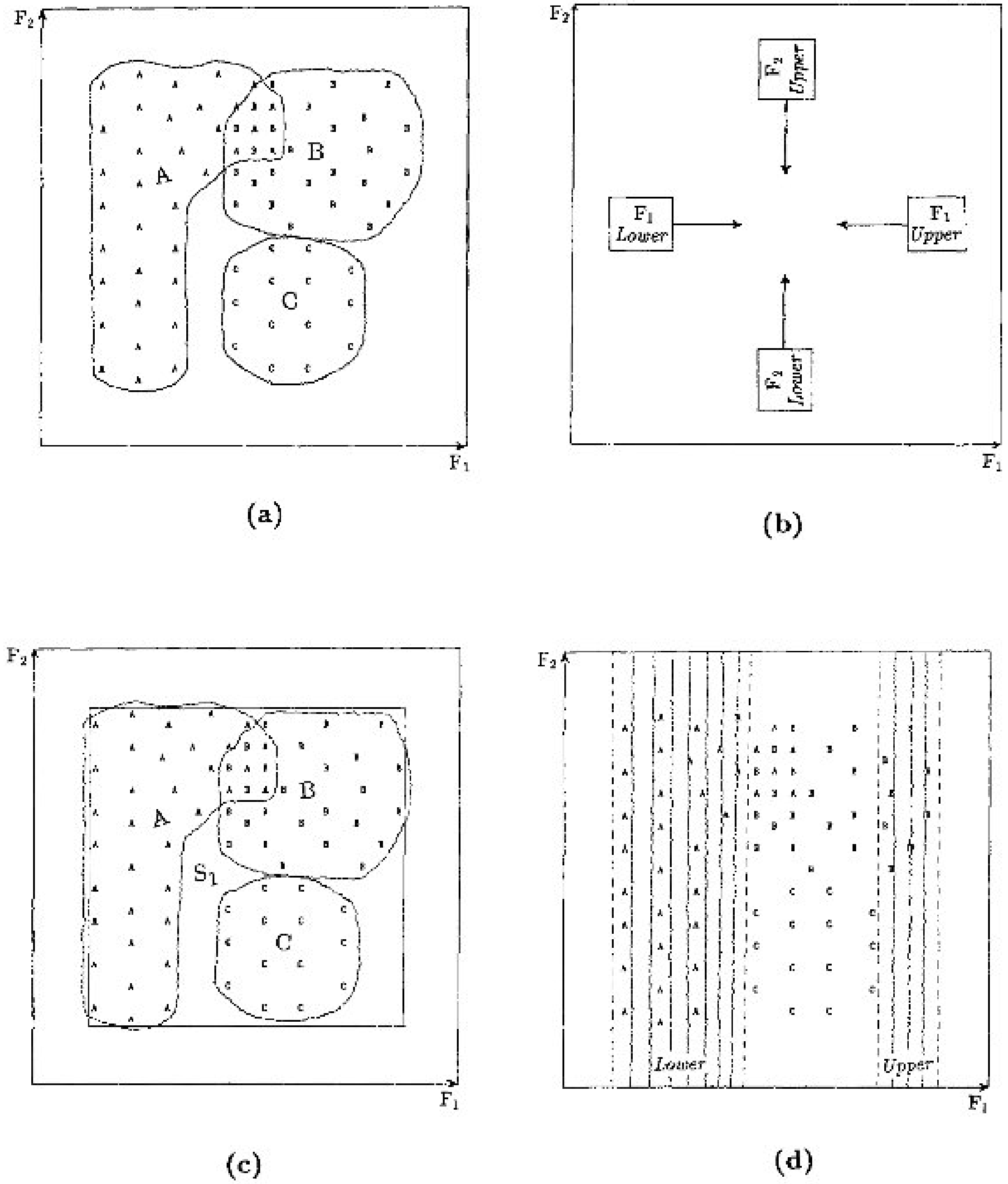
The operations of this section are fully dependent on the training samples. This section is divided into two blocks, namely, *feature space decomposer* and *if–then rule generator*. Representative hyperboxes in the feature space are obtained by the feature space decomposer block in order to handle the uncertainty of an input pattern. The if–then rule generator block finds the fuzzy if–then rules in terms of a relational matrix.

3.1. Feature space decomposer

In the proposed method, the training sample sets are initially subdivided into a few subclasses depending on the relative positions of the classes (training samples) in the feature space. Finally to represent the obtained subclasses (and therefore the original pattern classes), the whole feature space is decomposed into q hyperboxes $S_1, S_2, \ldots, S_h, \ldots, S_q$ where each S_h consists of N fuzzy sets [equation (4)]. All the hyperboxes are extended for the sake of incorporating the portions of the classes possibly uncovered by the training samples. Therefore, q overlapping hyperboxes are finally obtained here to represent the pattern classes to a better extent.

The proposed method of partitioning feature space is now illustrated considering a 3-class (denoted by A, B and C) and 2-feature (F_1 and F_2) problem (i.e. $M = 3$ and $N = 2$). This is shown in Fig. 3(a), where the classes A, B and C have 35, 24 and 12 samples respectively (the samples are shown by corresponding class characters). Note that there exists a overlap between the classes A and B. The proposed procedure always tries to extract the unambiguous (non-overlapping) or least ambiguous regions (in case there is no unambiguous region) from the feature space. This is done by comparing the possible regions obtained by proceeding from the lower and upper directions of each axis separately [as shown in Fig. 3(b)].

Initially only one hyperbox is assumed which includes all the pattern classes as shown in Fig. 3(c). The regions which can be extracted proceeding from the lower and upper sides of the features F_1 and F_2 are distinctly marked in Fig. 3(d) and (e) respectively. Among these regions, the two obtained considering the F_1 feature [Fig. 3(d)] are seen to be unambiguous (i.e. containing only one class). And between these two, the region obtained by proceeding from the lower direction of F_1 is seen to cover more samples and therefore, the samples in this region are kept in S_1 and the remaining samples are put into a newly generated hyperbox S_2. Correspondingly, the samples of the pattern class A are decomposed into two subclasses A_1 and A_2 such that S_1 includes only the subclass A_1 and S_2 consists of the samples from the (sub)classes A_2, B and C as shown in Fig. 3(f). Note that S_1 is now unambiguous by representing only the subclass



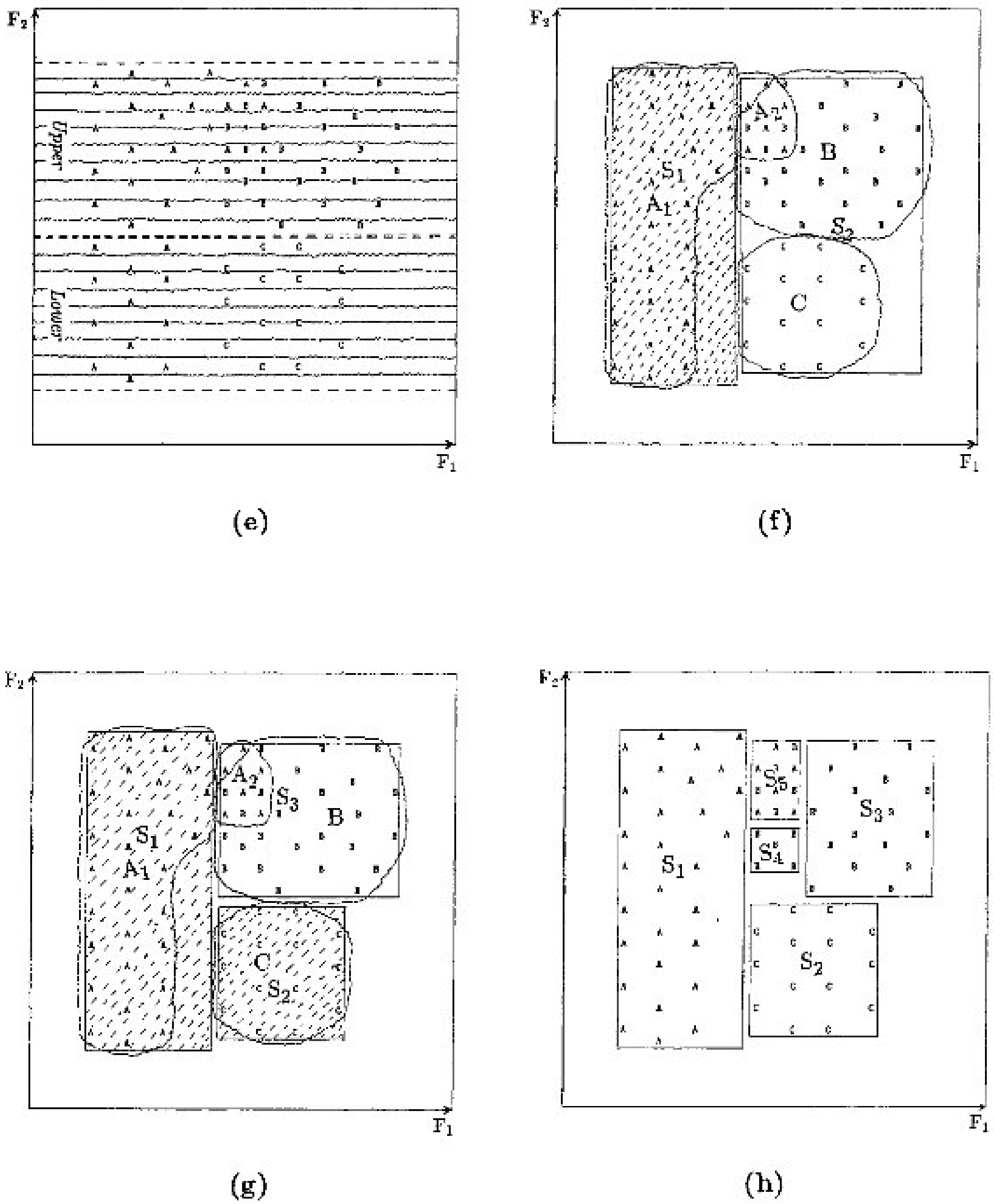
(a)

(b)

(c)

(d)

Fig. 3. (a)–(i). Illustrating the proposed feature space decomposition procedure.



