MACHINE LEARNING OF EXPERT DECISION OR SYSTEM BEHAVIOUR

Peter Otto

Institut für Automatisierungs- und Systemtechnik
Technische Universität Ilmenau
Gustav-Kirchhoff-Straße 1
D-98693 Ilmenau, Germany
Email: peter.otto@tu-ilmenau.de

Abstract: A fuzzy-modeling method for the emulation of expert decision behavior or for static as well as dynamic systems is presented. The input – output dataset of the system – or expert behavior is changed using fuzzy-sets into examples in linguistic form. These resulting examples build the fundament of the machine learning process for rule production (ID3). The fuzzy sets are optimized in order to minimize the mean square error between the model and the system output.

Keywords: Modeling, machine learning, decision trees, rules, fuzzy system.

1. INTRODUCTION

In many sophisticated, hardly comprehensible processes, which for different reasons can not be fully automatized, humans often require very long times, to make competitive decisions. Frequently such processes are very difficult to model and the determination of data dependencies for diagnosis, monitoring, control/regulation and decision-making is very complicated.

Since the data, which describe the behavior of such processes and the decisions of humans, are often very inaccurate and information can be seized partly from qualitative descriptions, fuzzy systems offer above all a feasible way for information acquisition and processing. It is possible to represent the relevant dependencies in form of rules, which at the same time contributes the understanding for humans. Therefore, the task is to optimally reproduce the process dependencies and/or the decisions of humans (fuzzy model).

This task can be solved effectively with methods for automatic knowledge acquisition (machine learning). These include cluster methods (Bezdek, 1981), multi-level methods (Vachkov, 1994) and the use of methods like ID3 or C4.5 from Quinlan (Srinivasan et al., 1993; Quinlan, 1992). They allow the production of rule-sets from large I/O-data sets. As a result, decision trees are produced, which include only significant dependencies as well as minimize the number of rules.

In this contribution a new method for the production of Fuzzy models for the above-mentioned tasks for static and dynamic dependencies, is presented.

2. RULE BASED MODELLING

2.1 Method and System Description

The decision behavior of an expert and the static and/or dynamic behavior of systems can be described by the following equations:

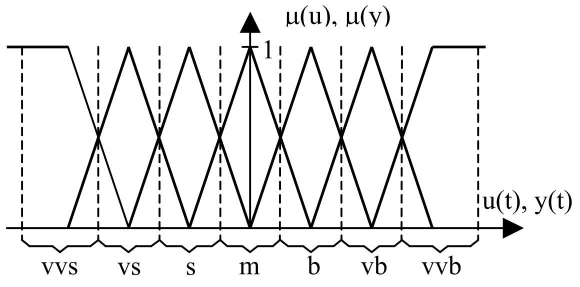
$$y = f(u_1, u_2, u_3, ..., u_n) \quad (1)$$

and

$$y(k) = f(y(k-1), ..., y(k-m), u(k-1), ..., u(k-n)) \quad (2)$$

where y is the output and $u_1, u_2, u_3, ..., u_n$ are the inputs of a static MISO-System,
y(k) is the output and y(k-1),..,y(k-m),u(k-1),.. ,u(k-n) are the sampled output and input values of a dynamic SISO-System and
f is a non-linear function.

The functional dependencies in equation (1) and equation (2) can be expressed in form of rules, if the input and output values of the system are described by linguistic attributes as a function of the measured values. In principle, there are two ways to determine the attributes for the inputs. In the first approach, the attributes are determined by dividing the entire range of the process inputs and outputs into a given number of *n* equal intervals. In the second approach only the output is divided into equal intervals, while a machine learning method optimally specifies the



v – very s – small m – middle b – big

Fig. 1. Signal Fuzzification

attribute borders for the inputs. For the fuzzification of the process variables triangular membership functions are assigned to the intervals in such a way that they intersect at the interval borders by $\mu = 0.5$. The edge intervals get one-side open membership functions (see Figure 1). Thus, a measured value with different membership values belong to two intervals. The measured values are then transformed according to the fixed intervals into linguistic attributes, so that a description of the static and/or dynamic expert and/or process behavior in form of linguistic expressions results (Table 1).

