
ELSEVIER International Journal of Approximate Reasoning 30 (2002) 149–179

INTERNATIONAL JOURNAL OF APPROXIMATE REASONING

www.elsevier.com/locate/ijar

Neuro-fuzzy approach to processing inputs with missing values in pattern recognition problems

Bogdan Gabrys *

Applied Computational Intelligence Research Unit, Division of Computing and Information Systems, The University of Paisley, High Street, Paisley PA1 2BE Scotland, UK

Received 1 August 2001; accepted 1 March 2002

Abstract

An approach to dealing with missing data, both during the design and normal operation of a neuro-fuzzy classifier is presented in this paper. Missing values are processed within a general fuzzy min–max neural network architecture utilising hyperbox fuzzy sets as input data cluster prototypes. An emphasis is put on ways of quantifying the uncertainty which missing data might have caused. This takes a form of classification procedure whose primary objective is the reduction of a number of viable alternatives rather than attempting to produce one winning class without supporting evidence. If required, the ways of selecting the most probable class among the viable alternatives found during the primary classification step, which are based on utilising the data frequency information, are also proposed. The reliability of the classification and the completeness of information is communicated by producing upper and lower classification membership values similar in essence to plausibility and belief measures to be found in the theory of evidence or possibility and necessity values to be found in the fuzzy sets theory. Similarities and differences between the proposed method and various fuzzy, neuro-fuzzy and probabilistic algorithms are also discussed. A number of simulation results for well-known data sets are provided in order to illustrate the properties and performance of the proposed approach.

Keywords: Missing data; Neuro-fuzzy classifier; Pattern recognition; Fuzzy systems

* Tel.: +44-141-848-3752; fax: +44-141-848-3542.
E-mail address: gabr-ci0@wpmail.paisley.ac.uk (B. Gabrys).

0888-613X/02/$ - see front matter © 2002 Elsevier Science Inc. All rights reserved.
PII: S0888-613X(02)00070-1

1. Introduction

There are many real-world pattern recognition problems where input feature vectors are incomplete [1–3,9,13,14,16–18,21,23,27,28]. The reasons for missing data can be multifold ranging from sensor failures in engineering applications to deliberate withholding of some information in medical questioners. And though such problems are very interesting from the practical and theoretical point of view, there are very few pattern recognition techniques which can deal with missing values in a straightforward and efficient manner. It is in a sharp contrast to the very efficient way in which humans deal with unknown data and are able to perform various pattern recognition tasks given only a subset of input features.

There are two distinctive problems when dealing with incomplete data which can be stated in the following way:

(a) How to use such data in the design stage of a pattern classifier, especially when one cannot afford to discard the inputs with missing values? and
(b) How to perform the classification of an incomplete input vector during the normal operation of the classifier?

Both learning on the basis of incomplete data and pattern classification using input vectors with missing features will be discussed in this paper. The proposed approach to handling incomplete data will be based on utilising hyperbox fuzzy sets within a general fuzzy min–max (GFMM) neural network architecture [5–12]. The facts that inputs to the GFMM can be given in a form of upper and lower limits and that the hyperboxes are also represented by a pair of minimum and maximum coordinates for each dimension of the input space will be heavily used in the way the missing values are processed during the learning and recall stages. This specific representation of inputs and cluster prototypes allows for the missing values to be handled very efficiently. It is achieved by representing the missing features as real valued intervals spanning the whole range of possible values. It facilitates the classification of incomplete input patterns without any need for modifications of the neural network structure or substituting of missing features with estimated values. It is argued that since there is a certain level of uncertainty associated with unobserved features, this uncertainty has to be quantified in some way during the classification process. What distinguishes the proposed method from a vast majority of the algorithms for dealing with missing data is that the primary goal of classification is specified as the reduction of a number of viable alternatives on the basis of observed features. Additionally the upper and lower degrees of class membership are calculated in order to quantify the level of uncertainty associated both with the classifier model built on the basis of incomplete training data and incomplete input patterns to be classified. The use of upper and lower degree of class membership is very similar to the plausibility and belief measures used in the theory of evidence [22] or possibility and necessity

values used in the fuzzy sets theory [19]. Despite using the upper and lower classification limits, it has to be stressed that if the observed features are discriminative enough a single class can be produced as a result of the classification. However, when a number of alternatives is found and one winning class must be specified, data frequency or hyperbox cardinality information can be used to select the most probable class. In this way the proposed method combines some ideas from the probabilistic methods and fuzzy sets in order to produce more informative classification results. Before concentrating on the description of the algorithms let us first review the existing approaches to handling missing data and highlight some similarities between them and our approach.

One of the most common ways of dealing with missing values is to substitute the missing features with their estimates [3]. These could include the mean value calculated over all examples (or k-nearest neighbours) for a particular feature. However, as it has been pointed out in many papers [1,13,17,18,27] such a "repaired" data set may no longer be a good representation of the problem at hand (as illustrated in Fig. 9) and quite often leads to the solutions that are far from optimal.

Another approach presented in [23] advocates the creation of a set of classifiers that would work on different subsets of input features. Unfortunately, this method of training classifiers for all possible combinations of input features very quickly explodes in complexity with an increasing number of features.

Within a probabilistic setting much better results have been obtained by estimating conditional probability distribution over all unknown features given the known features. The emphasis here has been put on the input data density estimation. Ghahramani and Jordan [13] used a mixture of gaussians as a basis for the density estimation and the Expectation–Maximisation algorithm for learning the parameters of this mixture model. A very similar approach has been presented in [27] where the input data density estimation has been carried out by employing Parzen windows and gaussians centered at training data points, though the training algorithm was based on the modified backpropagation method. Tresp et al. [27] also point out that their proposed normalised gaussian formula (very similar to the formula (18)) could be used with various approximations where the gaussians do not necessarily have to be centred at the training data points. It is especially important when one has a large number of training data points. In terms of data density estimation and using clustering methods as one of the alternatives for the above-mentioned approximation, the method proposed in this paper can be regarded as one such approximation with hyperbox fuzzy sets used instead of multidimensional gaussians. There are however some crucial differences which will be pointed out in the later sections. Yet another example of neural implementation of the approach based on building the input data density model can be found in [18] where a radial basis

Bolzmann machine has been used for conditional probability distribution estimation and modified learning rules have been proposed to deal with missing values during the training. And although all the methods discussed above are another form of estimating the missing values they proved to be more accurate than the heuristic approaches used before.

Apart from the above-mentioned probabilistic approaches there have been a number of fuzzy and neuro-fuzzy methods for dealing with missing values [1,17]. Fuzzy sets/fuzzy rule based classification systems naturally lend themselves to efficient processing of missing features. Both in [1] and [17], which discuss the processing of missing data within fuzzy rule based classifiers, the classification is only based on the known features without making any assumptions about the values of missing features. In other words, the assumption that any value for missing features is possible means that such a feature should not influence the computation of a degree of fulfilment of any of the classification rules. In [1,17] this was achieved by assigning 1.0 as a degree of membership to the missing features. The general idea of using only the known features for classification and ensuring that the missing features do not affect the generation of classification membership values is also used in this paper. The way of achieving this is however completely different though the final result is very similar. As mentioned earlier, the main goal of the classification is defined as a reduction of viable alternatives rather than producing one winning class without supporting evidence. It is argued here that one of the potential drawbacks of the methods based on the estimation of missing features is the fact that once the estimated value has been used (in implicit or explicit way) the output of the classifier does not differ in any way from the output generated for examples with all features present. An extreme example would be an attempt to classify an input pattern with all features missing which in case of probabilistic models discussed above would result in a guess based on the estimated data density model while for the fuzzy methods all classification rules would be fulfilled with a degree of 1.0 making all classes equally viable. One difference of the proposed method from [1,17] is that in the case of classification producing a number of viable alternatives the hyperbox cardinality information can be used to find the most likely class. The resulting formulas are very similar to the normalised gaussian formula presented in [27]. Additionally, by taking advantage of the interval representation of the inputs, a way of using some limits (if known) for missing features, which could help in the class discrimination, is proposed.

While classification on the basis of patterns with missing features is the same, the learning of fuzzy rules in [1] and [17] is performed differently. In [1] the rule model that evolves during the training to predict the most possible value for each of the missing features is used. These predictions are then subsequently used to complete the input pattern. In [17] when a missing feature is present all combinations of fuzzy sets, which are possible for such pattern,

are created. Depending on the granularity (the number of fuzzy sets) used to partition each variable this approach can initially lead to generation of a large number of rules, especially for high dimensional problems with relatively large proportion of missing values. However, this general approach of assuming that a missing feature can take any value will be utilised in the training stage of our method. But again the implementation of this idea will be based on a suitable representation of the missing feature as a real valued interval spanning the whole range of possible values.

Some other approaches to dealing with missing values within a context of classification decision trees and hyperbox based neural networks can be found in [20,28].

The remaining of this paper is organised as follows. Section 2 presents a summary of GFMM neural network with definitions of hyperbox fuzzy sets and associated fuzzy membership function. It also provides a short description of one of the learning algorithms which can be used to place and adjust hyperboxes in the input space. A way of dealing with missing features within the GFMM NN structure is described in Section 3. Both learning on the basis of incomplete data and GFMM operation when given an incomplete input vector are discussed. The simulation results for a number of well known data sets follow in Section 4. And finally the discussion and conclusions are presented in the last section.

2. An overview of GFMM neural network

GFMM neural network for clustering and classification [8,12] is a generalisation of and extension to the fuzzy min–max neural networks introduced in [24,25]. The main changes in GFMM constitute the combination of unsupervised and supervised learning, associated with problems of data clustering and classification respectively, within a single learning algorithm and extension of the input data from a single point in n-dimensional pattern space to input patterns given as lower and upper limits for each dimension, i.e. a hyperbox in n-dimensional pattern space.

The GFMM NN can be realised as a three layer feedforward neural network shown in Fig. 1. It consists of $2 * n$ input layer nodes, m second layer nodes representing hyperbox fuzzy sets and $p + 1$ output layer nodes representing classes.

