Intelligent Information Systems VIII
Proceedings of the Workshop held in
Ustroń, Poland, June 14-18 , 1999

Methodology of extraction, optimization and application of logical rules

Włodzisław Duch, Rafał Adamczak and Krzysztof Grąbczewski

Department of Computer Methods, Nicholas Copernicus University,
Grudziądzka 5, 87-100 Toruń, Poland.
E-mail: duch,raad,kgrabcze@phys.uni.torun.pl

Abstract. Methodology of extraction, optimization and application of sets of logical rules is described. The three steps are largely independent. Neural networks are used for initial rule extraction, local or global minimization procedures for optimization, and Gaussian uncertainties of measurements are assumed during application of logical rules. A tradeoff between rejection/error level is discussed. This methodology has been applied to a number of benchmark and real life problems with very good results.

Keywords: Neural networks, logical rules, fuzzy rules, optimization, medical diagnosis.

1 Introduction.

There is no reason why a simple model based on logical rules should always work, but in cases when it does it is certainly worth using. Our strategy is to use the simplest description possible, but not simpler. If the number of logical rules required for high accuracy of classification is too large other, more sophisticated models are needed. In many applications simple crisp logical rules proved to be more accurate and were able to generalize better than many machine and neural learning algorithms [1]. In other applications fuzzy rules proved to be more accurate. Using soft trapezoidal membership functions allows for smooth transition from crisp to fuzzy rules and enables natural interpretation of rules. Moreover, it is equivalent to using crisp rules for data which has Gaussian uncertainties (cf. Section 4).

Extraction of logical rules from data may be done using statistical, pattern recognition and machine learning methods, as well as neural network methods. In this paper we present complete methodology for extraction, optimization and application of logical rules. These last two steps

are largely neglected in the literature, with current emphasis being still on the extraction methods. Overlapping clusters may lead to non-zero probability of classification $p(C_i|X;M)$ for several classes. In medical diagnosis it is important to obtain an estimation of reliability of several possible classifications, not just a single decision. This is possible if the probability density functions (PDFs) modeling data clusters are approximated by combinations of products of "membership" functions. Using conjunction of all membership functions for clusters belonging to a given class classification probabilities may be calculated.

Classical crisp logic rules are obtained from fuzzy rules if all membership functions are rectangular (values 0 or 1). Rectangles allow to define logical linguistic variables for each feature by intervals or sets of nominal values and thus allow to express logical rules in simple sentences like "if the size is small then fruit is not ripe". If rules from different classes are not overlapping classification probabilities 1 or 0 are sufficient. If rectangular functions are softened or changed to trapezoidal membership functions natural generalization and fuzzy interpretation is obtained. Fuzzy logic classifiers are frequently based on triangular membership functions for each input feature, a further simplification comparing to trapezoidal functions. Membership functions usually do not depend on the region of the input space. Feature Space Mapping (FSM) [2], our neurofuzzy system, does not require such drastic simplifications. Using a difference of two sigmoidal functions $\sigma(x) - \sigma(x-b)$ one may obtain a soft trapezoid membership functions. Increasing the slope of sigmoids changes trapezoidal functions into rectangular functions. The shape of these functions depends on the location in the input space, providing context-dependent membership functions.

In the next section several neural rule extraction algorithms are described, the third section deals with simplification and optimization of logical rules. The fourth section concerns the application of rules and overcoming their brittleness and the fifth section illustrates our methodology on a few problems. The paper is finished with a short discussion.

2 Neural rule extraction

Several neural algorithms for feature selection and extraction of initial rules have been developed in our group. Since logical rules require linguistic inputs, selection of initial linguistic variables (symbolic inputs) is the first step of the rule extraction process.

Linguistic (logical) input variables s_k, for integer or symbolic variables x_i, taking values from a finite set of elements $X_i = \{X_i^{(j)}\}$, are defined as true if x_i belongs to a specific subset $X_{ik} \subseteq X_i$. Defining X as a subset of integers such linguistic variables as "prime-number" or "odd-number" may be defined. For continuos input features x_i linguistic variable s_k is obtained by specifying an open or closed interval $x_i \in [X_k, X_k']$ or some other predicate $P_k(x_i)$. Each continuos input feature x_i should be replaced by two or more linguistic values for which predicates P_k are true or false. If such representation is used at the pre-processing stage inputs to neural networks are already discretized and the actual selection of intervals is done independently of the rule extraction. An alternative is to include selection of linguistic variables in neural models.

