Design of interpretable fuzzy rule-based classifiers using spectral analysis with structure and parameters optimization

Alexandre G. Evsukoff\textsuperscript{a,*}, Sylvie Galichet\textsuperscript{b}, Beatriz S.L.P. de Lima\textsuperscript{a}, Nelson F.F. Ebecken\textsuperscript{a}

\textsuperscript{a}COPPE/Federal University of Rio de Janeiro, P.O. Box 68506, 21941-972 Rio de Janeiro, RJ, Brazil
\textsuperscript{b}LISTIC/ESIA, University of Savoie, BP 806, 74016 Annecy Cedex, France

Received 2 June 2008; received in revised form 7 August 2008; accepted 19 August 2008
Available online 7 September 2008

Abstract

This paper presents a design method for fuzzy rule-based systems that performs data modeling consistently according to the symbolic relations expressed by the rules. The focus of the model is the interpretability of the rules and the model’s accuracy, such that it can be used as tool for data understanding. The number of rules is defined by the eigenstructure analysis of the similarity matrix, which is computed from data. The rule induction algorithm runs a clustering algorithm on the dataset and associates one rule to each cluster. Each rule is selected among all possible combinations of one-dimensional fuzzy sets, as the one nearest to a cluster’s center. The rules are weighted in order to improve the classifier performance and the weights are computed by a bounded quadratic optimization problem. The model complexity is minimized in a structure selection search, performed by a genetic algorithm that selects simultaneously the most representative subset of variables and also the number of fuzzy sets in the fuzzy partition of the selected variables. The resulting model is evaluated on a set of benchmark datasets for classification problems. The results show that the proposed approach produces accurate and yet compact fuzzy classifiers. The resulting model is also evaluated from an interpretability point of view, showing how the rule weights provide additional information to help data understanding and model exploitation.

© 2008 Elsevier B.V. All rights reserved.

Keywords: Fuzzy system models; Linguistic modeling; Pattern recognition; Genetic algorithms; Learning; Fuzzy statistics and data analysis

1. Introduction

Fuzzy rule-based methods for data analysis and modeling are often acknowledged to allow the representation of dependencies among numerical variables by relations among corresponding linguistic concepts. Although sometimes the numerical accuracy of the results is much more emphasized, the interpretability of the rule base has always been one of the main motivations of fuzzy systems modeling [42] and recently has been brought to attention [4,5,16,18,22,23,25,31,35,40].

* Corresponding author. Tel.: +55 21 25627388; fax: +55 21 25627392.
E-mail addresses: alexandre.evsukoff@coc.ufrj.br (A.G. Evsukoff), sylvie.galicht@univ-savoie.fr (S. Galichet), bia@coc.ufrj.br (B.S.L.P. de Lima), nelson@ntt.ufrj.br (N.F.F. Ebecken).
The interpretability of a fuzzy rule base is related to a number of factors [22,23,35], some of them are more subjective while others are closely related to the complexity of the model. This work is mainly concerned with the latter, such that interpretability of the model is achieved by controlling:

- the number of input variables;
- the number and position of the fuzzy sets in the domain of each input variable and
- the number of fuzzy rules, which should describe locally relevant relationships.

On the other hand, as the complexity of the problem increases, interpretability and accuracy become incompatible features [42]. Thus, the complexity (in the sense of non-interpretability) of the model is a function of the complexity of the problem itself and/or the quality of the available data.

The relationship between the complexity of the problem, as it is presented in the observed data, and the complexity of the resulting model has been the subject of statistical learning theory (SLT) [38]. The notion of “complexity” of a model in the SLT has been formalized as an integer quantity, called the Vapnik–Chervonenkis (VC) dimension, which is a measure of the capacity of the model to correctly learn a dataset. The design of pattern recognition algorithms under the SLT is based on the structural risk minimization principle that provides a “trade-off between the quality of the approximation of the given data and the complexity of the approximation function” [38].

One of the main results of the SLT is the support vector machines (SVM) [3,37], where the VC dimension of a classifier is shown to be related to the margin of separation between two classes. In the standard linear SVM algorithm, the margin of separation is maximized, under the constraint that all training points are well classified. The solution to the nonlinear case is solved by operations through kernels, which correspond to the inner products in the embedding or feature space [3,37], the so-called “kernel trick”. This approach has also led to the extension of a number of conventional techniques (such as k-means, principal components analysis, linear discriminant analysis, etc.) to their kernel counterparts [37].

The similarities between fuzzy rule-based models and SVM have been a subject of recent investigation [7–9]. In such approaches, multi-dimensional membership functions are used to construct kernels of the standard SVM algorithm. In [8], the authors show that kernels constructed as the product of symmetric membership functions are positive definite and could be used in the SVM algorithm, such that the resulting support vectors can be used to design the fuzzy rules. The rules obtained from support vector are difficult to be directly interpretable since they are defined on a multi-dimensional domain. Moreover the support vectors represent only the borders of the region assigned to each class, such that the rules obtained from them do not represent the whole data as it would be desired for data understanding. Nevertheless, these works have pointed out that fuzzy rules can be used to reproduce the embedding or feature space.

In interpretable fuzzy rule-based models, the set of rules performs the transformation from the original (data) space to the feature or embedding space such that the available training data can be interpreted as a set of symbolic rules. The main problem then becomes the determination of an adequate number of rules, which is related to the complexity of the problem and the capacity of the resulting model. As the problem becomes more complex, more rules are necessary to adequately describe the data and still achieve a good accuracy.

Kernel-based methods are also related to spectral methods, as pointed out by Filippone et al. [21] in a recent review. Spectral methods are based on the graph theory and provide the tools for the eigenstructure analysis of data [32,34,41]. In this work, spectral analysis is used to derive the adequate dimension of the feature space, which is assigned to the number of rules of the fuzzy rule-based system. It is shown that the number of rules computed in this way is affected by the number of variables used in the premises, such that the proposed approach must be employed in conjunction with a variable selection algorithm.

The learning procedure is composed of three main tasks: structure selection, rule base identification and parameters estimation. The structure selection task includes the variable selection and the determination of the number of fuzzy sets in each variable fuzzy partition. The structure selection follows a wrapper approach, implemented by a genetic algorithm (GA), of which the fitness function is based on a trade-off between the accuracy and the model complexity. In the rule base identification task, the number of relevant fuzzy rules is computed by spectral analysis and a cluster algorithm computes the position of rules. The interpretable fuzzy rules are chosen to be as close as possible to the center of clusters. In parameters estimation task, the optimal rule confidence parameters are computed by a bounded quadratic optimization problem, which can be solved by efficient numerical algorithms.
This paper is focused on classification problems, but the proposed formulation and learning procedure can also be applied to regression problems. In the next section, the symbolic approach for fuzzy rule-based modeling is introduced to set up the notation, to position the work with respect to other approaches found in the literature and to present the key issues of the identification algorithm. Section 3 presents the algorithm used to induce fuzzy rules, which is based on spectral analysis and data clustering, and the formulation of the bounded quadratic optimization problem for parameters estimation, which are the main contributions of the work. Section 3 also presents a discussion of the proposed procedure in simple datasets to show the effect of the number of variables and the complexity of the problem in the number of rules. Section 4 describes the GA-based structure selection procedure, which is necessary to obtain understandable fuzzy rules for real applications. In Section 5 the performance results of the proposed approach in benchmark problems is discussed and compared to similar approaches presented in the literature. Section 5 also presents the model interpretation and exploitation based on the rule weight parameters from an analytic point of view. Section 6 presents the concluding remarks and the directions for future research.

2. Fuzzy symbolic modeling

2.1. Problem statement

Consider a dataset composed of \(N\) input–output samples \(T = \{(x(t), y(t)), \quad t = 1, \ldots, N\}\), where \(x \in \mathbb{R}^p\) is the vector of input variables and \(y \in \mathbb{R}^q\) is the vector of output variables. A nominal valued input (output) variable is modeled as a discrete variable \(x_i \in \mathcal{X}(y_i \in \mathcal{Y})\), where \(\mathcal{X}\) is the set of natural numbers. Consider also a set of input symbols \(\mathcal{A} = \{A_i, \quad i = 1, \ldots, n\}\) and a set of output symbols \(\mathcal{B} = \{B_j, \quad j = 1, \ldots, m\}\), of which each element can be assigned to a linguistic concept (or to a set of linguistic concepts connected by logic operators) that refers qualitatively to the values of input and output variables, respectively.

