Automatic generation of fuzzy logic rule bases: Examples I

Hartmut Surmann and Alexander Selenschtschikow

Abstract— **Learning fuzzy rule-based systems with genetic algorithms can lead to very useful descriptions of several problems. Many different alternative descriptions can be generated. In many cases, a simple rule base similar to rule bases designed by humans is preferable since it has a higher possibility of being valid in unforeseen cases. Thus, the main idea of this paper is to study the genetic fuzzy rule base learning algorithm FRBL [1] by examples from the machine learning repository [2] and to compare it with some other approaches.**

Keywords— **Fuzzy logic controller, genetic algorithm, entropy of fuzzy rules, machine learning.**

I. Introduction

The aim of control theory is to define a function f:X→Y with the intention to show that f(x) is the correct answer given the input x. On the base of pieces of fuzzy information (fuzzy granules) an approximation of such an ideal function f:X→Y is mentioned in fuzzy approaches to control applications [3]. This approximation is achieved by a system of fuzzy IF-THEN rules like:
If x is A THEN y is B, where A and B are labels for fuzzy subsets. Human system designer use such a rule as a logical implication A(x) → B(y) in our human unformalized logic. This usage is rather questionable but leads to good results in a huge number of practical applications.

Learning algorithms for the automatic generation of fuzzy rules are also well known [4]. These learning algorithms generate fuzzy rules on the base of learning vectors and learn/optimize the I/O behavior of the system. Novel approaches [5], [6], [7], [1] consider also the structure of the rule base. Cheong et. al. [5] use only a very simple rule base structure and they restrict the membership functions which leads to solutions far away from an optimum but they write correctly: *"We believe that their power have been blindly used to optimize the design of FLC's without regards to the simple and common sense reasoning structure of FLC's. Though the blind use of GA's can produce near optimal FLC's, it rises problems such as messy overlapping of fuzzy sets and rules not in agreement with common sense."* Castillo et. al. [6] coded one rule in a chromosome and selected the rule base on the base of the degree of completeness and the degree of consistency. Our approach [1] coded a rule base in one chromosome and leads to very useful descriptions [8], [9]. An important idea is inspired by neuro biology results. They postulate that human learning e.g. speech acquisition is also a process of reducing a rich parameter set. Therefore, the structure of the fuzzy rule base system is enlarged at the beginning and reduced during the learning process.

Nevertheless, all of these novel approaches provide better results i.e. the fuzzy rules are more readable by humans. Nevertheless they still have difficulties to model the human logical implication.

This paper shows some examples of generated fuzzy rule bases and is organized as follows. The next section gives a brief introduction to the basic algorithm and the last section shows the results of the FRBL algorithm learning four examples from the machine learning database [2].

Fraunhofer Institute for Autonomous intelligent Systems, Schloß Birlinghoven, D-53754 Sankt Augustin, Germany, Tel: ++492241/14-2518 hartmut.surmann@ais.fraunhofer.de, http://www.ais.fhg.de/ surmann/

II. Basic terms of Genetic Algorithms and Fuzzy Rule-Based Systems

Now a very brief introduction to GA and FRBS is given. An elaborated description can be found in [1].

A. Fuzzy Rule-Based Systems

Fuzzy Rule-Based Systems consist of K fuzzy rules of the form:

$$k: \text{IF } x_1 \text{ is } I_{1,k} \text{ AND } x_2 \text{ is } I_{2,k} \text{ AND } \ldots x_n \text{ is } I_{n,k} \text{ THEN } y \text{ is } O_k$$

where $x_1, \ldots, x_n$ and y represent input and output variables and $I_{1,k}, \ldots, I_{n,k}$, O_k their respective MFs. Then the extent to which a rule is activated is calculated as:

$$\alpha_k = \mu_{I_{1,k}}(x_1)\ AND\ \mu_{I_{2,k}}(x_2)\ AND\ \ldots\ AND\ \mu_{I_{n,k}}(x_n)$$

where $\mu_{I_{i,k}}(x_i)$ is the membership value of x_i in the $I_{i,k}$ fuzzy set. The result of the *AND* combination is used as a measure for the truth of the rule where the *AND* operation is a T-Norm e.g minimum. When a rule is activated with a truth value α_k, the inferencing process states that the MF of the output set is:

$$\mu_{O_k}^{inf}(y) = \alpha_k\ AND\ \mu_{O_k}(y),$$

which is the fuzzy result of the rule. Here, the minimum operator *min* is used as an *AND* operator for both cases.

