Knowl Inf Syst (2011) 29:203–221
DOI 10.1007/s10115-010-0340-x

SHORT PAPER

A general framework for designing a fuzzy rule-based classifier

Antanas Verikas · Jonas Guzaitis · Adas Gelzinis · Marija Bacauskiene

Received: 14 July 2009 / Revised: 28 December 2009 / Accepted: 31 August 2010 /
Published online: 16 September 2010

Abstract This paper presents a general framework for designing a fuzzy rule-based classifier. Structure and parameters of the classifier are evolved through a two-stage genetic search. To reduce the search space, the classifier structure is constrained by a tree created using the evolving SOM tree algorithm. Salient input variables are specific for each fuzzy rule and are found during the genetic search process. It is shown through computer simulations of four real world problems that a large number of rules and input variables can be eliminated from the model without deteriorating the classification accuracy. By contrast, the classification accuracy of unseen data is increased due to the elimination.

Keywords Classifier · Fuzzy rule · Genetic algorithm · Knowledge extraction · Variable selection · Evolving SOM tree

1 Introduction

Support vector machines (SVM) [1], neural networks [2], and relevance vector machines (RVM) [3] are probably the most popular data classification techniques. An SVM and RVM can provide near optimal performance. The advantages of SVM and RVM are the following:

A. Verikas (✉)
Intelligent Systems Laboratory, Halmstad University, Box 823, 30118 Halmstad, Sweden
e-mail: antanas.verikas@hh.se

A. Verikas · J. Guzaitis · A. Gelzinis · M. Bacauskiene
Department of Electrical & Control Equipment, Kaunas University of Technology,
Studentu 50, 51368 Kaunas, Lithuania

J. Guzaitis
e-mail: jonas.guzaitis@stud.ktu.lt

A. Gelzinis
e-mail: adas.gelzinis@ktu.lt

M. Bacauskiene
e-mail: marija.bacauskiene@ktu.lt

the ability to find the global minimum of the objective function, no assumptions made about the data, the complexity of a classifier depends on the number of support vectors, but not on the dimensionality of the input space. However, in spite of the attempts to explain decisions of such techniques [4,5], classifiers based on these techniques are not transparent enough and are often considered as "black boxes". The transparency is very important in some application areas, such as medical decision support or quality control.

By contrast, fuzzy rule-based systems and fuzzy decision trees [6,7] are known for their transparency and ability of accounting for uncertainty. Fuzzy rule-based classification methods can support rapid incremental learning from new instances without performance degradation on previous training data. ANFIS [8], fuzzy ARTMAP [9], and the fuzzy min–max classifier [10,11] are examples of the most prominent fuzzy logic-based systems. Fuzzy rule-based systems have been used in a variety of fields such as pattern recognition [12], image segmentation [13], data mining [14], process control [15], resource service selection [16], and system identification [8,17]. It is well known that designing of fuzzy rule-based systems in high-dimensional spaces is rather problematic. However, there are many problems characterized by a small or moderate number of variables. Moreover, quite often high-dimensional data vary in a much lower number of dimensions if compared to the dimensionality of the input space. System structure identification and parameter optimization are two main issues to consider when designing a fuzzy rule-based system [8,18]. Fuzzy partitioning, variable selection, and fuzzy reasoning are the tasks to be solved for identifying the system structure. Parameter optimization usually deals with tuning of parameters of fuzzy membership functions.

Various approaches have been used for dealing with the two main fuzzy rule-based system design issues. The initial system structure, often termed as fuzzy partitioning, is usually identified through K-Means [19], Fuzzy C-Means [20], Learning Vector Quantization (LVQ) [21] or SOM-based clustering [22–28] as well as incremental clustering [29,30] or by constructing a decision tree [31–33].

Variable selection techniques based on the output sensitivity to the input change [25,34], the output sensitivity combined with the correlation between variables [24], Fisher's interclass separability measure [35], variable correlation with the output [36] are the most popular and are applied. However, quite often, variable selection is not considered at all [23,26,27].

It seems that the simple gradient decent [23–25,27,37,38], error correction [39], and genetic search [26,31,35] are the most popular parameter optimization techniques utilized in various studies. The combined optimization of both structure and parameters has also been considered by applying genetic algorithms (GA) [40–44], unsupervised and reinforcement learning [45], or simple heuristics [36]. To reduce the evolution time in the GA-based technique, Chen et al. [46] proposed gathering similar chromosomes into k clusters and then using a representative chromosome in the evaluation process. In [42,43] the genetic search process focusses on "hard" data points by assigning a higher weight to such points. Such an approach has also been adopted for learning weights z_j^q of fuzzy rules [47]. In [48,49], genetic search-based multi-objective optimization was applied to design a fuzzy rule-based system. The task was to maximize $f_1(S)$, minimize $f_2(S)$, and minimize $f_3(S)$, where S is a set of fuzzy rules, $f_1(S)$ stands for correctly classified training samples, $f_2(S)$ is the number of fuzzy rules in S, and $f_3(S)$ is the total number of antecedent conditions in S. Thus, $f_3(S)$ can be considered as the total rule length. The optimization starts with all possible rules in the search space defined by the training patterns. In [50], it was proposed to combine several fuzzy rule-based classification systems into a committee through voting or weighted voting.

Generalization ability is an important issue to consider when designing a fuzzy rule-based classifier. The most popular technique applied to improve the generalization ability is rule

pruning based on similarity of fuzzy sets [24,31,35,51]. Other approaches utilized are the following: GA [28], simulated annealing [52], similarity of fuzzy sets combined with GA [26], through forgetting by decaying the grade of certainty of fuzzy rules [39], and pruning of rarely used rules [36].

1.1 Fuzzy rule-based and nearest neighbor techniques

The fuzzy rule-based classification techniques are closely related to nearest neighbor (NN)-based classification approaches. NN-based classification has a sound basis, since there is a considerable body of evidence from the literature that classification and recognition of patterns by humans are best explained as a form of interpolation between similar patterns [53]. NN methods are frequently criticized as requiring much greater use of memory than, for example, neural network algorithms. However, NN learning algorithms can reduce their memory usage by only retaining the full density of exemplars near to classification boundaries and thinning them in other regions [54–57]. As discussed above, various approaches to designing a fuzzy rule-based classifier exploit such techniques for determining an initial system structure through clustering. The location of fuzzy sets, reference patterns in the NN approach, can be further optimized by applying the LVQ techniques.

LVQ has been widely used to learn reference patterns for classification based on the NN approach. Each class C_j is described by several reference patterns $\mathbf{m}_j^l$ (fuzzy sets in the rule-based approach), which are properly placed within each class region. An unknown $\mathbf{x}$ is then determined to belong to the class k, if:

$$k = \arg\min_j [\min_l d(\mathbf{x}, \mathbf{m}_j^l), \quad l = 1, \ldots, N_j, \quad j = 1, \ldots, Q] \tag{1}$$

where Q is the number of classes, N_j is the number of reference patterns representing the class j and $d(\mathbf{x}, \mathbf{m}_j^l)$ is the distance between $\mathbf{x}$ and $\mathbf{m}_j^l$.