An unknown relationship \(y = f(x)\) is supposed to exist among input and output variables, such that the main purpose of the learning task is to compute an approximation \(\hat{y} = \hat{f}(x)\), represented by a set of rules \(A_i \rightarrow B_j\) in the form

\[
\text{if } x(t) \text{ is } A_i \text{ then } y(t) \text{ is } B_j.
\] (1)

In this work, only the classification problem is considered, such that there is only one \((q = 1)\) nominal valued output variable \((y \in \mathcal{Y})\) and a value \(y = j\) refers to a class \(B_j \in \mathcal{B}\).

The symbolic approach is based on an abstraction of the machine learning problem in three layers as shown in Fig. 1. The lower layer is the observation layer, represented by a set of observed input–output data collected from the application. The symbolic layer is implemented by interpretable symbolic rules \(A_i \rightarrow B_j\), which relates input and output variables. The third layer is the linguistic layer, where it is assumed that human experts are able to interpret and understand the symbols and the relations defined by the symbolic rules.

![Fig. 1. The three layers of abstraction in data analysis.](image-url)
The relationship among the observed data (objects) and their symbolic representation (symbols) with the linguistic concepts (meaning) assigned to the data by an interpreter is the subject of interest of semiotics. Although it is a very interesting discussion, it is not in the scope of this work. This work deals with the design of the fuzzy symbolic model (FSM) that performs input–output data modeling according to the input–output symbolic relations expressed by the rules.

As a usual fuzzy system, the FSM computes an approximation \( \hat{y}(t) = \hat{f}(x(t)) \) in three steps, which are implemented as the operators:

- **Fuzzification:**
  \[
  \hat{u}(t) = F(x(t), \alpha, \Xi). \tag{2}
  \]

- **Inference:**
  \[
  \hat{v}(t) = I(\hat{u}(t), \Phi). \tag{3}
  \]

- **Defuzzification:**
  \[
  \hat{y}(t) = D(\hat{v}(t)). \tag{4}
  \]

The operators and their parameters are presented in the following.

### 2.2. Fuzzification

The fuzzification is the mapping of the input variable domain to an \( n \)-dimensional embedding or feature space \( F : \mathcal{R}^p \to [0, 1]^n \), where \( p \) is the dimension of the input variable domain and \( n \) is the dimension of the feature space, which is also the number of rules in the FSM. Fuzzification is parameterized by the set of prototype vectors \( \alpha = \{ \alpha_k, k = 1, \ldots, p \} \) and the fuzzy rule matrix \( \Xi = \{ \xi_{ik}, i = 1, \ldots, n, k = 1, \ldots, p \} \), computed by the structure selection and rule induction algorithms, respectively.

An observed sample \( x(t) \in \mathcal{R}^p \) is mapped into a feature (or fuzzy) vector \( u(t) \in [0, 1]^n \), whose components are computed as

\[
   u(t) = [u_1(t), \ldots, u_n(t)] = [\mu_{A_1}(x(t)), \ldots, \mu_{A_n}(x(t))]. \tag{5}
\]

Each rule in the FSM is expressed in terms of the symbol \( A_i \) that represents a region (or a patch) on the multi-dimensional input variables’ domain. A key issue is the number of fuzzy rules that, in this work, is computed by the eigenstructure analysis of the dataset, as it is presented in Section 3.

The rule premise \( x(t) \) is \( A_i \) is computed as the conjunction of elementary fuzzy predicates \( x_k(t) \) is \( A_k^i \), where the term \( A_k^i \) represents the fuzzy set defined in the domain of the variable \( x_k \) that should be considered as a component of the multi-dimensional symbol \( A_i \). The components of the feature vector are computed by the combination of elementary membership functions \( \mu_{A_k^i}(x_k(t)), k = 1, \ldots, p \) in the Cartesian product:

\[
   u_i(t) = \mu_{A_i}(x(t)) = \mu_{A_1^i}(x_1(t)) \wedge \cdots \wedge \mu_{A_p^i}(x_p(t)), \tag{6}
\]

where \( \wedge \) stands for a t-norm operator, generally the product.

The output of the rule induction algorithm, presented in Section 3, is the rule matrix \( \Xi = \{ \xi_{ik}, i = 1, \ldots, n, k = 1, \ldots, p \} \), of which an element \( \xi_{ik} \in \Xi \) defines which symbol in the partition of the variable \( x_k \) should be associated to the component \( A_k^i \) in the rule \( i \). A value \( \xi_{ik} = j \) indicates that the fuzzy set \( A_{kj} \) (i.e. the \( j \)th fuzzy set in the fuzzy partition of variable \( x_k \)) should be used as the component \( A_k^i \) in the rule \( A_i \).

The elementary fuzzy membership functions of the fuzzy sets \( A_{kj} \) are computed by equally spaced Gaussian functions as

\[
   \mu_{A_k^j}(x_k(t)) = \exp \left( -\frac{(x_k(t) - \xi_{kj})^2}{\kappa \rho_k^2} \right), \quad k = 1, \ldots, p, \quad j = 1, \ldots, n_k, \tag{7}
\]

where \( \kappa = -1/(4 \log(0.5)) \) is a constant such that the intersection of adjacent membership functions are at a membership value equal to 0.5, \( \rho_k \) is the dispersion parameter, defined by the number of fuzzy sets in the partition of the variable.
nk and \( \mathbf{x}_k = (x_{k1}, \ldots, x_{kn_k}) \) is the prototype vector, of which the components are supposed to be equally spaced in the variable domain.

Each variable is supposed to be standardized in the \([-1,1]\) interval as

\[
x_k(t) = \frac{\hat{x}_k(t) - \bar{x}_k}{s_k},
\]

where \( \hat{x}_k(t) \) is the observed (raw) value, \( x_k \) and \( s_k \) are respectively the mean value and the standard deviation estimations.

The fuzzy partition can be fully parameterized by the number of fuzzy sets \( n_k \) considering the dispersion parameter \( \rho_k = 2/(n_k - 1) \). The fuzzy partition defined in this way results in very interpretable fuzzy sets as shown in Fig. 2 for \( n_k = 5 \), where the index \( k \) representing the variable number is omitted.

The result of the fuzzification is normalized before its use in the inference process as

\[
\hat{\mathbf{u}}(t) = \frac{\mathbf{u}(t)}{\sum_{i=1}^{n_k} u_i(t)},
\]

where \( u_i(t) \) is computed as (6).

2.3. Inference

The inference is the mapping \( I : [0, 1]^n \rightarrow [0, 1]^m \), where \( n \) is the number of rules and \( m \) is the number of classes. A model is built by the fuzzy relation matrix \( \Phi \in [0, 1]^{n \times m} \), of which each component \( \varphi_{ij} = \mu_{\Phi}(A_i, B_j) \) represents the confidence of the rule \( A_i \rightarrow B_j \), such that the weighted rule is written as

\[
\text{if } x(t) \text{ is } A_i \text{ then } y(t) \text{ is } (B_1/\varphi_{i1}, \ldots, B_m/\varphi_{im}).
\] (10)

The fuzzy inference is computed by the fuzzy composition operator:

\[
\hat{\mathbf{v}}(t) = \hat{\mathbf{u}}(t) \circ \Phi.
\] (11)

Usual fuzzy composition operators are the max–min and the sum–product composition operators. The sum–product composition operator is used in this work since it produces a linear mapping in the feature space, such that Eq. (11) can be written in the vector–matrix form as

\[
\hat{\mathbf{v}}(t) = \hat{\mathbf{u}}(t) \cdot \Phi.
\] (12)

In classification problems, the output fuzzy symbols represent the classes and can be considered independently, such that \( \Phi = [\varphi_1, \ldots, \varphi_m] \). In this case, the components of the vector \( \hat{\mathbf{v}} \in [0, 1]^m \), are computed as the
product:
\[ \hat{v}_j(t) = \hat{u}(t) \cdot \varphi_j, \quad j = 1, \ldots, m, \] (13)

where \( \varphi_j \) is the rule confidence related to the class \( B_j \) and \( \hat{v}_j(t) \) is the class membership computed for the sample \( x(t) \), such that \( \hat{v}_j(t) = \mu_{B_j}(x(t)) \).

The use of confidence factors to express the certainty of fuzzy rules has been recently investigated for classification problems [15,26,28–30,43]. Rule weights allow more flexibility in the design of the model and also provide additional information on the quality of the rules. In this work the rule weights are computed as the solution of a constrained quadratic programming problem, as shown in Section 4. The resulting weights are also used to interpret and exploit the model, as discussed in Section 5.

