Decision-theoretic approaches in fuzzy rule generation for diagnosis and fault detection problems

Sebastian Beck **Ralf Mikut** **Jens Jäkel** **Georg Bretthauer**
Forschungszentrum Karlsruhe, Inst. of Applied Computer Science, P.O. Box 3640, 76021 Karlsruhe, Germany
sebastian.beck@iai.fzk.de ralf.mikut@iai.fzk.de jens.jaekel@iai.fzk.de georg.bretthauer@iai.fzk.de

Abstract

A typical task in technical fault detection or medical diagnosis problems is to discriminate normal behavior from one or more types of abnormal behavior by means of different measured or computed features. This may lead to difficult classification problems due to extremely different a priori probabilities of classes and heterogeneous classes (e. g. unknown sub-classes for different errors to be detected). In this paper, an approach to design fuzzy classifiers is presented, which is based on decision-theoretic measures and uses a learning data set with feature values and given information about decision and classifier costs.

Keywords: decision theory, fuzzy modelling, diagnosis, fault detection, cost sensitive learning.

1 Motivation

Many classification problems are characterized by different costs for wrong decisions (”decision costs”) [1, 2, 3] and overlapping classes, i.e. there is no error-free solution of the classification problem. Usually, it is ”cheaper” to classify a normal sample as abnormal than vice versa. This consideration of costs is well established in crisp and fuzzy decision making [3, 4] but only few approaches include costs in classifier design [5]. Therefore, classifiers tend to fail in such tasks if they only try to reduce the number of misclassifications.

Other costs may include measurement and computation costs of features or user-defined preferences (e. g. due to interpretability) of features (”classifier costs”) [6]. The first is an important aspect regarding the offline classifier design. In case of limited sensor equipment and computational capacities one will intend to determine the feature combination that is most valuable for the overall classification costs. The latter aspect is for a deeper understanding of the essential problem if some features are easier to interpret than others. Consequently, a compromise between low costs for wrong decisions and low classifier costs has to be found.

In this paper we propose an approach for generating fuzzy classifiers based on decision-theoretic measures used in all design steps. It includes a tree-oriented rule generation with a subsequent pruning to generate generalized rules. Cooperating rules are selected as a rule base. An important feature is the possibility to interactively control the design process.

2 Rule generation and evaluation

We assume a data set for supervised learning with N samples, s features x_l, and one observed output variable y with m_y classes B_i. The fuzzy system to be generated contains rules with a general structure

$$R_r : \text{IF} \quad \underbrace{\underbrace{x_1 = A_{1,Rr}}_{\text{partial premise } P_{r1}} \quad \text{AND} \cdots \text{AND} \quad \underbrace{x_s = A_{s,Rr}}_{\text{partial premise } P_{rs}}}_{\text{premise } P_r}$$

$$\text{THEN } y = C_r, \quad r = 1,\ldots,r_{max}$$

and a default rule $R_{r_{max}+1}$: ELSE $y = C_{r_{max}+1}$.

The premise P_r consists of a conjunctive (AND) combination of partial premises $P_{r1},\ldots,P_{rs}$. The linguistic term $A_{l,Rr}$ can be a (primary) linguistic term $A_{l,i}$ $(i = 1,\ldots,m_l)$ of the feature x_l or a disjunctive (OR) combination of some neighboring or all linguistic terms of x_l, called derived linguistic term [7, 8]. In case of all terms, this partial premise has no influence on the rule activation and in the presentation of the rule it is omitted. Each rule conclusion C_r consists of one linguistic term $\hat{B}_j$. A maximum defuzzification chooses the best decision $\hat{B}_j$ resulting from feature values and the generated rule base.

The basis of our decision theoretic approach is the estimated expectation of the cost per decision$\hat{L}_T$ caused by misclassifications $\hat{L}_D$ in combination with the classifier cost per decision L_C. $\hat{L}_T$ can be estimated for the whole data set (all examples in the root node of the tree or covered by the rule base) or a part of it (examples in other nodes of the tree or covered by single rules):

$$\hat{L}_T = L_C + \underbrace{\sum_{i=1}^{m_y}\sum_{j=1}^{m_y} L(\hat{B}_j|B_i)\cdot\hat{p}(\hat{B}_j\wedge B_i)\,.}_{\hat{L}_D} \tag{1}$$