The basic idea of fuzzy min–max neural networks is to represent groups of input patterns using hyperbox fuzzy sets. A hyperbox fuzzy set is a combination of a hyperbox covering a part of n-dimensional pattern space and associated with it membership function. A hyperbox is completely defined by its min point and its max point. A membership function acts as a distance measure with input patterns having a full membership if they are fully contained within

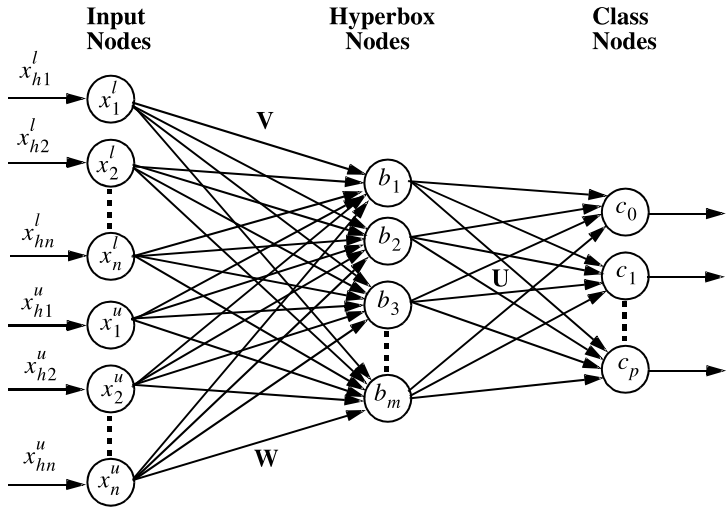

Fig. 1. GFMM neural network for clustering and classification.

a hyperbox and the degree of membership decreasing with the increase of distance from the hyperbox. Learning in the GFMM neural network for clustering and classification consists of creating and adjusting hyperboxes in the pattern space. In previous publications an on-line [12] and a batch (agglomerative) [8] versions of the training algorithm had been proposed. One of the two agglomerative learning schemes described in detail in [8] will be used in this paper and is summarised later in this section.

Once the network is trained the input space is covered with hyperbox fuzzy sets. Individual hyperboxes representing the same class are aggregated to form a single fuzzy set class. The cores of hyperboxes belonging to the same class are allowed to overlap while the cores of hyperboxes belonging to different classes are not allowed to overlap therefore avoiding the ambiguity of an input having full membership in more than one class.

The following are the definitions of input data format, hyperbox fuzzy sets, hyperbox membership function and hyperbox aggregation formula that are used within GFMM.

The input data used during the training stage of GFMM neural network is specified as a set of N ordered pairs

$$\{\boldsymbol{X}_h, d_h\}, \tag{1}$$

where $\boldsymbol{X}_h = [\boldsymbol{X}_h^{\mathrm{l}}\ \boldsymbol{X}_h^{\mathrm{u}}]$ is the hth input pattern in a form of lower, $\boldsymbol{X}_h^{\mathrm{l}}$, and upper, $\boldsymbol{X}_h^{\mathrm{u}}$, limit vectors contained within the n-dimensional unit cube I^n; and $d_h \in \{0, 1, 2, \ldots, p\}$ is the index of one of the $p+1$ classes, where $d_h = 0$ means that the input vector is unlabelled.

The jth hyperbox fuzzy set, B_j is defined as follows:

$$B_j = \{\boldsymbol{V}_j, \boldsymbol{W}_j, b_j(\boldsymbol{X}_h, \boldsymbol{V}_j, \boldsymbol{W}_j)\} \tag{2}$$

for all $j = 1, 2, \ldots, m$, where $\boldsymbol{V}_j = (v_{j1}, v_{j2}, \ldots, v_{jn})$ is the min point for the jth hyperbox, $\boldsymbol{W}_j = (w_{j1}, w_{j2}, \ldots, w_{jn})$ is the max point for the jth hyperbox, and the membership function for the jth hyperbox is

$$b_j(\boldsymbol{X}_h, \boldsymbol{V}_j, \boldsymbol{W}_j) = \min_{i=1..n}(\min([1 - f(x^{\mathrm{u}}_{hi} - w_{ji}, \gamma_i)], [1 - f(v_{ji} - x^{\mathrm{l}}_{hi}, \gamma_i)])), \tag{3}$$

where

$$f(x, \gamma) = \begin{cases} 1 & \text{if } x\gamma > 1, \\ x\gamma & \text{if } 0 \leqslant x\gamma \leqslant 1, \\ 0 & \text{if } x\gamma < 0, \end{cases} \quad \text{– two parameter ramp threshold function;}$$

$\gamma = [\gamma_1, \gamma_2, \ldots, \gamma_n]$ – sensitivity parameters governing how fast the membership values decrease; and $0 \leqslant b_j(\boldsymbol{X}_h, \boldsymbol{V}_j, \boldsymbol{W}_j) \leqslant 1$. A graphical example of the membership function is shown in Fig. 2.

Hyperbox fuzzy sets from the second layer are aggregated using the aggregation formula (4) in order to generate an output c_k which represents the degree to which the input pattern $\boldsymbol{X}_h$ fits within the class k. The transfer function for each of the third layer nodes is defined as

$$c_k = \max_{j=1}^{m} b_j u_{jk}, \tag{4}$$

where $\mathbf{U}$ is the binary matrix with values u_{jk} equal to 1 if the jth hyperbox fuzzy set is a part of the kth class and 0 otherwise; and $c_k \in [0, 1]$, $k = 0..p$, represent

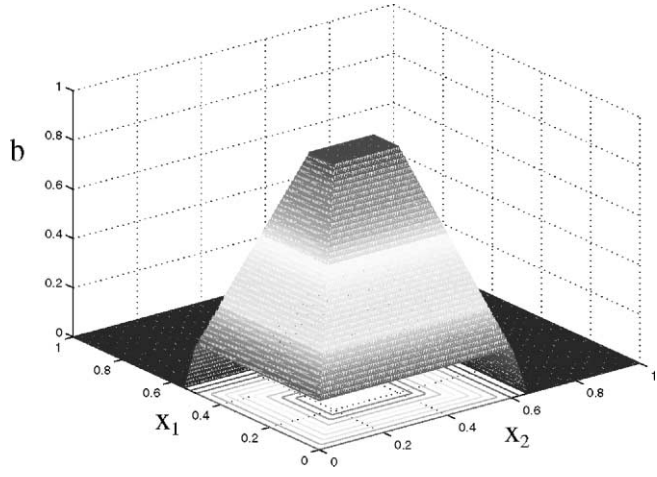

Fig. 2. A graphical illustration of the hyperbox membership function (3) for the values of $\boldsymbol{V} = [0.2\ 0.2]$, $\boldsymbol{W} = [0.3\ 0.4]$ and $\gamma = 4$.

the degrees of membership of the input pattern in the kth class with c_0 representing all unlabelled hyperboxes.

2.1. GFMM learning algorithm

The agglomerative learning for the GFMM neural network [8] initialises the min points matrix $\boldsymbol{V}$ and the max points matrix $\boldsymbol{W}$ to the values of the training set patterns lower $\boldsymbol{X}^{\mathrm{l}}$ and upper $\boldsymbol{X}^{\mathrm{u}}$ limits, respectively.

The hyperboxes are then agglomerated sequentially (one pair at a time) on the basis of the maximum similarity value calculated using the following similarity measure $s_{jh} = s(\boldsymbol{B}_j, \boldsymbol{B}_h) = s_j(\boldsymbol{B}_h, \boldsymbol{V}_j, \boldsymbol{W}_j)$ between two hyperbox fuzzy sets B_h and B_j:

$$s_{jh} = \min_{i=1..n}(\min([1 - f(w_{hi} - w_{ji}, \gamma_i)], [1 - f(v_{ji} - v_{hi}, \gamma_i)])), \tag{5}$$

which has been adopted directly from (3) and takes into account not only the proximity of two hyperbox fuzzy sets but also their sizes. Two other similarity measures for hyperbox fuzzy sets are defined in [8] and although any of the similarity measures could be used, for the clarity of further derivations, the results presented in this paper have been obtained using (5). Since in general when using (5) $s_{jh} \neq s_{hj}$, i.e. a degree of similarity of B_h to B_j is not equal to a degree of similarity of B_j to B_h (with exception when B_h and B_j are points and some other special cases), the selection of a hyperbox B_l, to be aggregated with B_j is made by finding the maximum value from either: (a) the minimum similarity values $\min(s_{jl}, s_{lj})$; or (b) the maximum similarity values $\max(s_{jl}, s_{lj})$ among all possible pairs of hyperboxes (B_j, B_l). In this paper we have used option (b) which leads to an agglomerative procedure somewhat resembling a single link agglomerative algorithm [26]. The process of finding B_l can be summarised as follows:

$$\underset{\substack{l=1\ldots m \\ l\neq j}}{\forall} s_{jh} = \max\left(\max(s_{jl}, s_{lj})\right). \tag{6}$$

The hyperboxes with the highest similarity value are only agglomerated if:

(a) newly formed hyperbox does not exceed the maximum allowable hyperbox size $0 \leqslant \Theta \leqslant 1$

$$\underset{i=l\ldots n}{\forall} (\max(w_{ji}, w_{hi}) - \min(v_{ji}, v_{hi})) \leqslant \Theta \tag{7}$$

and/or the similarity between hyperboxes B_h and B_j is greater than a certain user defined similarity threshold value $0 \leqslant s_{\min} \leqslant 1$

$$s_{jh} \geqslant s_{\min}, \tag{8}$$

(b) the agglomeration does not result in an overlap with any of the hyperboxes representing other classes (please see [12] for full details of the overlap test); and

(c) the hyperboxes B_h and B_j form a part of the same class or one or both are unlabelled

$$\begin{aligned} &\textit{if } class(B_h) = 0 \textit{ then aggregate } B_h \textit{ and } B_j \\ &\textit{else} \\ &\textit{if } class\ (B_j) = \begin{cases} 0 \Rightarrow \begin{array}{l} \textit{aggregate } B_h \textit{ and } B_j \\ class\ (B_j) = class\ (B_h) \end{array} \\ class\ (B_h) \Rightarrow \textit{ aggregate } B_h \textit{ and } B_j \\ \textit{else } \Rightarrow \begin{array}{c} \textit{take} \\ \textit{another pair of hyperboxes} \end{array} \end{cases} \end{aligned} \tag{9}$$

If the above conditions are met the aggregation is carried out in the following way:

(a) update B_j so that a new B_j will represent aggregated hyperboxes B_h and B_j

$$v_{ji}^{\text{new}} = \min(v_{ji}^{\text{old}}, v_{hi}^{\text{old}}) \quad \text{for each } i = 1, \ldots, n, \tag{10}$$

$$w_{ji}^{\text{new}} = \max(w_{ji}^{\text{old}}, w_{hi}^{\text{old}}) \quad \text{for each } i = 1, \ldots, n, \tag{11}$$

(b) remove B_h from a current set of hyperbox fuzzy sets (in terms of the neural network shown in Fig. 1 it would mean a removal of the hth second layer node).