The ID3-algorithm (Quinlan 1992) generates an optimal decision tree from the linguistic examples. Production rules of the form

IF (u_1 = vs AND u_3 = b) THEN y = m and
IF (y(k-1) = vb AND u(k-2) = s) THEN y(k) = vvb,

which describe the static or dynamic system/expert

Table 1: Learning Examples for the ID3-Algorithm

Static	y(k)	u_1(k)	u_2(k)	u_3(k)	u_4(k)
Dynamic	y(k)	y(k-1)	y(k-2)	y(k-3)	y(k-4)
k=1	vvs	vb	b	vb	s
k=2	vvb	vvs	vb	b	vb
k=3	b	vvb	vvs	vb	b
k=4	s	b	vvb	vvs	vb
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.

behavior, can then be derived from the decision tree. The decision tree is built in such a way that the information content of the attributes for the decision-making process drops with increasing depth and the irrelevant attributes remain unconsidered.

2.2 Generation of the Rule-Set

In analogy to the classical fuzzy design, the data based system design is characterized by an unknown relation between the input and output variables (rule-base). Generation of the rule base is based on the analysis of the existing signals of both input and output variables. In the automatized design method presented here, the rule base is generated using the well-established ID3 algorithm according to (Quinlan 1992; Otto 1995). In this algorithm, the fuzzy sets characterizing the signal are understood to be signal-to-symbol transformations. Back transformation, i.e. defuzzification, is performed in the opposite direction. The principle of the ID3 algorithm is based on the generation of a decision tree (equations (3), (4)). After starting the procedure with a primary configuration, the generated fuzzy sets are linked in the rules, and the latter are then checked for their "correctness" (mean information content).

$$H\left(C/a_j\right) = \sum_{i=1}^{N} P\left(X_i\right) H\left(C/X_i\right) \Rightarrow \min \quad (3)$$

This serves to determine those (if ... then...) rules, whose mean information content is maximum.

$$H\left(C/X_i\right) = -\sum_{j=1}^{n} P\left(C_j/X_i\right) \log_2 P\left(C_j/X_i\right) \quad (4)$$

$H(C/a_j)$	– entropy of occurence of the fuzzy set C_j for the input a_j
$P(C_j / X_i)$	– probability of occurence of the fuzzy set C_j of the output variable, provided that a class X_i of fuzzy sets exists

n	– number of fuzzy sets of the output variable
N	– number of equivelent classes of examples with the same fuzzy sets being active at the model input
a_j	– model input, where uncertainty (entropy) is smallest

In analogy to information processing, the entropy H expresses the expected value, i.e. it may be used as a criterion for evaluating the information content of the information source. After having calculated these minimums, the relations are deleted as are the relations with equivalent classes, where $H(C/X_i) = 0$. This procedure is repeated several times and the relations are combined in the decision tree. The ID3-algorithm infers decision trees by growing them from the root downward, greedily selecting the next best attribute for each new decision branch added to the tree.

ID3 searches a complete hypothesis space (i.e., the space of decision trees can represent any discrete-valued function defined over discrete-valued instances). It thereby avoids the major difficulty associated with approaches that consider only restricted sets of hypothesis: that the target function might not be present in the hypothesis space. The inductive bias implicit in ID3 includes a preference for smaller trees; that is, it searches through the hypothesis space, grows the tree only as large as needed in order to classify the available training examples adequately. Because the training examples are only a sample of all possible instances, it is possible to add branches to the tree that improve performance on the training examples while decreasing performance on other instances outside this set. Post-pruning permits to avoid overfitting as described in section 2.5 (Mitchell, 1997).

Due to the characteristics of this process, not all fuzzy set combinations are considered in the rule base. Consequently, the generated rule base is reduced compared to a complete one, as only those rules are applied, which are really required to simulate the process. The defuzzification is done using the center of gravity method, whereby the individual terms are determined by the minimum inference method.