Here, $L(\hat{B}_j|B_i)$ stands for the cost of decision$\hat{B}_j$ given the actual class B_i and $\hat{p}(\hat{B}_j\wedge B_i)$ is the estimated joint probability of this decision-class combination. We use this measure to rank features during tree induction, single rules evaluation or rule base selection and to find a cost-optimal conclusion C_r for a generated rule with the given premise P_r:

$$C_r = \operatorname*{argmin}_{\hat{B}_j}\sum_{i=1}^{m_y} L(\hat{B}_j|B_i)\cdot\hat{p}(B_i|P_r), \tag{2}$$

where $\hat{p}(B_i|P_r)$ denotes the estimated conditional probability of the class $y = B_i$ with the samples that are covered by the premise P_r.

The probability $\hat{p}(\hat{B}_j\wedge B_i)$ of joint occurrence of decision $\hat{B}_j$ and class B_i is estimated from the learning data set by

$$\hat{p}(\hat{B}_j\wedge B_i) = \sum_{r\in(C_r=\hat{B}_j)} \hat{p}(B_i|P_r)\cdot\hat{p}(P_r)\,. \tag{3}$$

The constraints

$$\sum_{i=1}^{m_y}\sum_{j=1}^{m_y}\hat{p}(\hat{B}_j\wedge B_i) = 1,\ \sum_{r=1}^{r_{max}+1}\hat{p}(P_r) = 1, \tag{4}$$

are met by the following estimation algorithm in (6) and the inference scheme proposed in [8] which handles overlapping premises.

All probabilities of fuzzy events are estimated by counting membership values in learning data and solving constrained optimization problems [7]. As an example, the probabilities $\hat{p}(P_r), \hat{p}(B_i), \hat{p}(B_i|P_r), \hat{p}(B_i\ \wedge\ P_r)$ are estimated by:

$$\hat{p}(P_r) = \frac{1}{N}\sum_{k=1}^{N}\mu_{P_r}(x_l[k]),\ \hat{p}(B_i) = \frac{1}{N}\sum_{k=1}^{N}\mu_{B_i}(y[k]) \tag{5}$$

$$E = \min_{\mathbf{R}_{B|P_r}} \|\underbrace{\mathbf{R}_{B|P_r}\cdot\boldsymbol{\mu}_{P_r}}_{\hat{\boldsymbol{\mu}}_B} - \boldsymbol{\mu}_B\|_F^2 \tag{6}$$

$$\begin{aligned}
&\text{s. t. } \mathbf{R}_{B|P_r} \geq \mathbf{0}_{m_y\times(r_{max}+1)},\ \mathbf{1}_{m_y}^T\mathbf{R}_{B|P_r} = \mathbf{1}_{r_{max}}^T\\
&\text{with } \mathbf{R}_{B|P_r} = ((\ \hat{p}(B_i|P_r)\ )) \in [0,1]^{m_y\times(r_{max}+1)},\\
\boldsymbol{\mu}_{P_r} =&((\ \mu_{P_r}(x_l[k])\ )) \in [0,1]^{(r_{max}+1)\times N},\\
\boldsymbol{\mu}_B =&((\ \mu_{B_i}(y[k])\ )) \in [0,1]^{m_y\times N},\\
\hat{p}(B_i\wedge P_r) =&\hat{p}(B_i|P_r)\cdot\hat{p}(P_r)\,.
\end{aligned}$$

After designing membership functions for each linguistic term $A_{l,i}$ (e. g. by using fast heuristic methods with a fixed number of terms and triangular membership functions like clustering or similar sample frequencies for each term), the proposed rule generation process of a fuzzy system consists of three steps.

In the first step, one or more decision trees are induced and rule hypotheses are extracted from them. In each node of the tree, the feature x_l leading to minimal estimated expectation cost in (1) for the samples in the node is chosen to split the data set. This cost is estimated by means of an auxiliary ”rule base” consisting of $r = 1,\ldots,m_l$ rules with premises $P_r = A_{l,r}$ whose rule conclusions are found by (2). If splitting the node does not lead to a cost reduction in comparison to the decision of a no-splitting rule$\hat{L}_{D,NS}$, the node is set to a terminal node with the optimal decision given by the no-splitting rule. The conclusion of the no-splitting rule is calculated by (2) with $\hat{p}(B_i|P_r) = \hat{p}(B_i)$ and the cost:

$$\hat{L}_{D,NS} = \min_{\hat{B}_j}\sum_{i=1}^{m_y} L(\hat{B}_j|B_i)\cdot\hat{p}(B_i)\,. \tag{7}$$

Otherwise, the algorithm creates m_l new nodes. The algorithm terminates when all nodes are terminal nodes or have been used for splitting the data set. The probabilities $\hat{p}(B_i|P_r)$, $\hat{p}(P_r)$, $\hat{p}(B_i)$ in (2), (3), (7) are estimated only with the samples in the node. In order to obtain a comprehensive rule set, decision trees with different features in the root node are induced by step-wise discarding the best features.