The process of agglomeration is repeated until there are no hyperboxes which can be aggregated. The agglomeration of hyperboxes can be controlled by specifying different values for the maximum hyperbox size Θ and hyperbox similarity threshold $s_{\min}$ during the training process. For instance, in order to encourage creation of clusters in the densest areas first (i.e. aggregation of the most similar hyperboxes) $s_{\min}$ can be initially set to a relatively high value and reduced in steps after all possible hyperboxes for a higher level of $s_{\min}$ have been aggregated. In this way we are able to produce (simulate) a hierarchy of nested clusterings. The optimal value of $s_{\min}$ when designing a GFMM classifier is selected during a cross-validation procedure where a system with the smallest error for the validation set is sought.

A similar effect can be obtained when using the maximum hyperbox size parameter Θ and starting with relatively small values of Θ initially allowing to create small hyperbox fuzzy sets, and increasing the value of Θ in subsequent steps with inputs to the next level consisting of the hyperbox fuzzy sets from the previous level.

Similarly to some other powerful classification methods [4] like unpruned decision trees, artificial neural networks, nearest neighbour classifiers or some *if ...then* fuzzy rule based classifiers the training set can be learned perfectly. This, however, usually leads to the training data overfitting and some suitable complexity control mechanism needs to be employed. In order to ensure good generalisation performance various approaches utilising statistical resampling

techniques [6] have been studied and used together with: pruning procedures [8], data editing techniques [7], and combinations of multiple copies of the GFMM classifier at the decision and model levels [5].

3. Incomplete data processing

The training procedure and operation of the GFMM classifier described in the previous sections assumed that the training data are complete, although a case of missing labels has been covered and is reflected in the training algorithm through suitably defined overlap testing procedure and the test for class compatibility expressed by (9). In the following sections the case of missing data is extended to a possibility of missing features. There are two major problems associated with the missing features which can be stated in the following form:

(1) How the classification of an incomplete input pattern should be performed given a trained GFMM classifier? and
(2) How the process of creating and adjusting hyperboxes should be modified if an incomplete input pattern is encountered?

The proposed solutions to the above questions will now follow. In both the classification and training based on incomplete data it is assumed that all values from the whole range for a missing feature are possible. In the proposed method the missing features are represented by a real valued intervals which can be naturally processed within the GFMM neural network. Only the observed values are used either for adjusting the hyperbox fuzzy sets during the training or to produce the degrees of class membership during the classification process. The uncertainty of the classification, resulting from the uncertainty present in the classifier model or uncertainty associated with the missing features of the input pattern to be classified, are reflected in the classifier output.

3.1. Classification of incomplete data patterns

It is assumed at this stage that the GFMM neural network has been trained to classify n-dimensional input patterns to one of the p classes. The classification is based on a similarity measure given by the hyperbox membership function (3) and the hyperbox aggregation formula (4).

The discriminative character of the hyperbox membership function is based on penalising the violations of hyperbox min and max values for each input dimension. The smaller x_{hi}^{l} than v_{ji} or the larger x_{hi}^{u} than w_{ji}, the smaller the membership value, $b_j(\boldsymbol{X}_h)$ for the jth hyperbox fuzzy set B_j. If the ith feature (dimension) of the input pattern is missing and assuming that all values for this

feature are possible, the hyperbox membership value associated with the missing feature should be 1.0 and therefore should not cause a decrease of the overall degree of membership. It can be assured by the following assignments for the missing ith feature:

$$x_{hi}^{l} = 1 \quad \text{and} \quad x_{hi}^{u} = 0. \tag{12}$$

In this way the lower limit of the missing feature x_{hi}^{l} will never be smaller than v_{ji} and the upper limit of the missing feature x_{hi}^{u} will never be greater than w_{ji}. The assignment (12) can be thought of as modelling the missing feature by a real valued interval spanning the whole range of values. By this substitution it is also ensured that the structure of the neural network will not have to be changed when processing inputs with missing dimensions and values in (12) can be considered as the default values for the GFMM neural network inputs.

The interval representation of the GFMM inputs has an additional advantage because when some limits for a missing feature are known they can be used in a straightforward way and contribute to the discriminative character of the membership function (3).

The following are the possible cases:

(a) if only the upper limit, $upper_i$, for the ith feature is known then assign:

$$x_{hi}^{l} = upper_i \quad \text{and} \quad x_{hi}^{u} = 0; \tag{13}$$

(b) if only the lower limit, $lower_i$, for the ith feature is known then assign:

$$x_{hi}^{l} = 1 \quad \text{and} \quad x_{hi}^{u} = lower_i; \tag{14}$$

(c) if both the upper, $upper_i$, and the lower, $lower_i$, limits for the ith feature are known then assign:

$$x_{hi}^{l} = upper_i \quad \text{and} \quad x_{hi}^{u} = lower_i. \tag{15}$$

The above considerations can be illustrated on a two dimensional example shown in Fig. 3.

The two depicted hyperboxes represent two classes. The input to be classified

$$\boldsymbol{X} = \begin{bmatrix} x_1^{l} & x_1^{u} \\ x_2^{l} & x_2^{u} \end{bmatrix} = \begin{bmatrix} 0.45 & 0.55 \\ ? & ? \end{bmatrix}$$

has the second feature missing.

The four considered cases of membership values calculation for the second feature include the examples where: (a) no limits are known – (12), (b) only an upper limit equal to 0.4 is known – (13), (c) only a lower limit equal to 0.6 is known – (14), and (d) both an upper and lower limits equal to 0.6 and 0.4, respectively are known – (15).

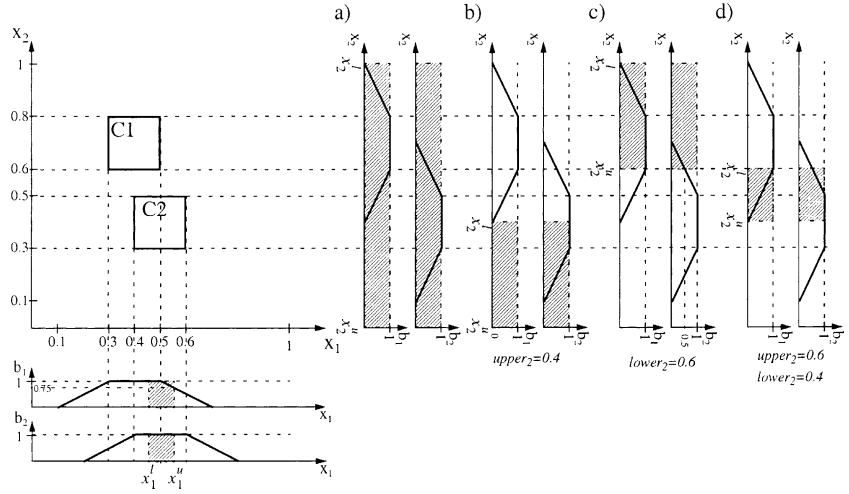

Fig. 3. A two-dimensional example of processing missing values and potential use of upper and lower limits for missing features.

The substitution values and the classification results are as follows:

(a) input pattern:

$$\boldsymbol{X} = \begin{bmatrix} 0.45 & 0.55 \\ 1 & 0 \end{bmatrix},$$

classification:

$$\boldsymbol{C} = \begin{bmatrix} 0.75 \\ 1 \end{bmatrix} - \text{class 2 selected;}$$

(b) input pattern:

$$\boldsymbol{X} = \begin{bmatrix} 0.45 & 0.55 \\ 0.4 & 0 \end{bmatrix},$$

classification:

$$\boldsymbol{C} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} - \text{class 2 selected;}$$

(c) input pattern:

$$\boldsymbol{X} = \begin{bmatrix} 0.45 & 0.55 \\ 1 & 0.6 \end{bmatrix},$$

classification:

$$\boldsymbol{C} = \begin{bmatrix} 0.75 \\ 0.5 \end{bmatrix} - \text{class 1 selected;}$$

(d) input pattern:

$$\boldsymbol{X} = \begin{bmatrix} 0.45 & 0.55 \\ 0.6 & 0.4 \end{bmatrix},$$

classification:

$$\boldsymbol{C} = \begin{bmatrix} 0.75 \\ 1 \end{bmatrix} - \text{class 2 selected.}$$

As we can see from Fig. 3 the second missing feature of the input pattern is processed differently from the first feature. Essentially when a feature is missing we adopt an optimistic approach and consider which values are possible. In the case of the considered example it translates into finding the maximum membership value within the shaded areas for the second feature illustrated as cases a, b, c and d in Fig. 3. It is in contrast to the first feature where we are looking for the minimum values within the shaded areas. Of course all these minimum and maximum values are found automatically just by setting the appropriate values for missing data as given by (12)–(14) or (15). Otherwise the processing is carried out as for the cases without any data missing as described in Section 2.