2.4. Defuzzification

In regression problems, defuzzification is the mapping \( D : [0, 1]^m \rightarrow \mathbb{R} \), where \( m \) is the dimension of the output feature space. In classification problems, defuzzification is the mapping \( D : [0, 1]^m \rightarrow \mathbb{N} \) and computes the class output index, based on the output fuzzy symbol. Generally the maximum rule is used, such that the class index is computed as the component with the greatest membership value:

\[ \hat{y}(t) = j : v_j(t) = \max(\hat{v}(t)). \] (14)

The design of the FSM is presented in the next two sections. The next section introduces the rule induction algorithm based on spectral analysis. Section 4 presents the GA for structure selection.

3. Induction of fuzzy rules

The design of fuzzy rule-based systems has been widely studied and many approaches have been proposed as described in excellent recent reviews [15,18,22,23,25,40]. One of the major key points of the design of fuzzy rule-based systems is the definition of the number of rules and how to avoid the “curse of dimensionality”. Although fuzzy systems’ design seems to be a mature field, new approaches have recently been proposed in connection to the SVM algorithm [7–9]. Such approaches follow a somewhat similar reasoning to early neuro-fuzzy system: a fuzzy system is defined to be equivalent to the SVM formulation and the resulting support vectors are used to design the rules. The main drawback of these methods is that the SVM (and thereby the equivalent fuzzy system) is designed to represent the decision surface and not the entire dataset. Consequently, a sample close to the decision boundary is classified in the same way as another sample inside the region related to the class. Moreover, the presence of outliers is known to be a problem in SVM algorithms [37].

From a data understanding point of view, it is desirable that the rule base provides a representation of the whole learning domain and, if possible, that additional information could be granted on the position of an unknown sample with respect to the learning data. The rule induction algorithm presented in this section is designed to represent as much as possible the training dataset and to provide optimized rule weights that present further information on the relevance of the rules with respect to the classification problem. The approach is based on spectral analysis and on the formulation of a bounded quadratic optimization problem for parameters estimation, as described next.

3.1. Spectral analysis

The main tools of spectral methods are derived from spectral graph theory [11] and, in recent years, spectral clustering has been established as an efficient and meaningful alternative to traditional clustering techniques [21,32,34,41]. A complete review of the main spectral clustering techniques was recently presented by Luxburg [34].

Li et al. [32] presented an algorithm based on spectral analysis to derive the number of clusters in a dataset. The objective of the spectral analysis in this work is to provide an estimation of the number of rules, which has not necessarily the same meaning than the number of clusters.

Consider a graph \( G(V, E) \) associated to the training dataset, of which the set of nodes \( V \) represent the input variable samples and the set of edges \( E \) is defined by the \( N \times N \) adjacency (or affinity) matrix \( A \). The graph is undirected and
weighted, such that the affinity matrix is symmetric, real valued and its elements represent the similarity between two input variable samples. In this work, the similarity metric is computed by the Gaussian function, such that each element of affinity matrix is computed as

\[ a_{ij} = \begin{cases} 
\exp \left( -\frac{\|x(i) - x(j)\|^2}{2\sigma^2} \right) & \text{if } i \neq j, \\
0 & \text{otherwise}, 
\end{cases} \tag{15} \]

where \( \sigma \) is a dispersion parameter that controls the spread of the similarity function. Note that the affinity matrix is equal to the Gaussian kernel matrix without the main diagonal.

The \( N \times N \) matrix \( D \) is a diagonal matrix whose elements are the degree of the nodes of \( G \), computed by the sum of similarities of neighbors of each node:

\[ d_{ii} = \sum_{j=1,\ldots,N} a_{ij}. \tag{16} \]

The spectral clustering problem is related to the graph cut problem where the objective is to separate (cut) the set of nodes into two subsets, minimizing the number of edges between the two subsets. The optimal solution has been proved to be an NP-hard problem [21], such that an approximated solution is obtained by computing the eigenvalues of a transformation of the adjacency matrix, called the Laplacian matrix, computed as \( L = D - A \). The Laplacian matrix is usually normalized and there are some definitions for the normalized Laplacian, each one with its own properties [34]. In this work, the normalized Laplacian analyzed by Li et al. [32] is adopted, which is computed directly by normalizing the adjacency matrix as

\[ L = D^{-1/2} A D^{-1/2}. \tag{17} \]

Ng et al. [36] used the same definition of the normalized Laplacian for spectral clustering. Their algorithm shares the same basic idea of most spectral clustering algorithms, i.e. it finds a new representation of the data based on the largest eigenvalues of the normalized Laplacian matrix and then performs the clustering on this new representation. The main question, which is particularly interesting for rule induction, is the meaning of the word “largest”, which is directly related to the number of clusters.

The spectral graph theory provides the tools to analyze the structure of a dataset by looking at the eigenvalues of the normalized Laplacian matrix, computed by the eigendecomposition:

\[ L = Z \Lambda Z^T, \tag{18} \]

where \( Z \) is the orthogonal matrix of the eigenvectors and \( \Lambda \) is a diagonal matrix of the eigenvalues, which are all real, since the normalized Laplacian matrix is symmetric. The columns of \( \Lambda \) (and \( Z \)) are ordered so that the eigenvalues \( \lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_N \). When the normalized Laplacian matrix is computed as (17), the following properties of its eigenvalues are derived from the spectral graph theory [11,32]:

(1) \( \lambda_1 = 1 \) and \( \sum_{i=1,\ldots,N} \lambda_i = 0 \),
(2) \( -1 \leq \lambda_i \leq 1, i = 1,\ldots,N \) and
(3) if the graph is connected then \( \lambda_2 < 1 \).

By properties (1) and (2) above, it can be concluded that there will always be an integer \( K \), such that \( \lambda_i \geq 0, 1 < i \leq K \) and \( \lambda_j < 0, K < j \leq N \), usually \( K \leq N \).

Li et al. [32] also show that, for a dataset containing \( K \) disjoint clusters and if the adjacency matrix achieves perfect clustering, i.e.:

\[ \hat{a}_{ij} = \begin{cases} 
1 & \text{if } \mathbf{x}(i) \text{ and } \mathbf{x}(j) \text{ are in same cluster}, \\
0 & \text{otherwise}. 
\end{cases} \tag{19} \]

Then the eigenvalues of the normalized Laplacian computed by adjacency matrix (19) are

\[ \hat{\lambda_i} = \begin{cases} 
1, & 1 < i \leq K, \\
0, & K < i \leq N. 
\end{cases} \tag{20} \]
If another similarity function is used to compute the adjacency matrix, for instance the one based on the Gaussian function (15), then the solution of the eigendecomposition (18) is such that $\lambda_i \rightarrow \hat{\lambda}_i$, $i = 1, \ldots, N$ [32]. The number of positive eigenvalues of the Laplacian can thus be used to estimate the number of clusters.

At this point it would be interesting to consider some examples of the solution of the eigenvalues problem (18) in synthetic and real datasets. In the first example, shown in Fig. 3, two synthetic datasets were constructed by four samples of Gaussian distribution random variables with different means and equal variances in two dimensions. On the left-hand side, the clusters are well separated and the correct number of clusters is equal to the number of positive eigenvalues. On the right-hand side, six clusters are estimated, but the fifth and the sixth positive eigenvalues are very small.

The relative differences of the Laplacian components can be observed in Fig. 3b, where color shades represent qualitatively the matrix values. It can be seen that smaller relative differences correspond to a greater number of positive eigenvalues. This kind of representation is very useful for multivariate data analysis and has been used for the assessment of clustering quality [2,24].

In the second example, shown in Fig. 4, the well-known Iris dataset is considered. On the left-hand side, the similarity matrix is computed using all four variables and six clusters are estimated by the number of positive eigenvalues of the Laplacian. On the right-hand side, only two variables (petal length and petal width) are used and the number of positive eigenvalues is equal to the actual number of classes.

These examples show two main aspects that are directly related to the number of positive eigenvalues: the geometric separation of the classes and the number of variables. In other words, the number of positive eigenvalues of the Laplacian matrix is related to the complexity of the dataset and, thus, it could be used to estimate the number of rules.

In this work, the number of rules is estimated at a subspace of the input variables space where the problem would be less complex. The number of positive eigenvalues is reduced by selecting the appropriate variables to compute the affinity matrix. For any two records in the dataset, the corresponding adjacency component computed by (15) will be greater if fewer variables are used in its computation. The relative differences of the normalized Laplacian matrix will also be greater resulting in a reduction in the number of positive eigenvalues. A similar effect is obtained by changing the dispersion parameter. A smaller dispersion parameter reduces the relative similarity and thus increases the number of positive eigenvalues. In this work, the dispersion parameter is fixed at $\sigma = 1$, as the variables are considered to be standardized (8). This approach has obtained the best results during the development of the method.