Initial values of the intervals for continuos linguistic variables may be determined by the analysis of histograms, dendrograms, decision trees or clusterization methods. **FSM**, our probability density estimation neurofuzzy network, is initialized using various clusterization methods [4, 3]. Optimization of linguistic variables is a part of the FSM learning process. These variables

are modeled using rectangular, triangular, trapezoidal, Gaussian or soft trapezoidal functions. The learning algorithm [4] finds optimal intervals and logical rules at the same time. These intervals are gradually increased if the number of errors does not grow, until the whole data range for a given feature is covered, indicating that the feature may be removed.

MLP (multilayered perceptron) neural models were modified by adding a special neural layer [5] of L-units (linguistic units) to find linguistic variables and logical rules simultaneously. L-units replace the continuos input variables with logical variables using rectangular (or soft trapezoid) filtering functions. They may also combine several inputs in the feature aggregation process. The simplest L-unit is a combination of two sigmoidal functions $L(x_i;b_i,b_i',\beta) = \sigma(\beta(x_i-b_i)) - \sigma(\beta(x_i-b_i'))$, parameterized by two biases b,b' determining the interval in which this function has non-zero value. The slope β of sigmoidal functions $\sigma(\beta x)$ is slowly increased during the learning process, transforming the fuzzy membership function ("soft trapezoid") into a window-type rectangular function [2, 5]. Similar smooth transformation is used in the FSM network using biradial transfer functions, which are combinations of products of $L(x_i;b_i,b_i')$ functions [6] with some additional parameters. Outputs of L-units $L(x_i;b_i,b_i',\beta)$ are usually combined and filtered through another sigmoid $\sigma(\sum_{ij} L(x_i;b_{ij},b_{ij}',\beta))$ or the product $\prod_{ij} L(x_i;b_{ij},b_{ij}',\beta)$ of these functions is used.

Initial linguistic variables have a strong influence on the logical rules extracted, especially if the learning process does not include optimization of their intervals. Therefore an iterative optimization process is used: neural networks with initial linguistic inputs are constructed, analyzed, logical rules are extracted, intervals defining linguistic variables are optimized using rules, and the whole process repeated until convergence is achieved. Usually two or three steps are sufficient, depending on the initial choice.

Construction and training of neural networks follows the initial definition of linguistic variables. We are using a constructive neural network algorithms, increasing their complexity to account for the incoming data. In FSM new neurons, corresponding to single rules, are created if the new data vector **X** is sufficiently far from the nearest center of the localized (for example rectangular) function realized by the existing neuron and if the activity of this function does not exceed certain minimum. MLP networks do not use localized representations and therefore should be transform first into something resembling a network performing logical operations (Logical Network, LN). In the constructive version of our method, called C-MLP2LN [7], we start from a single neuron connected to all L-units or linguistic inputs. This neuron is trained on all data minimizing the standard mean error function plus two penalty terms:

$$E(W) = E_0(W) + \frac{\lambda_1}{2}\sum_{i,j} W_{ij}^2 + \frac{\lambda_2}{2}\sum_{i,j} W_{ij}^2 (W_{ij}-1)^2 (W_{ij}+1)^2 \tag{1}$$

The first term, scaled by the λ_1 hyperparameter, encourages weight decay, leading to skeletonization of the network and elimination of irrelevant features. A relatively large value of λ_1 is set at the beginning of the training to ensure that only the most important features are left. The second term, scaled by λ_2, forces the remaining weights to approach ± 1 or 0, facilitating easy logical interpretation of the network function: 0 = irrelevant input, $+1$ = positive and -1 = negative evidence. As long as the slope β of sigmoidal functions in L-units is high or linguistic inputs are directly given the last condition does not have to be strictly enforced since it has