(e) (f)

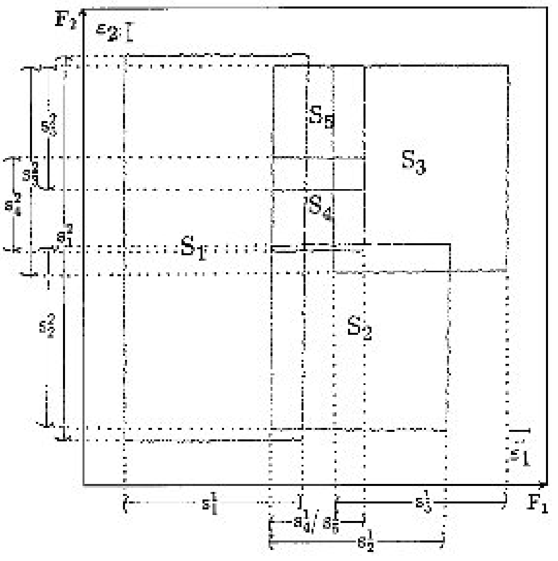
(g) (h)

Fig. 3. (Continued).

A_1. To process further, S_2 is decomposed into S_2 (keeps only the samples from class C) and S_3 (includes samples from the (sub)classes A_2 and B). This is shown in Fig. 3(g).

Proceeding similarly, the decomposition method ends with five regions or hyperboxes S_1, S_2, S_3, S_4 and S_5 where the first four regions are unambiguous representing only one (sub)class, and S_5 is ambiguous since it contains samples from two (sub)classes A_2 (i.e. A) and B as shown in Fig. 3(h). The generated regions are extended using coverage factors ε_1 and ε_2 corresponding to features F_1 and F_2 respectively to represent portions of the pattern classes possibly uncovered by the training samples and also to handle the overlap between the classes. The said extension makes the hyperboxes overlapping as shown in Fig. 3(i) where the feature ranges of the hyperboxes in individual feature axes are distinctly marked. Thus, the proposed decomposition method finally finds five (i.e. $q = 5$) overlapping hyperboxes corresponding to the classification problem in Fig. 3(a).

During the process of generating hyperboxes, we initially find candidate regions considering only one



(i)

Fig. 3. (Continued).

feature at a time and approaching from the lower and upper sides of each individual feature axis. That is, for N features, $2N$ candidate regions are obtained at any instance. Among these $2N$ candidates, one is selected which is optimum in terms of capturing the minimum number of classes (i.e., is least ambiguous) and at the same time incorporates maximum number of sample points. To find the candidate regions, we make use the coverage factors (ε_is) to avoid the formation of very small regions as such small regions are likely to incorporate negative effects on the classification of the proposed method.

The basic concepts of proposed feature space decomposition procedure are summarized below.

Basic concepts. Decomposition of feature space.

Step 1. Initially consider only one hyperbox with all the training samples and assume it as the current hyperbox.

Step 2. From the training samples in the current hyperbox, find candidate regions considering each of the features separately and also proceeding from both the lower and the upper sides of the feature axes.

Step 3. Among the candidates, one is selected which is optimum in terms of containing the minimum number of classes and incorporating the maximum number of sample points. Accordingly some subclasses are decomposed into two parts. The subclasses covered by the selected region are kept in the current hyperbox and the remaining subclasses (if any) are assigned to a newly generated hyperbox.

Step 4. In case the current hyperbox is found to be further decomposable (to decrease the ambiguity among the subclasses in the current hyperbox), go back to step 2 (i.e., repeat the same procedure with the current hyperbox). Otherwise processing of the current hyperbox is complete and its representative membership functions (T_h^i) corresponding to each axes are obtained.

Step 5. Processing is stopped if either all the hyperboxes are unambiguous or any new unambiguous or less ambiguous hyperbox can not be extracted from the existing ambiguous hyperboxes.

Therefore, the feature space decomposer block finds q overlapping hyperboxes using the training samples of M pattern classes. The detailed algorithm for decomposing feature space is furnished in the appendix.

3.2. *If–then rule generator*

Corresponding to each hyperbox (generated in the previous block) and each pattern class, one fuzzy if–then rule is generated which denotes the possibility of representing the pattern class by the hyperbox. As mentioned in Section 2, a relational matrix $\mathscr{R}$ [equation (6)] is introduced to represent the fuzzy if–then rules. The order of $\mathscr{R}$ is $q \times M$, and it denotes the compatibility of various pattern classes with the hyperboxes. Each column of $\mathscr{R}$ corresponds to a class and each row of that column denotes the degree of representing the class (based on the training samples) by the corresponding hyperbox.

Determination of $\mathscr{R}$. The relational matrix $\mathscr{R}$ is estimated based on the training samples of various pattern classes represented by different hyperboxes. Let the (h,j)th element of $\mathscr{R}$ (i.e. the element corresponding to hth hyperbox (S_k) and jth class (C_j)) be denoted by r_{hj}, where

$$r_{hj} = \begin{cases} 1 & \text{if } S_h \text{ represents only } C_j; \\ (0.5)^{ntt_h/ncs_h nt_j^h} & \text{if } S_h \text{ represents } C_j, \text{ along} \\ & \quad \text{with some other classes;} \\ 0 & \text{if } S_h \text{ does not represent } C_j \\ & \quad h = 1,2,\ldots,q, \\ & \quad j = 1,2,\ldots M., \end{cases} \tag{7}$$

Here ncs_k is the number of pattern classes represented by S_h; nt_j^h is the number of training samples from the class C_j represented by S_h, and ntt_h is the total number of training samples in S_h, i.e., $ntt_h = \sum_{j=1}^{M} nt_j^h$. If $ncs_h = 1$ and S_h represents C_j then $r_{hj} = 1$ and $r_{hk} = 0$ for all $k \neq j$. Otherwise (i.e. $ncs_h > 1$), S_h contains samples from more than one class. The factor $ntt_k/(ncs_h nt_j^h)$ is used as a density factor for the class C_j in S_h (overlapping). The number 0.5 is used in equation (7) for ambiguous regions as it represents the most ambiguous value in the fuzzy membership concept.

To illustrate the proposed way of determining the relational matrix $\mathscr{R}$, let us consider again the pattern class problem in Fig. 3(a). Recall that in this problem we obtained five hyperboxes $S_1, S_2, \ldots, S_5$ [Fig. 3(h) and (i)], where S_1 contains only class A, S_2 contains only class C, both S_3 and S_4 contain only the class B, and S_5 is an ambiguous hyperbox representing two classes A and B. In

S_5, there are six samples from class A and five samples from class B (i.e. the possibility of an unknown pattern to belong to A is slightly higher than that to belong to B). Using equation (7), one can find the relation matrix $\mathscr{R}$ in this case as follows:

$$\mathscr{R} = \begin{pmatrix} 1.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.0 \\ 0.0 & 1.0 & 0.0 \\ 0.0 & 1.0 & 0.0 \\ 0.81502 & 0.78235 & 0.0 \end{pmatrix}.$$

So the if–then rule generator block finds the relational matrix $\mathscr{R}$ which is utilized in the next section to decide the output of the proposed system.

4. FUZZY PROCESSOR

This section is divided into two parts, namely, *class possibility finder* and *output decider*. The class possibility finder uses the relational matrix $\mathscr{R}$ to determine the possibility values of an unknown pattern to different pattern classes. The final class assignment to one or more classes is carried out in the output decider block.

4.1. Class possibility finder

Here, we initially find the membership values of an unknown pattern X in each S_h. Recall that the feature space decomposer block finds q hyperboxes $S_1, S_2, \ldots, S_h, \ldots, S_q$, where each S_h $(h = 1, 2, \ldots, q)$ is the Cartesian product of the supports of the fuzzy sets $s_h^1, s_h^2, \ldots, s_h^i, \ldots, s_h^N$ [equation (4)]. Each of these fuzzy sets (s_h^i) is characterized by a symmetric piecewise linear triangular function $T_h^i\ (x_i; \alpha_h^i, \beta_h^i, \gamma_h^i)$ [equation (5)].

Characteristic vector. The membership values of an unknown pattern X to belong to the hyperboxes $\{S_h\}$ are denoted by a vector $V(X)$, named the characteristic vector, as

$$V(X) = (v_1(X), v_2(X), \ldots, v_h(X), \ldots, v_q(X))', \quad (8)$$

where $v_h(X)$ represents the membership value of X in S_h $(h = 1, 2, \ldots, q)$.

A typical pattern X consists of individual feature values, i.e. $X = (x_1, x_2, \ldots, x_N)$, where x_i denotes the value of ith feature F_i $(i = 1, 2, \ldots, N)$. Initially each individual feature information is considered separately to find the membership values to the fuzzy sets s_h^i $(i = 1, 2, \ldots, N;\ h = 1, 2, \ldots, q)$. Let $m_h^i(x_i)$ denote the membership value of x_i to the fuzzy set s_h^i. The value of $m_h^i(x_i)$ is obtained directly from the corresponding membership function as

$$m_h^i(x_i) = T_h^i(x_i; \alpha_h^i, \beta_h^i, \gamma_h^i), \quad i = 1, 2, \ldots, N, \\ h = 1, 2, \ldots, q. \quad (9)$$

Then the value of $v_h(X)$ is decided as the Geometric Mean (GM) of $m_h^1(x_1), m_h^2(x_2), \ldots, m_h^N(x_N)$ as

$$v_h(X) = (m_h^1(x_1) \times m_h^2(x_2) \times \cdots \times m_h^i(x_i) \\ \times \cdots \times m_h^N(x_N))^{1/N}, \quad h = 1, 2, \ldots, q. \quad (10)$$

Here instead of GM operator, some other operators such as the *AM* (Arithmetic Mean), *min* (Minimum) etc. could also be used.