One more drawback of classical NN and fuzzy rule-based methods is the often exhibited poor generalization performance when compared to neural networks, for example. Neural networks suffer from the "curse of dimensionality" to significantly less extent, since they are able to select useful input features from high-dimensional input vectors. In NN methods, by contrast, the degradation in performance often accompanies the addition of new unimportant features. However, there are many problems where different features are important in different regions of the input space. Figure 1 provides an example illustrating such a situation.

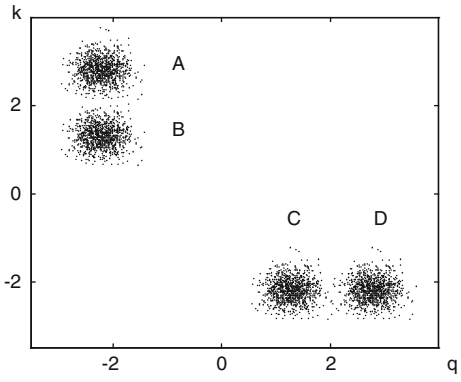

Fig. 1 Four decision classes in the two-dimensional space

The four data clusters illustrated in Fig. 1 represent four decision classes. It is obvious that the feature q is unimportant for discriminating the classes A and B, likewise the feature k is unimportant for discriminating the classes C and D. Thus, a subset of features used should be reference pattern or fuzzy rule dependent. However, in most of the known fuzzy rule-based classification algorithms, the feature selection problem is considered independently of the input space region or not considered at all. The objective of this work is to develop a fuzzy modeling framework capable of automatically generating a rule base for classification of numeric data, finding the optimal number of rules and input variables for each rule, and finding the optimal parameter values of fuzzy rules. The remainder of the paper is organized as follows. In the next section, the fuzzy model is described. The approach proposed is outlined in Sect. 3. Section 4 discusses the results of the experimental investigations. Section 5 presents the conclusions of the work.

2 The fuzzy model

We use the Mamdani model [58], which is the most popular fuzzy model applied in various studies for fuzzy reasoning [23–26,59]. Concerning the classification, the model is a collection of fuzzy rules R_j of the following form:

$$R_j : \texttt{IF}\ x_1\ \texttt{is}\ A_{j1}\ \texttt{AND}\ \ldots\ \texttt{AND}\ x_n\ \texttt{is}\ A_{jn}\ \texttt{THEN class}\ C_q\ \texttt{with}\ z_j^q \tag{2}$$

where $A_{ji} (i = 1, \ldots, n)$ are fuzzy sets defined over the input variables x_i; C_q is a class label and z_j^q is a rule weight. Each fuzzy set is represented by a membership function. A triangular or a Gaussian function of the form

$$\mu_{ji} = \exp\left(-\frac{(x_i - c_{ji})^2}{\sigma_{ji}^2}\right) \tag{3}$$

where c_{ji} and σ_{ji} are the center and the width of the Gaussian function, respectively, are common choices. We use Gaussian membership functions in this study. There are various ways to determine the rule weights z_j^q. In this work, the weight z_j^q is given by:

$$z_j^q = \max\left(\frac{\sum_{\mathbf{x}_p \in C_q} \mu_{\mathbf{A}_j}(\mathbf{x}_p) - \sum_{\mathbf{x}_p \notin C_q} \mu_{\mathbf{A}_j}(\mathbf{x}_p)}{\sum_{p=1}^{N} \mu_{\mathbf{A}_j}(\mathbf{x}_p)}, 0\right) \tag{4}$$

where N is the number of training patterns and the matching degree of the input pattern $\mathbf{x}_p$ with the antecedent part $\mathbf{A}_j = (A_{j1}, \ldots, A_{jn})$ is calculated using a T-norm

$$\mu_{\mathbf{A}_j}(\mathbf{x}_p) = T(\mu_{A_{j1}}(x_{p1}), \ldots, \mu_{A_{jn}}(x_{pn})) \tag{5}$$

We use the min T-norm operator in this work. Weights z_j^q of this type were studied in [48]. In [60], weights based on ROC analysis are advocated.

A winning rule is used to make a decision. Thus, given a rule base S consisting of L rules, an input pattern $\mathbf{x}_p$ is assigned to the class q if

$$q = \arg\max_k \{T[\mu_{\mathbf{A}_j}(\mathbf{x}_p), z_j^k],\ \ j = 1, \ldots, L\} \tag{6}$$

where T is the *product* T-norm operator, in this work.

Having defined the membership functions, we formulate the fuzzy modeling problem in the following way. Given N pairs of input-output patterns $(\mathbf{x}, y)$, create a minimal number of fuzzy rules r with the optimal number of features n_i for each rule and find the optimal values of parameters $(\mathbf{c}, \boldsymbol{\sigma}, \mathbf{z})$ of the fuzzy model $F(\mathbf{c}, \boldsymbol{\sigma}, \mathbf{z}, r, n_i, i = 1, \ldots, r)$.

3 The approach

The procedure to construct the fuzzy rule-based classifier consists of the following steps.

1. Divide the data set into learning and test subsets.
2. Cluster the learning set data by applying the evolving SOM tree.
3. Based on the evolved tree, generate a population of sub-trees. Each sub-tree defines the initial structure of one fuzzy rule-based classifier. The generation is accomplished by randomly cutting branches of the tree grown in Step 2. The cutting occurs approximately between 25 and 75% of the tree depth.
4. Represent each node in the sub-tree population by a set of fuzzy sets with the Gaussian membership functions.
5. Take one sub-tree (classifier of a given structure) from the sub-tree population and encode the structure, features (used/not used), and parameters of the membership functions of the classifier into a chromosome. When encoding, enable feature selection independently for each fuzzy rule.
6. Generate a population of chromosomes encoding individual classifiers of the given structure. The individual classifiers differ in features and values of the parameters.
7. Apply the modified LVQ-3 algorithm to the individuals of the population.
8. Evaluate the fitness of the individuals.
9. Apply genetic operations (to features and parameters) and generate a new population.
10. Repeat Steps 7–9 until convergence.
11. Take the best individual of the given structure.
12. Repeat Steps 5–11 for the whole population of sub-trees.
13. Apply genetic operations (to structure of sub-trees) and generate a new population of sub-trees.
14. Repeat Steps 5–13 for a given number of generations.

Next, we briefly describe the main topics of the technique.