little influence on the final shapes of decision borders. The hyperparameters determine the simplicity/accuracy tradeoff of the generated network and extracted rules. If very simple networks (logical rules) are desired, giving only rough description of the data, λ_1 is increased until the error will grow sharply: although one may estimate the relative size of the regularization term versus the mean square error (MSE) a few experiments are sufficient to find the largest value for which the MSE is still acceptable and does not decrease quickly when λ_1 is decreased. After training of the first neuron (in K-class problem $K-1$ neurons may be trained at the same time) the simplified, skeleton network is kept frozen and a second neuron is added. Thus the network expands and shrinks during the learning process. This procedure is repeated until most of the data is correctly classified or the number of new neurons starts to grow rapidly, indicating overfitting and noise in the data. We have also tried a fully numerical procedure replacing the gradient-based procedure by search-based procedure with quantized weight values (Grąbczewski, Duch, submitted).

Logical rule extraction: the slopes of sigmoidal functions are gradually increased to obtain sharp decision boundaries and the complex decision regions are transformed into simpler, hypercuboidal decision regions. In FSM networks rule extraction is immediate since each node corresponds directly to one rule. Rules R_k implemented by MLP networks are obtained in the form of logical conditions by considering contributions of inputs for each linguistic variable s. A combination of linguistic variables activating the hidden neuron above the threshold is a logical rule of the form: $R = (s_1 \wedge \neg s_2 \wedge ... \wedge s_k)$. After analysis of all weights a set of rules $R_1 \vee R_2 ... \vee R_n$ is found for each output class. Since the first neuron for a given class is trained on all data for that class the rules it learns are most general, covering largest number of instances. Therefore rules obtained by the C-MLP2LN algorithm are ordered, starting with rules that cover many cases and ending with rules that cover only a few cases. Initial knowledge about the problem may also be inserted directly into the network structure, defining initial conditions modified further in view of the incoming data. Insertion of partially correct rules to be refined by the learning process is quite straightforward because the final network structure is quite simple.

3 Optimization of rules

The neural network training process unfortunately does not guarantee that the globally optimal solution is found, therefore rules should be optimized. In this step an additional choice between accuracy and rejection rates is possible.

Simplification of rules: some rules obtained from analysis of the network may involve spurious conditions, more specific rules may be contained in general rules or logical expressions may be simplified if written in another form. We use a Prolog program for the simplification step.

Optimization of rules: optimal intervals for linguistic variables are found by maximization of a predictive power of the rule-based classifier. Let $P(C_i, C_j|M)$ be the confusion matrix, i.e. the number of instances in which class C_j is predicted when the true class was C_i, given the model M. Then for n samples $\mathbf{p}(C_i, C_j|M) = P(C_i, C_j|M)/n$ is the probability of (mis)classification. The best parameters of the model M are selected by maximizing the "predictive power" of rules $\max_M [\mathrm{Tr}\, P(C_i, C_j|M)]$ over all parameters M, or by minimizing the number of wrong predictions (possibly with some risk matrix $\mathbf{R}(C_i, C_j)$), $\min_M \left[\sum_{i \neq j} \mathbf{R}(C_i, C_j) P(C_i, C_j|M)\right]$. We use the

weighted combination of these two terms:

$$E(M) = \gamma \sum_{i \neq j} P(C_i, C_j|M) - \mathrm{Tr}\, P(C_i, C_j|M) \geq -n \tag{2}$$

which is minimized over parameters M without constraints. If γ is large the number of errors after minimization may become zero but some instances may be rejected (i.e. rules will not cover the whole input space). An alternative to minimization of the whole set of rules simultaneously is to define a cost function for each rule separately and minimize it over parameters M used only in the single rule R . We have used two global minimization techniques, Adaptive Simulated Annealing (ASA) and shuffled multisimplex method [7]. One problem with optimization of rules at this level is the brittleness of classification error. Neural systems have good generalization properties because they are wide margin classifiers. Their decision borders are obtained from the mean square error optimization of smooth function that extends over larger neighborhood contributing to the error. This allows for three important improvements: the use of inexpensive gradient method instead of global minimization, more robust rules with wider classification margins, and probabilistic estimation of confidence. These improvements are discussed below.