The total output fuzzy set is calculated by the compositional rule of inference as

$$\mu_O^{total}(y) = \mu_{O_1}^{inf}(y)\ OR\ \mu_{O_2}^{inf}(y)\ OR\ \cdots\ OR\mu_{O_K}^{inf}(y)$$

where it is supposed that the system is described by K rules. In order to get a crisp number as output usually a de-fuzzyfication method is used. The calculation of the fuzzy result function $\mu_O^{total}(y)$ and the final crisp value is the bottleneck during computation. Therefore a modified center of gravity algorithm is used to calculate the area A_i and the center of area M_i of each MF before runtime:
$A_i = \int O_i(y)dy$, $M_i = \int y \times \mu_{O_i}(y)dy$.
The output value is computed as

$$y_{crisp} = \frac{\sum_{i=0}^{K} \alpha_i \times M_i}{\sum_{i=0}^{K} \alpha_i \times A_i} \quad (1)$$

B. Costs and minimal FRBS

For an automatic learning algorithm which learns the function f:X→Y and the structure of the fuzzy rule base, the input and output variables as well as the MFs of the FRBS a function is demanded which represents the complexity of that problem:

Definition 1 (Complexity Cost Function, E) A FRBS with k fuzzy rules, n linguistic variables, $k*n$ subpremises and $m = m_1 + \ldots + m_n$ MFs is less complex than one with

1. $k+1$ fuzzy rules, $k*n$ subpremises and m MFs or
2. k fuzzy rules, $(k*n)$+1 subpremises and m MFs or
3. k fuzzy rules, $k*n$ subpremises and $m+1$ MFs.

The weighting of the above three terms depends on the application. A mathematical expression of the complexity function E is given later. On the base of the cost function E a two step learning procedure is described. First an "opening step" is any operation on a FRBS that increases the cost function E. Second a "closing step" is any operation on a FRBS that decreases the cost function E. Any opening step results in a new fuzzy system with more MFs. Whereas any closing step results in a new fuzzy system with less rules, MFs or subpremises. Finally we are interested in minimal fuzzy systems and we define them by using the cost function E:

Definition 2 (Minimal Fuzzy-System) Let us approximate a real system $f(\vec{x})$, using a fuzzy system $X(\vec{x})$ and let $\varepsilon > 0$ be a maximal accepted error limit. A fuzzy system X with $sup\{\mid f(\vec{x}) - X(\vec{x}) \mid \quad \mid \vec{x} \in U\} \leq \varepsilon$ is called minimal if there is no other fuzzy system Y with $sup\{\mid f(\vec{x}) - Y(\vec{x}) \mid \quad \mid \vec{x} \in U\} \leq \varepsilon$ and $E(Y) < E(X)$, where E is the complexity cost function.
Alternatively to the absolute metric the Euclidian metric can be used:
$(\sqrt{\sum(f(\vec{x}) - Y(\vec{x}))^2} \leq \varepsilon)$. The real system $f(\vec{x})$ is represented by referential vectors $\vec{r}$.

C. Genetic Algorithms

For the optimization of the FRBS we use a standard GA consisting of a number of encoded solutions, some genetic operators which produce new solutions and a fitness function which says how good a particular solution is [10]. A parallel version of the algorithm is implemented on the base of MPICH, a portable implementation of the messages passing interface MPI and the PGAPack V1.0 [11], [12].

Every *chromosome S* contains all fuzzy set parameters m_{ij} and σ_{ij}, where $i = 1,\ldots,n+m$ is the number of variables and $j = 1,\ldots,q_n$ is the number of respective MFs per variable.

m_{ij} and σ_{ij} are the means and sigmas of Gaussian like MFs. They are constructed from probability density functions by $\mu(x) = \lambda p(x)$. The constant λ is calculated using the constraint $\sup \mu = 1$. The Gaussian probability distribution $p(x) = \varphi(x; m, \sigma^2)$ is chosen, because this distribution fits a lot of real world problems. Therefore, the membership functions used in the system are defined by

$$\mu(x) = \sigma\sqrt{2\pi}\ \varphi(x; m, \sigma^2) = \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right). \quad (2)$$

The MFs are limited by $\mu(x) = 0\ for\ x \notin [m - 2.9\sigma, m + 2.9\sigma]$. Non–symmetric Gaussian MFs consist of two parts (left and right) with a common middle point but different σ_{left} and σ_{right} for the left and right side. Triangular or trapezoid MFs are special case of the Gaussian MFs by an approximation with two or three lines. The length of a string is: $L = 2 \times l \times (n+m) \times k$ for the symmetric and $L = 3 \times l \times (n+m) \times k$ for the non–symmetric case.

The used *fitness function* has to fulfill two constraints. The first part considers the I/O behavior and the second part the structure and complexity of the resulting fuzzy system. The structural part of the fitness function has three items.

$$OF = OF^{I/O} \times (OF^E \times OF^S \times OF^{UZ}). \quad (3)$$

OF is the fitness function that the GA has to maximize.

