BEYOND NEURO–FUZZY: PERSPECTIVES AND DIRECTIONS

Detlef Nauck
Department of Computer Science, Technical University of Braunschweig
Bueltenweg 74 – 75, D–38106 Braunschweig, Germany
Tel. +49.531.391–3155, Fax: +49.531.391–5936, email: nauck@ibr.cs.tu-bs.de

Abstract

The interest in neuro–fuzzy systems has grown tremendously over the last few years. First approaches concentrated mainly on neuro–fuzzy controllers, whereas newer approaches can also be found in the domain of data analysis. After successful applications in Japan neuro–fuzzy concepts also find their way into the European industries, though mainly simple models, like FAMs, still prevail. This paper shortly reviews some modern neuro–fuzzy concepts. After this a generic neuro–fuzzy model is presented, that serves a foundation for specific derived neuro–fuzzy applications, this is shown with a model for neuro–fuzzy data analysis, which we see as an important perspective for the neuro–fuzzy domain. The paper concludes with some thoughts on further research directions that go beyond simple neuro–fuzzy control applications.

1 Introduction

Neuro–fuzzy systems enjoyed an ever growing popularity over the recent years. The first approaches were mainly neuro–fuzzy controllers, but newer approaches can also be found in the domain of data analysis. Because there are so many models, and because the term "neuro–fuzzy" is often used inconsiderately some definitions of terms are necessary.

In this paper a "neuro–fuzzy" or "neural fuzzy" system is a combination of neural networks and fuzzy systems in such a way that neural networks, or neural learning algorithms, are used to determine parameters of fuzzy systems. This means, the main intention of a neuro–fuzzy approach is to automatically create or improve a fuzzy system by means of neural network methods. An even more important aspect is that the system should always be interpretable in terms of fuzzy if–then rules, because it is based on a fuzzy system reflecting vague knowledge.

On the other hand a "fuzzy neural network" is a neural network that uses fuzzy methods to learn faster, or to perform better. In this case the improvement of a neural network is the main intention. An interpretation in terms of fuzzy rules is neither important nor possible here, because the system is based on a neural network with black box characteristics. The term "fuzzy–neuro system" should be avoided, because it is not clear which of both approaches it refers to.

We propose the following taxonomy to describe different combinations of neural networks and fuzzy systems (for further details see [Nauck, 1994b; Nauck and Kruse, 1994a]):

- **Fuzzy Neural Networks:** This is not a neuro–fuzzy approach. Fuzzy methods are use to enhance the learning capabilities, or the performance of a neural network. This can be done, by using fuzzy rules to change the learning rate like in [Halgamuge et al., 1994], or by creating a network, that works with fuzzy inputs like in [Narazaki and Ralescu, 1991].
- **Concurrent "Neuro–Fuzzy Models":** A neural network, and a Fuzzy–System work together on the same task, but without influencing each other, i.e. neither system is used to determine parameters of the other one. Usually the neural network pre–processes the inputs to, or post–processes the outputs

This paper appears in the proceedings of the Third European Congress on Intelligent Techniques and Soft Computing (EUFIT'95), Aachen, August 28 – 31, 1995, pp. 1159–1164.

from the fuzzy system. These kind of models are strictly speaking neither real neuro–fuzzy approaches, nor are they fuzzy neural networks.

- **Cooperative Neuro–Fuzzy Models:** A neural network is used to determine the parameters (rules, rule weights and/or fuzzy sets) of a fuzzy system. After the learning phase, the fuzzy systems works without the neural network. This is a simple form of neuro–fuzzy systems, and the most simple form – determining rule weights by neural learning algorithms – is most widely used, even though semantical problems can arise as it is e.g. shown in [Nauck and Kruse, 1994a]. Cooperative models can be further divided into approaches that: a) learn fuzzy sets off–line, b) learn fuzzy rules off–line, c) learn fuzzy sets on–line, d) learn rule weights. See [Nauck, 1994b; Nauck and Kruse, 1994a] for details.
- **Hybrid Neuro–Fuzzy Models:** Modern Neuro–Fuzzy Approaches are of this form. A neural network, and a fuzzy system are combined into one homogeneous architecture. The system may be either interpreted as a special neural network with fuzzy parameters, or as a fuzzy system implemented in a parallel distributed form. Some examples for this kind of architecture are ANFIS [Jang, 1993], FuNe [Halgamuge and Glesner, 1993; Halgamuge and Glesner, 1994], Fuzzy Rule Net [Tschichold-Gürman, 1995], GARIC [Berenji and Khedkar, 1992], NEFCLASS [Nauck and Kruse, 1995a], and NEFCON [Nauck, 1994a; Nauck and Kruse, 1993; Nauck and Kruse, 1994b]. Some of these approaches are reinforcement learning types that are especially suited for control tasks (GARIC, NEFCON), and others are multi–purpose models (ANFIS, FuNe, Fuzzy Rule Net, NEFCLASS), which use supervised learning.