Possibility vector. The possibility (similarity) values of the pattern X to belong to various pattern classes are denoted by a vector $P(X)$, named the *possibility vector*, as

$$P(X) = (p_1(X), p_2(X), \ldots, p_j(X), \ldots, p_M(X))', \quad (11)$$

where $p_j(X)$ denotes the possibility that X belongs to class C_j $(j = 1, 2, \ldots, M)$. The value of $p_j(X)$ is

$$p_j(X) = \max_{h=1,2,\ldots,q} \{u_h^j(X)\}, \quad j = 1, 2, \ldots, M, \quad (12)$$

where

$$u_h^j = (v_h(X) \times r_{hj})^{1/2}.$$

Here r_{hj} is the (h,j)th element of the relational matrix $\mathscr{R}$. $u_h^j(X)$ in equation (12) may be interpreted as the membership of X in the class C_j with respect to S_h.

The operation in equation (12) is very similar to Zadeh's max–min compositional rule of inference.[13] The *min* operator of the max–min operation of Zadeh is replaced here by the geometric mean (GM) operator. In other words, in the proposed system we have incorporated Zadeh's compositional rule of inference in a slightly modified way, i.e. we have incorporated *max–GM compositional rule of inference*. The *min* operator finds the minimum of any two elements, i.e. it provides the connective information. On the other hand, the GM operator provides collective information and a slight change in any of the elements is reflected by the GM operator. Similar arguments hold for the arithmetic mean (AM) operator as done in reference (21).

Neighboring effect. If $u_h^j(X)$ is positive for more than one S_h for fixed j, the pattern X belongs to C_j through all those hyperboxes with varying possibilities. In such a case, all those hyperboxes are neighbors and combinely represent the class C_j. In other words, the said sample belongs to the central portion of a region formed by combining all those hyperboxes, i.e. X belongs to more inside of the class C_j than found through the hyperboxes. This in turn increases the possibility of the pattern X to be in class C_j as obtained by equation (12). This is referred to here as the *effect of neighboring regions* which is incorporated as

$$\hat{p}_j(X) = [p_j(X)]^{1-\rho}, \quad (13)$$

where

$$\rho = \max_{h=1,2,\ldots,q,\ u_h^j(X) \neq p_j(X)} \{u_h^j(X)\}, \quad j = 1, 2, \ldots, M.$$

To illustrate the concept of neighboring effect, let us consider again the classification problem of Fig. 3(a) where five overlapping hyperboxes $S_1, S_2, \ldots, S_5$ [Fig. 3(h) and (i)] were found. Here class A is represented by both S_1 and S_5, and class B is represented jointly by S_3, S_4 and S_5. With the use of the neighboring effect, the possibility value of a pattern X for class A increases in the region $S_1 \wedge S_5$ ($\wedge$ denotes intersection), and the possibility value of X for the class B increases in the regions

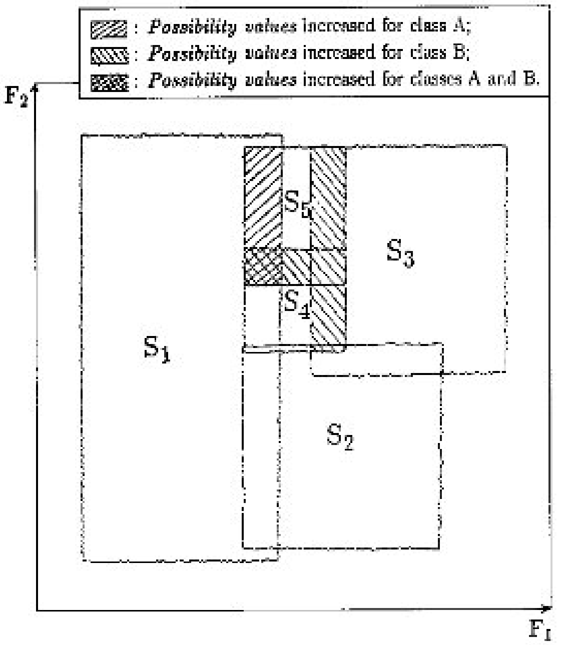

▨ : *Possibility values* increased for class A;
▧ : *Possibility values* increased for class B;
▩ : *Possibility values* increased for classes A and B.
F_2
S_5
S_3
S_4
S_1
S_2
F_1

Fig. 4. Illustrating the concept of neighboring effect.

$S_3 \wedge S_4$, $S_3 \wedge S_5$ and $S_4 \wedge S_5$ (whereas in the region $S_3 \wedge S_4 \wedge S_5$, the possibility value of X increases for both A and B). The possibility value of X for all the classes is unchanged with the neighboring effect in the remaining portion of the feature space. This is shown in Fig. 4. Note that operation in equation (13) is heuristic to a great extent.

4.2. *Output decider*

The possibility vector $P(X)$ is analyzed here to find the final class assignment of the unknown pattern X. The proposed classification system provides multiple class choices corresponding to input patterns belonging to the ambiguous feature regions, and it provides single class choices for inputs belonging to the unambiguous regions. Further, the system does not provide any class assignment for those input patterns belonging to the regions not represented by the hyperboxes (i.e. the similarities of such samples with the considered pattern classes based on the training samples are assumed to be insignificant to assign them to any of the classes). So, the proposed system decides as

$$\text{Assign } X \text{ to class } C_j \text{ if } \hat{p}_j(X) > \hat{p}_{j'}(X),$$
$$\text{for } j' = 1, 2, \ldots, M, j' \neq j. \qquad (14)$$

If X belongs to the nonoverlapping regions of the feature space, only one entry in $P(X)$ is positive and the corresponding class is taken as the output [equation (14)]. But if X lies in an overlapping region, more than one class in $P(X)$ would be positive and the class corresponding to the highest entry is taken as the output. In such cases, in addition to the output in equation (14) if the other classes for which the entries in $P(X)$ are positive are indicated as the *second* and *other choices* (in order of preference), the ambiguity of the samples lying in the ambiguous regions may be decreased. The output found by equation (14) is referred to as the *first choice* to distinguish from the *second* and *other choices*. Note that the *second* and *other choices* select only a few classes (in order of preference). Again, when X belongs to the feature regions not covered by the training sample sets (with extended portions), there will be no positive entry in $P(X)$. In such cases, the system does not assign the unknown pattern X to any of the classes, i.e. *null choice* is given.

So the output decider block finds the final class assignment of an unknown pattern X in terms of *first*, *second*, *other* and *null choices*. It is to be noted that the *first choice* may be considered as equivalent to the output of the conventional existing classifiers. Thus, to compare the performance of the proposed system with the existing classifiers, we consider only the *first choices*.

5. HANDLING IMPRECISE DATA

In the previous sections, we have described the proposed methodology considering only precise (numerical) data. But the data is sometimes found to be imprecise and/or incomplete in many real problems. Most conventional classifiers can only handle numerical data and patterns having imprecise data are usually ignored in their learning and classification phases.

On the other hand, the proposed classifier can handle various imprecisions in both learning and classification phases. We have grouped the imprecisions into four categories, namely, incomplete (partial), mixed, interval and linguistic. Different strategies are adopted to handle the imprecisions in each of these categories. Our strategies are described in this section.

5.1. *Incomplete data*

By incomplete data, we mean samples for which some of the feature values are not known. In many problems, the ability to deal with missing and/or uncertain data is crucial. A few attempts have recently been made to handle this problem. Ahmed and Tresp[29] discussed Bayesian techniques for extracting class probabilities from imprecise data with missing or noisy inputs. Ishibuchi *et al.*[30] transformed incomplete data with missing feature values into interval data (as the total feature range).

Our approach here is to ignore only the missing feature values of each sample and utilize all the available feature values in the learning phase (i.e. to decompose the feature space in generating fuzzy if–then rules). In the process, the subdivision of a (sub)class into two subclasses is carried out based on a particular selected feature at a time. If the value of the selected feature is missing for some samples, the samples are put in both the subclasses. On the other hand, if the missing feature is different from the considered feature, we carry out the decomposition process as usual.

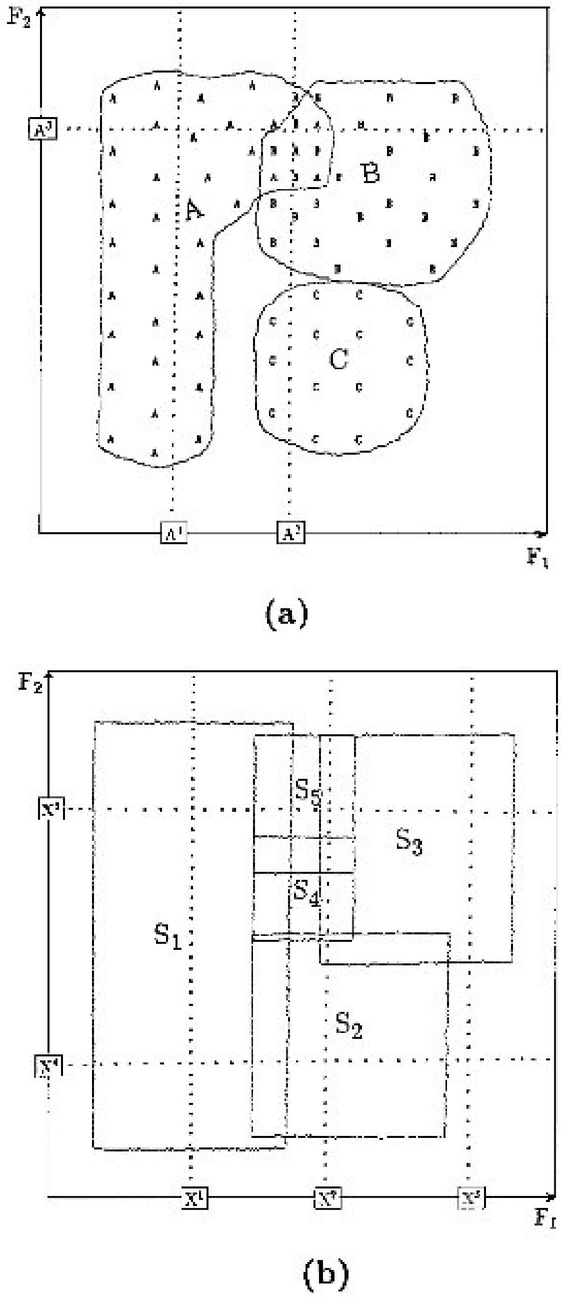


Fig. 5. Illustrating the proposed ways of handling incomplete data in (a) learning phase and (b) classification phase.