Another consequence of (12) and the other three substitution formulas is a possibility of an input with missing features having a full membership in more than one class. This, however, is something rather desirable since it aids in distinguishing from cases with all features present and reflects uncertainty associated with missing data. Generally, the more features missing the closer the membership value to 1 for more classes. The first objective of the classification here is to reduce a number of viable alternatives, however when a selection of the winning class is required additional information can be taken into account. During the agglomerative training process apart from the jth hyperbox parameters (i.e. min–max points coordinates and the hyperbox class label), the hyperbox cardinality information n_j is stored. By the hyperbox cardinality we mean a number of training data patterns fully contained within the jth hyperbox.

This cardinality information can be additionally used for finding the most probable class using the following formula:

$$\hat{P}(cl_k|\boldsymbol{X}_h) = \frac{\sum_{j \in j^k_{mx}} n_j b_j(\boldsymbol{X}_h, \boldsymbol{V}_j, \boldsymbol{W}_j)}{\sum_{i \in m_{mx}} n_i b_i(\boldsymbol{X}_h, \boldsymbol{V}_i, \boldsymbol{W}_i)}, \tag{16}$$

where $m_{mx} = \{i : \text{if } b_i(\boldsymbol{X}_h, \boldsymbol{V}_i, \boldsymbol{W}_i) = c_{k_{\text{win}}}\}$ is a set of indexes of all hyperbox fuzzy sets for which the degree of membership is equal to the degree of membership of the winning class found using the following formula:

$$c_{k_{\text{win}}} = \max_{j=1}^{p} c_j \tag{17}$$

and $j^k_{mx} = \{j : \text{if } class(B_j) = cl_k \text{ and } b_j(\boldsymbol{X}_h, \boldsymbol{V}_j, \boldsymbol{W}_j) = c_{k_{\text{win}}}\}$ is a subset of m_{mx} with indexes for the kth class and $\sum_{k=1}^{p} \hat{P}(cl_k|\boldsymbol{X}_h) = 1$.

If there is only one class with the degree of membership equal to $c_{k_{\text{win}}}$ it means that only one alternative has been found and $\hat{P}(cl_k|\boldsymbol{X}_h)$ for such class will be equal to 1.0 while for the other classes it will be equal to 0.0. This case is illustrated in the example of classification of the input patterns 1, 2, 3, 5 and 7 in Fig. 4. On the other hand when more than one alternative class is found using (17) the most probable class from the selected alternatives is found by utilising the hyperbox fuzzy sets cardinality values as given by (16). Please see the example of classification of the input pattern 4 in Fig. 4 below where due the second feature missing the hyperboxes 1 and 3 representing two different classes are regarded as equally viable alternatives according to (17) but class 2 is selected as more probable when using (16).

As we can see from the above description only a subset of the hyperbox fuzzy sets with the highest degrees of membership is used when deciding which of the alternative classes found using (17) should be selected as the winner. A generalisation of this definition of alternative classes will be given in the next section when we introduce the upper and lower degrees of class membership.

Alternatively all hyperbox fuzzy sets, which here can be regarded as an alternative to Parzen windows using gaussian window functions [27] or gaussian mixtures [13] as an input data distribution estimators, can be used to carry out the classification using the following formula:

$$P(cl_k|\boldsymbol{X}_h) = \frac{\sum_{j \in j^k} n_j b_j(\boldsymbol{X}_h, \boldsymbol{V}_j, \boldsymbol{W}_j)}{\sum_{i=1} n_i b_i(\boldsymbol{X}_h, \boldsymbol{V}_i, \boldsymbol{W}_i)}, \tag{18}$$

where $j^k = \{j : \text{if } class(B_j) = cl_k\}$ is a set of indexes of all the hyperboxes from the kth class and $\sum_{k=1}^{p} P(cl_k|\boldsymbol{X}_h) = 1$.

If (18) was to be implemented as a neural network shown in Fig. 1 the connections between the second and third layer nodes would not be binary values but equal to $u_{jk} = n_j / \sum_{i=1}^{m} n_i b_i(\boldsymbol{X}_h, \boldsymbol{V}_i, \boldsymbol{W}_i)$ if the jth hyperbox fuzzy set was part of the kth class and zero otherwise. The transfer function for the third layer nodes would be a sum of weighted membership values rather than a maximum as given in (4). In this case the GFMM would closely resemble the radial basis function (RBF) neural networks but with a crucial distinction that the inputs to the whole network can be given as the real number intervals.

Similarly to the width of the window in the standard Parzen windows density estimation [4], the parameter γ, governing the speed of decreasing of the membership function with the increasing distance from a hyperbox, has a crucial effect on how the classifier performs. It is especially important when using a weighted sum of *all* hyperbox fuzzy sets as in (18). However, as will be illustrated in the results section, the value of γ is not so crucial when using (16) because only a subset of hyperbox fuzzy sets with maximum membership values is selected.

When γ is approaching zero all b_j will have values approaching 1 and then P approaches the unconditional class probabilities

$$P(cl_k) = P(cl_k|\boldsymbol{X}_h)\frac{\sum_{j\in j^k} n_i}{\sum_{i=1}^{m} n_i}$$

estimated from the training data. This is also the case, both for (16) and (18), when one attempts to classify an input pattern with all features missing irrespective of the value of γ. On the other hand if γ is large the membership function decreases very quickly and the classification can be carried out only if the input pattern fall into one of the cores of hyperboxes or in a case of an input with missing features, all the observed feature values must fall between the min and max values of at least one hyperbox fuzzy set.

3.2. Training with incomplete data

The GFMM training algorithm in presence of incomplete data is virtually the same as for the training data with all features present. Once the substitution (12) (or (13)–(15)) has been made the remaining steps of the training algorithm ensure the use of observed features in the process of updating the hyperbox min and max points while the missing features are automatically ignored as a result of the hyperbox update rules (10) and (11).

The only modifications of the basic training algorithm concern the definition of the overlap test and the use of the assumption that all the values for the missing features are possible.

With respect to the overlap test only the hyperboxes for which $w_{ji} \geqslant v_{ji}$ for all $i = 1..n$ are tested for the undesired overlap after each agglomeration (update) of hyperbox fuzzy sets during the training process.

The second modification concerns the way how the missing dimensions in hyperboxes can affect the calculation of the hyperbox similarity values (5) (or hyperbox membership values (3)) used during the training process for selecting the most similar hyperboxes for agglomeration. Similarly to the classification of an input pattern with missing features, hyperbox similarity values are determined on the basis of observed features with missing dimensions not affecting (decreasing) the similarity value. This can be achieved by changing the hyperbox similarity measure (5) in the following way:

$$s_j(\boldsymbol{B}_h, \min(\boldsymbol{V}_j\boldsymbol{W}_j), \max(\boldsymbol{V}_j, \boldsymbol{W}_j)) = \min_{i=1\ldots n}(\min([1 - f(w_{hi} - \max(v_{ji}, w_{ji}), \gamma_i)], [1 - f(\min(v_{ji}, w_{ji}) - v_{hi}, \gamma_i)])). \tag{19}$$

If there are no missing data the above definition of the hyperbox similarity measure is exactly the same as (5) since all $w_{ji} \geqslant v_{ji}$.

The final classifier trained using incomplete data can be composed of a set of hyperbox fuzzy sets which can have some of the dimensions not set up during the training (please see the hyperbox 4 in the example in Fig. 4). Whether there will be some hyperboxes with "missing" dimensions or not depends on the composition of the training set. It is our way of representing the training data uncertainty in the classifier model.

Once the classifier is trained the potential uncertainty associated with the fact that the limits (i.e. min and max points) for some hyperboxes' dimensions have not been set during the training, can be quantified by producing the upper and lower degrees of class membership. This can be achieved by using (4) in the following way:

$$c_k^{\mathrm{u}} = \max_{j=1}^{m} b_j(\boldsymbol{X}_h, \min(\boldsymbol{V}_j, \boldsymbol{W}_j), \max(\boldsymbol{W}_j, \boldsymbol{V}_j))u_{jk}, \tag{20}$$

$$c_k^{\mathrm{l}} = \max_{j=1}^{m} b_j(\boldsymbol{X}_h, \boldsymbol{V}_j, \boldsymbol{W}_j)u_{jk} \tag{21}$$

for finding the upper c_u^k and lower c_k^{l} class membership values respectively.

The difference between (20) and (21) may only occur when some of $v_{ji} > w_{ji}$ signifying that some of the hyperbox min–max points have not been set. By taking $\min(\boldsymbol{V}_j, \boldsymbol{W}_j)$ and $\max(\boldsymbol{W}_j, \boldsymbol{V}_j)$ in (20) we effectively find the most optimistic class membership values. On the other hand, if we use (21) the most pessimistic class membership values are found. In a sense we can think of the values of c_k^{u} and c_k^{l} as similar to the possibility and necessity values [19] or the plausibility and belief values [22] to be found in the fuzzy set theory and the evidence theory respectively.

The fact that the classification result can now be given in a form of upper and lower class membership values requires a definition of what one would consider as a viable class alternative. In the previous section two or more classes were considered as viable alternatives if the class membership values for such classes were equal to the membership value of the winning class. Here, after finding c_k^{u} and c_k^{l} for all classes first the class is found with the maximum upper membership value

$$c_{k_{\mathrm{win}}}^{\mathrm{u}} = \max_{j=1}^{p} c_j^{\mathrm{u}}. \tag{22}$$

If there are more than one class with equal highest membership value the class with the minimum difference between upper and lower membership values

$$\min_{\substack{j=1..p \\ c_j^{\mathrm{u}}=c_{k_{\mathrm{win}}}^{\mathrm{u}}}} c_j^{\mathrm{u}} - c_j^{\mathrm{l}} \tag{23}$$

is selected.

Once such class has been found all classes for which

$$c_j^{\mathrm{u}} \geqslant c_{k_{\mathrm{win}}}^{\mathrm{l}} \quad \begin{matrix} j = 1 \ldots p \\ j \neq k_{\mathrm{win}} \end{matrix} \tag{24}$$

are regarded as viable alternatives.