In Japan neuro–fuzzy concepts were applied early [Asakawa and Takagi, 1994], and now they also find their way into the European industries [Nauck and Kruse, 1995b], though mainly simple models, like FAMs (fuzzy associative memory [Kosko, 1992]), still prevail. But there is no sign, that neuro–fuzzy approaches have grown to their full potential already. Until now the most applications were considered in the domain of fuzzy control, i.e. adapting a fuzzy controller until it performs sufficiently. These applications are usually only interested in the optimization capabilities of neuro–fuzzy models. But there is more to it: Neuro–fuzzy systems can also be viewed as a tool for knowledge acquisition. This aspect becomes especially clear in the domain of data analysis, where e.g. fuzzy rules describing a data set are of immediate interest.

2 The Fuzzy Perceptron

To compare different neuro–fuzzy approaches, or to derive special neuro–fuzzy applications with sound semantics, a generic model is useful. It defines the common foundations of neuro–fuzzy models, and defines some basic properties. We call our generic model *fuzzy perceptron*, because the basic structure resembles a 3–layer perceptron, and can also be extended to have more layers.

A fuzzy perceptron displays the architecture of an usual multilayer perceptron, but the weights are modelled as fuzzy sets and the activation, output, and propagation functions are changed accordingly. The intention of this model is to be interpretable in form of linguistic rules and to be able to use prior rule based knowledge, so the learning has not to start from scratch. In [Nauck and Kruse, 1995a] we used this generic model to derive the NEFCLASS model (NEuro Fuzzy CLASSification) which is shortly presented in the next section. The fuzzy perceptron can also be used to derive the NEFCON model [Nauck, 1994a; Nauck and Kruse, 1993; Nauck and Kruse, 1994b] for neuro–fuzzy control applications.

Definition 1 *A* **3–layer fuzzy perceptron** *is a 3–layer feedforward neural network* $(U, W, \mathrm{NET}, A, O, \mathrm{ex})$ *with the following specifications:*

(i) $U = \bigcup_{i \in M} U_i$ *is a non–empty set of units (neurons) and* $M = \{1, 2, 3\}$ *is the index set of* U*. For all* $i, j \in M, U_i \neq \emptyset$ *and* $U_i \cap U_j = \emptyset$ *with* $i \neq j$ *holds.* U_1 *is called input layer,* U_2 *rule layer (hidden layer), and* U_3 *output layer.*

(ii) The structure of the network (connections) is defined as $W : U \times U \rightarrow \mathcal{F}(\mathbb{R})$*, such that there are only connections* $W(u, v)$ *with* $u \in U_i,\ v \in U_{i+1} (i \in \{1, 2\})$ *(*$\mathcal{F}(\mathbb{R})$ *is the set of all fuzzy subsets of* $\mathbb{R}$*).*

(iii) A defines an activation function A_u *for each* $u \in U$ *to calculate the activation* a_u

(a) for input and rule units $u \in U_1 \cup U_2$*:* $A_u : \mathbb{R} \rightarrow \mathbb{R}, \quad a_u = A_u(\mathrm{net}_u) = \mathrm{net}_u,$

(b) *for output units* $u \in U_3$*:* $A_u : \mathcal{F}(\mathbb{R}) \to \mathcal{F}(\mathbb{R})$, $a_u = A_u(\mathrm{net}_u) = \mathrm{net}_u$.