4 Application of rules

Estimation of the accuracy of rules in important but difficult. Usually classification accuracy is tested using crossvalidation, each time including rule optimization on the training set. For rule-based classifiers this does not work because different sets of rules are produced for different partitions of the training set and in the end we do not know what confidence should be placed in the final set of rules. One may assign a confidence factor to each rule based on the number of correct classifications, for example $c_i = P(C_i, C_i|M) / \sum_j P(C_i, C_j|M)$, but such estimations are independent of the data classified and therefore are misleading.

One way to estimate confidence of rule-based classification is to use hierarchy of rules with different accuracy/rejection tradeoffs. Changing the value of γ during the optimization of rules leads to a series of models with higher and higher confidence of correct classification at the expense of growing rejection rate. A set of rules may classify some cases at the 100% confidence level; if some instances are not covered by this set of rules another set of (usually simpler) rules at a lower confidence level is used (confidence level is estimated as the accuracy of rules achieved on the training set). Estimation of classification confidence of a rejected vector $\mathbf{X}$ is possible if two sets of rules are considered, high reliability $R^{(1)}$ set rejecting vector $\mathbf{X}$, and lower reliability $R^{(2)}$ set assigning it to some class. Reliability of classification in the border region $R^{(2)} \setminus R^{(1)}$ should be estimated by looking at the number of training vectors belonging to this region and being in the neighborhood of $\mathbf{X}$ that fall into each class. A better method for estimation of confidence is given below.

The brittleness problem: good generalization requires the decision borders to be placed as far from the training data vectors as possible without increasing the error. Input values result usually from observations which are not quite accurate, therefore instead of the attribute value x a Gaussian distribution $G_x = G(y; x, s_x)$ centered around x with dispersion s_x should be given. This distribution may be treated as a membership function of a fuzzy number G_x. A Monte Carlo procedure may be performed sampling vectors from Gaussian distributions defined for

all attributes to compute probabilities $p(C_i|X)$. Analytical evaluation is based on the cumulative distribution function:

$$\rho(a-x) = \int_{-\infty}^{a} G(y;x,s_x)dy = \frac{1}{2}\left[1+\mathrm{erf}\left(\frac{a-x}{s_x\sqrt{2}}\right)\right] \approx \sigma(\beta(a-x)) \tag{3}$$

where erf is the error function and $\beta = 2.4/\sqrt{2}s_y$ makes the erf function similar to the standard unipolar sigmoidal function with the accuracy better than 2%. A rule $R_a(x)$ with single crisp condition $x \geq a$ is fulfilled by a Gaussian number G_x with probability:

$$p(R_a(G_x) = T) = \int_{a}^{+\infty} G(y;x,s_x)dy \approx \sigma(\beta(x-a)) \tag{4}$$

Incidentally, this is the reason why sigmoidal functions are so useful. Taking instead of the erf function a sigmoidal function changes the assumption about error distribution of x measurements from Gaussian to $\sigma(x)(1-\sigma(x))$, approximating Gaussian with $s^2 = 1.7$ within 3.5%. If the rule involves closed interval $[a,b], a \leq b$ the probability that it is fulfilled by a sample from the Gaussian distribution representing the data is:

$$p(R_{a,b}(G_x) = T) \approx \sigma(\beta(x-a)) - \sigma(\beta(x-b)) \tag{5}$$

Thus the probability that a given condition is fulfilled is proportional to the value of soft trapezoid function realized by L-units. Crisp logical rules with assumption that data has been measured with finite precision lead to soft L-functions that allow to compute classification probabilities that are no longer binary. In this way we may either fuzzify the crisp logical rules or obtain fuzzy rules directly from neural networks. The normalized product form $(\sigma(x+b)(1-\sigma(x-b)))/(\sigma(b)(1-\sigma(-b)))$ is equal to $(\sigma(x+b)-\sigma(x-b))/(\sigma(b)-\sigma(-b))$. If more conditions are present for the same feature x probabilities should be summed over all contributions. If conditions for other, independent features are present in a rule, probabilities should be multiplied. To obtain the same probabilities as those from the Monte Carlo procedure, for more than two conditions probabilities for combinations of pairs of conditions should be subtracted. Instead of the number of misclassifications the error function may include a sum over all probabilities:

$$E(M,s_x) = \frac{1}{2}\sum_X \sum_i (p(C_i|X;M) - \delta(C(X),C_i))^2 \tag{6}$$

where M includes intervals defining linguistic variables and dispersions s_x. Probabilistic confusion matrix allows for optimization of Eq. (2) using gradient-based methods. This minimization may be performed directly or may be presented as a neural network problem with special network architecture. Dispersions s_x of the values of features are additional adaptive parameters that may be optimized. We have used so far a very simple optimization with all s_x taken as a percentage of the range of feature x to perform one dimensional minimization of the error function independently of other steps. This approach to soft optimization may be used with any crisp logical rules to overcome the brittleness problem and to obtain robust wide margin rule-based classifiers.