In a case where there are more than one viable alternative and if only one winning class needs to be selected the hyperbox cardinality information is used by utilising (16) with b_j calculated as $b_j(\boldsymbol{X}_h, \min(\boldsymbol{V}_j, \boldsymbol{W}_j), \max(\boldsymbol{W}_j, \boldsymbol{V}_j))$ and a set of hyperbox fuzzy set indexes to be used in (16) is found as follows:

$$m_{mx} = \left\{ j : \text{if } b_j(\boldsymbol{X}_h, \min(\boldsymbol{V}_j, \boldsymbol{W}_j), \max(\boldsymbol{W}_j, \boldsymbol{V}_j)) \geqslant c_{k_{\mathrm{win}}}^{\mathrm{l}} \right\},$$

$$j_{mx}^{k} = \left\{ j : \text{if } class(B_j) = cl_k \right.$$
$$\left. \text{and } b_j(\boldsymbol{X}_h, \min(\boldsymbol{V}_j \boldsymbol{W}_j), \max(\mathbf{W}_j, \boldsymbol{V}_j)) \geqslant c_{k_{\mathrm{win}}}^{\mathrm{l}} \right\}$$

The membership values used in (18) are also calculated as $b_j(\boldsymbol{X}_h, \min(\boldsymbol{V}_j, \boldsymbol{W}_j), \max(\boldsymbol{W}_j, \boldsymbol{V}_j))$.

All the above derivations are a generalisation of the formulas from Section 3.1 and would simplify to the cases discussed in that section if for all the hyperboxes forming a classifier the values $w_{ji} \geqslant v_{ji}$ for all $i = 1..n$ and $j = 1..m$.

3.3. Example

To clarify some of the points introduced in the previous two sections and illustrate the way of dealing with missing values within the proposed algorithm a simple two dimensional example shown in Fig. 4 is now presented.

The classifier model consists of four hyperbox fuzzy sets representing two different classes. The parameters of the hyperboxes and some hypothetical cardinality values are shown in Table 1.

The fourth hyperbox is an example of a hyperbox with the second dimension which has not been set during the training (i.e. all the training data points used for the generation of this hyperbox had the second feature missing). Apart from the hyperboxes forming the GFMM classifier, seven input patterns to be classified are also shown. The first three patterns do not have any features missing, patterns 4 and 5 have the first feature missing while patterns 6 and 7 have the second feature missing. Additionally, the first feature of pattern seven is given in a form of lower and upper limit values. The numerical values for all

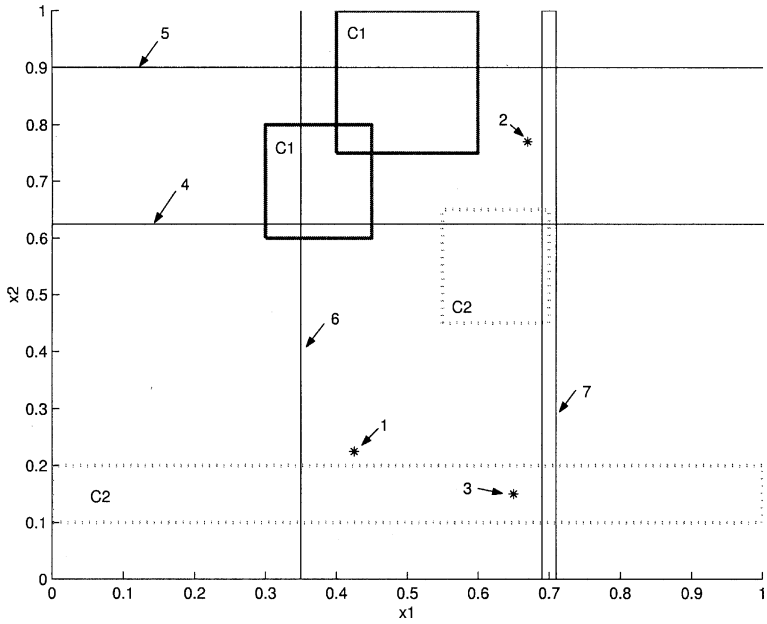

Fig. 4. Two-dimensional example of processing missing data within the proposed method.

Table 1
The parameters of hyperbox fuzzy sets shown in Fig. 4

Hyperboxes				
No. (j)	$\boldsymbol{V}_j$	$\boldsymbol{W}_j$	Class($\boldsymbol{B}_j$)	Cardinality of $\boldsymbol{B}_j$
1	[0.3 0.6]	[0.45 0.8]	1	12
2	[0.4 0.75]	[0.6 1]	1	20
3	[0.55 0.45]	[0.7 0.65]	2	15
4	[1 0.1]	[0 0.2]	2	10

the input patterns and classification results are shown in Table 2. All the degrees of membership have been calculated for the parameter $\gamma = 2$.

As it can be seen from Table 2 for input patterns 1, 2, 3, 5, and 7 only one alternative class has been found while for input patterns 4 and 6 both classes have been initially regarded as viable alternatives and the hyperbox cardinality information has been used to decide the more likely class. Note that when calculating P (last column in Table 2) all hyperboxes and their cardinalities are always used. Input pattern 4 provides an interesting case since according to (18) (a column marked P) the pattern should be classified as belonging to class 1 while according to $\boldsymbol{C}^{\mathrm{u}}$ and $\boldsymbol{C}^{\mathrm{l}}$ there is probably not enough information in the input pattern to choose one class over the other but after applying (16) (a column marked $\hat{P}$) the second class would be selected as a winner. This

Table 2
Input patterns and classification results for the example shown in Fig. 4

Input patterns			Classification			
No. (h)	$\boldsymbol{X}_h^{\mathrm{l}}$	$\boldsymbol{X}_h^{\mathrm{u}}$	$\boldsymbol{C}^{\mathrm{u}}$	$\boldsymbol{C}^{\mathrm{l}}$	$\hat{P}$	P
1	[0.425 0.225]	[0.425 0.225]	[0.25 0.95]	[0.25 0.55]	[0 1]	[0.15 0.85]
2	[0.67 0.77]	[0.67 0.77]	[0.86 0.76]	[0.86 0.76]	[1 0]	[0.68 0.32]
3	[0.65 0.15]	[0.65 0.15]	[0.1 1]	[0.1 0.4]	[0 1]	[0.07 0.93]
4	[1 0.625]	[0 0.625]	[1 1]	[1 1]	[0.44 0.56]	[0.62 0.38]
5	[1 0.9]	[0 0.9]	[1 0.5]	[1 0.5]	[1 0]	[0.8 0.2]
6	[0.35 1]	[0.35 0]	[1 1]	[1 0.6]	[0.55 0.45]	[0.61 0.39]
7	[0.69 1]	[0.71 0]	[0.78 1]	[0.78 0.98]	[0 1]	[0.46 0.54]

example also illustrates how the upper and lower degrees of class membership can provide additional level of information about the discriminative qualities of input data to be classified.

For input patterns 4 and 5 we can also observe that though in both cases the first feature is missing input pattern 5 can be quite confidently classified as belonging to class 1 just on the basis of the second feature value which is discriminative enough to do so, while input pattern 4 poses a much more difficult case as described above.

Other interesting cases include patterns 1, 3 and 6 the classification of which results in producing different values for $\boldsymbol{C}^{\mathrm{u}}$ and $\boldsymbol{C}^{\mathrm{l}}$ reflecting the fact that we are uncertain about the first dimension of hyperbox 4 which has not been set during the training and it is assumed that any value for feature one is possible.

4. Results

The testing of the proposed approach has been carried out on four well-known data sets, namely IRIS, Wine, Ionosphere, and SatImage obtained from the repository of machine learning databases (http://www.ics.uci.edu/~mlearn/MLRepository.html). The data sets have been selected due to their different characteristics concerning the number of samples, features and classes. They also represent four different classification problems drawn from different

Table 3
The sizes of data sets used in classification experiments

Data set	No. of inputs	No. of classes	No. of data points		
			Total	Train	Test
IRIS	4	3	150	75	75
Wine	13	3	178	90	88
Ionosphere	34	2	351	200	151
SatImage	36	6	6435	3219	3216

domains, i.e. classification of iris plants (IRIS data set), three types of wine (Wine data set), radar signals used to describe the state of ionosphere (Ionosphere data sets) and multi-spectral remotely sensed data of satellite images (SatImage). The sizes and splits into the testing and training data for all four data sets are shown in Table 3.

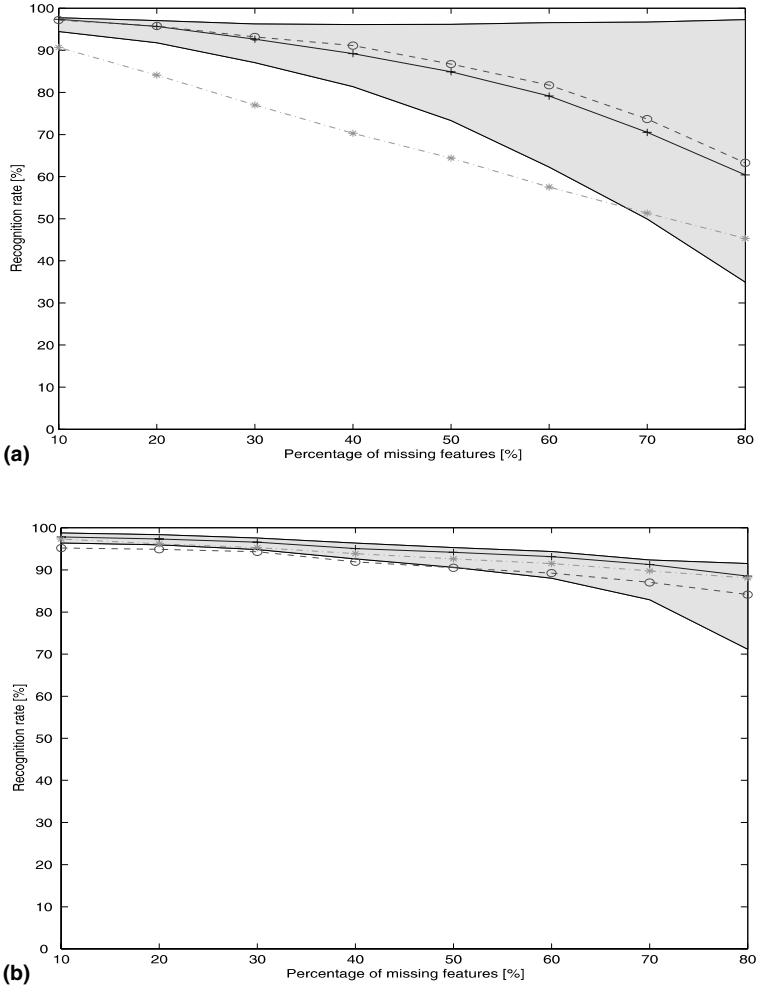

Fig. 5. Classifcation results of the IRIS data set for the proposed method and the nearest neighbour classifier with substituted mean value for the missing data: (a) training without missing values – testing for different levels of missing values; (b) training for different levels of missing values – testing without missing values; (c) training and testing for data with different levels of missing values. The solid line with '+' shows the classification results using (16); the dashed line with 'o' shows the classification results using (18); and the dash-dot line with '*' shows the classification results for the nearest neighbour classifier.