3.1 The algorithm

```
1.  create evolving SOM tree
2.  generate and encode a population of sub-trees tp
3.  for (given number of generations): structure evolution
4.      for (each chromosome tc in tp): parameter evolution
5.          generate a population pp encoding individual classifiers of the tc structure
6.          repeat
7.              for (each chromosome pc in pp)
8.                  apply the modified LVQ-3 algorithm to pc
9.                  evaluate the fitness of pc
10.             end for
11.             apply genetic operations to pp
12.         until (convergence)
13.         take the best individual of pp for tc in tp
14.     end for
15.     apply genetic operations to tp
16. end for
```

3.2 The evolving SOM tree

Like SOM, the evolving SOM tree [61] exhibits the self-organization property. The evolving tree structure enables the SOM tree to efficiently handle large-scale problems. Moreover, there is no need of choosing the map size beforehand. Like in ordinary SOM, each node of the SOM tree has a weight vector $\mathbf{w}_i$. When training the tree, for each training vector $\mathbf{x}$, the best matching unit (BMU) is found by a greedy tree search. BMU is always a leaf node. Weight vectors of the BMU and its neighbors are then updated using the SOM adaptation rule:

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) - h_{ci}(t)[\mathbf{x}(t) - \mathbf{w}_i(t)] \tag{7}$$

where $h_{ci}(t)$ is the neighborhood function. We used the Gaussian neighborhood function

$$h_{ci}(t) = \beta(t) \exp\left(\frac{\|\mathbf{r}_c - \mathbf{r}_i\|^2}{2s^2(t)}\right) \tag{8}$$

where $s(t)$ is the width of the Gaussian function, $\mathbf{r}_c$ and $\mathbf{r}_i$ denote location of nodes c and i, and $\beta(t)$ is the learning rate. The meaning of $s(t)$ and $\beta(t)$ is the same as in SOM [62], while the meaning of the norm $\|\mathbf{r}_c - \mathbf{r}_i\|$ is quite different. The basic idea of calculating the distance is to count how many "hops" are needed to get from the BMU to the considered node along the shortest path [61]. The distance $\|\mathbf{r}_c - \mathbf{r}_i\|$ is then given by the number of hops minus one.

Figure 2 presents an example of the evolving SOM tree generated to represent the two-dimensional data, where circles denote the center points of the SOM nodes. The deeper is a node in the tree, the smaller circle and the thinner line are used in the visualization. The same applies for the links. The leave nodes are denoted by triangles. Each parent node is split into three child nodes, in the example. However, other number of child nodes can be used. Each node is characterized by a center point and width computed from the data mapped

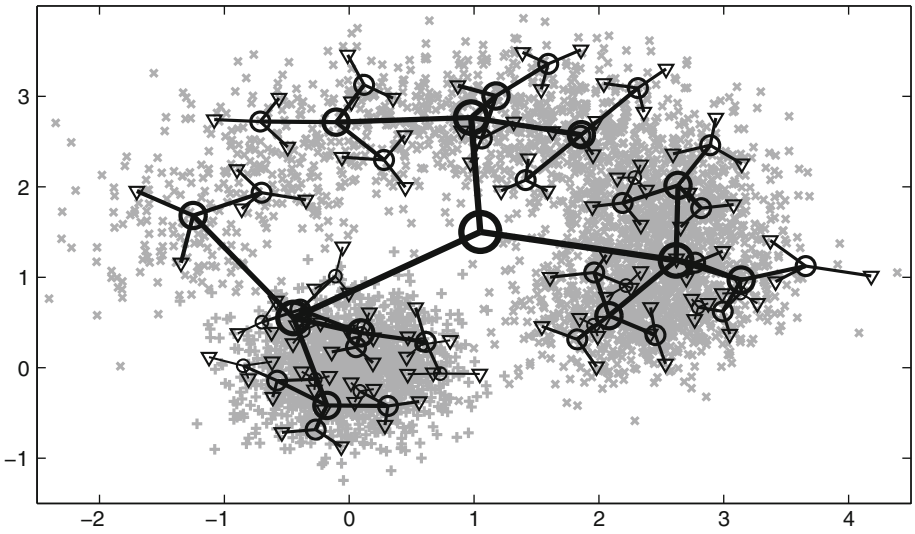

Fig. 2 An example of the evolving SOM tree generated to represent the two-dimensional data

onto the node. These parameters are used as initial values for the parameters of the Gaussian membership functions.

3.3 The modified LVQ-3 algorithm

Assume that d_i and d_j are the Euclidean distances from the pattern $\mathbf{x}$ to the reference patterns $\mathbf{m}_i$ and $\mathbf{m}_j$, respectively. Then, $\mathbf{x}$ is defined to fall into a window of the relative width λ, if

$$\min\left(\frac{d_i}{d_j}, \frac{d_j}{d_i}\right) > \frac{1-\lambda}{1+\lambda} \tag{9}$$

For all $\mathbf{x}$ falling into the window adapt:

$$\mathbf{m}_i(t+1) = \mathbf{m}_i(t) - \alpha(t)[\mathbf{x}(t) - \mathbf{m}_i(t)] \tag{10}$$

$$\mathbf{m}_j(t+1) = \mathbf{m}_j(t) + \alpha(t)[\mathbf{x}(t) - \mathbf{m}_j(t)] \tag{11}$$

where $\alpha(t)$ decreases with time and $0 < \alpha(t) < 1$, $\mathbf{m}_i$ and $\mathbf{m}_j$ are two closest reference patterns to $\mathbf{x}$, whereby $\mathbf{x}$ belongs to the same class as $\mathbf{m}_j$, but not as $\mathbf{m}_i$. If $\mathbf{x}$, $\mathbf{m}_i$ and $\mathbf{m}_j$ belong to the same class:

$$\mathbf{m}_k(t+1) = \mathbf{m}_k(t) + \gamma\alpha(t)[\mathbf{x}(t) - \mathbf{m}_k(t)] \tag{12}$$

for $k \in \{i, j\}$. If $\mathbf{x}$ belongs to a different class than $\mathbf{m}_i$ and $\mathbf{m}_j$:

$$\mathbf{m}_k(t+1) = \mathbf{m}_k(t) - \gamma\alpha(t)[\mathbf{x}(t) - \mathbf{m}_k(t)] \tag{13}$$

for $k \in \{i, j\}$. The optimal value of the parameter γ depends on the size of the window, the value is smaller for narrower windows [21]. Values between 0.1 and 0.5 are suggested for γ [21].

The algorithm performs fine tuning of the centers of membership functions and helps reducing the time of genetic search. The last adaptation step is not used in the original version of the LVQ-3 algorithm. We have found that the use of the step quite noticeably improved the accuracy of the algorithm.

3.4 Encoding

Structure, features, and parameters of the membership functions are to be encoded. The structure is determined by a sub-tree and is encoded as a connected graph. Figure 3 presents an example of the evolving SOM tree along with two chromosomes encoding two hypothetical sub-trees. The hypothetical leaf nodes of the two sub-trees are shown connected by the bold solid and the dashed line, respectively, in Fig. 3. There are as many *sections* in the chromosome, as there are leaf nodes in the corresponding sub-tree (the number of fuzzy rules in the classifier).