5 Illustrative Applications

Wisconsin breast cancer data.

We have already analyzed the Wisconsin cancer dataset before giving detailed comparison with other results [1]. It contains 699 instances, with 458 benign (65.5%) and 241 (34.5%) malignant cases. Although the values of the 9 attributes are quantized in the range 1-10 we may assume that they are continuos and imprecisely measured. The simplest set of rules obtained from optimization includes only two rules for malignant class:

$$f_2 \geq 7 \vee f_7 \geq 6$$

The confusion matrix, including the ELSE condition for the benign class, is: $P = \begin{pmatrix} 215 & 10 \\ 26 & 448 \end{pmatrix}$, and the overall accuracy is 94.9%. Using rules in such form will immediately show their brittleness. If $f_2 \geq 7$ then any vector sampled from the left side of Gaussian distribution centered at 7 will give an error, i.e. any uncertainty in the measurement sharply increases the number of errors. The simplest solution is to shift the intervals to 6.5 and 5.5 for example. The number of errors on the whole dataset is not changed since these rules have single conditions, but in several erroneously classified cases significant probabilities for alternative classifications are obtained. Higher accuracy set of five rules (percentage of correct classifications for each rule is given in parenthesis):

R_1: $f_2 < 6 \wedge f_4 < 3 \wedge f_8 < 8$ (99.8)%
R_2: $f_2 < 9 \wedge f_5 < 4 \wedge f_7 < 2 \wedge f_8 < 5$ (100)%
R_3: $f_2 < 10 \wedge f_4 < 4 \wedge f_5 < 4 \wedge f_7 < 3$ (100)%
R_4: $f_2 < 7 \wedge f_4 < 9 \wedge f_5 < 3 \wedge f_7 \in [4,9] \wedge f_8 < 4$ (100)%
R_5: $f_2 \in [3,4] \wedge f_4 < 9 \wedge f_5 < 10 \wedge f_7 < 6 \wedge f_8 < 8$ (99.8)%

give 99.00% overall accuracy with the confusion matrix $P = \begin{pmatrix} 240 & 1 \\ 6 & 452 \end{pmatrix}$. Assuming uncertainties of up to 5% times the data range the same accuracy is achieved. After this operation one benign vector misclassified as malignant has only 54% probability of belonging to the wrong class. Unfortunately the 6 misclassified malignant cases have almost 100% confidence to belong to the wrong class, perhaps indicating that the data is really noisy.

The Ljubliana cancer data.

The Ljubliana cancer data [8] contains 286 cases, of which 201 are no-recurrence-events (70.3%) and 85 are recurrence-events (29.7%). There are 9 attributes, with 2-13 different values each. A single logical rule for the recurrence-events:

NOT involved nodes=[0,2] $\wedge$ Degree malignant = 3

with ELSE condition for the second class gives over 77% accuracy in crossvalidation tests. This rule is easy to interpret: the number of involved nodes is bigger than 2 and the cells are highly malignant. More accurate and optimized rules:

R_1: Degree malignant = 3 $\wedge$ breast=left $\wedge$ node caps=yes
R_2: (Degree malignant = 3 $\vee$ breast=left) $\wedge$ age = NOT [30-49] $\wedge$ tumor size = [35-54]

give slightly higher accuracy of 78%. Since the dataset is small many different sets of rules may give similar accuracy. It would be hard to improve this simple result, most methods give significantly lower accuracies using more complex models. For example LERS, a machine learning

technique based on rough sets, gives after optimization almost 100 rules achieving only 69.4% of accuracy (below the majority rate).