In the second step (”pruning”), candidates for a generalization of the extracted rules are step-wise generated by adding disjunctions with neighboring terms or deleting partial premises (see Section 3 for an example). In each step, the best alternative among the original rule and the pruning candidates is accepted.

An optimal single rule with conclusion $C_r = \hat{B}_r$ should cover all examples of class B_r and none of the other classes. Especially in problems with non-compact classes, the rule base has to contain several rules with the same conclusion in order to obtain minimum decision costs. During the generalization of single rules, the rules extracted from the tree are processed one after another. It is therefore difficult to decide whether one ore more rules for class B_i should be included in the rule base. From this point of view, a compromise between covering all samples of B_i with one premise and receiving as few misclassifications as possible has to be found. However, the more important point during rule generalization is to avoid misclassifications.

There are two straightforward approaches to evaluate the pruning candidates using the estimated expectancy of the decision cost: Firstly, only the cost for those samples within the rule premise are considered. From (1) we get:

$$\hat{L}_{D,P_r} = \sum_{i=1}^{m_y} L(C_r|B_i) \cdot \hat{p}(B_i|P_r). \tag{8}$$

Secondly, the whole set of data is considered and each single rule is handled as an auxiliary ”rule base” with two rules whose premises are P_r and $\overline{P_r}$ with $\hat{p}(\overline{P_r}) = 1 - \hat{p}(P_r)$. The conclusions are calculated according to (2) and the ”rule base” is evaluated using (1). The first approach is not appropriate as generalization of a rule premise does not result in a cost reduction as long as there is no misclassified example in the extracted rule and the candidates. In that case, the estimated expectation of the decision cost remains constant: $\hat{L}_{D,P_r} = L(C_r|B_r)$. Thus, there is no incentive for an error-free rule premise to be generalized.

The second approach may also be not appropriate as it tends to generalize rules to much. That is because of several reasons. Let C_r be the optimal decision for premise P_r and $C_{\bar{r}}$ the optimal decision for $\overline{P_r}$. If the cost for misclassification of a third class B_k is equal for decision C_r and $C_{\bar{r}}$, the estimated cost expectation for this rule is independent of the classification of the B_k-examples. This means, using the second evaluation approach the algorithm does not notice certain misclassifications.

Another point occurs e.g. in a two-class scenario with at least one non-compact class B_r. Here, at least two rules are necessary in the rule base to minimize the number of misclassifications. Depending on the ratios of cost between the two possible misclassifications the estimated cost expectation for a single rule with $C_r = \hat{B}_r$ can decrease during generalization although the number of misclassifications increases. This happens as long as more examples of the class B_r are captured by the premise than necessary to compensate the cost for misclassified examples of the other class that are also captured by the premise. In such problems, the algorithm would try to cover the whole class with one rule.

Thus, we have to modify the cost criterion for single rules. To give an incentive for generalization of error-free premises we add to the estimated cost expectation of the premise the potential misclassification cost for those examples that belong to the rules conclusion C_r but are not covered by the premise:

$$\begin{aligned}\hat{L}_{D,r} = \sum_{i=1}^{m_y} [L(C_r|B_i) \cdot \hat{p}(B_i|P_r) + \\ \hat{p}(C_r|\overline{P_r}) \cdot L(C_{\bar{r}}|B_i) \cdot \hat{p}(B_i|\overline{P_r})].\end{aligned} \tag{9}$$

Here, $\hat{p}(C_r|\overline{P_r})$ is the probability of an example of the class specified in the rule conclusion not to be covered by the rule premise.