Each chromosome section consists of three *sub-sections*: $\mathbf{f}$, $\mathbf{c}$, and $\mathbf{s}$, encoding features ("feature mask"), centers and widths of the membership functions μ, respectively. Figure 4 illustrates sub-sections of the jth node. There are n bits in the $\mathbf{f}_j$ sub-section, where n is the dimensionality of the input space. A bit in the $\mathbf{f}_j$ sub-section set to 0/1 means that the corresponding feature is *used/not used* in the rule encoded in the corresponding section of the chromosome. Centers and widths of the membership functions μ are stored as real/integer numbers in slots of the subsections $\mathbf{c}_j$ and $\mathbf{s}_j$. For example, integer numbers can be used to

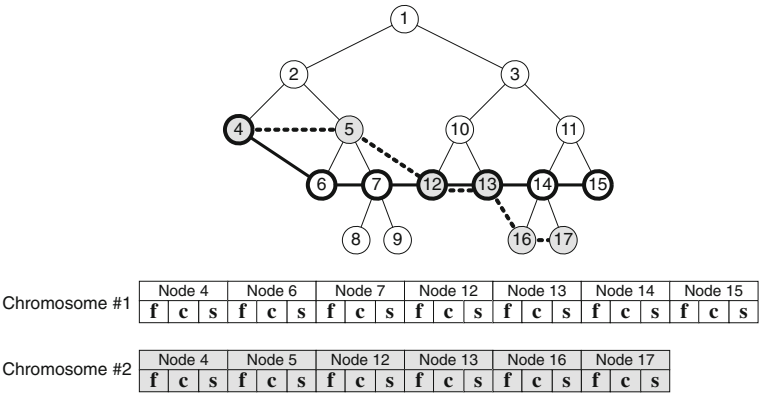

Fig. 3 An example of the evolving SOM tree along with two chromosomes encoding two hypothetical sub-trees, the leaf nodes of which are connected by the *bold solid* and *dashed lines*

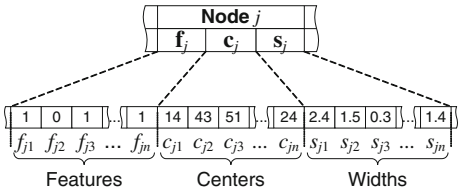

Fig. 4 Three sub-sections ($\mathbf{f}_j$, $\mathbf{c}_j$, and $\mathbf{s}_j$) of the jth node encoding features, centers and width of the membership functions, respectively

encode centers of the membership functions in the applications related to image analysis. There are n slots in each, $\mathbf{c}_j$ and $\mathbf{s}_j$, sub-section.

3.5 Genetic operations

Crossover and mutation are the genetic operations applied in both loops of genetic evolution: the loop concerning structure evolution and the loop concerning features and parameters of the membership functions. The crossover and mutation operations are executed with the probability of crossover p_c and the mutation probability p_m, respectively.

3.5.1 Mutation of chromosomes encoding structure

Mutation in structure evolution amounts to taking one step up or down (the direction is selected randomly) along a randomly selected branch of the tree. To select the direction, an integer ϑ is selected randomly from the set $\{-1, 1\}$ for each node. The node undergoing the mutation is replaced with its parent node, if $\vartheta < 0$ or is split into children, if $\vartheta > 0$. Figure 5 illustrates the mutation operation in structure evolution.

Nodes being active before the mutation operation are shown in gray and connected by the dashed line, in Fig. 5. A node is said to be active if it is used to define the classifier structure at a current moment. Nodes being active after the mutation operation are shown connected by the bold line, in Fig. 5. Arrows in Fig. 5 indicate the direction of mutation. As can be seen, nodes *7, 10*, and *14* mutate toward their children, node *15* mutates toward its parent, while nodes *4* and *6* do not mutate.

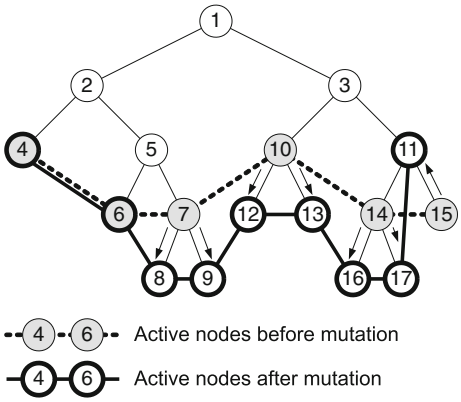

Fig. 5 A part of the evolving SOM tree illustrating the mutation operation in structure evolution

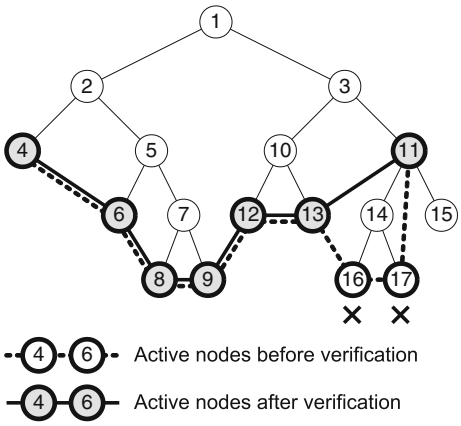

Fig. 6 A part of the evolving SOM tree illustrating the verification operation in structure evolution

3.5.2 Verifying the structure consistency

After the genetic operations, the sub-trees are verified for consistency. The redundant active nodes are deactivated during the verification. Figure 6 illustrates the verification operation. Nodes being active before the verification operation are shown in gray and connected by the dashed line, in Fig. 6. Nodes being active after the verification operation are shown connected by the bold line. As can be seen from Fig. 6, nodes *4, 6, 8, 9, 12*, and *13* are left unaffected, while nodes *16* and *17* are deactivated, since node *11*, predecessor of nodes *16* and *17*, was activated during the mutation operation. The deactivated nodes are labeled by $\times$, in Fig. 6.

3.5.3 Crossover of chromosomes encoding structure

When performing crossover for structure evolution, parts of two sub-trees are exchanged. The crossover point is selected randomly. Figure 7 illustrates two sub-trees (*Parent #1* and *Parent #2*) before the crossover operation with the crossover points indicated by the dashed lines. The active nodes of the trees encoded in the corresponding chromosomes are shown connected by the bold lines. The resulting sub-trees after the crossover operation are labeled as *Unverified offspring #1* and *Unverified offspring #2*, in Fig. 7. The verification procedure is