The proposed approach to handle incomplete samples in the learning phase is illustrated here with the classification problem of Fig. 3(a). We have included now three incomplete samples A^1, A^2 and A^3 (from class A) where the first two are of the form $(x_1, ?)$ and the third one is of the form $(?, x_2)$ (here "?" indicates that the corresponding feature value is missing). For a better realization of the situation, the feature diagram of Fig. 3(a) is reproduced in Fig. 5(a) where the three incomplete values are shown in the axes of the available features. Recall that to find the first hyperbox S_1 [Fig. 1(f)], the training samples of class A are decomposed into two subclasses A_1 and A_2 based on F_1 feature values. In this stage, the incomplete training samples A^1 and A^2 would be assigned to the subclasses A_1 and A_2 respectively based on only their F_1 values (as F_2 values are missing). But the sample A^3 would be assigned to both A_1 and A_2 as its F_1 value is missing.

In the classification phase, we assign some low (say, 0.2) membership value to the individual feature ranges of all hyperboxes corresponding to any missing feature value of an input pattern. It is done to decide the output based on the available partial (or incomplete) data. The logic behind assigning low membership values for missing data is to bring down the confidence of the system's output and subsequently to allow in classifying such samples based on the available feature values.

To illustrate the adopted approach to classify incomplete samples, the feature diagram of Fig. 3(i) is reproduced here in Fig. 5(b) by adding five missing input samples $X^1, X^2, \ldots, X^5$. The samples X^1, X^2 and X^3 have only the F_1 feature values (F_2 values are missing) where as the samples X^4 and X^5 have only the F_2 feature values (F_1 values are missing). Based on the available feature values, one is likely to classify these samples [consulting the feature diagram of Fig. 5(b)] as shown on Table 1. The proposed approach finds the same classification as in Table 1 for the missing samples $X^1, X^2, \ldots, X^5$ in the classification problem of Fig. 5(b).

5.2. *Mixed data*

Data in this form is a mixture of linguistic hedges[31] and numerical terms, i.e. linguistic hedges applied to numerical terms. The linguistic hedges considered here are "more or less", "about", "approximately" etc., each of which represents an approximate feature range around the numerical term. In other words, linguistic hedges add some impreciseness to the numerical term. Examples of this form are "x_i is more or less 10", "x_i is about 10" etc.

In the learning phase (i.e. to decompose the feature space), we ignore the linguistic hedges and process it with only the numerical term alone.

But in the classification phase, the membership values of data in this form for different feature ranges of the hyperboxes are decreased from that of the membership values of the data with numerical value alone. The

Table 1. Possible classification of five incomplete samples of Fig. 6(b)

Missing samples	Responding hyperboxes	Possible classes
X^1	S_1	A
X^2	S_2, S_3, S_4, S_5	A, B, C
X^3	S_3	B
X^4	S_1, S_2	A, C
X^5	S_1, S_3, S_5	A, B

amount of decrease is determined according to the linguistic hedges. As an example, for the sample with ith feature value as "x_i is about a", the membership value corresponding to s_h^i is assigned as

$$m_h^i(x_i \text{ is about } a) = \{m_h^i(x_i \text{ is } a)\}^{1.25}, \qquad (15)$$

where $m_h^i(\cdot)$ represents the membership value corresponding to s_h^i.

Similar to the missing data, the logic behind adopting the previous modifications [e.g. equation (15)] of the membership values is also to bring down the confidence of the system's output.

5.3. *Interval data*

Like the mixed form, the data in interval form is also mixture of linguistic hedges and numerical terms. The basic difference lies with the type of linguistic hedge used. Hedges considered for this form are "less than", "more than", "between" etc. which represent interval data sets. Examples of this form are "x_i is less than 15", "x_i is between 10 and 15" etc.

Let us first consider here a sample with ith feature value as "x_i is less than a". At the time of decomposing the (sub)class into two subclasses (in the learning phase), if the selected feature is F_i then we put the sample in two subclasses in case the threshold value is less than a. In case the threshold value is more than a, the sample is put in one subclass which includes the samples with the ith feature values less than the threshold value.

Processing of other linguistic hedges like "more than", "between" etc. is similarly carried out in the learning phase.

In the classification phase, either a low (say, 0.2) or zero membership value is assigned to an individual feature range of the hyperboxes (as obtained in the learning phase) depending on whether the interval data set (of the sample) has any intersection with the said individual feature range by such a sample or not. The other operations remain the same.

5.4. *Linguistic data*

Data in this form is completely linguistic (i.e. with only linguistic hedges and linguistic variables) such as "x_i is small" or "x_i is more or less high".

To handle linguistic samples, the system assumes only three primary variables, namely, small, medium and high, and the corresponding membership functions considered as $1-S$, π and S functions[14,31], respectively. Using the *a priori* knowledge, the values of the parameters of the membership functions are assigned.

As long as the membership functions are chosen properly, one recovers[32] the entirety of the classical logic for the designations of true and false. This implies that the system finds an interval of the truth values corresponding to a linguistic data. Here the system assumes that true $= [0.5,1]$, false $= [0,0.5]$ and extended this particular logic by adding

$$\begin{aligned} \text{very true} &\equiv [0.8, 1.0], \\ \text{more less true} &\equiv [0.6, 0.8), \\ \text{neither true nor false} &\equiv [0.4, 0.6), \\ \text{more or less false} &\equiv [0.2, 0.4), \\ \text{very false} &\equiv (0.0, 0.2). \end{aligned} \qquad (16)$$

So corresponding to this type of interval based truth values [equation (16)], one can find an interval of feature values that can be considered an equivalent of any linguistic data. That means, the system converts linguistic data into interval data and processes them as in the interval form.

6. EXPERIMENTAL RESULTS

The effectiveness of the proposed classification system has been verified with some synthetic data as well as with four real-world tasks. We find it convenient to test the proposed methodologies first on the synthetic data sets in two-dimensional feature space as the results can easily be realized. Finally we have considered four real data sets to show the practical applicability of the proposed system.

6.1. *Synthetic data*

To verify the effectiveness of the proposed system, different possible pattern classes were generated artificially and the proposed algorithm was implemented on them. Results were found to be quite satisfactory in all the cases. Results are demonstrated here with four such typical pattern class problems in two-dimensional feature space as shown in Fig. 6(a)–(d). In Fig. 6(a), there are five classes (denoted by C_1, C_2, C_3, C_4 and C_5) where the classes are elliptical in shape and are assumed to follow truncated Gaussian distributions with different mean vectors and variance–covariance matrices. There are three classes (denoted by C_1, C_2 and C_3) in Fig. 6(b) and two classes (denoted by C_1 and C_2) in both Fig. 6(c) and (d). The classes in Fig. 6(b)–(d) are assumed to follow uniform distribution over the regions shown in the figures.

For implementing the proposed algorithm, five different training sample sets were initially generated for each of the previous four pattern class problems. The sizes of the sample sets are 250 (with 50 samples from each class), 200 (with 100, 50 and 50 samples from the classes C_1, C_2 and C_3, respectively), 200 (with 100 samples from each class) and 150 (with 100 and 50 samples from the classes C_1 and C_2 respectively) corresponding to the problems in Fig. 6(a)–(d) respectively. All these samples were generated following the underlying distributions of the classes.

The test data sets consist of 10,000 samples for each of the problems drawn randomly using their underlying distributions. The recognition scores for the considered four cases are shown in Tables 2–5 as obtained by averaging those corresponding to five different training sets. The recognition scores are grouped here into four categories, namely, *first correct*, *second correct*, *other correct* and *fully wrong choices*. The *first correct choice*