(iv) O *defines for each* $u \in U$ *an output function* O_u *to calculate the output* o_u

(a) *for input and rule units* $u \in U_1 \cup U_2$*:* $O_u : \mathbb{R} \to \mathbb{R}$, $o_u = O_u(a_u) = a_u$,

(b) *for output units* $u \in U_3$*:* $O_u : \mathcal{F}(\mathbb{R}) \to \mathbb{R}$, $o_u = O_u(a_u) = \mathrm{DEFUZZ}_u(a_u)$, *where* DEFUZZ_u *is a suitable defuzzification function.*

(v) NET *defines for each unit* $u \in U$ *a propagation function* NET_u *to calculate the net input* net_u

(a) *for input units* $u \in U_1$*:* $\mathrm{NET}_u : \mathbb{R} \to \mathbb{R}$, $\mathrm{net}_u = ex_u$,

(b) *for rule units* $u \in U_2$*:* $\mathrm{NET}_u : (\mathbb{R} \times \mathcal{F}(\mathbb{R}))^{U_1} \to [0,1]$, $\mathrm{net}_u = \underset{u' \in U_1}{\top} \{W(u',u)(o_{u'})\}$, *where* $\top$ *is a t–norm,*

(c) *for output units* $u \in U_3$*:* $\mathrm{NET}_u : ([0,1] \times \mathcal{F}(\mathbb{R}))^{U_2} \to \mathcal{F}(\mathbb{R})$, $\mathrm{net}_u : \mathbb{R} \to [0,1]$, $\mathrm{net}_u(x) = \underset{u' \in U_2}{\perp} \{\top(o_{u'}, W(u',u)(x))\}$, *where* $\perp$ *is a t–conorm.*

If the fuzzy sets $W(u',u)$, $u' \in U_2$, $u \in U_3$, *are monotonic on their support, and* $W^{-1}(u',u)(\tau) = x \in \mathbb{R}$ *such that* $W(u',u)(x) = \tau$ *holds, then the propagation function* net_u *of an output unit* $u \in U_3$ *can alternatively be defined as*

$$\mathrm{net}_u(x) = \begin{cases} 1 & \text{if } x = \dfrac{\sum_{u' \in U_2} o_{u'} \cdot m(o_{u'})}{\sum_{u' \in U_2} o_{u'}} \\ 0 & \text{otherwise} \end{cases}$$

with $m(o_{u'}) = W^{-1}(u',u)(o_{u'})$. *To calculate the output* o_u *in this case* $o_u = x$, *with* $net_u(x) = 1$ *is used instead of (iv.b).*

(vi) $\mathrm{ex} : U_1 \to \mathbb{R}$, *defines for each input unit* $u \in U_1$ *its external input* $\mathrm{ex}(u) = \mathrm{ex}_u$. *For all other units* ex *is not defined.*

A fuzzy perceptron can be viewed as a usual 3–layer perceptron that is "fuzzified to a certain extent". Only the weights, the net inputs, and the activations of the output units are modelled as fuzzy sets. A fuzzy perceptron is like a usual perceptron used for function approximation. The advantage lies within the interpretation of its structure in the form of linguistic rules, because the fuzzy weights can be associated with linguistic terms. The network can also be created partly, or in the whole, out of linguistic (fuzzy if–then) rules.

For this reason the fuzzy perceptron has been defined as a 3–layer feedforward network, but one can also think of models where the output layer inputs to an additional hidden (rule) layer, after which a new output layer is placed. This way fuzzy perceptrons can be build with an odd number of hidden layers that alternately have different functionalities. Such a model could be interpreted as a system of cascaded linguistic rules, and it would also allow to create a system consisting of several modules for different tasks.

3 Beyond Simple Neuro–Fuzzy Models: Data Analysis

Simple Neuro–Fuzzy Approaches like e.g. FAMs are used to quickly optimize a fuzzy system, usually a fuzzy controller. In this case only the performance of the fuzzy controller is considered, but semantical aspects are overlooked. This can lead to "optimized fuzzy systems" that cannot be interpreted anymore in terms of fuzzy rules [Nauck and Kruse, 1992].