The only property for the I/O behavior of the FRBS is the maximization of the reciprocal of the Mean Square Error (MSE).

$$OF_{pv}^{I/O} = pv / \sum_{i=1}^{pv} \sum_{j=1}^{n_{out}} (r_{i,j} - o_{i,j})^2, \quad (4)$$

where pv is the number of referential data pairs, n_{out} is the dimension of the output vector, $o_{i,j}$ is the current computed output value of the FRBS and $r_{i,j}$ is the respective referential value ($i = 1 \ldots pv$, $j = 1 \ldots n_{out}$). For other application fields this part of the fitness function should be adapted to the application class. For control processes the system response parameter e.g. rise time RT, overshoot

OV, steady-state error SSE, settling time ST, etc. can be used. Classifier systems are coded as a stair function. Each class is identified by a number (1,2, ... #of classes) and the above MSE criteria is used during the learning.

Three criteria for the structure of the FRBS are also considered. The first criterion (OF^E) for the structure of a FRBS is the *degree of fuzziness* or the *entropy* of a FRBS. It is defined by the average number of activated rules:

$$R_\phi = \frac{1}{pv} * \sum_{i=1}^{pv} R_{cur,i}, \tag{5}$$

where $R_{cur,i}$ is the current number of activated rules for the i–th input/output pair and pv the number of those pairs. If the number of activated rules is minimal i.e. one, then a FRBS is maximally understandable by humans. FRBSs with a high number of activated rules behave like a neural network i.e. a lot of rules (neurons) determine the output values. To decrease the entropy of a FRBS and with it the overlap of the MFs, a maximal number R_{max} and a desired number R_{act} of activated fuzzy rules has to be defined and the fitness function is extended as:

$$OF^E = \frac{1}{(\frac{R_\phi}{R_{max}} - 1)a + (\frac{R_\phi}{R_{act}} - 1)b + 1} \tag{6}$$

where a and b are predefined weighting factors.

The second criterion (OF^S) for the structure of the FRBS is the number of MFs. For that MFs with high degrees of overlap are counted. Such nearly equal MF pairs are desirable, because they can be easily unified to only one MF which reduces the cost function (Def. 1(3)) The criterion for the highest possible number of similar MFs is :

$$OF^S = \left(\frac{s_{no}}{(n+m)R_{total}}\right)\gamma + 1 \tag{7}$$

where s_{no} is the number of similar MFs, $n+m$ is the total number of input and output variables, R_{total} is the total number of rules and $\gamma \in \mathbf{R}$ is a predefined weighting factor for the number of similar MFs.

The third criterion (OF^{UZ}) for the structure of the FRBS deals with *never activated* ($\mu(x) = 0, \forall x \in U$) and *always completely activated* ($\mu(x) = 1, \forall x \in U$) MFs. On one hand a subpremise of a rule can be eliminated if the belonging MF is always completely activated for all of the training pairs so that the cost function E decreases (Def. 1(2)). On the other hand MFs which are never activated can be eliminated together with all the rules that they participated to so that the cost function E also decreases (Def. 1(1)). The criterion for the highest possible number of always zero and always one MFs with the predefined weighting factors $\zeta, \eta \in \mathbf{R}$ is:

$$OF^{UZ} = \left(\frac{u_{no}}{(n+m)R_{total}}\right)\zeta + \left(\frac{z_{no}}{(n+m)R_{total}}\right)\eta + 1, \tag{8}$$

where u_{no} is the number of always one MFs, z_{no} is the number of always zero MFs. Always one MFs are replaced by trapeze MFs covering the total variable domain, while zero MFs are replaced by impulses.

Beside the well known mutation and crossing over operators we use two new operators: *Set zero/one* is used to produce MFs which are always one or always zero. It randomly selects a MF for each input/output variable and changes its sigma value to infinity ($+\infty$) in case of an always one MF or to zero in case of an always zero MF. The other genetic operator called *set similar*, selects randomly for each input/output variable a MF and makes it equal to the MF that is most similar to it.

III. Examples

This section shows some examples of the learning algorithm FRBL. Most of the datasets can be found at the machine learning repository database of the UCI [2]. A continously updated result page can be found at http://www.ais.fhg.de/ surmann/fuzzy/ [13].

A. Gas Furnace Data

The data set from Box & Jenkins' consists of 296 pairs of input/output observations. The input is the gas flow rate into the furnace and the output is the concentration of CO_2 in the exhausted gas.

As well as in literature, two input variables $n_{in} = 2$: $x(t-\tau_1)$, $y(t-\tau_2)$ and one output variable $y(t)$ are chosen. Here, $x(t-\tau_1)$ denotes the input at time $t-\tau_1$ and $y(t-\tau_2)$ the output of the process at $t-\tau_2$. With $\tau_1 = 4$ and $\tau_2 = 1$ the referential data set contains $p = 296 - \tau_1 = 292$ elements. The above data set is used for training and as validation set for testing. Table II shows some results for different numbers of rules and entropy and table I some results from the literature.

TABLE I
RESULTS FOR THE GAS FURNACE DATA FROM LITERATURE.