In the third step, a rule base is chosen by adding step-wise a rule (resulting from the second step) such that (1) is minimized. The conclusion of the default rule (all samples not yet covered by a premise) can be either manually fixed to a decision $\hat{B}_j$ or automatically set by (2) (with $\hat{p}(P_{r_{max}+1}) = 1 - \sum_{r=1}^{r_{max}} \hat{p}(P_r)$). A third possibility is a rejection class $\hat{B}_{m_y+1}$. Thus, the algorithm tends to find at least one rule for each class B_i. The cost for the decision $\hat{B}_{m_y+1}$ is calculated as mean value of the lowest and second lowest cost for each class:

$$\begin{aligned}L(\hat{B}_{m_y+1}|B_i) = \\ \frac{1}{2} \cdot \Big(\min_{j \text{ with } \hat{B}_j \neq \arg\min_j L(\hat{B}_j|B_i)} L(\hat{B}_j|B_i) + \min_j L(\hat{B}_j|B_i) \Big).\end{aligned} \tag{10}$$

Thus, it is cheaper to put examples in the rejection class than to misclassify them. At the end of the design process the examples that are still assigned to the rejection class are turned into the default rule with automatically fixed conclusion. The algorithm stops if no further rule reducing (1) is found. To avoid large rule bases, a threshold for improvement can be defined.

Because all different design steps (tree induction, pruning, rule base selection) only create suboptimal solutions, the resulting rule base is generally also suboptimal. However, a concurrent complete search over all possible rule bases is not practicable because of the combinational explosion of the search space.

In Table 1 the different cost criteria and the samples they are applied on are displayed for the three design steps. Besides the decision cost $\hat{L}_D$, we optionally integrate classifier cost L_C into the evaluation criteria. The key problem with classifier cost during rule generation is the deleting of promising candidate rules caused by none or too low improvements of decision cost in comparison to a higher classifier cost. This effect may stop the development of rules before the possible reduction of the decision cost during specialization (tree induction) and pruning (deleting of subpremises instead adding terms) is reached. Finally, such rule bases tend to be too simple. Hence, we consider classifier cost only in the third design step, where the rule generalization is finished and the algorithm can choose those rules with the best cost-information ratio. Alternative approaches which partially integrate classifier cost into tree induction and pruning will be investigated in future research.

Table 1: L_D criterion during design phase.

Design step	Criterion	Samples
Tree induction	cost expectation (1)	1st node: all following: split
Pruning	cost expectation modified (9)	all of class C_r and errors in P_r
Rule base selection	cost expectation (1)	all

The classifier costs include the costs for all features l_P used in at least one rule premise:

$$L_C = \sum_{l \in l_P} L_{C,l} - L_{CD,l}(l_P). \qquad (11)$$

The feature costs per data set $L_{C,l}$ includes both fixed and variable costs. The fixed costs consist of the investment (engineering, asset cost e. g. for sensors and microcontrollers, and commissioning) prorated to the number of years the equipment is in use and the operational fixed costs per year (e. g. staff, maintenance, energy). The fixed costs arise whether the equipment is in use or not. In contrast, the variable costs are directly related to the generation of a single sample (e. g. consumable material). Thus the feature cost per data set $L_{C,l}$ is the sum of the total fixed costs divided by the number of samples per year and the variable costs. The costs may be reduced by $L_{CD,l}$ if other features are used simultaneously. As an example, the cost for a feature is smaller if an other feature based on the same sensor information is already chosen and there is no need for another sensor. If $L_{C,l}$ is not precisely known, a rough estimation is reasonable to rank different classifiers and features in a qualitative way. In addition, virtual costs like interpretability aspects and user preferences may be included. Considering the feature costs in the design process leads to classifiers using mostly those feature with a reasonable cost-information ratio (see example in the next section).

3 Example

The method will be explained by a simple illustrative example with $m_y = 2$ classes and $s = 4$ features. The class B_1 (abnormal) with $N_1 = 60$ is non-compact and consists of two subclasses B_{1a}, B_{1b}. But this subdivision is not labelled in the learning data set. The class B_2 (normal) contains $N = 300$ samples. The samples for both classes are produced by a constant mean value $\overline{x_i}(B_{1a}) = [2.5,\ 3,\ 1,\ -2.5]$, $\overline{x_i}(B_{1b}) = [-2,\ 2,\ 1,\ 2]$ and $\overline{x_i}(B_2) = [1,\ 2,\ 1,\ -1]$ with an additional non-correlated normal distributed noise. The third feature x_3 is not useful for classification as the mean values for both classes are identical. The fourth feature x_4 is highly correlated with x_1 and gives only almost redundant information.