For the rule induction algorithm to be effective at producing understandable rules, it must be coupled with a variable selection algorithm [22,23,31,35]. The rule induction algorithm and the rule weights parameters estimation are presented in the next two subsections, where additional requirements are obtained for the variable selection algorithm, which is presented in Section 4.

3.2. The rule induction algorithm

The combination of one-dimensional fuzzy sets is easier to understand than a multi-dimensional fuzzy set. Thus, the idea of the rule induction algorithm is to run a clustering algorithm on the dataset and then to associate a rule to each cluster. For each cluster, the nearest combination to the cluster center, among all possible combinations of fuzzy sets, is chosen as the rule to be included in the model.

This approach for rule induction is highly flexible and a large number of clustering strategies could be used [21]. In this work, three approaches for clustering are investigated:

- Spectral clustering according to the approach proposed by Ng et al. [36].
- Standard k-means applied on the input variables space.
- Hierarchical clustering using the complement of the adjacency matrix (15) as the distance matrix and the Ward’s linkage method.

In the spectral clustering approach, the k-means algorithm is applied over the matrix $\tilde{Z}$, computed as the first $K$ columns of the eigenvector matrix $Z$. Recall that the columns of $Z$ are ordered according to the corresponding eigenvalues, such that the first columns correspond to the largest eigenvalues. On this approach, $K$ clusters are computed on a $K$-dimensional space, so that the initialization may be computed by the $K$-dimensional identity matrix, avoiding the instabilities due to random initialization. The coordinates of clusters’ centers in the input variables space are computed
Fig. 3. Synthetic 2D examples: (a) the data in the 2D plane; (b) image display of the graph Laplacian and (c) the positive eigenvalues of the Laplacian.
by the average of the coordinates of the records assigned to each cluster. This is the main drawback of this approach, since the coordinate average in the input space does not correspond to the cluster center in the feature space.

In the second approach, the standard k-means is applied directly to the input variables space and the clusters’ centers are obtained as output. This approach has the advantage of computing the clusters directly in the input variable space. The well-known problems associated to random initialization must be avoided since the procedure is computed within the GA iteration. Thus, the eigenvectors of the Laplacian matrix are used to provide a deterministic initialization for the k-means algorithm as

\[ W_0 = X^T \tilde{Z} \]  

(21)

where \( X^T = [x(1)^T, \ldots, x(N)^T] \) is the data matrix, \( \tilde{Z} \) is the matrix computed as the first \( K \) columns of the eigenvector matrix \( Z \) and \( W_0 \in \mathbb{R}^{p \times K} \) is the initial guess for the clusters’ centers.

In the hierarchical clustering approach, the hierarchical cluster tree is cut at the level corresponding to the number of clusters specified by the spectral analysis. The Ward’s link method is used to join two clusters, such that the within sum of squares distances increases at each iteration. It is generally used when the distance function is not the Euclidean
distance, as in the present case. The cluster center coordinates are computed by the average of the coordinates of the records assigned to each cluster.

In this work, one fuzzy rule is computed as the combination of one-dimensional fuzzy sets nearest to each cluster center. This approach has been chosen because it is simple and efficient, but any other approach to derive interpretable fuzzy rules from multi-dimensional clusters can be adopted. For instance, Castellano et al. [6] have proposed an approach that also adjusts the fuzzy sets prototypes with respect to clusters’ centers.

The one-dimensional fuzzy membership functions, as defined by (7), depend only on the number of fuzzy sets in each input variable fuzzy partition. The rule induction algorithm is thus supposed to run within an outer loop for structure optimization, described in Section 4, which defines both the variables that should be used by the model and the number of fuzzy sets in each variable’s domain.

At each iteration, the structure optimization algorithm (cf. next section) provides the candidate structure represented by the vector \( \gamma = [\gamma_1, \ldots, \gamma_p] \), where \( 2 \leq \gamma_k \leq \gamma_{\text{max}} \) is the number of fuzzy sets in the fuzzy partition of the variable \( x_k \) and a value \( \gamma_k = 1 \) indicates that the variable \( x_k \) should not be considered in the model. A similar approach was proposed by Cordon et al. [14]. In their work, besides a different fuzzy inference procedure, the variable selection was not considered and the fuzzy rules were computed by the GA. The rule induction algorithm is sketched out in Algorithm 1.

**Algorithm 1.** Rule induction.

**input:** \( \gamma = [\gamma_1, \ldots, \gamma_p]; \alpha = \{x_k, k = 1, \ldots, p\}; T = \{(x(t), y(t)), t = 1, \ldots, N\} \)

**output:** \( \Xi = \{\tilde{z}_{ik}, i = 1, \ldots, n, k = 1, \ldots, p\}; \)

```plaintext
01 begin
02 // spectral analysis
03 compute the matrix A
04 compute the matrix D
05 compute \( L = D^{-1/2}AD^{-1/2} \)
06 n ← \( K; \lambda_i \geq \lambda_2 \geq \cdots \geq \lambda_N \)
07 \( W \leftarrow \text{clustering}(n) \)
08 // assign a rule to each cluster
09 for \( i = 1, \ldots, n \)
10 for \( k = 1, \ldots, p \)
11 if \( (\gamma_k > 1) \)
12 \( \tilde{z}_{ik} = j^k: x_{kj*} = \text{argmin}(x_{kj} - w_{ik}) \); // compute the nearest prototype
13 end
14 end
15 end
```

The number of rules in line 06 is chosen according to a parameter \( \delta \) to avoid small fluctuations around zero. The value \( \delta = 0.01 \) was used in the examples discussed in Section 5. The matrix \( W \in \mathbb{R}^{p \times n} \) in line 07 stores the cluster center coordinates, computed by either one of the three approaches discussed above. The rules are actually chosen in line 11, by the nearest prototype to the corresponding cluster center coordinates.

An example of the rule induction algorithm results is presented in Fig. 5 where, for the same two variables of the Iris dataset, the cluster center coordinates computed by the three clustering approaches can be slightly different. The dashed lines represent the location of the fuzzy sets’ prototypes in each one-dimensional domain, considering \( n_1 = n_2 = 5 \). The intersections of the dashed lines are the possible locations for the rules. The rules to be included in the model are indicated by the arrows. In this example, the different clustering approaches produce the same rules, but it is not the case in more general problems.

The set of rules issued by the rule induction algorithm defines the premises of (1) and represents a linguistic description of the learning dataset. The same set of rules is used for all classes, with the classification being assigned by the rule weights, which are computed as described next.
3.3. Rule weights optimization

Several heuristic approaches have been proposed to compute the rules’ parameters in weighted fuzzy systems [26,28–30,43]. In this work, the rules’ weights are computed as the solution of a bounded quadratic problem [19].

The rules’ parameters estimation corresponds to the determination of rule confidence weights \( \phi_j \) that relates the rule premise symbol \( A_i \) to the rule consequent (class) \( B_j \). Optimal rule weights \( \phi_j \) can be computed in the least squares sense, by solving the following bounded quadratic programming problem for each class \( B_j \):

\[
\begin{align*}
\text{minimize} & \quad \|U \phi_j - V_j\|^2 \\
\text{subject to} & \quad 0 \leq \phi_{ij} \leq 1, \quad i = 1, \ldots, n,
\end{align*}
\]

(22)

where \( U = [u(1), \ldots, u(N)]^T \), \( U \in [0, 1]^{N \times n} \), is the matrix of which each line is the fuzzy vector computed for each data sample as (5) and \( V_j = [v_j(1), \ldots, v_j(N)]^T \) is the column vector with the corresponding observed memberships of class \( B_j \), such that \( v_j(t) = \mu_{B_j}(y(t)) \). The bounds attempt to constrain the weights within the \([0,1]\) interval, such that they can be interpreted as fuzzy rule confidence. The bound constrains also avoid very high values for the rule weight parameters, providing the smoothness of the solution.

Expanding the quadratic term in the objective function and ignoring the constant terms, the problem (22) can be reformulated as

\[
\begin{align*}
\text{minimize} & \quad \frac{1}{2} \phi_j^T K \phi_j - C_j^T \phi_j \\
\text{subject to} & \quad 0 \leq \phi_{ij} \leq 1, \quad i = 1, \ldots, n,
\end{align*}
\]

(23)

where \( K = U^T U \) is a strictly positive definite matrix and \( C_j^T = V_j^T U \).

The quadratic optimization problem defined by (23) is widely known and the solution can be computed by efficient numerical algorithms [12,37].