Method	Accuracy %	Reference
C-MLP2LN, 2 rules	78.0	This paper
Assistant-86	78.0	Ref. [10]
CART	77.3	Ref. [11]
CART, PVM, C-MLP2LN single rule	76.2	Ref. [11]
Naive Bayes rule	75.9	Ref. [11]
MLP with backpropagation	71.5	Ref. [11]
AQ15	66-72	Ref. [12]
Weighted network	68-73.5	Ref. [13]
LERS (rough sets)	67.1	Ref. [9]
k-NN	65.3	Ref. [11]

Table 1. Results for the Ljubliana cancer dataset.

Psychometric data

The psychometric data was collected in the Academic Psychological Clinic of Nicholas Copernicus University and in several psychiatric hospitals around Poland. In the present version of our project computerized version of the Minnesota Multiphasic Personality Inventory (MMPI) test was used (hundreds of books and papers about this test exist, cf. review in [15]), consisting of 566 questions with yes/no/don't know answers. Questions range from general health problems and specific neurological symptoms to personal political and moral opinions. Questionnaries, collected from 750 people, including 100 people considered as "normal", were evaluated in a standard way computing 14 coefficients. These coefficients are often displayed as a histogram (called "a psychogram") allowing skilled psychologists to diagnose specific problems, such as neurosis, drug addiction or criminal tendencies. The first four coefficients are used for control, measuring consistency of answers or the number of "don't know" answers, allowing to find malingerers. The next 10 coefficients form clinical scales, developed to measure tendencies towards hypochondria, depression, hysteria, psychopathy, paranoia, schizophrenia etc. For example values between 70 and 80 in the hypochondria scale may be interpreted as "very strong worries about own health, leading to psychosomatic reactions". A large number of simplification schemes has been developed to make the interpretation of psychograms easier. They may range from rule-based systems derived from observations of characteristic shapes of psychograms, statistical discrimination functions, or systems using smaller number of aggregated coefficients.

Although many computerized versions of the MMPI test exist to assist in information acquisition, evaluation of results is still done by an experienced clinical psychologist. Our goal is to provide automatic psychological diagnosis. Rule based system is most desirable because a detailed interpretation, including description of personality type, may be assigned to each diagnosis. We have worked with different datasets containing up to 34 classes (normal, neurotic, drug

addicts, schizophrenic, psychopaths, organic problems, malingerers etc.) determined by expert psychologists. Our initial logical rules achieved about 93% accuracy on the whole set, increasing to over 95% after some optimization. For comparison we have also used C4.5 [14] decision tree, a very good classification system which may also generate logical rules. For this application C4.5 gives comparable accuracy but most rules use open intervals and therefore are harder to interpret. For different data (man, woman, mixed) sets of 50-60 rules were generated. On average 2.5 logical rules per class were derived, involving between 2 and 9 features. For most classes there were only a few errors and it is quite probable that they are due to the psychologists interpreting the psychogram data. The only exception is the class of organic problems, which leads to answers that are frequently confused with symptoms belonging to other classes. A typical rule has the form:

$$\text{If } Hy \in [72,88] \wedge Ps \in [74,80] \wedge Pt \in [81,90] \text{ Then Organic problem}$$

These rules are used with assumption about accuracy of the measurement in each of the scales corresponding to a Gaussian dispersion of about 5 units. Each rule has detailed interpretation associated with it. An expert system using these rules should be evaluated by clinical psychologist in the near future.

6 Discussion

We are sure that in all cases, independently of the final classifier used, it is advantageous to extract crisp logical rules. First, in our tests logical rules proved to be highly accurate; second, they are easily understandable by experts in a given domain; third, they may expose problems with the data itself. However, if the number of rules is too high or the accuracy of classification is too low other methods should be used. For small datasets logical rules may provide very good results but, due to a large statistical error which is inevitable for such datasets, only the most robust rules should be trusted.

In this paper complete methodology for construction and application of logical models of data has been presented. Although we use neural networks to obtain initial sets of rules the other two steps, optimization and application of rules, may be used independently from the extraction step. We prefer to start from crisp logical rules and fuzzify them by assuming uncertainties in the inputs, obtaining probabilities of classification.