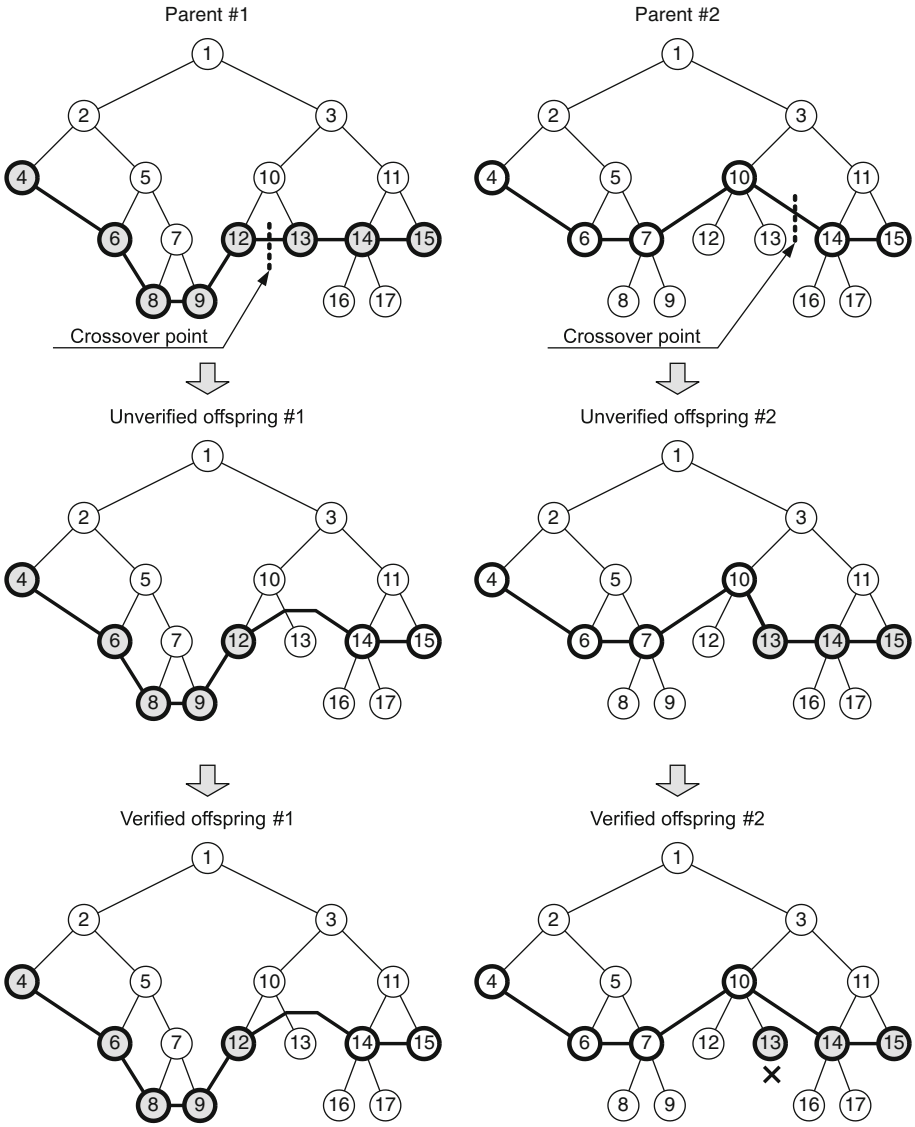

Fig. 7 A part of the evolving SOM tree illustrating the crossover operation in structure evolution

then applied to the unverified offsprings. The verification results into the verified offsprings with one deactivated node, namely node *13* of the offspring #2.

3.5.4 Crossover and mutation of chromosomes encoding features and parameters

Crossover of two chromosomes encoding features and parameters of the membership functions are performed at a randomly selected point by exchanging parts of the chromosomes, as it is illustrated in Fig. 8. Two chromosomes selected for the crossover operation are to encode the same number of nodes.

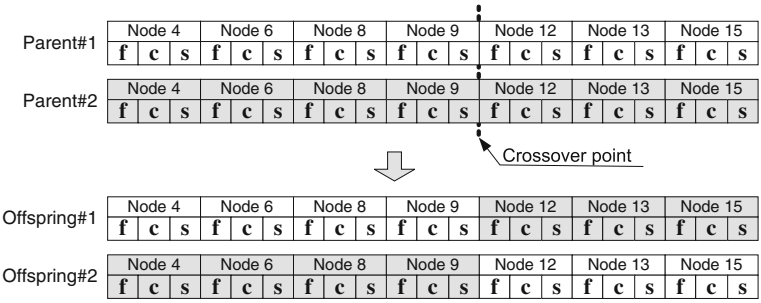

Fig. 8 Parameters of two parent chromosomes are exchanged at the crossover point

The mutation operation is accomplished by reversing the value of a bit in the "feature mask" (**f** sub-section) and by adding a random value from a given, symmetric around zero, interval to parameters stored in the slot (gene) selected for mutation in the **c** and **s** sub-sections. Selection of genes for mutation is performed independently in the three sub-sections.

3.6 Fitness function

The fitness value of the ith chromosome f_i is given by

$$f_i = \chi_i - \eta \frac{\sum_{j=1}^{r_i} n_j}{r_0 \times n} \tag{14}$$

where χ_i is the classification accuracy obtained from the classifier encoded in the ith chromosome, η is a parameter determining the degree to which classifiers with a large number of features are penalized, n is the total number of available features, n_j is the number of features used by the jth rule, r_0 is the number of rules in the initial tree, and r_i stands for the number of rules used by the ith classifier. A value of $\eta = 0.1$ worked well in all the tests.

Chromosomes are selected for genetic operations with some probability. The selection probability of the ith chromosome p_i is given by

$$p_i = \frac{f_i}{\sum_{j=1}^{M} f_j} \tag{15}$$

where M is the population size. The roulette selection principle was applied.

4 Experimental investigations

4.1 Data used

Four data sets have been used in the tests.

US congressional voting records problem. The United States Congressional voting records data set consists of the voting records of 435 congressman on 16 major issues in the 98th Congress. The votes are categorized into one of the three types of votes: (1) (*Yea*), (2) (*Nay*), and (3) (*Unknown*). The task is to predict the correct political party affiliation of each congressman. The 98th Congress consisted of 267 Democrats and 168 Republicans.

Wisconsin diagnostic breast cancer problem. There are 30 real-valued features. The features are computed from a digitized image of a fine needle aspirate of a breast mass and

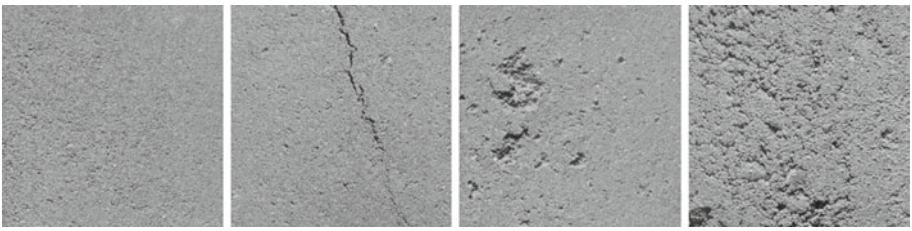

Fig. 9 Examples of pavement tile surfaces: a quality surface on the left and three defective surfaces

describe characteristics of the cell nuclei present in the image. There are 569 instances, 357 benign and 212 malignant.