MSE	Init. Rules	Rules	Author
0.469	7x6	19	Tong [14]
0.355	5x5	6	Sugeno et.al [15]
0.320	9x9	81	Pedryz [16]
0.328	5x5	25	Xu-Lu [17]
0.172	7x7	38	Abreu et.al. [18]
0.138	3x5	15	Surmann [8]

A.1 Discussion of the result

The *first example* shows an optimization with a target of very low costs E (Def. 1). It starts with 10 fuzzy rules and 10 MFs per variable and ends with only 2 rules and two MFs per variable and a mean square error of 0.172

1. IF X1 is small **AND** X2 is large **THEN** Y1 is small
2. IF X1 is large **AND** X2 is small **THEN** Y1 is large

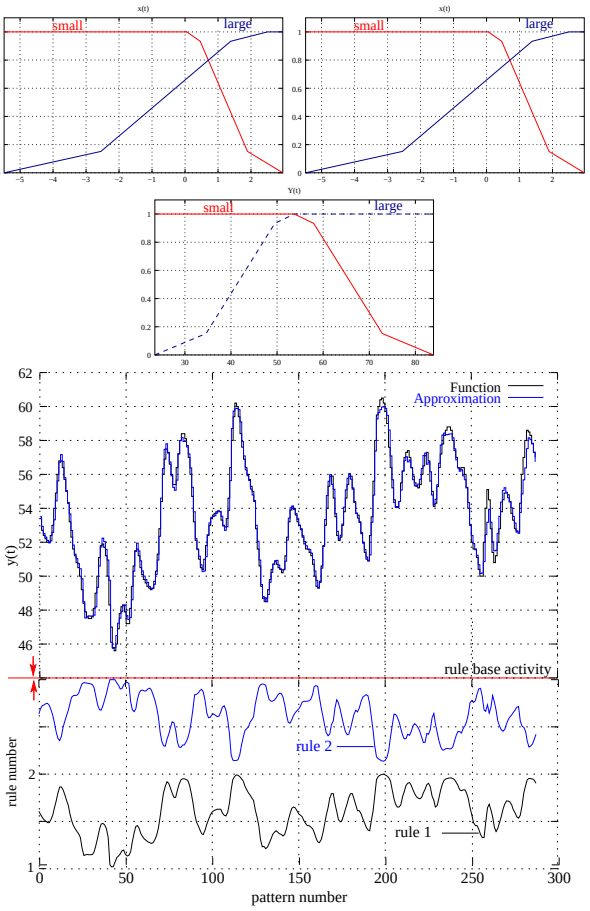

Fig. 1. Top: The two fuzzy rules and six MFs for the first example of the Gas Furnace data. Bottom: The function and approximated curve (MSE = 0.172) together with the activation of the rules.

(fig. 1). The rule activation shows the contraditionary part of the rules.

The *second example* shows an optimization with a target of low costs E (Def. 1) but higher entropy. It results in a better approximation behavior. It also starts with 10 fuzzy rules and 10 MFs per variable and ends with 5 rules and 5 MFs per variable (Fig. 2). The number of activated rules is always high (four), but the fuzzy rule base does not behave like a neural network. The rule activation (Fig. 2 bottom) shows that mainly one rule is highly activated and that the other three rules are only activated to a small degree. Nearly all input values of X2 = y(t-4) lie in the overlapping area of the three MFs. The means square error (MSE = 0.117) is distinctly better then in example one.

1. IF X1 is very-small **AND** X2 is very-large **THEN** Y1 is large
2. IF X1 is small **AND** X2 is medium **THEN** Y1 is medium
3. IF X1 is medium **AND** X2 is small **THEN** Y1 is very-small
4. IF X1 is large **AND** X2 is large **THEN** Y1 is very-large
5. IF X1 is very-large **AND** X2 is very-small **THEN** Y1 is small