The resulting rules’ weights can be used to interpret the model. Typically, in the solution of (23), three types of rule weights values can occur:

- \( \phi_{ij} = 0 \): Meaning that the rule \( A_i \rightarrow \neg B_j \) is certain and that the region defined by the symbol \( A_i \) cannot be assigned to class \( B_j \).
- \( 0 < \phi_{ij} < 1 \): Meaning that the rule is uncertain and that the symbol \( A_i \) represents a region on the border of the class \( B_j \).
For a fixed model structure, the more the solution values $\varphi = 1$, the more certain is the classifier and also the greater is the norm of the weight vector, defined by

$$\|\varphi_j\|^2 = \varphi_j^T \varphi_j, \quad j = 1, \ldots, m. \quad (24)$$

The sum $\sum_{j=1}^m \|\varphi_j\|^2$, when compared with the number of rules $n$, is a measure of the amount of certain rules and can thus be interpreted as a measure of the classifier quality.

Other rule quality measures have also been studied in the domain of fuzzy association rules. The most typical measures are support and confidence, defined as [17]

$$supp(A_i \rightarrow B_j) = \sum_{t=1}^N \mu_{A_i}(x(t)) \cdot v_j(t). \quad (25)$$

$$conf(A_i \rightarrow B_j) = \frac{\sum_{t=1}^N \mu_{A_i}(x(t)) \cdot v_j(t)}{\sum_{t=1}^N \mu_{A_i}(x(t))}. \quad (26)$$

Ishibushi and Yamamoto [28,29] proposed the use of the rule confidence measure as the rules’ weights, but within a different inference procedure. In this work, the confidence measure (26) is used as an initial guess to the solution of the bounded quadratic problem (23).

The complexity of the rule induction algorithm is dominated by the spectral analysis, which is a very computational intensive task. The time and storage complexity of the affinity matrix computation is $O(N^2)$, while the full eigendecomposition of a symmetric matrix requires at least $O(N^3)$ floating point operations. However, only the first eigenvalues are necessary, such that efficient iterative subspace decomposition algorithms could be used, allowing to reduce the time complexity of the spectral analysis to $O(N^2)$. The time complexity of the k-means clustering algorithm is $O(NMn)$, where $M$ is the maximum number of interactions. The complexity of the hierarchical clustering algorithm is $O(N)$ since it uses the affinity matrix already computed for spectral clustering as the distance matrix. An evaluation of the processing time of the proposed approach in benchmark datasets is presented in Section 5.

The rule induction algorithm presented above is supposed to run within a structure selection algorithm described next.

4. Structure selection

The structure selection algorithm should provide an estimation of the structure vector $\gamma = [\gamma_1, \ldots, \gamma_p]$, which defines the variables that must be included in the model and also the number of fuzzy sets in the domain of each variable.

The variable (feature) selection problem has been widely studied and a large number of algorithms have been proposed [33]. Generally speaking, there are three main approaches for variable selection: the filter, the wrapper and the hybrid approaches. The generalized filter and wrapper approaches are very similar in the sense that a search strategy evaluates subset candidates using an independent performance measure, for the filter approach, or a data mining algorithm, for the wrapper approach. The hybrid approach combines the previous approaches using an independent measure to decide the best subsets for a given number of variables and the data mining algorithm to select the variable subset. The wrapper approach is generally more suitable for data mining applications although it can also be more time consuming.

In a variable selection algorithm, the search strategy defines which candidate subsets must be evaluated. The simplest search strategy is the incremental forward (or backward) search, in which each variable is added to (or subtracted from) the model according to its ability to improve the model’s result. Incremental strategies have also been used to define the number of fuzzy sets in the domain of each variable for regression problems [20]. In this work, the problem of selecting the appropriate number of fuzzy sets for each variable is included in the variable selection problem, such that the number of possible solutions becomes huge and unfeasible for an incremental strategy. Stochastic search strategies become more suitable when the search space becomes very large.
GAs have been widely used to design fuzzy rule-based systems [5,13,28]. Different approaches have been proposed for variable selection, design of the rule base and also the shape of fuzzy sets’ membership functions. Although it is an intensive research area, the main focus of this work is on the design of the rule base by spectral analysis and parameters optimization, such that sophisticated genetic operators and complex variables encodings are not used. The simplest GA is adopted to solve the structure selection problem, so that the procedure can be implemented in any standard, accessible and general purpose GA code.

When applying a GA to a new problem, besides the GA’s intrinsic operators and parameters, two main issues must be defined by the user: the codification of the variables into individuals and the fitness or objective function.

In the structure selection problem, each individual of the GA represents the structure vector \( \gamma = [\gamma_1, \ldots, \gamma_p] \). For each candidate solution generated by the GA, a rule base is computed as described by Algorithm 1 (cf. Section 3), considering the selected variables and the number of fuzzy sets for each variable specified by the structure vector.

One of the main advantages of the GA is the freedom to select the objective function. In this work, the fitness function is based on a trade-off between the model performance and the complexity of the approximation function. The objective function to be minimized is the total risk \( R(\gamma) \), composed of two terms:

\[
R(\gamma) = R_e(\gamma) + R_c(\gamma),
\]

where \( R_e(\gamma) \) is the empirical risk, based on the classification error, and \( R_c(\gamma) \) is the complexity risk that must penalize models with high number of rules.

In order to be consistent with the rules’ parameters estimation, the empirical risk is computed as the objective function of the bounded quadratic problem (22). The complexity term is computed by rules’ weight vector norm (24). The total risk functional is then computed by summing the corresponding terms for all classes:

\[
R(\gamma) = \frac{1}{2N} \sum_{j=1,\ldots,m} \|U(\gamma)\Phi_j(\gamma) - V_j\|^2 + \frac{1}{2N} \sum_{j=1,\ldots,m} \|\Phi_j(\gamma)\|^2,
\]

where the term \( 1/2N \) is used to rescale the risk functional to the \([0,1]\) interval.

The risk functional (28) has the form of the regularized functional often used to solve ill-conditioned problems [37]. It has the effect of a trade-off between the model’s accuracy and capacity according to the required complexity of the problem. For a very simple problem, the empirical risk term is small and the structure selection will select a model with a small number of rules in order to reduce the complexity term. On the other hand in a more complex problem, it can be expected that a model with a higher number of rules would have better accuracy but, also, more certain rules, resulting in a greater value of the complexity term, so that the complexity and the accuracy terms must “negotiate” to find equilibrium.

A similar formulation of the fitness function was proposed by Cordón et al. [15], where the complexity term was simply the number of rules and the relative importance of the two terms were weighted by parameters. More recent approaches consider the two terms in multi-objective optimization problem [22,28]. Ishibushi and Yamamoto [28] also consider a third factor that penalizes the number of variables in the fuzzy rules. The multi-objective approach could be more useful when integrating several objectives [27,39]. The fitness function (28) is not only a function of the number of rules; it is mainly a function of the number of certain rules, which is related to the complexity of the problem and, consequently, to the accuracy of the model. The functional (28) represents the complexity term as a penalty to the accuracy in one single objective.

At the GA initialization, one single variable FSM is computed for each variable, with the number of fuzzy sets varying from 2 to \( \gamma_{\text{max}} \), without using the rule induction algorithm presented in Section 3.2. From the set of all results, the best individuals were chosen to start the evolutionary process.

All tests discussed in the next section were performed within the Matlab environment, using the standard routine provided in the software to compute the full eigenvalue decomposition in spectral analysis. The genetic wrapper was implemented using the GA Toolbox version 1.2 [10], a former free source code that became a commercial product. The population was fixed at 25 individuals for all tests and the standard crossover and mutation operators were used with 70% and 5% of crossover and mutation rates, respectively. The maximum number of iterations was fixed at 500, but the process was halted after 50 iterations without a change in the best fitness value. The structure vector was represented using binary coding, with 3 bits per variable, which allows up to \( \gamma_{\text{max}} = 8 \) fuzzy sets per variable (cf. [35]).
5. Results and discussion

5.1. Performance results on benchmark datasets

The performance evaluation of the FSM classifier was based on a set of benchmark classification datasets obtained from the UCI machine learning repository [1]. The selected datasets for the analysis are shown in Table 1. The accuracy results obtained for the same datasets by the SMO (sequential minimal optimization algorithm for training SVM) classifier implemented in the Weka\(^1\) data mining suite, using all default parameters, are also presented in Table 1. Results for similar fuzzy models presented in the literature are also presented, but these results should be considered as reference values as they were not obtained under the same experimental conditions.

The performance evaluation is based on the comparison of the results of the three following approaches for the FSM design, according to the clustering algorithm:

- **FSM-SC**: Spectral clustering according to the approach proposed by Ng et al. [36].
- **FSM-KM**: Standard k-means on the input variables space.
- **FSM-HW**: Hierarchical clustering using the complement of the affinity matrix (15) as the distance matrix and the Ward’s linkage method.