The diabetes diagnosis problem. *The Pima Indians Diabetes* data set contains 768 samples taken from patients who may show signs of diabetes. Each sample is described by eight features: (1) *Number of times pregnant*, (2) *Plasma glucose concentration*, (3) *Diastolic blood pressure*, (4) *Triceps skin fold thickness* , (5) *Two-hour serum insulin*, (6) *Body mass index*, (7) *Diabetes pedigree function*, and (8) *Age*. There are 500 samples from patients who do not have diabetes and 268 samples from patients who are known to have diabetes. These three data sets are available at: http://archive.ics.uci.edu/ml/.

Pavement tiles surface inspection problem. A pavement tile surface is to be assigned into a *quality* or *defective* class. Features for the classification are extracted from a camera image. Five features characterizing the image texture and the gray level distribution [63] have been used to design a classifier. Figure 9 presents four examples of pavement tile surfaces used in the study. In total, 200 quality and 200 defective surfaces were available.

4.2 Experimental setup

We randomly assign the available data points into the learning D_L and test D_T data sets. The data are normalized to have zero mean and unit variance. We run an experiment 30 times with different random splits into the sets D_L and D_T. The results obtained are averaged over the 30 runs.

4.3 Optimization parameters

The optimal size of the LVQ-3 window depends on the number of training samples. If a large number of samples is available, a narrow window would guarantee the most accurate location of the decision boundary. For good statistical accuracy, however, the number of samples falling into the widow must be sufficient [21]. The optimal value of γ depends on the size of the window, being smaller for narrower windows [21,64]. After some experiments, the following values of the LVQ-3 parameters have been used: $\lambda = 0.05$, $\alpha = 0.02$, and $\gamma = 0.4$.

There are two loops of genetic evolution: the outer loop concerning structure evolution and the inner loop concerning features and parameters of the membership functions. The genetic search lasted for 100 generations (for both loops) with the following parameters: the population size was set to 50 and the number of offsprings produced for creating the next population was equal to 50. The number of generations was determined experimentally by monitoring changes of the fitness function value. The number chosen was such that no fitness function value increase was observed in the last 10 generations. The values of crossover and

mutation probabilities were found experimentally. The following values worked well in the tests: $p_c = 0.95$ and $p_m = 0.01$. The appropriate value of the parameter η was found to be $\eta = 0.05$.

4.4 Results

In the first set of experiments, the feature selection has not been applied and the classification accuracy obtained from the fuzzy rule-based classifier was compared with the accuracy achieved by other techniques. The multilayer perceptron (MLP), k-NN, and LVQ-3 classifiers have been used for the comparison. The appropriate number of hidden nodes in the MLP and the k value of the k-NN classifier were found experimentally. The leaf nodes created by the evolving SOM tree were used as initial reference patterns for the LVQ-3 classifier. Table 1 presents the average test data set classification accuracy (%) obtained from the different classifiers using all available features. As can be seen from Table 1, the proposed fuzzy rule-based classifier provided the highest classification accuracy for all the problems studied.

In the next set of experiments, feature selection was activated and features specific for each rule were found through the genetic search. Table 2 presents the average test data set classification accuracy obtained from the approach proposed using the selected features. The classification accuracy obtained using all the available features is also presented for the sake of comparison. The obtained improvement in classification accuracy should be obvious from Table 2. Assuming that the classification errors are log-normally distributed and applying the t-test it was found that the difference between the classification accuracy obtained using the selected and all features is significant with 95% confidence, except for the Breast cancer data. In Table 2, we also provide the average test data set classification accuracy obtained by other authors for the same public data sets in recent studies. In [65–67], evolutionary techniques have also been used to design the classifiers. As can be seen from Table 2, the proposed technique outperformed the other approaches on all the three data sets.

Table 3 presents information on the number of rules and features used to classify the data. In the parentheses given are the number of initial rules and the number of available features. Ranges in the "Features" column indicate the minimum and the maximum number of features used by different rules. As can be seen from Table 3, the number of features used by different

Table 1 The average test data set classification accuracy (%) obtained from different classifiers using all available features

Data set\classifier	k-NN	MLP	LVQ-3	Proposed
Voting	91.24	93.78	77.41	94.68
Breast cancer	88.73	97.18	75.87	98.54
Diabetes	71.19	71.49	64.22	71.92
Surface inspection	77.30	81.63	78.13	84.13

Table 2 The average test data set classification accuracy (%) obtained from the approach proposed using all and selected features, along with the average accuracy obtained by other authors in recent studies

Data set\features	All features	Selected features	Comparison
Voting	94.68	98.68	96.98 [65], 95.31 [66]
Breast cancer	98.54	99.02	95.55 [67], 94.38 [47]
Diabetes	71.92	75.92	75.19 [67], 75.00 [60]
Surface inspection	84.13	99.63	–

rules varies significantly. Observe that even if the number of features used by two different rules is the same, the features used are often different. Thus, features used are rule specific, indeed.

Below given is an example of a fuzzy rule (one out of ten) generated by the proposed technique for classification of the Diabetes set data.

R_1 : IF x_2 is ABOUT 119.0 AND x_4 is ABOUT 38.1 THEN class *Healthy* with 0.84

where "ABOUT" is a fuzzy variable, 119.0 and 38.1 are the center points of the Gaussian membership functions, x_2 stands for "plasma glucose concentration", and x_4 means "triceps skin fold thickness". Knowing meaning of the variables $x_1, \ldots, x_8$, the rules are easy to interpret for a medical doctor.

One may wonder what classification accuracy would be achieved, if an ordinary GA-based feature selection procedure—not specific for each fuzzy rule—were used. Table 4 presents the results obtained from such an experiment. Comparing the results presented in Tables 2, 3 and 4 we can see that the fuzzy rule-specific feature selection procedure results into the decreased average number of rules and features and the increased classification accuracy. While the difference between the average number of rules and features utilized by the two techniques is rather small, the difference between the classification accuracies is statistically significant with 95% confidence for all the data sets.

Next, the influence of crossover and mutation probabilities, p_c and p_m, on classification accuracy was studied. The same p_c and p_m values were used for both structure and parameter evolution. To reduce the computation time and to decouple the influence of p_c and/or p_m, and feature selection on the classification accuracy, the studies were performed without employing feature selection. A similar performance was observed for p_m values raging from 0.005 to 0.05. A value of $p_m = 0.01$ was selected. When studying the influence of p_c, p_m was set to $p_m = 0$. Table 5 presents the average test set classification accuracy obtained for different p_c values. The interval of $p_c = [0.8, 1.0]$ was studied additionally and was found that the highest and similar classification accuracy is obtained for $p_c = 0.9$ and $p_c = 0.95$. Thus, p_c values close to unity are recommended.