2.3 Optimization of the Membership Functions

Since, for the production of the rules, homogeneous membership functions were used for all attributes, a further improvement of the fuzzy models by using variable membership functions is expected. In addition, the optimization of the fuzzy sets is done so that the mean square error between model and system output is minimized:

$$Q = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \qquad (5)$$

where: $\hat{y}_i$ – model output y_i – system output
n – number of patterns

For this purpose, the membership functions of the input characteristics are described by four pivot places, so that arbitrary, rectangle, triangle and/or trapezoidal membership functions can be defined. The other conditions are that maximally 2 attributes are assigned to a measured value and the sum of the membership functions $\mu(u)$ is equal to one. The parameters of these membership functions are therefore given by the start and end points of the intervals of the full membership (roof points η_1, η_2 ...,η_{2n-2}). The bottom (foot) points are equal to the points of the roof of the membership functions of the neighboring attributes (see Figure 2). As a result 2(n-1) parameters, η_1, η_2 ...,η_{2n-2} must be optimized for a fuzzy set with n membership functions.

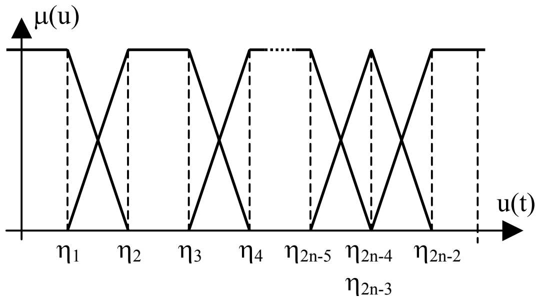


Fig. 2. Membership Functions of an Input Variable

This is a constrained non-linear optimization problem. To solve this problem, the Powell-Method (Powell, 1964) as well as the optimization method using evolutionary algorithm proposed by (Schwefel, 1977) are applied.

The method, which determines the optimal attribute borders for the inputs, does not change the number of attributes during the optimization process, so that the determined rules are also fully valid after the optimization of the membership functions. With the method, which divides the inputs into equally large attribute ranges, the allocation of the measured values to the linguistic descriptions is no longer valid after the optimization of the membership functions due to the shift of the originally selected attribute borders, so that the rules determined from it do not have to be optimal likewise any longer. For this reason the entire procedure of the rule determination and fuzzy set optimization must be repeated, until no further improvement in quality can be registered. Hereby, the optimized membership functions of the preceding step are used as initial membership functions for the next learning step. According to experience, 2-4 iterations are sufficient, in order to determine the optimal fuzzy model.

2.4 Determination of the number of output classes

During the evaluation of the performance of the models, one can find that the mean square error continuous decreases for the learning data with increasing number of output classes n. However, the complexity of the models also enlarges simultaneously by enlargement of the number of rules. A heuristic criterion is employed for the definition of the initial classes here in the form of the predicted squared error (PSE) for independent data which are not used for training. The PSE is given by:

$$\mathrm{PSE}=\mathrm{FSE}+\mathrm{KP} \qquad (6)$$

where FSE is the fitting squared error of the model on the training data and KP is a complexity penalty. Figure 3 shows the relationship between the FSE, PSE, and KP.

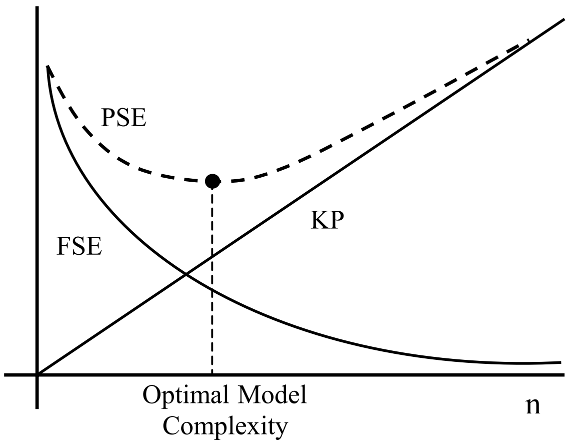


Fig. 3. Determination of the number of output classes

The complexity penalty, KP, is determined by the equation:

$$\mathrm{KP}=\mathrm{CPM}\times\frac{n}{N}S^2 \qquad (7)$$

where n, N and S^2 are determined by the database of training examples and CPM is a adjustable Complexity Penalty Multiplier. n is the number of output classes, N is the number of training data and S^2 is an a-priori estimate of the true unknown model error variance. The number of output classes for which the PSE becomes minimal is designated as optimal.