The feature costs for $x_1 - x_4$ are $L_{C,1} = L_{C,2} = L_{C,3} = 0.05$, $L_{C,4} = 0.03$. There is no discount for simultaneous use of features: $L_{C,P} = 0$. The decision cost matrix is

$$\mathbf{L} = \begin{pmatrix} 0 & L(\hat{B}_1|B_2) \\ L(\hat{B}_2|B_1) & 0 \end{pmatrix} \qquad (12)$$

$$L_{Ratio} = L(\hat{B}_2|B_1)/L(\hat{B}_1|B_2) \qquad (13)$$
$$\text{with } \min(L(\hat{B}_1|B_2), L(\hat{B}_2|B_1)) = 1$$

where the decision $\hat{B}_j$ are in the rows and the actual classes B_i are in the columns.

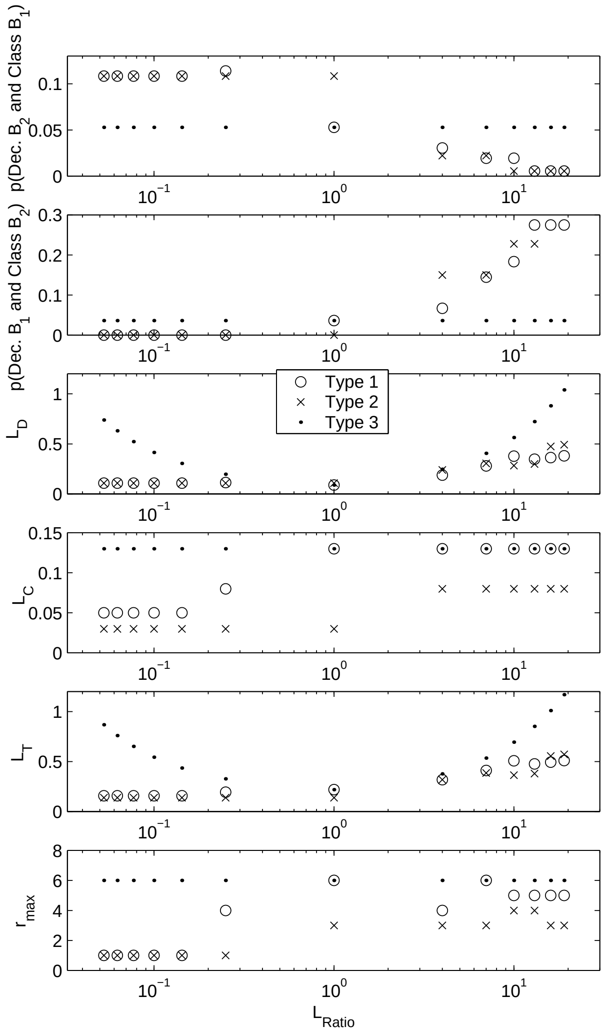

Figure 1: Estimated probabilities for misclassifications $\hat{p}(\hat{B}_2 \wedge B_1)$, $\hat{p}(\hat{B}_1 \wedge B_2)$, decision costs $\hat{L}_D$, classifier costs L_C, total cost $\hat{L}_T$ and number of rules in the rule base r_{max} (top down) as functions of the decision cost ratio L_{Ratio} for different cost approaches

The results of the proposed method for different ratios L_{Ratio} (13) are shown in Fig. 1. Here, estimated probabilities for misclassifications $\hat{p}(\hat{B}_2 \wedge B_1)$, $\hat{p}(\hat{B}_1 \wedge B_2)$, decision costs $\hat{L}_D$, classifier costs L_C, total costs $\hat{L}_T$ and the number of rules in the rule base are compared for three different approaches. The rule bases of all three classifier types are evaluated by (1) with (12) and the given feature costs. But in the design phase, only Type 1 and 2 use the cost matrix in (12) and only Type 2 includes the feature costs (see Table 2). For Type 3 $L_{Ratio} = 1$ is used in the design process.

Table 2: Cost parameters during design phase.