One and the same loop can be used for both structure and parameter evolution of the classifier, presumably at the expense of computation time. An experiment was performed to compare these two implementations using the *Diabetes* data set. A very similar test data set

Table 3 The number of rules and features used to classify data from the different data sets

Data set	# Rules	# Features
Voting	15 (25)	05–09 (16)
Breast cancer	10 (14)	12–20 (30)
Diabetes	10 (16)	02–08 (8)
Surface inspection	09 (13)	03–04 (5)

Table 4 The average number of rules and features used to classify data from the different data sets and the average test data set classification accuracy (%) obtained using an ordinary GA-based feature selection procedure

Data set	# Rules	# Features	Accuracy
Voting	16 (25)	08 (16)	95.12
Breast cancer	10 (14)	17 (30)	96.91
Diabetes	11 (16)	06 (8)	73.23
Surface inspection	10 (13)	04 (5)	98.11

Table 5 The average test set classification accuracy (%) obtained for different p_c values

Data set	$p_c = 0.25$	$p_c = 0.50$	$p_c = 0.75$	$p_c = 1.0$
Voting	92.89	93.01	93.21	94.68
Breast cancer	94.12	93.32	94.78	95.02
Diabetes	67.21	68.45	69.53	71.92
Surface inspection	82.01	82.10	83.02	84.13

classification accuracy was achieved in both implementations. However, the evolution based on the two-loop implementation was about five times faster.

5 Conclusions

Proposed is a general framework for designing a fuzzy rule-based classification system. The developed two-stage GA partitions the search space and enables evolving both structure and parameters of the classifier. Salient input variables, specific for each fuzzy rule, are also found during the search process.

Computer simulations of four real world problems have shown that the performance obtained from the classifier is comparable or even higher than the best performance obtained by other authors when using "black box" as well as fuzzy rule-based models. The proposed variable selection tool allowed to significantly increase the classification accuracy if compared to the case of using all the available input variables. It was shown through computer simulations that a large number of rules and input variables can be eliminated from the model without deteriorating the classification accuracy. Moreover, the classification accuracy of the test set data increased due to the reduction.

Acknowledgments We acknowledge the support from the agency for international science and technology development programmes in Lithuania (COST IC0602).

References

1. Vapnik VN (1998) Statistical learning theory. Wiley, New York
2. Bishop CM (2006) Pattern recognition and machine learning. Springer, Singapore
3. Tipping ME (2001) Sparse bayesian learning and the relevance vector machine. J Mach Learn Res 1: 211–244
4. Hamel L (2006) Visualization of support vector machines with unsupervised learning. In: Proceedings of IEEE symposium computational intelligence and bioinformatics and computational biology, 2006 CIBCB'06, IEEE 1–8
5. Strumbelj E, Bosnic Z, Kononenko I, Zakotnik B, Kuhar CG (2010) Explanation and reliability of prediction models: the case of breast cancer recurrence. Knowl Inf Syst 24(2):305–324
6. Janikow CZ (1998) Fuzzy decision trees: issues and methods. IEEE Trans Syst Man Cybern Part B Cybern 28(1):1–14
7. Olaru C, Wehenkel L (2003) A complete fuzzy decision tree technique. Fuzzy Sets Syst 138:221–254
8. Jang JR (1993) ANFIS: adaptive-network-based fuzzy inference system. IEEE Trans Syst Man Cybern 23:665–685
9. Carpenter GA, Grossberg S, Markuzon N, Reynolds JH, Rosen DB (1992) Fuzzy ARTMAP: a neural network architecture for incremental supervised learning of analog multidimensional maps. IEEE Trans Neural Netw 3(5):698–713
10. Simpson KP (1992) Fuzzy min-max neural networks—Part 1: classification. IEEE Trans Neural Netw 3(5):776–786
11. Abe S, Lan MS (1995) A method for fuzzy rules extraction directly from numerical data and its application to parten classification. IEEE Trans Fuzzy Syst 3(1):18–28

12. Pedrycz W (1990) Fuzzy sets in pattern recognition: methodology and methods. Pattern Recognit 23(1–2):121–146
13. Jaffar MA, Hussain A, Mirza AM (2010) Fuzzy entropy based optimization of clusters for the segmentation of lungs in CT scanned images. Knowl Inf Syst 24(1):91–111
14. Denton AM, Wu J (2009) Data mining of vector-item patterns using neighborhood histograms. Knowl Inf Syst 21:173–199
15. Takagi T, Sugeno M (1985) Fuzzy identification of systems and its applications to modeling and control. IEEE Trans Syst Man Cybern 15(1):116–132
16. Tao F, Zhao D, Zhang L (2009) Resource service optimal-selection based on intuitionistic fuzzy set and non-functionality QoS in manufacturing grid system. Knowl Inf Syst doi:10.1007/s10115-009-0263-6
17. Wang LX (1997) Modeling and control of hierarchical systems with fuzzy systems. Automatica 33(6):1041–1053
18. Nauck D, Klawon F, Kruse R (1997) Foundations of neuro-fuzzy systems. Wiley, Chichester
19. Duda RO, Hart PE, Stork DG (2001) Pattern classification. 2nd edn. Wiley, New York
20. Bezdek JC (1981) Pattern recognition with fuzzy objective function algorithms. Plenum Press, New York
21. Kohonen T (1990) The self-organizing map. Proc IEEE 78(9):1461–1480
22. Yoshinari Y, Pedrycz W, Hirota K (1993) Construction of fuzzy models through clustering techniques. Fuzzy Sets Syst 54(2):157–165
23. Wang Y, Rong GA (1999) self-organizing neural-network based fuzzy system. Fuzzy Sets Syst 103:1–11
24. Chen MY, Linkens DA (2001) A systematic neuro-fuzzy modeling framework with application to material property prediction. IEEE Trans Syst Man Cybern Part B 31(5):781–790
25. Castellano G, Castiello C, Fanelli AM, Mencar C (2005) Knowledge discovery by a neuro-fuzzy medeling framework. Fuzzy Sets Syst 149:187–207
26. Zhou E, Khotanzad A (2007) Fuzzy classifier design using genetic algorithms. Pattern Recognit 40(12):3401–3414
27. Nishina T, Hagiwara M (1997) Fuzzy inference neural network. Neurocomputing 14:223–239
28. Chang PC, Liao TW (2006) Combining som and fuzzy rule base for flow time prediction in semiconductor manufacturing factory. Appl Soft Comput 6:198–206
29. Kasabov N (2001) Evolving fuzzy neural networks for supervised/unsupervised on-line, knowledge-based learning. IEEE Trans Syst Man Cybern 31(6):902–918
30. Minku FL, Ludermir TB (2008) Clustering and co-evolution to construct network ensembles: an experimental study. Neural Netw 21:1363–1379
31. Abonyi J, Roubos JA, Szeifert F (2003) Data-driven generation of compact, accurate, and linguistically sound fuzzy classifiers based on a decision-tree initialization. Int J Approx Reason 32(1):1–21
32. Kbir MM, Benkirane H, Maalmi K, Benslimane R (2000) Hierarchical fuzzy partition for pattern classification with fuzzy if-then rules. Pattern Recognit Lett 21:503–509
33. Pulkkinen P, Koivisto H (2007) Identification of interpretable and accurate fuzzy classifiers and function estimators with hybrid methods. Appl Soft Comput 7:433–520
34. Verikas A, Bacauskiene M (2002) Feature selection with neural networks. Pattern Recognit Lett 23(11):1323–1335
35. Roubos JA, Setnes M, Abonyi J (2003) Learning fuzzy classification rules from labeled data. Inf Sci 150:77–93
36. Nauck D, Kruse R (1999) Obtaining interpretable fuzzy classification rules from medical data. Artif Intell Med 16(2):149–169
37. Lin CT, Lu YC (1995) A neural fuzzy system with linguistic teachnig signals. IEEE Trans Fuzzy Syst 3(2):169–189
38. Tung WL, Quek C, Cheng P (2004) GenSo: a novel neural-fuzzy based early warning system for predicting bank failures. Neural Netw 17:567–587
39. Nozaki K, Ishibuchi H, Tanaka H (1996) Adaptive fuzzy rule-based classification systems. IEEE Trans Fuzzy Syst 4(3):238–250
40. Wang CH, Hong TP, Tseng SS (1998) Integrating fuzzy knowledge by genetic algorithms. IEEE Trans Evol Comput 2(4):138–149
41. Tsang CH, Kwong S, Wang H (2007) Genetic-fuzzy rule mining approach and evaluation of feature selection techniques for anomaly intrusion detection. Pattern Recognit 40(9):2373–2391
42. Ozyer T, Alhajj R, Barker K (2007) Intrusion detection by integrating boosting genetic fuzzy classifier and data mining criteria for rule pre-screening. J Netw Comput Appl 30(1):99–113
43. Hoffmann F (2004) Combining boosting and evolutionary algorithms for learning of fuzzy classification rules. Fuzzy Sets Syst 141:47–58
44. Wang CH, Hong TP, Tseng SS (2000) Integrating membership functions and fuzzy rule sets from multiple knowledge sources. Fuzzy Sets Syst 112:141–154