More recent approaches in neuro–fuzzy control like e.g. NEFCON [Nauck, 1994a; Nauck and Kruse, 1993] take the semantical aspects of optimizing fuzzy sets or rules into account, and use special techniques to constraint the changes inflicted by a learning algorithm. Neuro–fuzzy approaches, where the semantical aspects are even more important, can be found in the domain of data analysis. In this case a set of fuzzy rules have to be found to describe a data set, e.g. classification. In this domain models like FuNe [Halgamuge and Glesner, 1994], Fuzzy Rule Net [Tschichold-Gürman, 1995] and NEFCLASS [Nauck and Kruse, 1995a] can be found.

NEFCLASS (**NE**uro **F**uzzy **CLASS**ification) especially concentrates to find a small but sufficient number of fuzzy rules to classify patterns, so the obtained classification can be easily interpreted. NEFCLASS is derived from the fuzzy perceptron. The exact definition of the NEFCLASS model can be found in [Nauck and Kruse, 1995a]. Due to lack of space, we will only present some main features.

The patterns to be classified are vectors $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ and a class C is a (crisp) subset of $\mathbb{R}^n$. We assume the intersection of two different classes to be empty. The pattern feature values are represented by fuzzy sets, and the classification is described by a set of linguistic rules. For each input feature x_i there are q_i fuzzy sets $\mu_1^{(i)}, \ldots, \mu_{q_i}^{(i)}$, and the rule base contains k fuzzy rules $R_1, \ldots, R_k$.

The rule base is an approximation of an (unknown) function $\varphi : \mathbb{R}^n \rightarrow \{0,1\}^m$ that represents the classification task where $\varphi(\mathbf{x}) = (c_1, \ldots, c_m)$ such that $c_i = 1$ and $c_j = 0$ ($j \in \{1, \ldots, m\}, j \neq i$), i.e. $\mathbf{x}$ belongs to class C_i. Because of the mathematics involved the rule base actually does not approximate φ but the function $\varphi' : \mathbb{R}^n \rightarrow [0,1]^m$. We will get $\varphi(\mathbf{x})$ by $\varphi(\mathbf{x}) = \psi(\varphi'(\mathbf{x}))$, where ψ reflects the interpretation of the classification result obtained from a NEFCLASS system. In our case we will map the highest component of each vector $\mathbf{c}$ to 1 and its other components to 0, respectively.

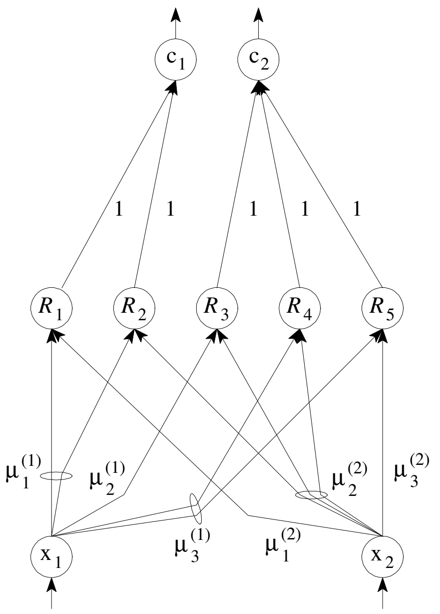

Figure 1: A NEFCLASS system with two inputs, five rules and two output classes

The fuzzy sets and the linguistic rules which perform this approximation, and define the resulting NEFCLASS system, will be obtained from a set of examples by learning. The Fig. 1 shows a NEFCLASS system that classifies input patterns with two features into two distinct classes by using five linguistic rules.

Its main feature are the *shared weights* on some of the connections. This way we make sure, that for each linguistic value (e.g. "x_1 is *positive big*") there is only one representation as a fuzzy set (e.g. $\mu_1^{(1)}$ in Fig. 1), i.e. the linguistic value has only one interpretation for all rule units (e.g. R_1 and R_2 in Fig. 1). It cannot happen that two fuzzy sets that are identical at the beginning of the learning process develop differently, and so the semantics of the rule base encoded in the structure of the network is not affected [Nauck and Kruse, 1992]. Connections that share a weight always come from the same input unit. The connections from the rule units to the output unit are fixed at unity. We refrain from using rule weights, because they would destroy the semantics of the model.