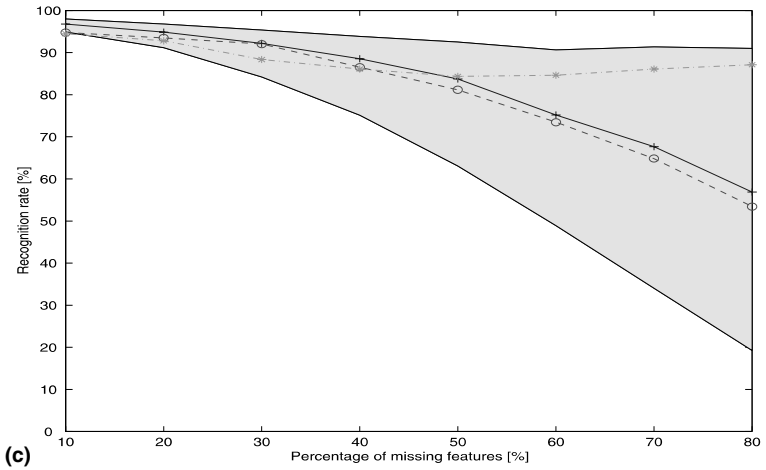

Fig. 5 (*continued*)

The results obtained using the proposed method and their comparison with the nearest neighbour classifier, with missing values substituted with the mean values calculated from the training set, are shown at Figs. 5–8. The testing of the influence of missing data on classification performance has been divided into three separate experiment groups.

In the first group, the GFMM neural network was trained using training data sets without missing values. Subsequently, the testing was conducted on all testing data sets for a factor of missing features ranging from 10% to 80%. At each level, the average values over 100 testing runs, each for randomly chosen missing features have been calculated. The results are shown in Figs. 5(a), 6(a), 7(a) and 8(a).

In the second group of experiments the GFMM was trained on the basis of training data sets with a factor of missing features ranging from 10% to 80%. The training has been repeated 50 times for IRIS, Wine and Ionosphere data sets and 20 times for SatImage data set, each time with randomly chosen missing features. The testing has been performed for the testing sets without any features missing. The results are shown in Figs. 5(b), 6(b), 7(b) and 8(b).

In the third group of experiments both training and testing sets contained the same level of missing features again between 10% and 80% of all features. The results are shown in Figs. 5(c), 6(c), 7(c) and 8(c).

The greyed area in all figures represents cases where 2 or more classes have been identified as viable alternatives (according to Eqs. (22)–(24)) with the correct class always present. The area below the greyed area refers to the unique, correct classification where only one, correct class has been chosen. The area above the greyed area represents misclassified cases. The solid line marked with '+' within the greyed area represents the classification rates obtained when

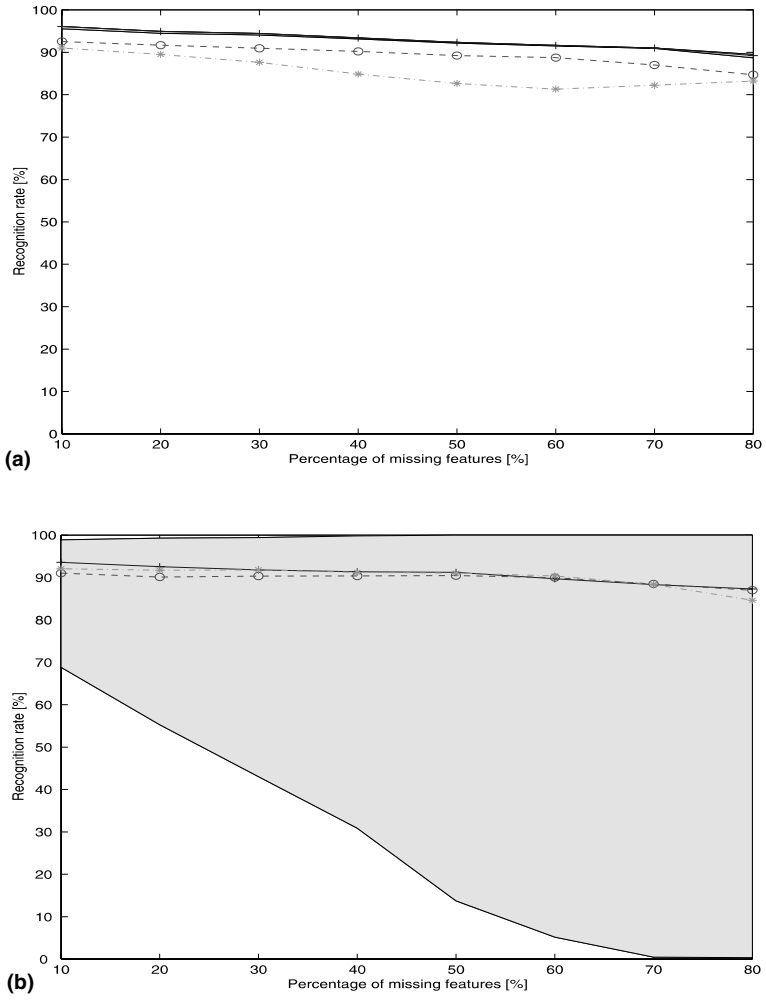

Fig. 6. Classification results of the Wine data set for the proposed method and the nearest neighbour classifier with substituted mean value for the missing data. Please see the caption of Fig. 5 for additional description.

hyperbox cardinality information is used according to (16). The dashed line with '○' shows the classification results using (18) and the dash-dot line with '*' shows the classification results for the nearest neighbour classifier.

The first comment which can be made on the basis of Figs. 5(a), 6(a), 7(a) and 8(a) is that the substituting of the mean value for the missing features and classifying using the nearest neighbour classifier produced much worse results than the proposed method. This is not surprising since it is just a confirmation

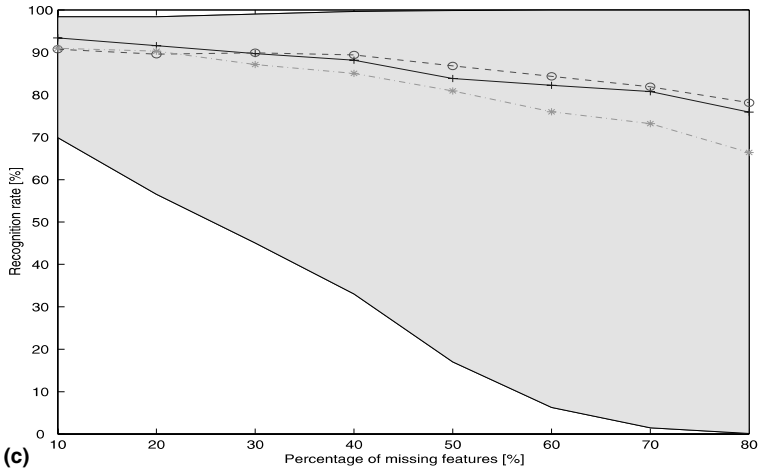

Fig. 6 (*continued*)

of what has been found by other researchers investigating such substitutions. It was though very surprising that the nearest neighbour method performed very well in the second group of experiments where the training data was generated with various levels of missing features and there were no missing data in the testing set. The reason for this can be explained on the basis of the two dimensional example shown in Fig. 9. The data are the normal mixtures data introduced by Ripley [20] and used in many other publications [5–8,15]. The training data consists of two classes with 125 points in each class. Each of the two classes has bimodal distribution and the classes were chosen in such a way as to allow the best-possible error rate of about 8%. An independent testing set of 1000 samples has been drawn from the same distribution. In the example shown in Fig. 9 the training data with 80% of missing features have been utilised. It means that out of the total number of 500 feature values only 100 have been used. As we can see from Fig. 9(a) substituting the missing values with mean values completely changes the distribution of the data which makes such "repaired" data completely inappropriate to use for training of other classifiers though the performance of the 1-nn classifier on the testing set does not decrease dramatically. On the other hand, the hyperboxes shown in Fig 9(b) generated on the basis of the same training data with 80% of missing features cover the original training data very well and the classification performance is also very good.