45. Er MJ, Zhou Y (2008) Automatic generation of fuzzy inference systems via unsupervised learning. Neural Netw 21(10):1556–1566
46. Chen CH, Tseng VS, Hong TP (2008) Cluster-based evaluation in fuzzy-genetic data mining. IEEE Trans Fuzzy Syst 16(1):249–262
47. Nakashima T, Schaefer G, Yokota Y, Ishibuchi H (2007) A weighted fuzzy classifier and its application to image processing tasks. Fuzzy Sets Syst 158:284–294
48. Ishibuchi H, Nojima Y (2007) Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning. Int J Approx Reason 44(1):4–31
49. Ishibuchi H, Nakashima T, Murata T (2001) Three-objective genetics-based machine learning for linguistic rule extraction. Inf Sci 136:109–133
50. Ishibuchi H, Nakashima T, Morisawa T (1999) Voting in fuzzy rule-based systems for pattern classification problems. Fuzzy Sets Syst 103:223–238
51. Lei Z, Ren-hou L (2008) Designing of classifiers based on immune principles and fuzzy rules. Inf Sci 178:1836–1847
52. Mohamadi H, Habibi J, Abadeh MS, Saadi H (2008) Data mining with a simulated annealing based fuzzy classification system. Pattern Recognit 41(5):1824–1833
53. Lowe DG (1995) Similarity metric learning for a variable-kernel classifier. Neural Comput 7:72–85
54. Chang CL (1974) Finding prototypes for nearest neighbour classifiers. IEEE Trans Comput 23: 1179–1184
55. Tomek I (1976) An experiment with the edited nearest-neighbour rule. IEEE Trans Syst Man Cybern 6:448–452
56. Tomek I (1976) Two modifications of CNN. IEEE Trans Syst Man Cybern 6:769–772
57. Verikas A, Bacauskiene M, Malmqvist K (2003) Learning an adaptive dissimilarity measure for nearest neighbour classification. Neural Comput Appl 11(3–4):203–209
58. Mamdani EH, Assilian S (1975) An experiment in linguistic synthesis with a fuzzy logic controller. Int J Man-Mach Stud 7(1):1–12
59. Lin Y, Cunningham GA (1995) A new approach to fuzzy-neural system modeling. IEEE Trans Fuzzy Syst 3(2):190–198
60. Zolghadri MJ, Mansoori EG (2007) Weighting fuzzy classification rules using receiver operating characteristics (ROC) analysis. Inf Sci 177:2296–2307
61. Pakkanen J, Iivarinen J, Oja E (2004) The evolving tree—A novel self-organizing network for data analysis. Neural Process Lett 20:199–211
62. Kohonen T (2001) Self-organizing maps. 3rd edn. Springer, Berlin
63. Guzaitis J, Verikas A (2008) An efficient technique to detect visual defects in particleboards. Informatica 19(3):363–376
64. Song HH, Lee SW (1996) LVQ combined with simulated annealing for optimal design of large-set reference patterns. Neural Netw 9(2):329–336
65. Kalina A, Mezyk E (2008) Accuracy boosting induction of fuzzy rules with artificial immune systems. In: Proceedings of the international multiconference on computer science and information technology vol 3, IEEE, pp 155–159
66. Halavati R, Shouraki SB, Lotfi S, Esfandiar P (2009) Symbiotic evolution of rule based classifier systems. Int J Artif Intell Tools 18(1):1–16
67. Ho SY, Chen HM, Ho SJ (2004) Design of accurate classifiers with a compact fuzzy-rule base using an evolutionary scatter partition of feature space. IEEE Trans Syst Man Cybern Part B 34(2):1031–1043

Author Biographies

Antanas Verikas is currently holding a professor position at both Halmstad University Sweden and Kaunas University of Technology, Lithuania. His research interests include image processing, pattern recognition, artificial neural networks, fuzzy logic, and visual media technology. He is a member of the International Pattern Recognition Society, European Neural Network Society, International Association of Science and Technology for Development, Swedish Society of Learning Systems, and a member of the IEEE.

Jonas Guzaitis is a Ph.D. student at the department of Electrical and Control Equipment, Kaunas University of Technology, Lithuania. His research interests include image processing, pattern recognition, artificial neural networks, and applied soft computing.

Adas Gelzinis received the M.S. degree in electrical engineering from Kaunas University of Technology, Lithuania, in 1995. He received the Ph.D. degree in computer science from the same university, in 2000. He is a senior researcher in the Department of Electrical and Control Equipment at Kaunas University of Technology. His research interests include artificial neural networks, kernel methods, pattern recognition, signal and image processing, texture classification.

Marija Bacauskiene is a senior researcher in the Department of Electrical and Control Equipment at Kaunas University of Technology, Lithuania. Her research interests include artificial neural networks, image and signal processing, pattern recognition, and fuzzy logic. She participated in various research projects and published numerous papers in these areas.