A NEFCLASS system can start with no hidden units (rules) at all, and it begins with equally distributed membership functions to partition the domains of the input variables. The learning algorithm scans the training set, and includes a rule unit each time a pattern cannot be classified correctly. The antecedent fuzzy sets are chosen such that each input feature of the current pattern obtains the highest possible membership degree. The conclusion of the rule is determined by the class of the current pattern. After all patterns have been processed, the number of rules is reduced, by finding the best rules, and deleting all other rules. The "best" rules are found by a given criterion, e.g. order the rules by the number of correct classifications they are involved in, and keep the first n rules.

After this the fuzzy sets are optimized by a supervised learning algorithm that is derived from the fuzzy error backpropagation algorithm already used for the NEFCON model [Nauck, 1994a; Nauck and Kruse, 1993]. The algorithm is explained in detail in [Nauck and Kruse, 1995a] and can be summarized as follows: After the current pattern has been processed, for each rule the antecedent fuzzy set is changed that is responsible of the activation of the rule node. The changes are done in a way, that these fuzzy sets will produce a higher membership value (or lower one in case of misclassification), when the same pattern is propagated again. There are constraints on the changes to the fuzzy sets, such that a fuzzy set may not pass a neighbor, that a fuzzy set stays within the domain, etc.

The idea of neuro–fuzzy systems like NEFCLASS is to find a small number of fuzzy rules that can correctly classify a set of patterns, by using a local neural like learning algorithm, that causes only local changes in the system. Small local changes make it easy to remain in control of the architecture, and to apply constraints that ensure proper interpretability of the learning result. In addition the learning procedure is computationally inexpensive. NEFCLASS can apply prior knowledge in form of fuzzy rules and fuzzy sets, and it is possible not to learn some parts of the system, e.g. rules that are known for sure, and which must not be changed. On the other hand NEFCLASS can only be applied to supervised learning problems, i.e. the classification for the training data must be known.

Some results of the performance of NEFCLASS can be found in [Nauck and Kruse, 1995a], and the respective software can be obtained at ftp://ftp.ibr.cs.tu-bs.de/pub/local/nefclass/nefclass.zip.

4 Directions and Perspectives

There are two future perspectives for neuro–fuzzy models. For neuro–fuzzy control models the range of applications in Europe and the USA will surely rise. Taking into account that the "fuzzy wave" hit Europe and the USA a few years after Japan, it is obvious that Japanese applications of neuro–fuzzy controllers will be copied by other countries in the very near future. Major manufacturers of fuzzy development tools begin to include, or already have included neuro–fuzzy approaches in their products. The most important aspect of neural fuzzy control is easy and fast optimization of a fuzzy system by using data samples. The interpretation of the learning result is usually not considered, because one is only interested in a working application. But it is still important to use semantically sound models, which are well understood. Neuro–fuzzy control models derived from the fuzzy perceptron like NEFCON [Nauck, 1994a; Nauck and Kruse, 1993] are a step in this direction.

The other perspective that is more important to the ongoing research in neuro–fuzzy systems is data analysis or data mining. Neuro–fuzzy systems can be used to obtain fuzzy rules and fuzzy sets from a database of samples. In this case the result of the data analysis, e.g. a classification result is not the most important thing, but the obtained knowledge. The classification result may not be optimal, because of the semantical constraints applied to the learning algorithm that are necessary to obtain an interpretable neuro–fuzzy system. Other data analysis methods may provider better results, but they are usually not interpretable in a linguistic way, and cannot use prior knowledge.

Future research in neuro–fuzzy systems should concentrate on the knowledge acquisition aspect. For this it is very important to have a well defined architecture. The above mentioned fuzzy perceptron [Nauck and Kruse, 1995a] can serve as a generic model, and allows to derive new models or to compare existing ones. Neuro–fuzzy models may differ in the constraints on their architectures (e.g. shared weights) and on their learning algorithms (e.g. fuzzy sets may not pass each other), but they have the same goal: to provide a framework with clear semantics that is used to transfer data into knowledge.

Local neural learning algorithms allow us to keep track of the changes in the system, if they are properly constrained. It is therefore advisable to keep the learning procedures and architectures simple. This is even more important when more complex neuro–fuzzy systems perhaps consisting of several modules arranged in a hierarchical manner (e.g. a network of NEFCLASS systems) have to be considered, because the problems to be solved become more difficult.