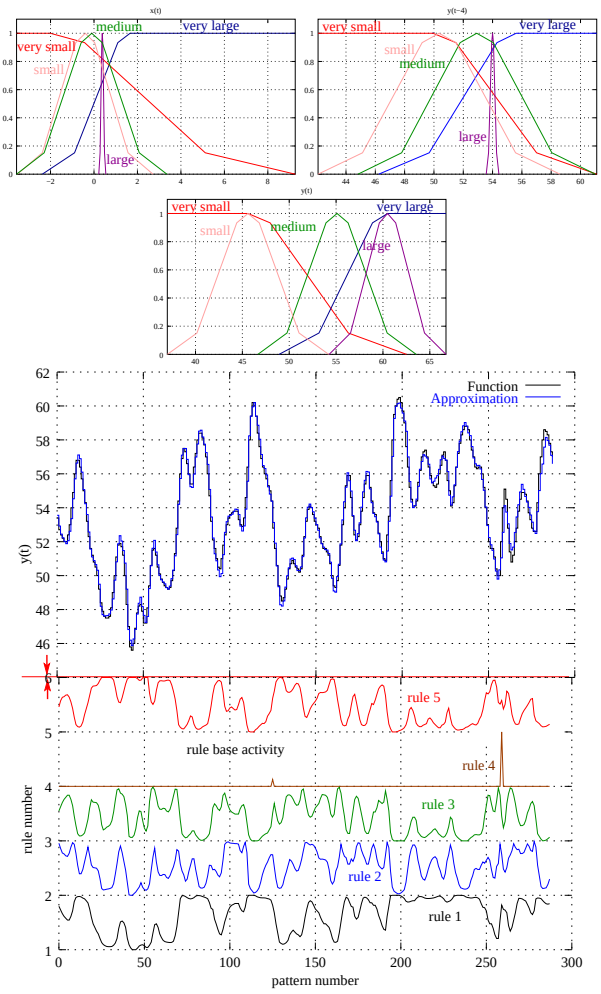

Fig. 2. Top: The five fuzzy rules and six MFs for the second exmaple of the Gas Furnace data. Bottom: The function and approximated curve (MSE = 0.117) together with the activation of the rules.

B. Wine data

This data is the results of a chemical analysis of wines from the same region but with different types of grapes, using 13 continuous variables, 178 examples splitted in 3 classes. Castillo et. al. [6] use this dataset to show the performance of the SLAVE algorithm for fuzzy rule generation.

The data was used with many others for comparing various classifiers. The classes are separable, though only RDA has achieved 100% correct classification. RDA : 100%, QDA 99.4%, LDA 98.9%, 1NN 96.1% (z-transformed data). All results use the leave-one-out technique. In a classification context, this is a well posed

TABLE II

RESULTS FOR THE GAS FURNACE DATA FROM THE FRBL ALGORITHM (SYMMETRIC MFS).

MSE	#rules	entropy
0.172	2	2.0
0.163	3	2.90
0.144	4	2.85
0.132	4	2.98
0.164	4(5)	2.73
0.150	3(5)	2.96
0.117	5	3.91

1. IF X1 is medium **AND** X2 is medium **AND** X4 is medium **AND** X5 is medium **AND** X7 is medium **AND** X8 is medium **AND** X10 is large **AND** X12 is medium **AND** X13 is small **THEN** Y1 is large
2. IF X1 is small **AND** X2 is medium **AND** X4 is large **AND** X5 is small **AND** X7 is small **AND** X8 is medium **AND** X10 is small **AND** X12 is small **AND** X13 is large **THEN** Y1 is small
3. IF X1 is large **AND** X2 is medium **AND** X4 is small **AND** X5 is large **AND** X7 is large **AND** X8 is medium **AND** X10 is medium **AND** X12 is large **AND** X13 is medium **THEN** Y1 is medium

Fig. 3. The three fuzzy rules for the wine data.

problem with ”well behaved” class structures. It is a good data set for first testing of a new classifier, but not very challenging [2].

TABLE III

RESULTS FOR THE WINE DATA FROM THE FRBL ALGORITHM AND FROM LITERATURE.

Class.	MSE	#rules/MFs	entropy	author
100%	0.001931	3 / 26	1.70	FRBL
100%	0.001079	3 / 32	1.72	FRBL
96.76%	–	5.2 / –	–	[6]

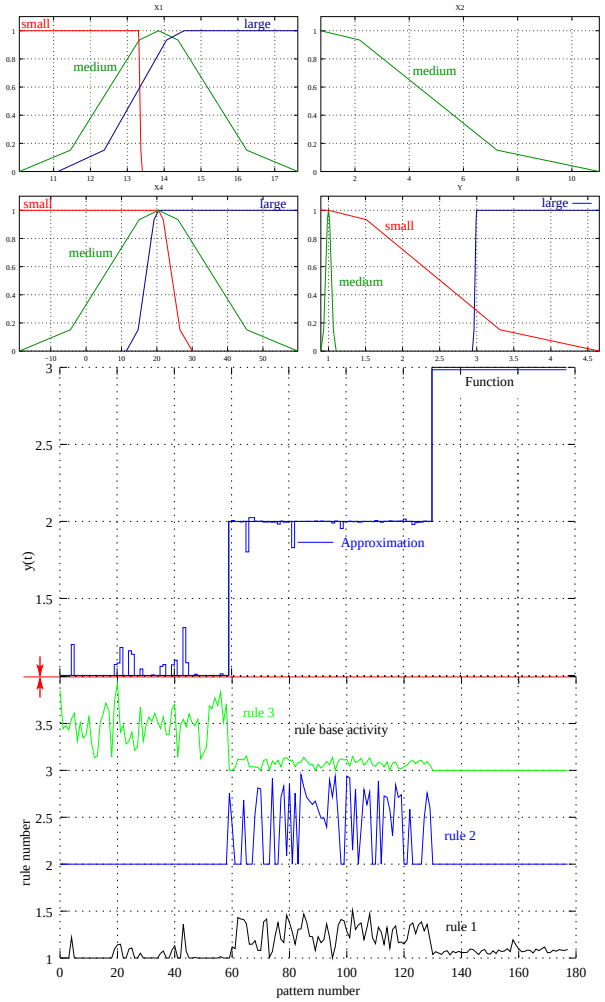

Fig. 4. A subset MFs for the wine data. Bottom: The function and approximated curve (MSE = 0.0019, 100% correct classified) together with the activation of the rules.