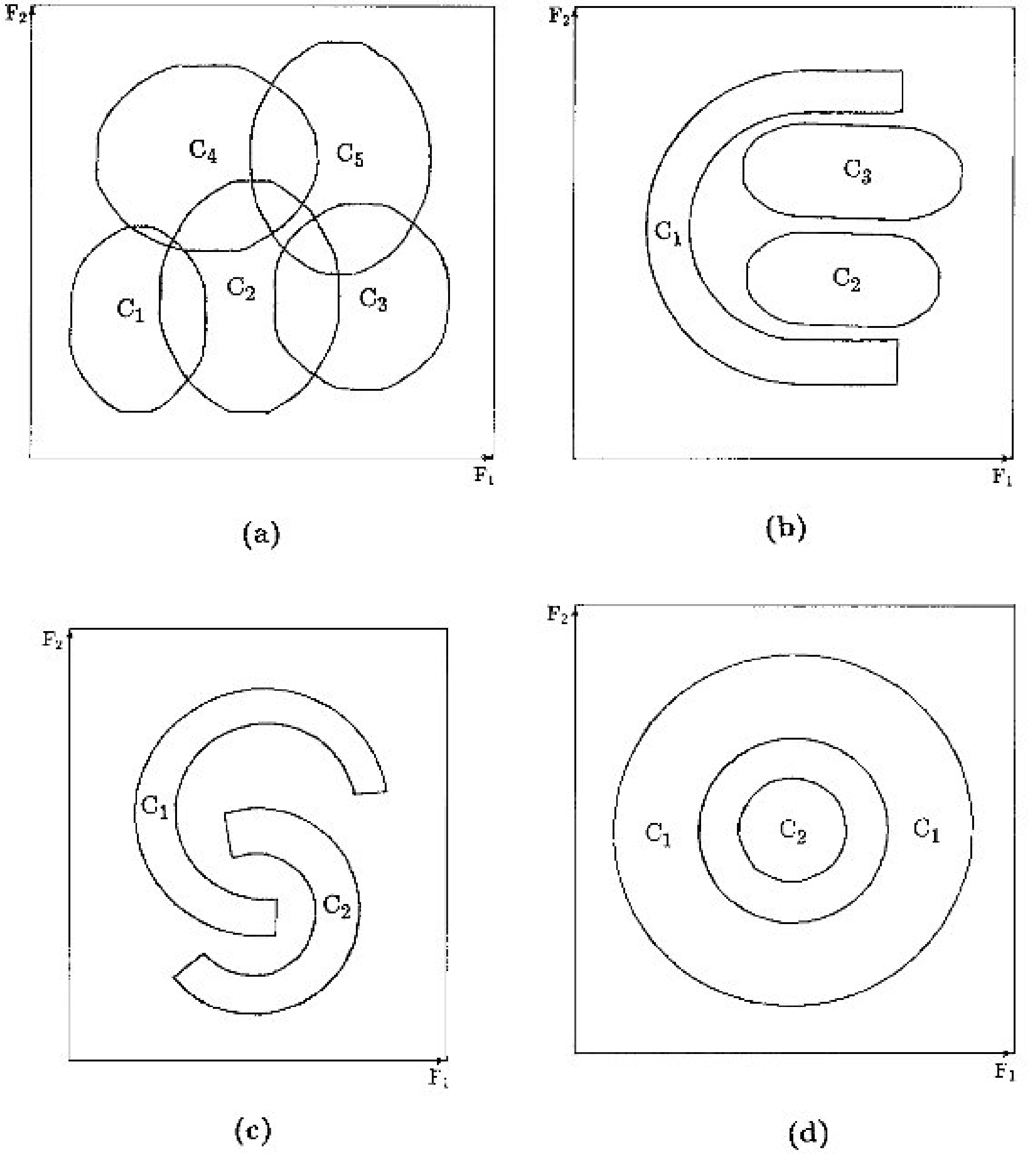


Fig. 6. (a)–(d). Four synthetic data sets.

set includes those samples for which the system's *first choices* correspond to their actual classes. Depending on whether the *second* or *other choices* correspond to their actual classes, the samples are put either in *second* or *other correct choice* group. Samples not falling under the aforesaid categories are termed as misclassification or *fully wrong choices*. Note that the *second correct choice* and *other correct choice* categories are omitted from Tables 2,3,4 and 5 as all the samples were correctly classified by the *first choices*.

In order to observe the relative performance of the proposed method with the existing classifiers, we have tried to implement the Bayes classifier[1] on the synthetic data sets [Fig. 6(a)–(d)]. Note that the Bayes classifier is the most well known and established classifier. If the classes are of regular shapes and if their distributions can be obtained nicely, the results of the proposed classifier (with only *first correct choices*) are quite comparable with that of the Bayes classifier. For example, the classes in Fig. 6(a) are of regular (elliptical) shapes and the recognition score of the Bayes classifier (assuming Gaussian class density) was found to be 81.76% where as the recognition score of the proposed classifier with *first correct choices* is 83.12%. Again analyzing the results, it has been found that our output decisions with various output forms are more natural and justified.

When the pattern classes are not of regular shapes [Fig. 6(b)–(d)], it is extremely difficult to find their

Table 2. Recognition score for the pattern classes as in Fig. 6(a)

Various group	% Recognition score					
of	Actual classes					Overall
choices	C_1	C_2	C_3	C_4	C_5	score
First Correct Choice	87.92	73.17	86.91	82.98	84.61	83.12
Second Correct Choice	9.56	13.58	8.99	12.27	11.21	11.12
Other Correct Choice	2.52	12.32	3.83	4.61	4.12	5.48
Fully Wrong Choice	0.00	0.93	0.27	0.14	0.06	0.28

Table 3. Recognition score for the pattern classes as in Fig. 6(b)

Various group	% Recognition score			
of	Actual classes			Overall
choices	C_1	C_2	C_3	score
First Correct Choice	100.0	100.0	100.0	100.0
Fully Wrong Choice	0.0	0.0	0.0	0.0

Table 4. Recognition score for the pattern classes as in Fig. 6(c)

Various group	% Recognition score		
of	Actual classes		Overall
choices	C_1	C_2	score
First Correct Choice	100.0	100.0	100.0
Fully Wrong Choice	0.0	0.0	0.0

Table 5. Recognition score for the pattern classes as in Fig. 6(d)

Various group	% Recognition score		
of	Actual classes		Overall
choices	C_1	C_2	score
First Correct Choice	100.0	100.0	100.0
Fully Wrong Choice	0.0	0.0	0.0

distribution functions.[7] However, in such cases the densities can be estimated by some non-parametric methods [e.g. Parzen[33] and Cacoullus[34] window methods, k-nearest neighbor density estimation method[35]] for the application of the Bayes classifier. On the other hand, the proposed system provides good results [Tables 2,3,4 and 5] irrespective of the shapes and class densities of the data sets.

For the pattern classes in Fig. 6(b)–(d), the 1-nearest neighbor decision rule[36] may give very satisfactory results. However, the purpose of considering the pattern classes in Fig. 6(a)–(d) is to show that the proposed system can be applied satisfactorily to different possible structures of the pattern classes.

6.2. *Real-world problems*

To examine the practical applicability of the proposed system, the algorithm has been implemented on four real data, namely, the Iris data set, an appendicitis data set, a speech data set and a hepatic disease data set. Our results are found to be better than the existing results on all these data sets.

Iris data.[37] In this problem, three classes of Iris flowers are to be discriminated using four continuous valued features that represent physical characteristics of the flowers. The data set consists of 150 samples, 50 for each class.

We have evaluated the performance of the proposed algorithm on the Iris data set using the leaving-one-out method. In this method, one sample is considered as the test data and the remaining 149 samples are used as the training set. The method is repeated 150 times by selecting each of the samples as the test data. The results are shown in Table 6, where all the samples were classified correctly by the *first choices*.

Appendicitis data.[38] This data set is of the patients admitted to an emergency room with a tentative diagnosis

Table 6. Recognition score for the Iris data set

Various group	% Recognition score			
of	Actual classes			Overall
choices	C_1	C_2	C_3	score
First Correct Choice	100.0	100.0	100.0	100.0
Fully Wrong Choice	0.0	0.0	0.0	0.0

Table 7. Recognition score for the appendicitis data set

Various group	% Recognition score		
of	Actual classes		Overall
choices	C_1	C_2	score
First Correct Choice	100.0	100.0	100.0
Fully Wrong Choice	0.0	0.0	0.0

of acute appendicitis. The problem here is to discriminate the true appendicitis patients (C_1) from the healthy person (C_2). The samples consists of eight diagnostic tests (i.e. eight features) and 106 patients (85 and 21 samples from classes C_1 and C_2 respectively). Here some samples are incomplete as those do not have one particular test (feature) data.

The performance of the proposed system on this data set is shown in Table 7 using leaving-one-out method where the recognition is seen to be 100%.

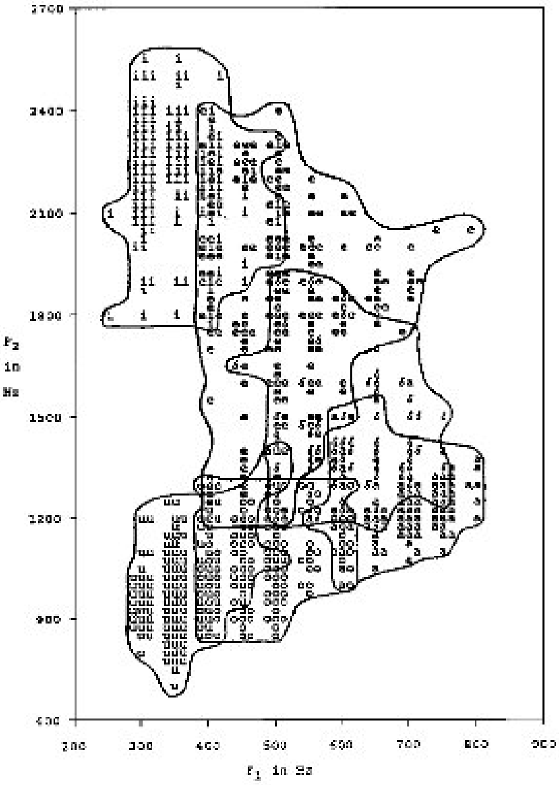


Fig. 7. Vowel classes in the $F_1 \times F_2$ plane.

Speech data.[16] This data set consists of Indian Telugu vowel sounds in consonant-vowel-consonant context uttered by three speakers in the age group 30–35 yr. Fig. 7 shows the typical feature space of six vowel classes /ð/, /a/, /i/, /u/, /e/ and /o/ with 72, 89, 172, 151, 207 and 180 samples, respectively corresponding to the features F_1 and F_2. Here F_1 and F_2 denote the first and second formant frequencies which were obtained through spectrum analysis of the speech data. Details of the feature extraction procedure can be found in reference (16). The classes are seen to be overlapping and their boundaries are ill-defined (fuzzy).

The data set includes the aforementioned 871 precise samples and 102 imprecise (incomplete) samples. These imprecise samples on F_1 and F_2 were coded by the trained personnel. These imprecise samples were ignored by some of the earlier works[16,39] that were incapable of handling them.

To find the performance of the proposed algorithm on this data set, the total available 973 samples (including both precise and imprecise data) is divided 10 disjoint groups where seven are of size 97 and three are of size 98. The algorithm was repeated 10 times considering 10 different training sets. In each iteration, one group was considered as the training set and the remaining nine groups were considered as to constitute the test set. The results obtained by averaging those corresponding to 10 different iterations are shown in Table 8.

Hepatic disease data.[40,41] This data set is concerned with the medical diagnosis of hepatic diseases. It consists of 568 patients (samples) and 20 laboratory tests (features), and to be classified into five hepatic disease classes.