Another interesting observation based on Figs. 5(a), 6(a), 7(a) and 8(a) is that the level of misclassified cases remains roughly the same for a whole range of missing features. The increasing level of missing features is reflected in a

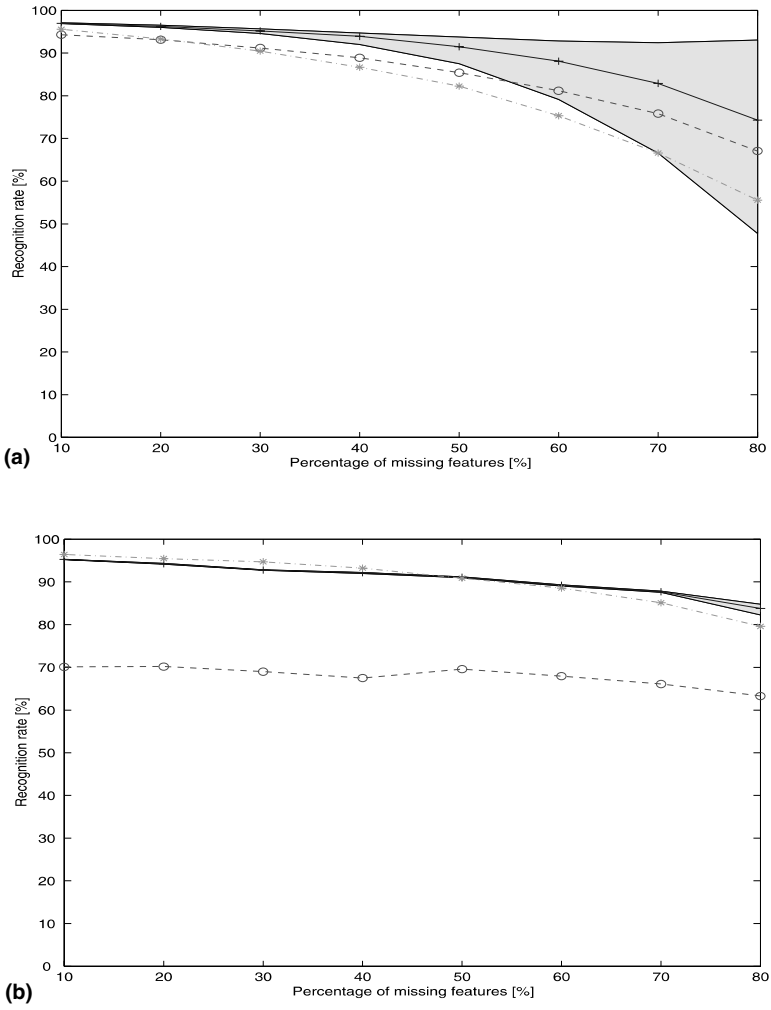

Fig. 7. Classification results of the ionosphere data set for the proposed method and the nearest neighbour classifier with substituted mean value for the missing data. Please see the caption of Fig. 5 for additional description.

higher percentage of cases for which classification resulted in producing more than one viable alternative but almost always including the correct class. This can be particularly observed for IRIS and Wine data sets. Please note that 80% of missing features for IRIS data set means that a significant proportion of the testing cases will have all features missing.

However, as it can be seen for Ionosphere and SatImage (up to 60% of missing features) data sets even for a high level of missing features, one winning

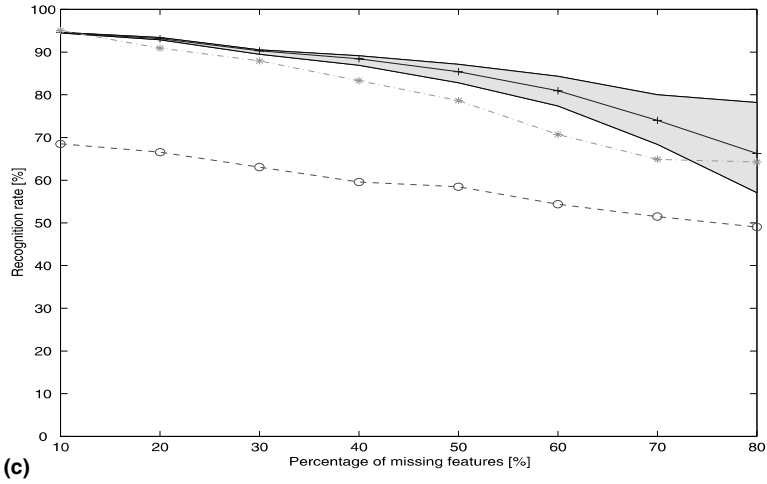

Fig. 7 (*continued*)

class is produced for a vast majority of testing cases (i.e. there is a very small grey area). This is an example of how the classification with missing features can result in a one class being selected if only the known features are discriminative enough to do so.

Another effect characteristic for the method used is illustrated for the Ionosphere data set in Fig. 7(b) and (c). It is an example for which a large number of hyperboxes have been retained in the final classifier model many of which did not have some of the dimensions set during the training. The upper and lower degrees of class membership overlap for both classes for a large number of testing cases which according to the test defined in Section 3.2 means that the two classes are initially treated as viable alternatives (large greyed areas). The use of cardinality information and Eqs. (16) and (18) result in a very good classification performance even for high ratios of missing features.

When all the hyperbox fuzzy sets are used for classification as in (18) it is very important that the parameter γ is chosen carefully as it decides how wide an influence of a hyperbox fuzzy set around the core hyperbox will be. It is essentially the same problem when one needs to select the Parzen window width or more specifically the variance of a gaussian when multidimensional gaussians are used for the data density approximation as in [27]. While there are procedures for finding a good estimate for γ based on cross-validation it does not change the fact that the results of applying (18) are very sensitive to the changes in γ. The illustration of this phenomena is given for SatImage data set in Fig. 8(a) and Wine data set in Fig. 6(b)–(c). The value of parameter $\gamma = 1$ was clearly too small (the hyperbox membership values did not decrease

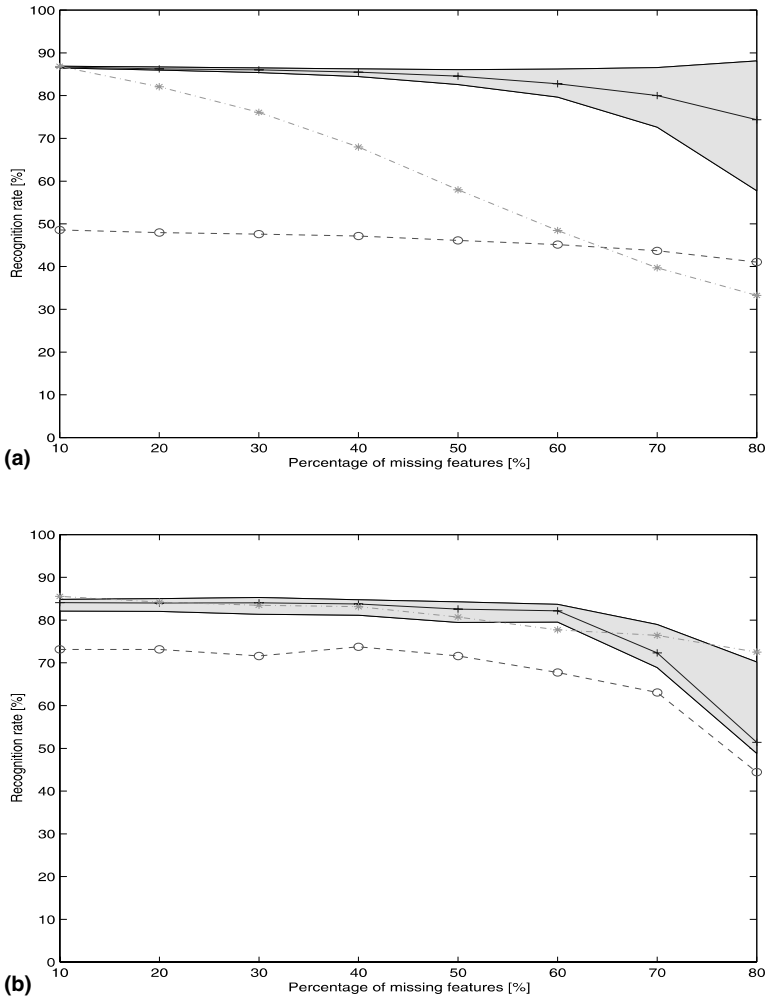

Fig. 8. Classification results of the SatImage data set for the proposed method and the nearest neighbour classifier with substituted mean value for the missing data. Please see the caption of Fig. 5 for additional description.

quickly enough) since the performance of (18) was significantly worse than (16). By increasing γ to 2, therefore making the hyperbox fuzzy sets more local, a significant improvement could be noticed as shown in Fig. 8(b)–(c) for Satlmage data set and in Fig. 6(a) for the Wine data set. It is worth noting that the performance based on (16) was not affected by changes in γ. It is because the hyperboxes to be used in (16) are selected as the best locally performing ones in the first place.

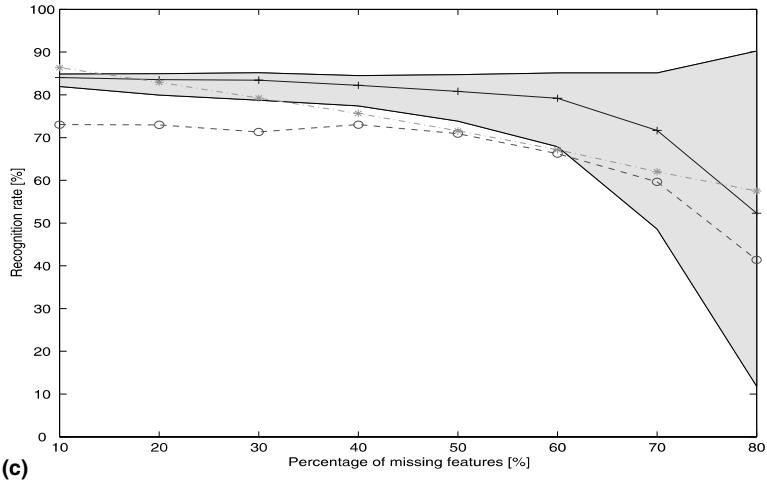

Fig. 8 (*continued*)

The performance of the GFMM classifier utilising formula (16) on IRIS data set is similar to the results reported in [13,18] and for the SatImage data set to the results reported in [1] though it has to be said that the emphasis in the above-mentioned publications had been on the learning with missing values while the testing was carried out on testing sets without missing features which in our case refers to results in Figs. 5(b) and 8(b) respectively.