2.5 Treatment of noise in the data

The ID3 algorithm attempts always to classify all examples of the learning data correctly, independent of the strength of the noise that is contained in the data. As a result, the complexity of the decision tree increases and the classification rate for not learned examples decreases (Overfitting). For avoidance of this "Overfitting", a cutting of the tree (Pruning) is necessary. The success of cutting is checked at a separate test dataset. The pruning contains the following steps:

1. Consider each of the decision nodes in the tree to be candidates for pruning.
2. Removing the subtree rooted at that node, making it a leaf node, and assigning it the most common classification of the training examples affiliated with that node.
3. Nodes are removed only if the resulting pruned tree performs no worse than the original over the test dataset.
4. Nodes are pruned iteratively, always choosing the node whose removal most increases the decision tree accuracy over the test dataset.
5. Pruning of nodes continues until further pruning is harmful.

The impact of reduced-error pruning on the accuracy of the decision tree is illustrated in Figure 4.

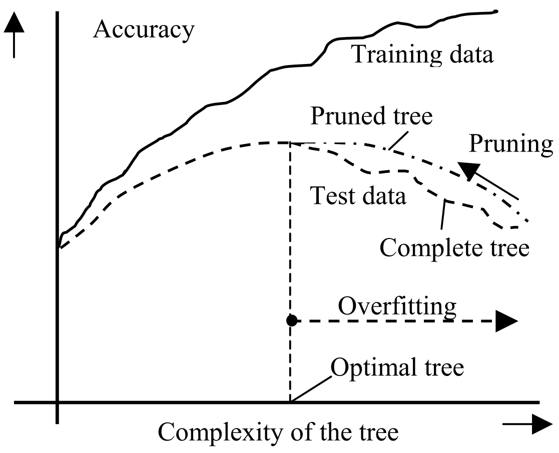


Fig. 4. Determination of the optimal complexity of the tree

One quite successful method for finding high accuracy hypothesis is a technique called rule post-pruning (Mitchell, 1997). Rule post-pruning involves the following steps:

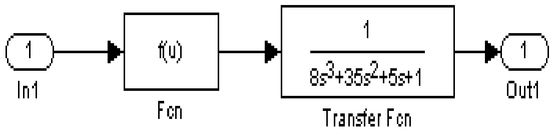
1. Infer the decision tree from the training set, growing the tree until the training data is fit as well as possible and allowing overfitting to occur.
2. Convert the learned tree into an equivalent set of rules by creating one rule for each path from the root node to a leaf node.
3. Prune (generalize) each rule by removing any preconditions that result in improving its estimated accuracy.
4. Sort the pruned rules by their estimated accuracy, and consider them in this sequence when classifying subsequent instances.

2.6 Simulation Results

The above described algorithm for the fuzzy concept was realized in the program system FuzzyOpt (Dung et al. 1997) (equal large attribute ranges for the inputs) and FuzzyMod (optimal attribute borders), which can both be used separately or in combination with the Fuzzy Control Design Toolbox for MATLAB® (Koch et al. 1996). Here the function of the system is going to be demonstrated on an example of a simulated nonlinear dynamic system with Hammerstein-structure:

Simulink®-Model where:

$T_S = 1s$; $-1 \leqslant u \leqslant 1$; ($T_S$ = Sampling time)



non-linearity: $f(u) = 1.36*u-2.06*u^3+1.7*u^5$

Fig. 5. Simulated non-linear System

The fuzzy model was determined from a training sequence with 2500 randomly distributed input values with the method, which classifies the model input values optimally and automatically. The number of output classes was preset to 75. The optimal number of the output classes was determined with an independent test-data set by increasing stepwise the number of classes until the mean square error reached a minimum.
A part of the generated rule set is represented in the following:

1. IF (y(k-1) = Term_3_0) THEN y(k) := Term_0 ;
2. IF (y(k-1) = Term_3_1) THEN y(k) := Term_1 ;
3. IF (y(k-1) = Term_3_2) THEN y(k) := Term_2 ;
4. IF (y(k-1) = Term_3_3) THEN y(k) := Term_0 ;
5. IF (y(k-1) = Term_3_4) AND (y(k-2) = Term_4_0) THEN y(k) := Term_3;
6. IF (y(k-1) = Term_3_4) AND (y(k-2) = Term_4_1 OR Term_4_2 OR Term_4_3 OR Term_4_4 OR Term_4_5 OR Term_4_6 OR Term_4_7 OR Term_4_8 OR Term_4_9 OR Term_4_10 OR Term_4_11 OR Term_4_12 OR Term_4_13 OR Term_4_14 OR Term_4_15 OR Term_4_16 OR Term_4_17 OR Term_4_18 OR Term_4_19 OR Term_4_20 OR Term_4_21 OR Term_4_58 OR Term_4_59 OR Term_4_60 OR Term_4_61 OR Term_4_62 OR Term_4_63 OR Term_4_64 OR Term_4_65 OR Term_4_66 OR Term_4_67) THEN y(k) := Term_1 ; …..

526. IF (y(k-1) = Term_3_228) THEN y(k) := Term_74 ;

Altogether 526 rules were generated.

In Figure 6 a comparison of the simulated and the output values estimated by the fuzzy model for an independent test sequence, is presented.

A number of practical applications have shown that the presented identification procedure can be applied to various problems, including modelling the decision behaviour of human experts.

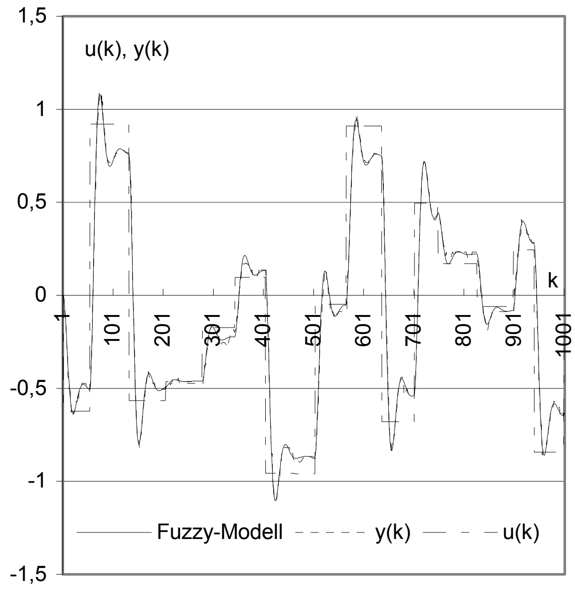


Fig. 6. Comparison of the Simulated and Estimated Output Values

Figure 7 shows a part of the determined optimal membership functions.

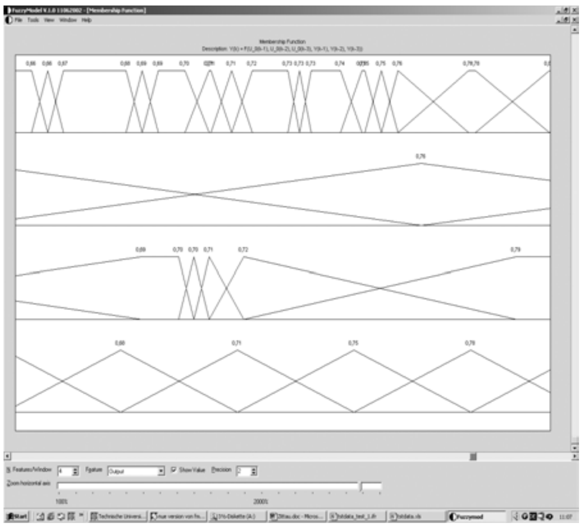


Fig. 7. Part of the Determined Optimal Membership Functions

3. CONCLUSION

This paper has shown that fuzzy models can describe nonlinear static and dynamic systems very well. By using machine learning methods (ID3-algorithm proposed by Quinlan), the declarative knowledge in form of the rule base can be determined problem-free. The structure of such systems also allows for an interpretation of the interactions between the system variables.

A further improvement of the model is possible by an additional optimization of the membership functions. Since, in the method with equally large intervals for the inputs, the arrangement of the measured values to the attributes can change, an iterative approach is necessary in this case. This leads after approx. 2 to 4 optimization steps for the given number of attributes to an "optimal" fuzzy model in the sense of the minimization of the mean square error between model and system output.