The available 568 samples were divided in reference (40,41) into two parts: the training set with 468 samples and the test set with 100 samples (20 from each class). We have also considered the same training and test data set to implement the proposed system on this data set. One of the important characteristics of these data is that a

Table 8. Recognition score for the vowel classes as in Fig. 7

Various group	% Recognition score						
of	Actual classes						Overall
choices	/∂/	/a/	/i/	/u/	/e/	/o/	score
First Correct Choice	67.04	88.67	94.24	93.82	88.61	81.37	85.00
Second Correct Choice	22.47	10.26	5.29	5.16	18.31	17.96	12.99
Other Correct Choice	8.74	1.07	0.87	1.02	2.44	0.67	1.70
Fully Wrong Choice	1.75	0.00	0.00	0.00	0.64	0.00	0.31

Table 9. Recognition score for the hepatic disease data set.

Various group	% Recognition score					
of	Actual classes					Overall
choices	C_1	C_2	C_3	C_4	C_5	Score
First Correct Choice	100.0	80.0	85.0	80.0	80.0	85.0
Second Correct Choice	0.0	10.0	5.0	15.0	15.0	9.0
Other Correct Choice	0.0	5.0	10.0	5.0	5.0	5.0
Fully Wrong Choice	0.0	5.0	0.0	0.0	0.0	1.0

Table 10. Misclassification rates (ERR_{LV1}) of some existing fuzzy classifiers on Iris and appendicitis data sets[19,12]

DATA	Existing fuzzy classifiers								
sets	PERP	QUAD	FHS	MIN	FcM	HFcM	HIER	FkNN	PFS
Iris	4.7	3.3	8.0	4.0	6.7	6.7	4.7	3.3	3.3
Appendicitis	20.8	13.2	15.1	13.2	28.8	21.7	19.8	13.2	–

significant number of samples does not have all the values of laboratory tests (the only tests carried out may be sufficient to decide the disease by a physician). In a sense, these samples are useful to verify the system's capability of handling incomplete (missing) samples and also of handling large number of features. The recognition system is learned by the above training set and obtained 104 fuzzy if–then rules. The results on the test set are furnished in Table 9.

6.3. *Comparison of results*

Results of the proposed system on four real data sets has been provided previously. These results are now compared with some well known fuzzy and classical classification algorithms based on their reported results in the literature.

Results of some existing classification algorithms based on fuzzy set theory on both Iris and appendicitis data sets were reported in reference (19) in terms of the estimated misclassification rates considering different methods such as hold out (learning and test on the whole data set), leaving-one-out and 2-fold cross validation (divide at random the data set into two parts, training respectively on each part and test on the whole) etc. The leaving-one-out method (ERR_{LV1}) is considered as the most representative to estimate the performance of a classification system. So we have used here the ERR_{LV1} values of various fuzzy classifiers from reference (19) for the proposed comparison on the Iris and appendicitis data.

In reference (19), the misclassification rates (ERR_{LV1}) were reported for four methods based on fuzzy pattern matching procedures[42] (fuzzy integral with perceptron (PERP), quadratic (QUAD) criteria, fast heuristic search (FHS) with Sugeno integral and fuzzy pattern matching with minimum operator (MIN)), three methods based on fuzzy clustering procedures[13] (fuzzy c-means (FcM), fuzzy c-means with histogram (HFcM) and hierarchical fuzzy c-means (HIER)) and the fuzzy k-nearest neighbor algorithm (FkNN).[43] Again, ERR_{LV1} of a recent fuzzy approach based on partitioning of feature space (PFS)[12] on Iris data set is also readily available for comparison purpose. The best results of estimated error rates (ERR_{LV1}) for these fuzzy approaches on both Iris and appendicitis data sets are furnished in Table 10.

For classical methods, we have used the results as available in reference (44,45) corresponding to the following methods: linear discriminant (LD) and quadratic discriminant (QD),[37] nearest neighbor (NN),[36] Bayes independent (BI) and Bayes second order (B2),[1,2] neural networks with back propagation (BP)[46] and ordinary differential equation of BP (ODE),[47] decision tree methods like optimal rule of length 2 (OR2),[48] classification and regression trees (CART),[49] predictive value maximization rule (PVMR).[50] The best values of

Table 11. Misclassification rates (ERR_{LV1}) of some existing classical classifiers on Iris and appendicitis data sets[44,45]

DATA	Existing classical classifiers									
sets	LD	QD	NN	BI	B2	BP	ODE	OR2	CART	PVMR
Iris	2.0	2.7	4.0	6.7	16.0	3.3	2.7	2.0	4.7	4.0
Appendicitis	13.2	26.4	17.9	17.0	18.9	14.2	13.2	10.4	15.1	10.4

Table 12. Overall recognition score (%) on Iris data set with different accuracy factors (δ)

Various group of choices	Accuracy factors (δ)							
	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08
First Correct Choice	100.0	100.0	100.0	100.0	100.0	100.0	100.0	99.33
Second Correct Choice	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.67
Fully Wrong Choice	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.00

ERR_{LV1} on both Iris and appendicitis data sets for these classical methods (as reported in references (44,45)) are reproduced in Table 11. It is to be mentioned here that as one feature of the appendicitis data was having some missing values, the results reported in references (19,44,45) were based on only the other seven feature values.

For comparing the performance of our procedure with the above two-state methods, we considered only the *first correct choices* as the correct decisions and the remaining are considered as the misclassification. So the values of ERR_{LV1} for the proposed method are found to be null (i.e. 0) on both Iris (Table 6) and appendicitis data sets (Table 7). Our recognition scores are seen to be higher than those of the existing methods as considered in references (19,44,45). We therefore can conclude that the proposed system is better (performance wise) than most of the existing classical and fuzzy classification methods.

It is to be observed from the vowel recognition problem that the confusion in recognizing a sample considering the *first choices* lies, in general, only with the neighboring classes constituting a vowel triangle. The similar finding were also obtained with the previous investigations[16,39] considering precise input/output. The overall recognition score of the proposed system is also found to be better than the earlier results[16,39] on the vowel data set.

The recognition score of our proposed algorithm on hepatic disease data set is far better than those of reference (40,41). In those approaches, the actual feature values were coded there with 3–6 numbers (0 ~ 5) for their implementation. Again, seven features from the available 20 features were first of all selected there using rough sets.[51] The recognition scores in references (40,41) were found to be 62.0 and 74.0%, respectively. In contract to this, we did not find any difficulty to utilize all 20 features with their actual (and missing) values and the recognition score (with only *first choices*) was found to be 85.0% with only 104 fuzzy if–then rules. For comparing the performance of the proposed system, we have also implemented the system based on the this data set with the same seven feature values (not codes) as used in references (40,41) and found 93 fuzzy if–then rules while the recognition score with *first choices* was observed as 81.0%.

Effect of accuracy factor (δ). In each of our experiments, the value obtained by the equation (2) is considered as the accuracy factor (δ). But the value of δ can be any real number satisfying the inequality (1). For example, for the Iris data set the value of δ was considered as 0.0441581 using the equation (2) whereas it can be any real number lying between 0.0066667 and 0.0816492 [equation (1)]. To show the effect of δ in the proposed method on the Iris data set, we have considered different values of δ and correspondingly estimated the performance of our algorithm on the Iris data set using leaving-one-out method as shown in Table 12.

The smaller values of δ are seen to provide better results than the higher values of δ on the Iris data set. But the smaller values of δ may find some samples to be classified as the null choices in many other problems (when the training set is not a good representation of their pattern classes). So we prefer to consider the value of δ as in equation (2).

7. CONCLUSIONS AND DISCUSSION

The present article proposed an efficient fuzzy partition of a feature space to generate fuzzy if–then rules for pattern classification. The basic idea of the method is to decompose the whole feature space into some overlapping hyperboxes depending on the relative positions of the pattern classes represented by their training samples. Therefore, a feature space is automatically divided here into a few hyperboxes of different sizes using the training samples. Ultimately a classification system has been formulated which has the flexibility of accepting the precise (numerical) as well as the imprecise (ambiguous) patterns both in learning and classification phases. The imprecise information are grouped into four categories, namely, incomplete, mixed, interval and linguistic in form.

The effectiveness of the system has been demonstrated on some synthetic data sets. The practical applicability of

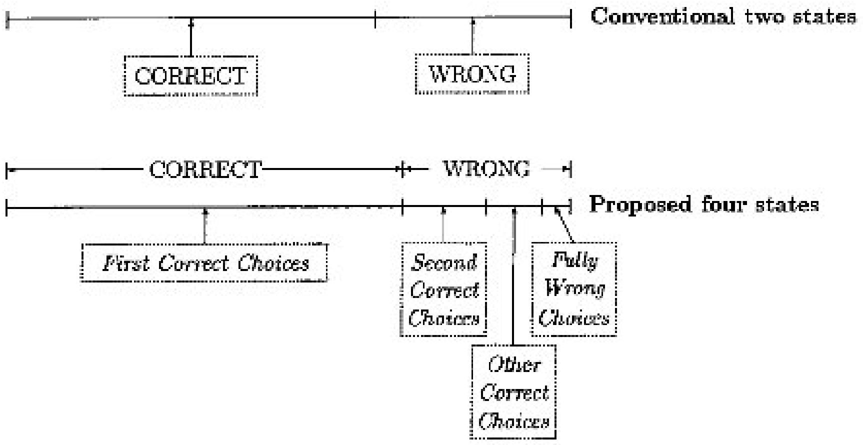


Fig. 8. Conventional two state versus proposed four state output.

the system is verified on four real data sets such as the Iris data set, an appendicitis data set, a speech data set and a hepatic disease data set. The basic characteristics of such data sets are that these consist of many features and moreover a good number of samples were seen to be incomplete and imprecise. The performance of the proposed system are found to be superior than the existing classification approaches on these data sets.

The output decisions of the existing conventional classifiers are usually categorized into two hard states as *correct* and *wrong*. Each of these classifiers is designed to apply to some particular situations. If they are applied in the situations different from their aimed ones, then satisfactory results, in general, are not obtained. Again, there exist some fuzzy set theoretic approaches in the literature which generates fuzzy if–then rules from numerical (precise) data for pattern classification problems. One of the serious shortcomings in these approaches is that these are not applicable to higher dimensional feature spaces. Most of the approaches are also incapable of handling imprecise data.