B.1 Discussion of the result

The FRBL algorithm classifies 100% with only 3 rules (fig. 3 and fig. 4). The number of MFs could be further reduced. Rule 3 classifies the first class, rule 1 the third class. Class 2 is classified by rule 2 and 1. The automatic labeling algorithm mislabeled two output MFs (small and medium fig. 4).

C. Ionosphere data

Classification of radar returns from the ionosphere, 34 attributes, 200 training and 151 test examples, 2 classes. Original they investigated using backprop and the perceptron training algorithm on this database. Using the first 200 instances for training, which were carefully split into almost 50% positive and 50% negative examples, they found that a ”linear” perceptron attained 90.7%, a ”non-linear” perceptron attained 92% and backprop an average of over 96% accuracy on the remaining 150 test instances. The test instances consist of 123 ”good” and only 24 ”bad” instances. Accuracy on ”good” instances was much higher than for ”bad” instances. Backprop was tested with several different numbers of hidden units and incremental results were also reported (corresponding to how well the different variants of backprop classify after a periodic number of epochs) [2].

David Aha (aha@ics.uci.edu) briefly investigated this database. He found that nearest neighbor attains an accuracy of 92.1%, that Ross Quinlan’s C4 algorithm attains 94.0% (no windowing), and that IB3 (Aha & Kibler, IJCAI-1989) attained 96.7% (parameter settings: 70% and 80% for acceptance and dropping respectively). Castillo et. al. [6] use this dataset to show the performance of the SLAVE algorithm for fuzzy rule generation.

TABLE IV
RESULTS FOR THE IONOSPHERE DATA FROM THE FRBL ALGORITHM AND THE LITERATURE.

Class.	sets	#rules/MFs	author
98,57%	total (351)	2 / 35	FRBL
98.0%	learning (200)	2 / 35	FRBL
99.34%	testing (151)	2 / 35	FRBL
92.88%	total (351)	4 / –	[6]

1. IF X1 is medium **AND** X2 is large **AND** X3 is large **AND** X4 is large **AND** X5 is large **AND** X6 is large **AND** X7 is large **AND** X8 is large **AND** X9 is large **AND** X10 is large **AND** X11 is small **AND** X12 is large **AND** X13 is large **AND** X14 is large **AND** X15 is large **AND** X16 is large **AND** X17 is large **AND** X18 is large **AND** X19 is large **AND** X20 is small **AND** X21 is large **AND** X22 is small **AND** X23 is small **AND** X24 is large **AND** X25 is small **AND** X26 is small **AND** X27 is small **AND** X28 is small **AND** X29 is large **AND** X30 is large **AND** X31 is small **AND** X32 is large **AND** X33 is small **AND** X34 is small **THEN** Y1 is large
2. IF X1 is medium **AND** X2 is small **AND** X3 is small **AND** X4 is small **AND** X5 is small **AND** X6 is small **AND** X7 is small **AND** X8 is small **AND** X9 is small **AND** X10 is small **AND** X11 is large **AND** X12 is small **AND** X13 is small **AND** X14 is small **AND** X15 is small **AND** X16 is small **AND** X17 is small **AND** X18 is small **AND** X19 is small **AND** X20 is large **AND** X21 is small **AND** X22 is large **AND** X23 is large **AND** X24 is small **AND** X25 is large **AND** X26 is large **AND** X27 is large **AND** X28 is large **AND** X29 is small **AND** X30 is small **AND** X31 is large **AND** X32 is small **AND** X33 is large **AND** X34 is large **THEN** Y1 is small

Fig. 5. Fuzzy rules for the Ionosphere data set.

C.1 Discussion of the result

The FRBL algorithm classifies 98.57% of the complete dataset and 99.34% of the test dataset with only 2 rules (fig. 5 and fig. 6). The number of MFs could also be further reduced. Rule 1 classifies the first class, rule 2 the second.

D. Iris data

In his introduction of the method of discriminant analysis, R.A. Fisher presented an analysis of measurements on Irises. There were 50 flowers from each of three species of Iris - Iris setosa, Iris versicolor, and Iris virginica. There were four measurements on each flower - petal length, petal width, sepal length and sepal width. Since species type is known, the problem is one of statistical pattern recognition and the data can be analyzed using discriminant analysis or a supervised learning approach. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant.

This is perhaps the best known database to be found in the pattern recognition literature. Fishers paper is well known in the field and is referenced frequently to this day. (See Duda & Hart, for example.) One class is linearly separable from the other two classes; the latter are NOT linearly separable from each other [2].