In order to evaluate the improvement provided by the spectral analysis, the Wang and Mendel rule induction algorithm [40] was included in the analysis. A similar approach using triangular fuzzy sets was proposed in an earlier work [19]. In the FSM design based on the Wang and Mendel algorithm (FSM-WM), the rule matrix \( \mathbf{E} \) is computed from each data record by the most activated fuzzy set of each variable fuzzy partition, provided that no repeated rules are included in the model. The fuzzy sets, the rule inference, the rule base weights optimization and the GA wrapper are computed as described above and are the same for all four design approaches.

All the results were generated under the same experimental conditions: the tests were performed in 10-fold cross-validation analysis, i.e. for each benchmark dataset, the GA structure selection algorithm was run 10 times. The classification accuracy results are shown in Table 2, where the best values obtained by the FSM approaches discussed in this work are highlighted. The results of the SVM-SMO algorithm are repeated in Table 2 to ease the comparison between the methods. The accuracy results presented in Table 2 were computed by summing the results obtained in all 10 test sets during the cross validation.

The results are quite similar; each one of the classifiers has obtained the best accuracy result in some tests. Nevertheless, in most of the cases, the proposed approach has obtained better accuracy results than other fuzzy models recently reported in the literature (cf. Table 1).

The clustering approach has thus an important influence on the accuracy results and could be included into the GA wrapper. The spectral clustering approach (FSM-SC) has obtained the best results for the last two datasets, which are the ones with the greatest number of variables. The model based on the Wang and Mendel rule induction algorithm (FSM-WM) has also obtained the best results in three benchmarks (Iris, Balance and Heart), although for two of them (Iris and Heart), it was tied with another model based on spectral analysis.

The accuracy values presented by the FSM classifier are also very similar to those reported in the literature by similar approaches. A direct comparison among such values should be made with care since experimental conditions were not the same. The FSM classifier has also obtained similar, and often better, results than the SVM classifier implemented in Weka (SMO algorithm) with default parameters.

The average and the standard deviation of the number of rules computed in the models generated by the 10-fold cross validation are shown in Table 3. It can be seen that in the second dataset (Balance), for which the FSM-WM approach has obtained the best accuracy result, it was done with a number of rules that exceeds strongly the spectral approaches. The result is the opposite in the last benchmark (Sonar), where the FSM-WM approach has obtained a very compact model, but with a poor accuracy. In the other benchmarks, even with a greater number of rules, the FSM-WM approach did not obtain better accuracy.

---

1 http://www.cs.waikato.ac.nz/ml/.
Table 1
Benchmark datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>pm</th>
<th>m</th>
<th>N</th>
<th>SMO</th>
<th>[15]</th>
<th>[28]</th>
<th>[29]</th>
<th>[30]</th>
<th>[27]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris</td>
<td>4</td>
<td>3</td>
<td>150</td>
<td>96.7</td>
<td>95.7</td>
<td>96.4</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Balance</td>
<td>4</td>
<td>3</td>
<td>625</td>
<td>60.2</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Diabetes</td>
<td>8</td>
<td>2</td>
<td>768</td>
<td>77.3</td>
<td>74.6</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>75.80</td>
</tr>
<tr>
<td>Cancer</td>
<td>9</td>
<td>2</td>
<td>286</td>
<td>96.9</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>68.2</td>
<td>53.27</td>
</tr>
<tr>
<td>Glass</td>
<td>9</td>
<td>6</td>
<td>214</td>
<td>57.0</td>
<td>–</td>
<td>–</td>
<td>68.2</td>
<td>53.27</td>
<td>61.64</td>
</tr>
<tr>
<td>Wine</td>
<td>13</td>
<td>3</td>
<td>178</td>
<td>98.9</td>
<td>92.9</td>
<td>97.2</td>
<td>95.5</td>
<td>99.44</td>
<td>93.48</td>
</tr>
<tr>
<td>Heart (Statlog)</td>
<td>13</td>
<td>2</td>
<td>270</td>
<td>82.9</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Image</td>
<td>18</td>
<td>7</td>
<td>210</td>
<td>89.5</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>82.86</td>
<td>–</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>34</td>
<td>2</td>
<td>351</td>
<td>88.6</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Sonar</td>
<td>60</td>
<td>2</td>
<td>208</td>
<td>75.9</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>81.25</td>
<td>75.71</td>
</tr>
</tbody>
</table>

Table 2
Accuracy results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SMO</th>
<th>FSM-SC</th>
<th>FSM-KM</th>
<th>FSM-HW</th>
<th>FSM-WM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris</td>
<td>96.7</td>
<td>95.33</td>
<td>95.33</td>
<td>96.00</td>
<td>96.00</td>
</tr>
<tr>
<td>Balance</td>
<td>60.2</td>
<td>76.00</td>
<td>77.44</td>
<td>77.12</td>
<td>86.40</td>
</tr>
<tr>
<td>Diabetes</td>
<td>77.3</td>
<td>75.52</td>
<td>76.82</td>
<td>75.52</td>
<td>75.26</td>
</tr>
<tr>
<td>Cancer</td>
<td>96.9</td>
<td>96.63</td>
<td>96.19</td>
<td>96.93</td>
<td>95.61</td>
</tr>
<tr>
<td>Glass</td>
<td>57.0</td>
<td>67.29</td>
<td>68.22</td>
<td>66.36</td>
<td>65.89</td>
</tr>
<tr>
<td>Wine</td>
<td>98.9</td>
<td>94.94</td>
<td>95.51</td>
<td>96.07</td>
<td>91.01</td>
</tr>
<tr>
<td>Heart (Statlog)</td>
<td>82.9</td>
<td>80.74</td>
<td>82.96</td>
<td>80.00</td>
<td>82.96</td>
</tr>
<tr>
<td>Image</td>
<td>89.5</td>
<td>86.19</td>
<td>86.63</td>
<td>87.79</td>
<td>84.76</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>88.6</td>
<td>92.15</td>
<td>86.67</td>
<td>87.14</td>
<td>83.72</td>
</tr>
<tr>
<td>Sonar</td>
<td>75.9</td>
<td>81.73</td>
<td>76.44</td>
<td>79.81</td>
<td>72.60</td>
</tr>
</tbody>
</table>

Table 3
Number of rules

<table>
<thead>
<tr>
<th>Dataset</th>
<th>FSM-SC</th>
<th>FSM-KM</th>
<th>FSM-HW</th>
<th>FSM-WM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris</td>
<td>3.0 ± 0.00</td>
<td>3.0 ± 0.00</td>
<td>3.0 ± 0.00</td>
<td>5.1 ± 0.32</td>
</tr>
<tr>
<td>Balance</td>
<td>4.9 ± 0.32</td>
<td>5.0 ± 0.00</td>
<td>4.8 ± 0.42</td>
<td>33.6 ± 18.37</td>
</tr>
<tr>
<td>Diabetes</td>
<td>12.3 ± 2.26</td>
<td>12.2 ± 1.69</td>
<td>11.7 ± 2.36</td>
<td>18.4 ± 3.66</td>
</tr>
<tr>
<td>Cancer</td>
<td>7.7 ± 2.11</td>
<td>7.8 ± 1.32</td>
<td>8.3 ± 0.82</td>
<td>8.7 ± 2.21</td>
</tr>
<tr>
<td>Glass</td>
<td>10.0 ± 1.25</td>
<td>10.1 ± 0.99</td>
<td>10.1 ± 0.88</td>
<td>21.4 ± 4.77</td>
</tr>
<tr>
<td>Wine</td>
<td>8.2 ± 2.25</td>
<td>8.1 ± 2.02</td>
<td>7.5 ± 1.27</td>
<td>13.9 ± 3.14</td>
</tr>
<tr>
<td>Heart (Statlog)</td>
<td>13.6 ± 1.78</td>
<td>14.1 ± 2.23</td>
<td>13.9 ± 1.73</td>
<td>13.8 ± 1.99</td>
</tr>
<tr>
<td>Image</td>
<td>12.7 ± 1.42</td>
<td>13.8 ± 1.48</td>
<td>12.4 ± 2.17</td>
<td>20.9 ± 3.00</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>25.0 ± 3.02</td>
<td>24.7 ± 3.02</td>
<td>24.8 ± 3.26</td>
<td>11.5 ± 5.56</td>
</tr>
<tr>
<td>Sonar</td>
<td>35.8 ± 2.52</td>
<td>28.3 ± 4.06</td>
<td>34.8 ± 2.74</td>
<td>5.3 ± 3.47</td>
</tr>
</tbody>
</table>

All the models, except the FSM-WM have used the same eigenstructure analysis to determine the number of rules. However, the clustering approach in the rule induction algorithm affects the performance and, consequently, the selection of variables. The number of variables has a strong influence on the number of rules, such that the clustering approach indirectly affects the number of rules. Nevertheless, the numbers of rules obtained for the three spectral analysis-based FSM classifiers were very similar, except for the last benchmark dataset (Sonar).