It is to be observed that the proposed system does not assume any distribution of the pattern classes. If the training samples represent the pattern classes to a reasonable extent, the proposed system is found to give good results in all possible situations. Moreover, the implementation of the proposed system on a problem (with many features and classes) is quite simple as can be observed from the detailed algorithm furnished in the appendix. The concept of accuracy factor (also coverage factors) is found to be very effective. In one side, it controls to represent the possible uncovered portions of the pattern classes by the training samples. On the other hand, it manages to avoid the formation of very small hyperboxes which could produce unexpected effects on the proposed classification method.

The recognition score of the proposed classification system is categorized into four states, namely, *first correct*, *second correct*, *other correct* and *fully wrong choices*. In other words, the *correct choices* of the conventional systems are equivalent to the *first correct choices* of the proposed system and the *wrong choices* have been decomposed here into *second correct*, *other correct* and *fully wrong choices*. This is explained in Fig. 8. Note further that if the training samples represent the classes to a reasonable extent, the *fully wrong choice* set becomes nearly null as observed from the experimental results in the previous section. Because of the flexibility, the proposed system has a provision of improving its efficiency significantly by incorporating *second* and *other choices* under the control of a supervisory scheme.

Acknowledgements—The author gratefully acknowledges Professors S. K. Pal and H. Tanaka for their interests in the work, and Dr H. Ishibuchi for various discussions with him which helped me to formulate the present classification scheme. The author also thanks Dr N. R. Pal for his constructive criticisms which greatly help to improve the presentation of the article.

APPENDIX A. FEATURE SPACE DECOMPOSITION PROCEDURE

The detailed algorithm of the proposed feature space decomposition procedure is described here. Let t_j be the number of training samples from class C_j and x_{ij}^k denote the ith feature value of the kth sample from class C_j $(i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M;\ k = 1, 2, \ldots, t_j)$. Assume that l_{ij} and u_{ij} denote respectively the lower most and upper most sample value of feature F_i in class C_j.

Initially depending on the relative positions of the classes (training samples) in the feature space, the training sample sets are subdivided into few subclasses. Assume that m represents the number of such subclasses and org(j) represents the parent (original) class of jth subclass $(j = 1, 2, \ldots, m)$.

In the proposed procedure, the whole feature space is decomposed into q overlapping hyperboxes. Let $ncss_h$ denote the number of subclasses represented by the hth hyperbox S_h $(h = 1, 2, \ldots, q)$, and $ss(h, k)$ denote the subclass corresponding to the kth member represented by S_h $(k = 1, 2, \ldots, ncss_h)$.

During the process of generating hyperboxes, we find candidate regions proceeding from the lower and upper sides of each individual feature axis. Assume that npr_d^i represents the number of classes contained in the candidate region considering direction d ($0 =$ lower; $1 =$ upper) of feature F_i ($i = 1, 2, \ldots, N$), and $\mathrm{pr}(d, i, k)$ denotes the subclass corresponding to the kth member in that candidate region. Assume also that $\mathrm{density}_d^i$ represents the number of sample points captured by the candidate region considering direction d along the feature axis F_i.

The basic concepts of the proposed feature space decomposition procedure have been introduced in Section 2. We now describe its detailed algorithm.

Algorithm: Decomposition of feature space.

Step 1: (Initialization)

(a) $i = 1$.
(b) $j = 1$.
(c) $l_{ij} = \min\limits_{1 \le k \le t_i} \left\{x_{ij}^k\right\}$; $u_{ij} = \min\limits_{1 \le k \le t_i} \left\{x_{ij}^k\right\}$.
(d) If $j < M$ then $j = j + 1$ and go to 1(c). Otherwise

$$\varepsilon_i = \left[\max_{1 \le j \le M} \{u_{ij}\} - \min_{1 \le j \le M} \{l_{ij}\}\right] \times \delta.$$

(e) If $i < N$ then make $i = i + 1$ and go to 1(b).
(f) $m = M$; $q = 1$; $h = 1$; $ncss_h = m$; $ss(h, k) = k$ for $k = 1, 2, \ldots, ncss_h$. (Here h represents the hyperbox currently being processed.)

Step 2: (Finding candidate regions)

(a) $i = 1$.
(b) $d = 0$ (Lower direction)

$$lb_i^d = \min_{j = ss(h,k),\, k = 1,2,\ldots,ncss_h} \{l_{ij}\}.$$

$npr_i^d = 0$.

$npr_i^d = npr_i^d + 1$ and $pr(d, i, npr_i^d) = h$

if $l_{ij} - lb_i^d \le \varepsilon_i$ where $j = ss(h, k)$

for $k = 1, 2, \ldots, ncss_h$.

Find

$$U_1 = \min_{j = ss(h,k),\, j \notin pr(d,i),\, k = 1,2,\ldots,ncss_h} \{l_{ij}\},$$
$$U_2 = \min_{j = pr(d,i,k),\, k = 1,2,\ldots,npr_i^d} \{u_{ij}\}.$$

If $(U_1 < U_2)$ and $((U_2 - U_1) > \varepsilon_i)$ then $ub_i^d = U_1$;

Otherwise if $(U_1 > U_2)$ and $((U_1 - U_2) > \varepsilon_i)$ then $ub_i^d = U_2$;

Otherwise $ub_i^d = (U_1 + U_2)/2$.

$density_i^d = 0$.
$density_i^d = density_i^d + t_j$

if $u_{ij} - ub_i^d < \varepsilon_i$ where $j = ss(h, k)$

for $k = 1, 2, \ldots, ncss_h$.

Again,

$density_i^d = density_i^d + 1$

if $u_{ij} - ub_i^d > \varepsilon_i$
and if $x_{ij}^{k1} \le ub_i^d$ where $j = ss(h, k)$

for $k = 1, 2, \ldots, ncss_h$ and $k1 = 1, 2, \ldots, t_j$.

(c) $d = 1$ (Upper direction).

$$ub_i^d = \max_{j = ss(h,k),\, k = 1,2,\ldots,ncss_h} \{u_{ij}\}.$$

$npr_i^d = 0$.
$npr_i^d = npr_i^d + 1$ and $pr(d, i, npr_i^d) = h$

if $ub_i^d - u_{ij} \le \varepsilon_i$ where $j = ss(h, k)$

for $k = 1, 2, \ldots, ncss_h$.

Find

$$L_1 = \max_{j = ss(h,k),\, j \notin pr(d,i),\, k = 1,2,\ldots,ncss_h} \{u_{ij}\},$$
$$L_2 = \max_{j = pr(d,i,k),\, k = 1,2,\ldots,npr_i^d} \{l_{ij}\}.$$

If $(L_1 > L_2)$ and $((L_1 - L_2) > \varepsilon_i)$ then $lb_i^d = L_1$;

Otherwise if $(L_1 < L_2)$ and $((L_2 - L_1) > \varepsilon_i)$ then $lb_i^d = L_2$;

Otherwise $lb_i^d = (L_1 + L_2)/2$.

$density_i^d = 0$.
$density_i^d = density_i^d + t_j$

if $lb_i^d - l_{ij} \le \varepsilon_i$ where $j = ss(h, k)$

for $k = 1, 2, \ldots, ncss_h$.

Again,
$density_i^d = density_i^d + 1$

if $lb_i^d - l_{ij} > \varepsilon_i$ and if $x_{ij}^{k1} \ge lb_i^d$ where $j = ss(h, k)$

for $k = 1, 2, \ldots, ncss_h$ and $k1 = 1, 2, \ldots, t_j$.

(d) If $i < N$ then make $i = i + 1$ and go to 2(b).

[In Step 2, various candidate regions are found proceeding from the lower (i.e. $d = 0$ as in 2(b)) and upper (i.e., d=1 as in 2(c)) sides of each individual feature axis. Note that in this stage we make use of the coverage factors (ε_is) by which the formation of very small regions are avoided. Otherwise such small regions could provide some unexpected effect on the classification method.]

Step 3: (Selection of optimum hyperbox from the candidate regions)

$$mincl = \min_{d = 0,1,\, i = 1,2,\ldots,N} \{npr_i^d\}.$$
$$maxdens = \max_{d = 0,1,\, i = 1,2,\ldots,N,\, npr_i^d = mincl} \{\mathrm{density}_i^d\}.$$

Find the direction D alongwith the feature axis I for which

$$\mathrm{density}_I^D = maxdens \text{ and } npr_I^D = maxcl.$$

Now the subclasses C_j's belonging to the set $pr(D,I)$ and satisfying $(u_{lj} - ub_I^D) > \varepsilon_I$ are decomposed into two subclasses. In such cases, the training samples for which their Ith feature value less than or equal to ub_I^D are kept in the same subclass. Then a new subclass is generated as $m = m + 1$ and $org(m) = org(j)$, and the remaining samples are put in this new subclass. As soon as we make a new subclass, the number of subclasses in the current hyperbox automatically increases to $ncss_h = ncss_h - 1$ and $ss(h, ncss_h) = m$. The values of l_{ij}, u_{ij}, l_{im} and u_{im} $(i = 1, 2 \ldots, N)$ are recalculated for future processing.

If $ncss_h > npr_I^D$, a new hyperbox is generated (i.e. $q = q + 1$), and in the new hyperbox, put those subclasses which belong to the set $ss(h)$ but not belong to the set $pr(D, I)$. Subsequently the remaining subclasses (i.e. belong to the set $pr(D, I)$) are only kept in the current hyperbox and make $ncss_h = npr_I^D$.

Step 4: (Determination of representative membership function)

If a new hyperbox is generated in the previous step and $ncss_h > 1$ then go to step 2.

Otherwise processing of the current hyperbox is considered to be completed. So the parameters of its membership functions $T_h^i(\alpha_h^i, \beta_h^i, \gamma_h^i)$ [equation (5)] $(i = 1, 2, \ldots, N)$ are now determined as follows:

$$L_i = \min_{j=ss(h,k),\, k=1,2,\ldots,ncss_h} \{l_{ij}\};$$

$$U_i = \max_{j=ss(h,k),\, k=1,2,\ldots,ncss_h} \{u_{ij}\};$$

$$\alpha_h^i = (U_i + L_i)/2; \beta_h^i = (U_i - L_i)/2$$

and

$$\gamma_h^i = \varepsilon_I.$$

Step 5: (Stopping condition)

Make $h = h + 1$.