5. Discussion and conclusions

Though seemingly very different the two groups of algorithms with their roots in probabilistic and fuzzy settings discussed in Section 1 have one major common characteristic. They all use a mechanism for building an input data model using a combination of local basis functions. In probabilistic approaches the data density estimation is carried out based on a number of multidimensional gaussians while in fuzzy approaches the triangular or trapezoidal membership functions serve a similar purpose of covering the input space. In other words after learning has been completed the input data space is covered by "patches" which are then combined to give a final classification model. The approach discussed in this paper also provides a mechanism for building an input data model by covering the input space with hyperbox fuzzy sets.

It has been shown that the GFMM neural network, with its definition of the inputs given in a form of real number intervals and hyperbox fuzzy sets used to represent input data clusters, provides a natural and efficient way of

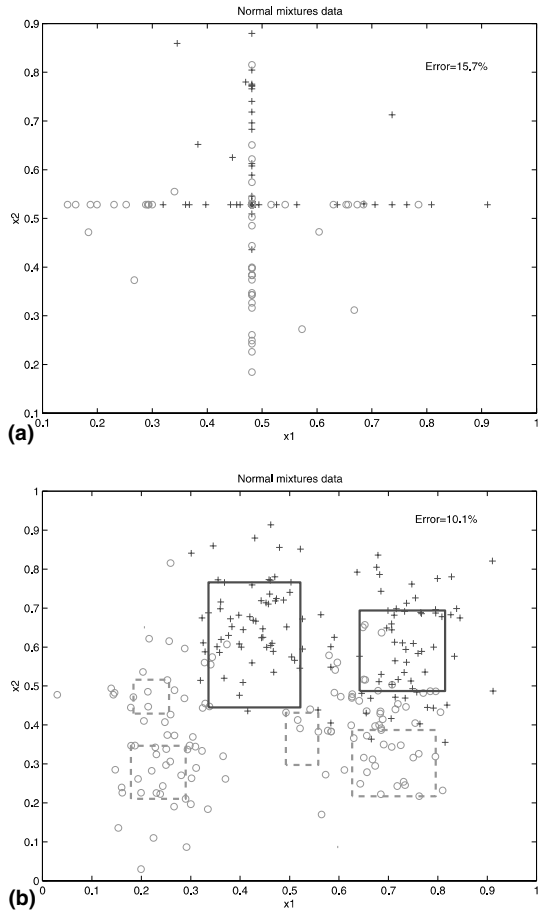

Fig. 9. The normal mixtures data with 80% of feature values missing: (a) nearest neighbour with mean values used as substitutions; (b) hyperboxes created for the training data with 80% of features missing. The complete original training data points are also shown. The error rates for an independent testing set of 1000 samples are shown in the right corners of (a) and (b).

representing and processing of missing data. No assumptions about the missing features or substitutions with fixed values have been used either during the classification or training of the classifier based on incomplete input patterns. What is also distinctive about our method is that the process of placing and adjusting the hyperboxes in the input space can lead to creation of some hyperbox fuzzy sets for which certain dimensions are not set. Such a hyperbox would effectively represent a subspace of the original input space. This could occur, for instance, when all the training data for one class would have consistently one or more features missing. Such a "not missing at random" ex-

ample could cause serious problems for the methods which learn the parameters of the classifier model by using random initialisation and assume that some values have to observed for each feature and all the classes. In contrast such an example would be efficiently processed and is catered for within the GFMM training algorithm which is virtually the same whether the missing values are present or not. The classifier model uncertainty resulting from training on incomplete data (like in the example above) can be quite easily quantified by generating upper and lower class membership values during the GFMM classifier operation.

It has been argued that the primary goal of the classification when missing data are involved should be a reduction of the viable class alternatives based on the evidence present in the observed features. This can lead to the selection of a single winning class but in cases when two or more viable alternative classes are identified we have also proposed formulas utilising the hyperbox cardinality values (the data frequency information) to find the most probable class. Such a procedure leads to a more informative classification and could be crucial in situations when important decisions have to be made on the basis of such classification.

One more advantage of the GFMM classifier, shared with the decision trees or rule based classifiers, is its transparency and ability to provide easily interpretable explanation of the suggested classification. The hyperbox min–max values can be easily converted into a set of rules understandable to a non-technical user. In cases when the classification results in a number of possible alternatives this would also provide an opportunity to highlight and suggest the features which would be most useful in reducing the number of alternatives. An example could include a hypothetical loan approval application where on the basis of initial processing of incomplete data a customer could be asked to provide an additional information, suggested by the classification system, which would reduce the risk/ambiguity of the decision to be made. The use of the proposed techniques within the medical diagnosis and financial decision support systems are currently under investigations.

Acknowledgements

Research reported in this paper has been supported by the Nuffield Foundation grant (NAL/00259/G).

References

[1] M.R. Berthold, K.-P. Huber, Missing values and learning of fuzzy rules, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems vol. 6 (2) (1998) 171–178.

[2] J.C. Bezdek, J. Keller, R. Krisnapuram, N.R. Pal, Fuzzy Models and Algorithms for Pattern Recognition and Image Processing, Kluwer Academic Publishers, Dordrecht, 1999.

[3] J.K. Dixon, Pattern recognition with partly missing data, IEEE Transactions on Systems, Man, and Cybernetics SMC-9 (10) (1979) 617–621.
[4] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, Wiley, New York, 2000.
[5] B. Gabrys, Combining Neuro-Fuzzy Classifiers for Improved Generalisation and Reliability, in: Proceedings of the Int. Joint Conference on Neural Networks (IJCNN '2002), a part of the WCCI '2002 Congress, Honolulu, USA, May 2002, pp. 2410–2415, ISBN 0-7803-72786.
[6] B. Gabrys, Learning hybrid neuro-fuzzy classifier models from data: to combine or not to combine? in: Proceedings of the EUNITE'2001 Conference, Tenerife, Spain, 2001.
[7] B. Gabrys, Data editing for neuro-fuzzy classifiers, in: Proceedings of the SOCO'2001 Conference, ISBN: 3-906454-27-4, Abstract page 77, Paper no. # 1824-036, Paisley, Scotland, 2001.
[8] B. Gabrys, Agglomerative learning algorithms for general fuzzy min-max neural network, Special issue of the Journal of VLSI Signal Processing Systems entitled Advances in Neural Networks for Signal Processing, 2002, to appear.
[9] B. Gabrys, Pattern classification for incomplete data, in: Proceedings of fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies KES'2000, Brighton, ISBN 0-7803-6400-7, vol. 1, August, 2000, pp. 454–457.
[10] B. Gabrys, Neural network based decision support: modelling and simulation of water distribution networks, Ph.D. Thesis, The Nottingham Trent University, 1997.
[11] B. Gabrys, A. Bargiela, Neural networks based decision support in presence of uncertainties, ASCE Journal of Water Resources Planning and Management 125 (5) (1999) 272–280.
[12] B. Gabrys, A. Bargiela, General fuzzy min–max neural network for clustering and classification, IEEE Transactions on Neural Networks 11 (3) (2000) 769–783.
[13] Z. Ghahramani, M.I. Jordan, Supervised learning from Incomplete DAta via an EM approach, in: J.D. Cowan, G. Tesauro, J. Alspector (Eds.), Advances in Neural Information Processing Systems, vol. 6, Morgan Kaufman, San Mateo, CA, 1994.
[14] J. Kittler, Classification of pattern vectors using modified discriminant functions, IEEE Transactions on Computers C-27 (4) (1978) 367–375.
[15] L.L. Kuncheva, Fuzzy Classifler Design, Physica, Heidelberg, 2000.
[16] R.J.A. Little, D.B. Rubin, Statistical Analysis with Missing Data, Wiley, New York, 1987.
[17] D. Nauck, R. Kruse, Learning in neuro-fuzzy systems with symbolic attributes and missing values, in: Proceedings of the International Conference on Neural Information Processing – ICONIP'99, Perth, 1999, pp. 142–147.
[18] M.J. Nijman, H.J. Kappen, Symmetry breaking and training from incomplete data with radial basis Boltzmann machines, International Journal of Neural Systems 8 (3) (1997) 301–315.
[19] W. Pedrycz, F. Gomide, An Introduction to Fuzzy Sets: Analysis and Design, MIT Press, Cambridge, MA, 1998.
[20] B.D. Ripley, Pattern Recognition and Neural Networks, Cambridge University Press, Cambridge, 1996.
[21] J.L. Shafer, Analysis of Incomplete Multivariate Data, Kluwer Academic Publishers, Dordrecht, 1997.
[22] G. Shafer, A Mathematical Theory of Evidence, Princeton University Press, Princeton, NJ, 1976.
[23] P.K. Sharpe, R.J. Solly, Dealing with missing values in neural network-based diagnostic systems, Neural Computing and Applications 3 (1995) 73–77.
[24] P.K. Simpson, Fuzzy min–max neural networks – Part 1: Classification, IEEE Transactions on Neural Networks 3 (5) (1992) 776–786.
[25] P.K. Simpson, Fuzzy min–max neural networks – Part 2: Clustering, IEEE Transactions on Fuzzy Systems 1 (1) (1993) 32–45.
[26] S. Theodoridis, K. Koutroumbas, Pattern Recognition, Academic Press, San Diego, CA, 1999.

[27] V. Tresp, R. Neuneier, S. Ahmad, Efficient methods for dealing with missing data in supervised learning, in: G. Tesauro, D.S. Touretzky, K. Leen (Eds.), Advances in Neural Information Processing Systems, MIT Press, Cambridge, MA, 1995.
[28] N. Tschichold-Gurman, V.G. Dabija, Meaning-Based handling of don't care attributes in artificial neural networks, in: Proceedings of the International Conference on Neural Networks, San Francisco, CA, ICNN'93, vol. 1, 1993, pp. 281–286.