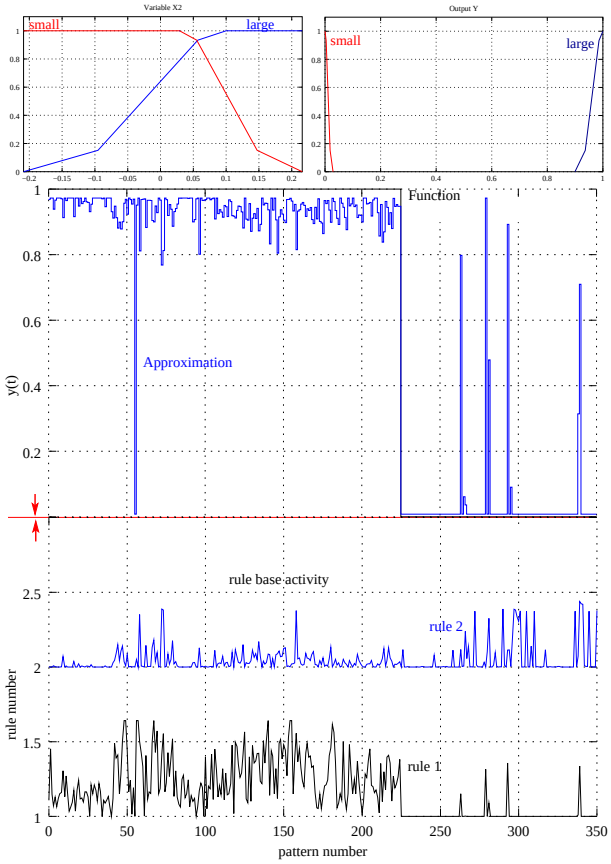

Fig. 6. A subset of the MFs for the Ionosphere data. Bottom: The function and approximated curve (98.57% correct classified, 5 errors) together with the activation of the rules.

TABLE V
RESULTS FOR THE IRIS DATA FROM THE FRBL ALGORITHM.

Class.	sets	#rules/MFs	author
97.33%	total (150)	2 / 11	FRBL

D.1 Discussion of the result

The FRBL algorithm classifies 97.33% of the iris dataset with only 3 rules (fig. 7 and fig. 8). Rule 1 classifies the first class, rule 1,2 and 3 the second class and rule 3 the third class.

Acknowledgment Special thanks to Claudia Hirsch for hints and corrections.

REFERENCES

[1] Hartmut Surmann and Michail Maniadakis, "Learning feed-forward and recurrent fuzzy systems: a genetic approach," *Journal of Systems Architecture, Special issue on evolutionary computing*, vol. 47, no. 7, pp. 535 – 556, 2001.
[2] URL:, "UCI machine learning repository," in *http://www.ics.uci.edu/∼mlearn/MLRepository.html*, 2001.
[3] Lotfi A. Zadeh, "Fuzzy sets," *Information and Control*, vol. Vol. 8, pp. 338 – 353, 1965.
[4] F. Herrera and J.L. Verdegay, Eds., *Studies in Fuzzines and Soft*

1. IF X1 is medium **AND** X2 is medium **AND** X3 is small **AND** X4 is small **THEN** Y1 is small
2. IF X1 is medium **AND** X2 is medium **AND** X3 is medium **AND** X4 is medium **THEN** Y1 is medium
3. IF X1 is medium **AND** X2 is medium **AND** X3 is large **AND** X4 is large **THEN** Y1 is large

Fig. 7. Fuzzy rules for the iris data.