Using the early version of our hybrid methodology simplest logical description for several benchmark problems was obtained [1]. For many medical datasets very simple and highly accurate results were obtained. Recently we have improved further our neural method of linguistic variable determination [16] obtaining good logical description of some datasets that were previously hard to analyze, such as the diabetes dataset. Considering the simplicity of decision borders and difficulties in finding the best coverings it is not quite clear why logical rules work so well. For example in the hypothyroid or the Wisconsin breast cancer case the accuracy is better than that of any other classifier. One reason for such performance of rule-based systems is due to the good control of the complexity of data representation. Another possible explanation for the medical data is that the classes labeled “sick” or “healthy” have really fuzzy character. If the doctors are forced to make yes-no diagnosis they may fit the results of tests to specific intervals, implicitly using crisp logical rules.

Acknowledgments

Support by the KBN, grant 8 T11F 014 14, is gratefully acknowledged. We would like to thank J. Gomuła and T. Kucharski for providing the psychometric data.

References

1. W. Duch, R. Adamczak, K. Grąbczewski, G. Żal. *Hybrid neural-global minimization logical rule extraction method for medical diagnosis support*, Intelligent Information Systems VII, Malbork, Poland, 15-19.06.1998, pp. 85-94
2. W. Duch, G.H.F. Diercksen. *Feature Space Mapping as a universal adaptive system*, Computer Physics Communication, 87: 341–371, 1995;
3. W. Duch, R. Adamczak, N. Jankowski. *Initialization of adaptive parameters in density networks*, 3rd Conf. on Neural Networks, Kule, Oct. 1997, pp. 99-104
4. W. Duch, R. Adamczak, N. Jankowski. *New developments in the Feature Space Mapping model*, 3rd Conf. on Neural Networks, Kule, Poland, Oct. 1997, pp. 65-70
5. W. Duch, R. Adamczak and K. Grąbczewski. *Constrained backpropagation for feature selection and extraction of logical rules*, in: Proc. of "Colloquia in AI", Łódź, Poland 1996, p. 163–170
6. W. Duch and N. Jankowski. *New neural transfer functions.* Applied Math. & Comp. Science 7 (1997) 639-658
7. W. Duch, R. Adamczak and K. Grąbczewski. *Extraction of logical rules from backpropagation networks.* Neural Processing Letters 7, 1-9 (1998)
8. C.J. Mertz, P.M. Murphy. *UCI repository of machine learning databases*, http://www.ics.uci.edu/pub/machine-learning-data-bases.
9. J.W. Grzymała-Busse, T. Soe. *Inducing simpler rules from reduced data.* Intelligent Information Systems VII, Malbork, Poland, 15-19.06.1998, pp. 371-378
10. G. Cestnik, I. Konenenko, I. Bratko. *Assistant-86: A Knowledge-Elicitation Tool for Sophisticated Users.* In: I. Bratko and N. Lavrac (Eds.) *Progress in Machine Learning*, pp. 31-45, Sigma Press 1987
11. S.M. Weiss, I. Kapouleas. *An empirical comparison of pattern recognition, neural nets and machine learning classification methods.* In: J.W. Shavlik and T.G. Dietterich, *Readings in Machine Learning*, Morgan Kauffman Publ, CA 1990
12. R.S. Michalski, I. Mozetic, J. Hong, N. Lavrac. *The Multi-Purpose Incremental Learning System AQ15 and its Testing Application to Three Medical Domains.* In: Proc. 5th National Conf. on AI, pp. 1041-1045, Philadelphia, PA: Morgan Kaufmann 1986
13. M. Tan, L. Eshelman. *Using weighted networks to represent classification knowledge in noisy domains.* Proc. 5th Intern. Conf. on Machine Learning, pp. 121-134, Ann Arbor, MI 1988
14. J.R. Quinlan. *C4.5: Programs for machine learning.* San Mateo, Morgan Kaufman 1993
15. J.N. Butcher, S.V. Rouse. *Personality: individual differences and clinical assessment.* Annual Review of Psychology 47, 87 (1996)
16. W. Duch, R. Adamczak and K. Grąbczewski. *Neural optimization of linguistic variables and membership functions*, ICONIP'99, Perth, Australia (submitted)