The experimental results presented above were obtained on an Intel Quad-Core 2.66GHz workstation running the Windows XP 64 bits operational system. The mean processing time, in seconds, computed to generate each model in the cross-validation procedure is shown in Table 4, where the greatest values are highlighted. The total time for the model validation for each dataset is thus 10 times the value presented in Table 4.
Table 4
Processing time (in seconds)

<table>
<thead>
<tr>
<th>Dataset</th>
<th># records</th>
<th>FSM-SC</th>
<th>FSM-KM</th>
<th>FSM-HW</th>
<th>FSM-WM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris</td>
<td>135</td>
<td>92.9</td>
<td><strong>109.2</strong></td>
<td>83.3</td>
<td>32.5</td>
</tr>
<tr>
<td>Balance</td>
<td>563</td>
<td>1276.8</td>
<td>1175.9</td>
<td><strong>1327.1</strong></td>
<td>82.9</td>
</tr>
<tr>
<td>Diabetes</td>
<td>691</td>
<td><strong>3100.5</strong></td>
<td>2931.6</td>
<td>2846.4</td>
<td>82.2</td>
</tr>
<tr>
<td>Cancer</td>
<td>257</td>
<td>1931.9</td>
<td>2183.8</td>
<td><strong>2936.6</strong></td>
<td>64.0</td>
</tr>
<tr>
<td>Glass</td>
<td>193</td>
<td>351.8</td>
<td><strong>426.6</strong></td>
<td>316.8</td>
<td>181.5</td>
</tr>
<tr>
<td>Wine</td>
<td>160</td>
<td><strong>317.2</strong></td>
<td>308.6</td>
<td>249.6</td>
<td>82.4</td>
</tr>
<tr>
<td>Heart (Statlog)</td>
<td>243</td>
<td>625.3</td>
<td><strong>799.0</strong></td>
<td>411.4</td>
<td>112.0</td>
</tr>
<tr>
<td>Image</td>
<td>189</td>
<td>1031.6</td>
<td><strong>1109.8</strong></td>
<td>736.7</td>
<td>411.4</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>316</td>
<td><strong>1422.6</strong></td>
<td>1378.5</td>
<td>865.9</td>
<td>155.0</td>
</tr>
<tr>
<td>Sonar</td>
<td>187</td>
<td><strong>831.4</strong></td>
<td>592.0</td>
<td>388.4</td>
<td>139.4</td>
</tr>
</tbody>
</table>

The results in Table 4 show that the time required by the Wang and Mendel fuzzy rule induction algorithm is much smaller than the time required by the algorithms based on spectral analysis for all the benchmark datasets. It can be concluded that the proposed rule induction algorithm generate more compact models than the Wang and Mendel algorithm, but the spectral analysis is much more time consuming. Moreover, the GA wrapper and the optimization of the confidence rules, as proposed in this work, have allowed the fuzzy model generated by the Wang and Mendel rule induction algorithm to achieve good accuracy results.

Among the algorithms using the spectral analysis, each one of the k-means-based approaches (FSM-SC and FSM-KM) has had the greatest time in four datasets, while the hierarchical clustering approach (FSM-HW) has had the greatest time in two datasets. This is certainly due to the fact that the hierarchical clustering approach uses the affinity matrix computed for spectral analysis as the dissimilarity matrix.

The number of records in the training set during cross validation (90% of the entire dataset) is also shown in Table 4. It can be seen that the processing time required for the whole procedure is not proportional to the size of the problem. This is due to the convergence criterion that halts the GA iteration after 50 generations without a change in the best fitness value.

The quantitative results presented above are very important when evaluating the usefulness of the proposed method. Nevertheless, the main motivation of the work was the rule base interpretation and exploitation as discussed next.

5.2. Model interpretation and exploitation

For model interpretation, one FSM model was generated for each benchmark dataset using all records in the datasets and the spectral clustering option (FSM-SC) for the rule induction algorithm. The same parameters of the GA used in the cross-validation analysis were used to generate the models.

The results are shown in Table 5, where \( p \) is the original number of input variables, \( \tilde{p} \) is the selected number of variables, \( n \) is the number of rules (using \( \delta = 0.01 \)), Acc. is the accuracy of the model and \( \gamma \) the structure vector that defines the number of variables and also the number of fuzzy sets in the partition of each variable. The structure vector is actually the specification of the model, such that all the remaining parameters of the model can be computed from it, as described above.

Three benchmark datasets were chosen to present the analysis of model interpretation and exploitation: Wine, Heart and Sonar. These datasets present problems with growing complexity in terms of overlapping classes and number of variables. The Wine and the Heart datasets present equal numbers of variables but the latter is much more complex. The Sonar dataset presents an application of the method for a problem with a large number of variables.

The rule table (matrix \( \Xi \)) and the rule confidence weights (matrix \( \Phi \)) for the model obtained with the Wine dataset are shown in Table 6. Each line in the matrix \( \Xi \) corresponds to a rule premise. Each entry is the index of the fuzzy set in the fuzzy partition of the variable assigned in the column that should be included the rule. For instance, the first rule is defined by \( A_{1,4} \land A_{4,1} \land A_{7,3} \land A_{9,5} \land A_{10,3} \land A_{11,3} \land A_{12,4} \land A_{13,4} \), where \( A_{ij} \) stands for the \( j \)th fuzzy set in the fuzzy partition of the variable \( x_i \). Each fuzzy set corresponds to a symbol that can be associated to a linguistic concept in the context of an application. The qualitative concept associated to each symbol depends only on the order among the symbols, such that it can be derived from the index shown in Table 6.
Table 5
Benchmark models

<table>
<thead>
<tr>
<th>Dataset</th>
<th>p</th>
<th>( \hat{p} )</th>
<th>n</th>
<th>Acc.</th>
<th>( \gamma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iris</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>96.67</td>
<td>[3,1,7,5]</td>
</tr>
<tr>
<td>Balance</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>81.92</td>
<td>[5,5,4,5]</td>
</tr>
<tr>
<td>Diabetes</td>
<td>8</td>
<td>6</td>
<td>14</td>
<td>78.13</td>
<td>[3,6,1,2,4,2,4]</td>
</tr>
<tr>
<td>Cancer</td>
<td>9</td>
<td>5</td>
<td>8</td>
<td>97.51</td>
<td>[5,7,1,1,4,2,2,1]</td>
</tr>
<tr>
<td>Glass</td>
<td>9</td>
<td>9</td>
<td>12</td>
<td>75.23</td>
<td>[7,7,8,6,3,8,5,6,8]</td>
</tr>
<tr>
<td>Wine</td>
<td>13</td>
<td>8</td>
<td>10</td>
<td>99.44</td>
<td>[6,1,1,2,1,1,5,1,8,6,5,6,6]</td>
</tr>
<tr>
<td>Heart (Statlog)</td>
<td>13</td>
<td>12</td>
<td>21</td>
<td>87.78</td>
<td>[5,5,6,6,5,1,8,4,2,6,2,8,7]</td>
</tr>
<tr>
<td>Image</td>
<td>18</td>
<td>14</td>
<td>14</td>
<td>91.9</td>
<td>[2,8,1,4,1,1,3,1,6,6,3,6,5,4,4,6,6,7]</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>34</td>
<td>18</td>
<td>26</td>
<td>95.06</td>
<td>[4,1,8,2,5,3,1,5,1,1,1,3,5,1,1,5,6,1,2,1,4,5,3,7,1,1,1,2,1,6,5]</td>
</tr>
<tr>
<td>Sonar</td>
<td>60</td>
<td>31</td>
<td>39</td>
<td>89.42</td>
<td>[5,5,1,5,5,5,1,3,3,1,7,1,2,1,1,2,5,2,3,1,1,2,1,6,3,1,1,3,1,1,1,1,4,2,1,8,1,1,1,2,1,2,5,2,6,1,1,1,1,3,2,1,2,5,3,5]</td>
</tr>
</tbody>
</table>

Table 6
The Wine dataset model

<table>
<thead>
<tr>
<th>( \Xi )</th>
<th>( \Phi )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>( B_1 )</td>
</tr>
<tr>
<td>( A_1 )</td>
<td>4</td>
</tr>
<tr>
<td>( A_2 )</td>
<td>5</td>
</tr>
<tr>
<td>( A_3 )</td>
<td>3</td>
</tr>
<tr>
<td>( A_4 )</td>
<td>3</td>
</tr>
<tr>
<td>( A_5 )</td>
<td>3</td>
</tr>
<tr>
<td>( A_6 )</td>
<td>2</td>
</tr>
<tr>
<td>( A_7 )</td>
<td>3</td>
</tr>
<tr>
<td>( A_8 )</td>
<td>3</td>
</tr>
<tr>
<td>( A_9 )</td>
<td>4</td>
</tr>
<tr>
<td>( A_{10} )</td>
<td>4</td>
</tr>
</tbody>
</table>

It can be seen that, although the structure selection algorithm defined a great number of fuzzy sets for the fuzzy partition of some variables, the rules do not use all the fuzzy sets. For instance, six fuzzy sets were defined for the fuzzy partition of variable \( x_1 \) (cf. Table 5), but only the fuzzy sets numbered from 2 to 5 were effectively used in the rules. This is certainly due to a search for the best location of fuzzy sets prototypes, which cannot be freely adjusted in order to preserve the interpretability, so that the structure selection algorithm increases the number of fuzzy sets in the partition of some variables.