Computing: Genetic Algorithms and Soft Computing (8), Physica-Verlag, 1996.
[5] France Cheong and Richard Lai, "Constraining the optimization of a fuzzy logic controller using an enhanced genetic algorithm," *IEEE Transaction on systems, man and cybernetics – part B: Cybernetics.*, vol. 30, no. 1, pp. 31–46, February 2000.
[6] Luis Castillo, Antonio Gonzalez, and Raul Perez, "Includinga simplicity criterion in the selection of the best rule in a genetic fuzzy learning algorithm," *Fuzzy Sets and Systems*, vol. 120, pp. 309 – 321, 2001.
[7] Ingo Renners, Adolf Grauel, and Ernesto Saavedra, "Methodology for optimizing fuzzy classifiers based on computational intelligence," in *Computational Intelligence: Theory and Applications, Lecture Notes in Computer Science 2206, Springer Verlag, ISBN 3-540-42732-5*, 2001, pp. 412 – 419.
[8] Hartmut Surmann, Andreas Kanstein, and Karl Goser, "Self-organizing and genetic algorithms for an automatic design of fuzzy control and decision systems," in *Proceedings of the First European Congress on Fuzzy and Intelligent Technologies, EUFIT'93, Aachen*, 7–10 Sept. 1993, pp. 1097–1104.
[9] Hartmut Surmann, *Genetic Optimizing of Fuzzy Rule-Based Systems*, vol. 8 of *Studies in Fuzziness and Soft Computing*, chapter Optimization of Fuzzy Controllers, pp. 389 – 402, Physica-Verlag, Sep. 1996.
[10] David E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Massachusetts, 1989.
[11] William Gropp and Ewing Lusk, "Guide to mpich, a portable implementation of mpi version 1.2.2," *http://www-unix.mcs.anl.gov/mpi/mpich/*, 2001.
[12] David Levine, "Users guide to the pgapack parallel genetic algorithm library," *ftp.mcs.anl.gov/pub/pgapack/*, 1996.
[13] URL:, "Examples of the FRBL learning algortihm," in *http://www.ais.fhg.de/∼surmann/fuzzy/*, 2001.
[14] R. M. Tong, "The evaluation of fuzzy models derived from experimental data," *Fuzzy Sets and Systems*, vol. Vol. 4, pp. 1 – 12, 1980.
[15] Michio Sugeno and Takahiro Yasukawa, "Linguistic modeling based on numerical data," in *Proceedings of IFSA '91, Brüssel*, 1991.
[16] Witold Pedrycz, "An identification algorithm in fuzzy relational systems," *Fuzzy Sets and Systems*, vol. Vol. 13, pp. 153 – 167, 1984.
[17] Chen-Wei Xu and Yong-Zai Lu, "Fuzzy model identification and self-learning for dynamic systems," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. smc-17, pp. 683 – 689, 1987.
[18] Antonio Abreu, Luis Custodio, and Carlos Pinto-Ferreira, "A fuzzy modelling: a rule based approach," in *Fifth IEEE International Conference on Fuzzy Systems, FUZZ-IEEE 96*, 1996, pp. 162–168.

Hartmut Surmann received his diploma in computer science from the Department of Computer Science, University of Dortmund and his Ph.D. in electrical engineering from the Faculty of Electrical Engineering, University of Dortmund, Germany, in 1989 and 1994, respectively. He is a senior researcher at FHG (former GMD) Autonomous intelligent System Institute in St. Augustin, Germany. His primary research interests are robotics and computational intelligence. He has published several papers in the area of design, application and hardware for robotics and fuzzy logic controllers. He is a member of the GI and VDE.

Dr. Surmann received the FUZZ-IEEE/IFES'95 robot intelligence award for a fuzzy system for indoor mobile robot navigation and the Ph.D. award for his thesis from the German AI institutes in 1996.

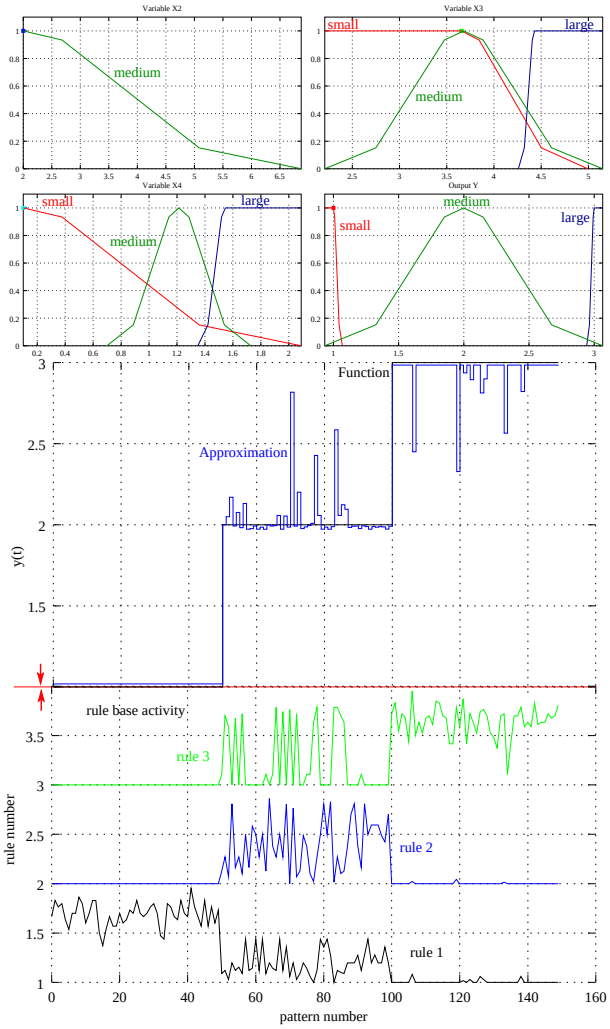

Fig. 8. A subset of the MFs for the iris data. Bottom: The function and the approximated curve (97.33% correct classified, 4 errors) together with the activation of the rules.

Alexander Selenschtschikow studies computer science and applied mathematics in Moscow Railway University, Institute for computer science and social management. From 1996 to 1999 he worked as a programmer for Internet-Design company in Moscow. In 2000-2001 he studies computer science at Fh-Gummersbach, Germany and works as a research assistant at FHG-AiS. He writes his diploma thesis on computational intelligence. Furthermore he is interested in optimization, data analyzation and robotics.