Classifier	L_{Ratio} (13)	$L_{C,l}$
Type 1	0.05 - 20	0
Type 2	0.05 - 20	0.03-0.05
Type 3	1	0

With the proposed approach in Type 1 and Type 2, the classifiers avoid more expensive misclassifications in uncertain situations and turns these estimated probabilities to zero or nearby. As a consequence, both types accept higher probabilities of cheaper misclassifications. Because Type 3 does not use the different misclassification cost, it generates always the same solution. This leads to high decision costs at very small and very high values of the cost ratio in comparison to the other types.

The main difference between the classifier design in Type 1 and Type 2 in comparison to Type 3 is the acceptance of pruning candidates resulting in misclassifications as shown in Fig. 2 and the selection of differentially generalized rules in the third step.

In addition, Type 2 is able to reduce the classifier costs by preferring the cheaper feature x_4 in comparison to the more expensive feature x_1 which contains almost the same information and partially by skipping x_2 with some loss of information. It reduces classifier costs without significant increase of decision costs. Some small differences for $L_{Ratio} \approx 10$ in $\hat{L}_D$ between Type 1 and Type 2 (Type 2 causes less misclassification using less features) are caused by the suboptimality of our approach. In combination with the reduced number of used features the rule bases of Type 2 tend to consist of less rules. As expected none of the classifiers uses feature x_3.

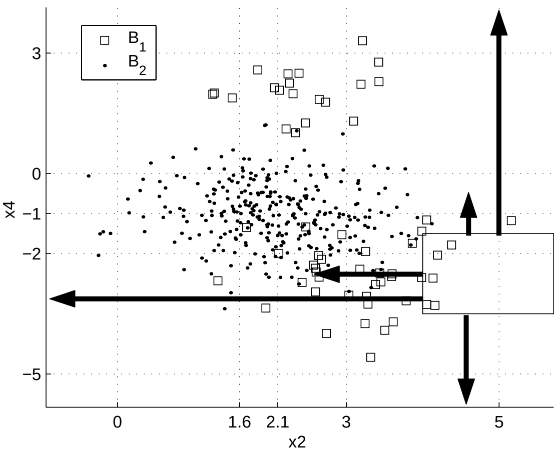

Figure 2: Extracted rule from decision tree with α-cut of premise (IF $x_4 = A_{4,2}$ AND $x_2 = A_{2,5}$, solid line), parameters of membership functions (dotted lines) and pruning options (arrows)

4 Conclusions

The proposed method fully integrates decision-theoretic measures in the data-based design of fuzzy classifiers. In applications with asymmetric costs for misclassifications and classifier costs, premises and conclusions of the found rule base depend on costs. Most alternative methods ignore these costs or only change rule conclusions for given premises depending on costs. Consequently, the proposed method is often able to reduce costs in comparison to other methods.

The application of this method to a fault detection benchmark problem in quality supervision of car gears (with asymmetric decision costs) and a medical diagnosis problem of gait analysis (with different feature costs by interpretability restrictions) is underway.

References

[1] S. Merler, C. Furanello, B. Larcher, and A. Sboner. Automatic model selection in cost-sensitive boosting. *Information Fusion*, 4:3–10, 2003.

[2] Provost, F. and Fawcett, T. Analysis and Visualization of Classifier Performance: Comparison under Imprecise Class and Cost Distributions. *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97).* American Association for Artificial Intelligence, 1997.

[3] R.L. Keeney and H. Raiffa. *Decisions with Multiple Objective*. John Wiley & Sons, Inc., 1976.

[4] R.E. Bellman and L.A. Zadeh. Decision making in a fuzzy environment. *Management Science*, 17(4):141–163, 1970.

[5] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, Belmont, Ca., 1984.

[6] P. Turney. Types of cost in inductive concept learning. In *Proc. Workshop on Cost-Sensitive Learning at the 17th International Conference on Machine Learning (WCSL at ICML-2000)*, pp. 15–21. Stanford University, California., 2000.

[7] J. Jäkel, L. Gröll, and R. Mikut. Tree-oriented hypothesis generation for interpretable fuzzy rules. In *Proc. 7th Europ. Congr. on Intelligent Techniques and Soft Computing EUFIT'99*, pp. 279–280, Aachen, 1999. CD-ROM.

[8] R. Mikut, J. Jäkel, and L. Gröll. Inference methods for partially redundant rule bases. In R. Hampel, M. Wagenknecht, and N. Chaker (Ed.), *Fuzzy Control: Theory and Practice*, Advances in Soft Computing, pp. 177–185, Heidelberg, 2000. Physica.