This method has the advantage that ascertaining the rules is limited to the part of state space relevant to the description of system behavior so that a combinatorial explosion which is possible in the case of other methods is avoided. The generated rule base is reduced compared to a complete one, as only those rules are applied, which are really required to simulate the process.

The running time of the algorithm is only proportional to the number of the training examples. For systems with stronger noises, an overfitting can be avoided by pruning of the decision tree or of the rules.

The described algorithm for Fuzzy modeling was implemented in the program system FuzzyOpt (Dung et al. 1997) (equal large attribute ranges for the inputs) and FuzzyMod (optimal attribute borders), which can both be used separately or in combination with the Fuzzy Control Design Toolbox for MATLAB® (Koch et al. 1996).

Further examples of the application of the Fuzzy modeling method introduced here are given in (Koch *et al.*, 1995; Heine, 1999; Malberg *at al.*, 2001; Malberg *at al.*, 2002; Mönch and Otto, 2002).

REFERENCES

Bezdek, J.C. (1981). *Pattern Recognition with Objective Function Algorithm.* Plenum Press, New York.

Dung, L.T., M. Koch and P. Otto. (1997). FuzzyOpt– ein Werkzeug zum Entwurf optimaler Fuzzy-Systeme. *at* **45** 11, 555-556.

Heine, K. (1999). Beschreibung von Deformationsprozessen durch Volterra- und Fuzzy-Modelle sowie Neuronale Netze, *DEUTSCHE GEODÄTISCHE KOMMISSION bei der Bayerischen Akademie der Wissenschaften,* Reihe C Dissertationen Heft Nr. 516, Verlag der Bayerischen Akademie der Wissenschaften, München.

Koch, M., P. Otto and J. Wernstedt (1995). Entwurf von Talsperrensteuerungen auf der Grundlage von Lernverfahren *at* **43** 6, 305-315.

Koch, M., Th. Kuhn and J. Wernstedt. (1996). *Fuzzy Control.* R. Oldenbourg Verlag München Wien.

Malberg, H., P. Otto, A. Voss, J. Walther and J. Wernstedt (2001). Fuzzy-System zur Darstellung des Einflusses genetischer Veränderungen auf die autonome Regulation des Herz-Kreislauf-Systems, AUTOMED 2001, *3. Workshop Automatisierungstechnische Methoden und Systeme für die Medizin, 17.-18. September 2001 an der Ruhr-Universität Bochum, Beiträge*, 64-65.

Malberg, H., P. Otto, T. Walther and J. Wernstedt. (2002). Identifikation der Baroreflex-Dynamik mit Hilfe eines Fuzzy-Modells.“, *at* 50 05, 212-219.

Mitchell, M.M. (1997). *Machine Learning.* McGraw-Hill Companies, Inc.

Mönch, L. and P. Otto (2002). Scheduling Jobs on Parallel Batch Processing Machines Using Dispatching Rules and Machine Learning Techniques. *4th Middle East Symposium on Simulation and Modeling* (A.-A. Marwan, Ed.), 192-196, SCS Publication.

Otto, P. (1995). Fuzzy-Modelling of Nonlinear Dynamic Systems by Inductive Learned Rules. *Proceedings of the Third European Congress on Intelligent Techniques and Soft Computing*, Vol. 2, 858-864, EUFIT `95, Aachen.

Powell, M.J.D. (1964). An efficient method for finding variables without calculating derivatives. *Computer Journal* , **7**, 155-162.

Quinlan, J.R. (1992). *C 4.5. Programs for Machine Learning.* Morgan Kaufmann Publishers, San Mateo, California.

Schwefel, H.P. (1977). *Nichtlineare Optimierung von Computermodellen mittels Evolutionsstrategie.* Birkhäuser Basel und Stuttgart.

Srinivasan, A., C. Batur and Ch.Ch.Chan (1993). Using Inductive Learning to Determine Fuzzy Rules for Dynamic Systems”. *Engng. Applic. Artif. Intell.*, **Vol. 6**, No. 3, 257-264, Pergamon Press Ltd.

Vachkov, G. (1994). Multilevel Fuzzy Rule Based Modeling”. *Proceedings of the Second European Congress on Intelligent Techniques and Soft Computing*, **Vol. 1**, 240-245. EUFIT `94, Aachen.