If $h \leq q$, then go to step 2.

Otherwise stop.

The above algorithm automatically generates q hyperboxes based on the training samples of M classes.

REFERENCES

1. R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*. Wiley, New York (1973).
2. P. A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*. Prentice-Hall, London (1982).
3. K. S. Fu, *Syntactic Pattern Recognition and Applications*. Academic Press, London (1982).
4. M. Sugeno, An introductory survey of fuzzy control, *Inform. Sci.* **36**, 59–83 (1985).
5. C. C. Lee, Fuzzy logic in control systems; Fuzzy logic controller—Part I and Part II, *IEEE Trans. Systems Man Cybernet.* **SMC-20**, 404–435 (1990).
6. D. G. Burkhardt and P. P. Bonissone, Automated fuzzy knowledge base generation and tuning, in *Proc. FUZZ-IEEE*, San Diego, California, pp. 179–188 (1992).
7. T. Tagani and M. Sugeno, Fuzzy identification of systems and its application to modeling and control, *IEEE Trans. Systems Man Cybernet.* **SMC-15**, 116–132 (1985).
8. L. X. Wang and J. M. Mendel, Generating fuzzy rules from numerical data with applications, Report No. 169, University of Southern California (1991).
9. I. Hayashi, H. Nomura, H. Yamasaki and N. Wakami, Construction of fuzzy inference rules by NDF and NDFL, *Int. J. Appr. Reasoning* **6**, 241–266 (1992).
10. J. S. R. Jang, Fuzzy controller design without domain experts, in *Proc. FUZZ-IEEE'92*, San Diego, California, pp. 289–287 (1992).
11. H. Ishibuchi, N. Nozaki and H. Tanaka, Distributed representation of fuzzy rules and its application to pattern classification, *Fuzzy Sets and Systems* **52**, 21–32 (1992).
12. H. Ishibuchi, N. Nozaki and H. Tanaka, Efficient fuzzy partition of pattern space for classification problems, *Fuzzy Sets and Systems* **59**, 295–304 (1993).
13. L. A. Zadeh, Fuzzy logic and approximate reasoning, *Synthese* **30**, 407–428 (1977).
14. L. A. Zadeh, Fuzzy sets, *Inform. Control* **8**, 338–353 (1965).
15. J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithm*. Plenum Press, New York (1981).
16. S. K. Pal and D. D. Majumder, *Fuzzy Mathematical Approach to Pattern Recognition*. Wiley, Halsted, New York (1986).
17. W. Pedrycz, Fuzzy sets in pattern recognition: Methodology and methods, *Pattern Recognition* **23**, 121–146 (1990).
18. J. C. Bezdek and S. K. Pal, eds, *Fuzzy Models for Pattern Recognition: Method That Search for Structures in Data*. IEEE Press, New York (1992).
19. M. Grabisch and F. Dispot, A comparison of some methods of fuzzy classification on real data, in *Proc. IIZUKA'92*, Iizuka, Japan, pp. 659–662 (1992).
20. D. P. Mandal, A multivalued approach for uncertainty management in pattern recognition problems using fuzzy sets, Ph.D. Thesis, Indian Statistical Institute, Calcutta, India (1992).
21. D. P. Mandal, C. A. Murthy and S. K. Pal, Formulation of a multivalued recognition system, *IEEE Trans. Systems Man Cybernet.* **SMC-22**, 607–620 (1992).
22. D. P. Mandal, C. A. Murthy and S. K. Pal, Determining the shape of a pattern class from sampled points in $\mathbb{R}^2$, *Int. J. General Systems* **20**, 307–339 (1992).
23. D. P. Mandal, C. A. Murthy and S. K. Pal, Determining the shape of a pattern class: Extension to $\mathbb{R}^N$, *Int. J. General Systems*, to appear (1996).
24. S. K. Pal and D. P. Mandal, Linguistic recognition system based on approximate reasoning, *Inform. Sci.* **61**, 135–161 (1992).
25. S. Abe and M. Lan, A method for fuzzy rules extraction directly from numerical data and its application to pattern classification, *IEEE Trans. Fuzzy Systems* **3**, 18–28 (1995).
26. C. A. Murthy, On consistent estimation of classes in $\mathbb{R}^2$ in the context of cluster analysis, Ph.D. Thesis, Indian Statistical Institute, Calcutta, India (1988).
27. U. Grenander, *Abstract Inference*. Wiley, New York (1981).
28. L. A. Zadeh, Calculus of fuzzy restriction, in *Fuzzy Sets and Their Application to Cognitive and Decision Processes*, L. A. Zadeh, ed., pp. 1–39. Academic Press, New York (1975).
29. S. Ahmad and V. Tresp, Classification with missing and uncertain inputs, *Proc. IEEE-ICNN*, San Francisco, pp. 1949–1954 (1993).
30. H. Ishibuchi, A. Miyazaki, K. Kwon and H. Tanaka, Learning from incomplete training data with missing values and medical application, *Proc. IJCNN*, Nagoya, Japan, pp. 1871–1874 (1993).
31. L. A. Zadeh, The concept of linguistic variable and its application to approximate reasoning—II, *Inform. Sci.* **8**, 301–357 (1975).

32. D. G. Schwartz, The case for an interval based representation of linguistic truth, *Fuzzy Sets and Systems* **17**. 153–165 (1985).
33. E. Parzen, On the estimation of a probability density function and the mode, *Ann. Math. Statist.* **33**, 1065–1076 (1962).
34. E. Cacoullos, Estimation of a multivariate density, *Ann. Inst. Statist. Math.* **18**, 178–189 (1966).
35. D. D. Loftsgaarden and C. P. Quensenberry, A nonparametric estimate of a multivariate density function, *Ann. Math. Statist.* **36**, 1049–1051 (1965).
36. T. Cover and P. Hart, Nearest neighbor pattern classification, *IEEE Trans. Inform. Theory* **IT-13**, 21–27 (1967).
37. R. A. Fisher, The use of multiple measurements in taxonomic problems, *Ann. Eugenics* **7**, 179–188 (1936).
38. A. Marchand, F. Van Lente and R. Galen, The assessment of laboratory tests in the diagnosis of acute appendicitis, *Amer. J. Clinical Pathology* **80**, 369–374 (1983).
39. S. K. Pal and D. D. Majumder, Fuzzy sets and decision making approaches in vowel and speaker recognition. *IEEE Trans. Systems Man Cybernet.* **SMC-7**, 625–629 (1977).
40. H. Tanaka, H. Ishibuchi and N. Matsuda, Fuzzy expert system based on rough sets and its application to medical diagnosis, *Int. J. General Systems* **21**, 83–97 (1992).
41. H. Tanaka, H. Ishibuchi and T. Shigenaga, Fuzzy inference system based on rough sets and its application to medical diagnosis, in *Intelligent Decision Support*, R. Slowinski, ed., pp. 111–117. Kluwer Academic Publishers, London (1993).
42. M. Grabisch and M. Sugeno, Multi-attribute classification using fuzzy integral, in *Proc. IEEE Conf. on Fuzzy Systems*, San Diego, pp. 47–54 (1992).
43. J. M. Keller, M. R. Gray and J. A. Givens, Jr, A fuzzy *k*-nearest neighbor algorithm, *IEEE Trans. Systems Man Cybernet.* **SMC-15**, 580–585 (1985).
44. S. M. Weiss and I. Kapouleas, An empirical comparison of pattern recognition, neural nets and machine learning classification methods, in *Proc. Int. Jt. Conf. on Artificial Intelligence*, Detroit, pp. 781–787 (1989).
45. S. M. Weiss and C. A. Kulikowski, *Computer Systems That Learn*. Morgan Kaufmann Publishers, California (1991).
46. D. E. Rumelhart, J. L. McClelland and the PDP Research Group, *Parallel Distributed Processing*, 1. MIT Press, Cambridge, Massachusetts (1986).
47. A. Owens and D. Filkin, Efficient training of the back propagation network by solving a systems of stiff ordinary differential equations, in *Proc. Int. Conf. on Neural Networks*, Washington, DC, pp. 381–386 (1989).
48. J. Quinlan, Introduction of decision trees. *Mach. Learning* **1**, 81–106 (1986).
49. L. Breiman, J. Friedman, R. Olshen and C. Stone, *Classification and Regression Trees*. Warsworth, Monterey, California (1984).
50. S. Weiss, R. Galen and P. Tadepalli, Maximizing the predictive value of production rules, *Artif. Intell.* **45**, 47–71 (1990).
51. Z. Pawlak, Rough sets. *Int. J. Inform. Comput. Sci.* **11**. 341–356 (1984).

About the Author — DEBA PRASAD MANDAL born in 1963 in Bagula village in West Bengal, India. He obtained B.Sc. (honors) degree in Statistics from Kalyani University, West Bengal, India, in 1984. In 1988, he received Master of Computer Applications (MCA) degree from Jawaharlal Nehru University, New Delhi. He got Ph.D. degree (in Computer Sciences) from Indian Statistical Institute, Calcutta, in 1993. During February 1994 to March 1996, he visited the Department of Industrial Engineering, University of Osaka Prefecture, Japan, with a Japanese Government Postdoctorate Fellowship. At present, he is a lecturer in the Machine Intelligence Unit at the Indian Statistical Institute, Calcutta. His research interests mainly include Pattern Recognition, Image Processing, Fuzzy Sets and Systems, Remote Sensing, and Neural Networks. Dr Mandal received the Young Scientist Award in Computer Sciences from Indian Science Congress Association in 1992. He is listed in *World's Who's Who of Men and Women of Distinction* and *International Directory of Distinguished Leadership*. He is a life members of the Indian Science Congress Association (ISA) and Indian Statistical Institute (ISI), and members of Indian Society for Fuzzy Mathematics and Information Processing (ISFUMIP) and Indian Unit for Pattern Recognition and Artificial Intelligence (IUPRAI).