The symbols \( B_1, B_2 \) and \( B_3 \) in matrix \( \Phi \) refers, respectively, to Class 1, Class 2 and Class 3 that, in the Wine classification problem, refers to three different cultivars of wine in the same region of Italy. The domain expert can evaluate the variable values according to the class. For instance, low values of variable \( x_4 \) (Alkalinity) refer to Class 1 (rules \( A_1 \) and \( A_2 \)) or to Class 2 (rules \( A_3 \) and \( A_7 \)) while high values of this variable refer to Class 2 or Class 3 (rules \( \{ A_4, A_5, A_6 \} \) and \( \{ A_8, A_9, A_{10} \} \)).

The rule support and confidence weights of the FSM for the Wine dataset are shown in Fig. 6. The problem is well separated and all the 10 rules are certain. The dataset is also well distributed and all the rules have similar support values. This model has obtained 99.4% of accuracy with respect to the training dataset, as only one record was misclassified.

The relationship among the data complexity, the number of variables and the resulting number of rules generated by the model can be evaluated when the Wine dataset model is compared with the Heart dataset model. Both datasets have the same number of variables, but the latter is a more difficult problem than the former, as it can be observed from the relative differences in the affinity matrix, shown in Fig. 7. Consequently, the number of selected variables and the number rules for the Heart dataset is greater than then number of rules found for the Wine dataset.
The rule support and the confidence weights of the FSM for the Heart dataset are graphically depicted in Fig. 8. In this problem, the rules are not well distributed such that some rules have a very small support. The great overlap between the two classes resulted in a large number of uncertain rules.

The effect of uncertain rules can be observed in Fig. 9, where the computed classes’ membership is plotted against the data records, rearranged in decreasing order of Class 1 membership, showing the location of the misclassified records. A smaller confidence weight results in a smaller class membership value (cf. Eq. (12)), since the same fuzzy membership vector is computed in the rule premise for all classes. Consequently, the FSM classifier provides a smooth transition from one class to another. The classification errors are thus more likely to occur within the overlapping region, which correspond to the uncertain rules, where the computed class membership values are smaller.
The FSM models for the Wine and Heart datasets remain readable, since the number of rules (10 and 21, respectively) is not very large. The FSM model for the Sonar dataset, which resulted in 39 rules relating 31 variables, is more difficult to be completely understood linguistically. Nevertheless, as it can be seen in Fig. 10, the rules can still be interpreted as certain or uncertain by their rule confidence weights. The analysis of rules’ weights allow the separation of the patches fully inside each class, represented by the certain rules, from the patches located on the border between the classes, represented by the uncertain rules. Like the previous example, most of the errors occurred in the region of the uncertain rules, as shown in Fig. 11.

When the model is exploited, the correct class information is not available. However, if there are more certain than uncertain rules, the classifier could work as a filter, by separating the records that activate the certain rules, of which a
better classification can be expected, from the records that activate the uncertain rules. The evaluation of three unlabeled 
records by the FSM is shown in Fig. 12. In the examples shown in Fig. 12(a) and (c) the records activate certain rules 
for Class 1 and Class 2, respectively. In the example shown in Fig. 12(b) the record activates an uncertain rule such 
that the classification provided by the model could not be trusted. The latter case could have a different treatment in 
the context of the application.

The analysis presented by Fig. 12 can be done in batch mode, so that the set of records that activate uncertain rules 
are not predicted. This is a great advantage of the FSM classifiers in real application that cannot be achieved by most 
of other types of pattern recognition models.
6. Conclusions

This work presented a fuzzy classifier based on a symbolic formulation of fuzzy systems, in which the fuzzification is seen as a nonlinear mapping from the input variables space to a feature space. Each dimension of the feature space is assigned to a rule (i.e. a symbol), representing a conjunction of fuzzy sets defined on the domain of each variable. The feature space can be seen as a symbolic space and the model is thus called the fuzzy symbolic model (FSM). The FSM inference is a linear transformation from input to output symbolic spaces, which is computed by the rule confidence weights for each output symbol (class). The dimension of the feature space is defined from eigendecomposition of the Gaussian affinity matrix computed from data. The rule confidence weights are optimized through a bounded quadratic problem and are very useful for model interpretation and exploitation.

The design of FSM is focused on the interpretability of the fuzzy rules, such that interpretability constrains the search for accuracy. Interpretability is closely related to the number of variables as so as the number and position of fuzzy sets in the domain of each variable. The fuzzy membership functions were arbitrarily chosen to be equally spaced within each input variable domain, in order to enhance interpretability. Nevertheless, any other choice for membership functions positioning could be used in the rule induction algorithm.
The model structure is optimized by a genetic algorithm wrapper, which simultaneously defines the variables that should be included into the model and the number of fuzzy sets in the fuzzy partition of each variable. The simplest genetic algorithm, without any specialized operators or coding, was adopted in this work, such that a prototype can be implemented using any standard library.

The main contribution of this work is on the utilization of recent results of spectral methods in determining the number of interpretable rules in a fuzzy system. This work also proposed the optimization of rule base confidence weights, which can be used to interpret rules as certain or uncertain. The user can thus understand the model and recognize which rules are certainly associated to each class and which rules are related to the boundary region among classes.

The experiments show that the FSM has good accuracy results when compared to similar models presented in the literature. The proposed rule induction algorithm based on spectral analysis improved the model accuracy when compared with the Wang and Mendel algorithm, by keeping all the other model characteristics unchanged and tested in the same experimental conditions. It was shown that the clustering approach used in the rule induction algorithm has a considerable influence in the accuracy result of the classifier. This result suggests that the clustering approach could be coded into the chromosome and optimized within the outer loop of the genetic algorithm wrapper. Moreover other clustering approaches could be investigated and evaluated.

The model interpretation and exploitation analysis showed that most errors occur in the region represented by uncertain rules, as was expected. In a given application, the classifier could work as a filter separating the records that activate the certain rules from the records that activate the uncertain rules. The latter could be considered as uncertain and may be treated differently in the context of the application.

The main drawback of the proposed approach is the great computational cost associated to the genetic algorithm wrapper and the numerical computation of eigenvalues. Nevertheless, with the recent advances in hardware technology and the growing availability of multi-core processor-based workstations, processing time can be substantially reduced, since the genetic algorithm and the numeric computations can be processed in parallel. Moreover, computing only the largest eigenvalues of the Laplacian using iterative subspace decomposition algorithms can also reduce the processing time, such that the limit for the problem size depends only on the available hardware.

The future directions of this research should address specifically the computational cost of the structure selection. One possible direction is to adopt a faster implementation of the proposed method in order to evaluate the robustness and the stability of the structure selection algorithm as well as the use of more sophisticated genetic operators and also other evolutionary approaches. Another possible extension is to replace the genetic algorithm wrapper by a greedy incremental approach. This solution would also allow the induction of a rule base with a different number of variables in each rule, such that only the necessary input variables would be considered for each rule, what could improve interpretability and also accuracy.

The extension of the proposed method for function approximation is straightforward. The eigenstructure analysis of the affinity matrix can also be used to derive pattern recognition models directly from the feature space. This kind of model could also be used for classification problems and would allow evaluating the accuracy loss due to interpretability constraints. These models are currently under investigation and will be considered in a forthcoming publication.

Acknowledgments

The authors are grateful to the Brazilian Research Agencies, CAPES, CNPq and FINEP for the financial support for this research.

References