5 Conclusions

In this paper we have argued that neuro–fuzzy research will go beyond simple neuro–fuzzy control applications. The "neuro" aspect is provided by local neural–network–like learning algorithms, and it is often supported by transferring a fuzzy system in a feed-forward network structure. In future systems the optimization capabilities will be less prominent, because data analysis and data mining, i.e. the transformation from raw data into knowledge will be more and more important.

Neuro–fuzzy systems are capable of using prior knowledge in form of fuzzy rules, adapting it to the actual situation reflected in the data, and creating new additional fuzzy rules from the data. So they are an excellent tool for knowledge acquisition. The possible applications are numerous: medical diagnosis, credit worthiness, quality insurance, etc. – all this are areas, where not only the result is of interest, but more important the way it was reached, so it can be checked for plausibility.

For future neuro–fuzzy systems in data analysis domains it will also be important to be able to process missing values, and crisp information. Their combination with symbolic knowledge processing [Kasabov, 1995] will probably also be of major interest in future research.

References

K. Asakawa and H. Takagi (1994). *Neural Networks in Japan.* Communications of the ACM, 37(3):106–112.

H. R. Berenji and P. Khedkar (1992). *Fuzzy Rules for Guiding Reinforcement Learning.* In *Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'92)*, pages 511–514, Mallorca.

S. K. Halgamuge and M. Glesner (1993). *The Fuzzy Neural Controller FuNe II with a New Adaptive Defuzzification Strategy Based on CBAD Distributions.* In *Proc. First European Congress on Fuzzy and Intelligent Technologies (EUFIT93)*, pages 852–855, Aachen.

S. K. Halgamuge and M. Glesner (1994). *Neural Networks in Designing Fuzzy Systems for Real World Applications.* Fuzzy Sets and Systems, 65:1–12.

S. K. Halgamuge, A. Mari and M. Glesner (1994). *Fast Perceptron Learning by Fuzzy Controlled Dynamic Adaption of Network Parameters.* In R. Kruse, J. Gebhardt and R. Palm, eds.: *Fuzzy Systems in Computer Science*, pages 129–139. Vieweg, Wiesbaden.

J.-S. R. Jang (1993). *ANFIS: Adaptive-Network-Based Fuzzy Inference Systems.* IEEE Trans. Systems, Man & Cybernetics, 23:665–685.

N. K. Kasabov (1995). *Hybrid Fuzzy Connectionist Rule-based Systems and the Role of Fuzzy Rules Extraction.* In *Proc. IEEE Int. Conf. on Fuzzy Systems 1995*, pages 49–56, Yokohama.

B. Kosko (1992). *Neural Networks and Fuzzy Systems. A Dynamical Systems Approach to Machine Intelligence.* Prentice-Hall, Englewood Cliffs.

H. Narazaki and A. L. Ralescu (1991). *A Synthesis Method for Multi-Layered Neural Network using Fuzzy Sets.* In *IJCAI-91: Workshop on Fuzzy Logic in Artificial Intelligence*, pages 54–66, Sydney.

D. Nauck (1994a). *Building Neural Fuzzy Controllers with NEFCON–I.* In R. Kruse, J. Gebhardt and R. Palm, eds.: *Fuzzy Systems in Computer Science*, Artificial Intelligence, pages 141–151. Vieweg, Wiesbaden.

D. Nauck (1994b). *Fuzzy Neuro Systems: An Overview.* In R. Kruse, J. Gebhardt and R. Palm, eds.: *Fuzzy Systems in Computer Science*, Artificial Intelligence, pages 91–107. Vieweg, Wiesbaden.

D. Nauck and R. Kruse (1992). *Interpreting Changes in the Fuzzy Sets of a Self-Adaptive Neural Fuzzy Controller.* In *Proc. Second Int. Workshop on Industrial Applications of Fuzzy Control and Intelligent Systems (IFIS'92)*, pages 146–152, College Station, Texas.

D. Nauck and R. Kruse (1993). *A Fuzzy Neural Network Learning Fuzzy Control Rules and Membership Functions by Fuzzy Error Backpropagation.* In *Proc. IEEE Int. Conf. on Neural Networks 1993*, pages 1022–1027, San Francisco.

D. Nauck and R. Kruse (1994a). *Choosing Appropriate Neuro-Fuzzy Models.* In *Proc. Second European Congress on Fuzzy and Intelligent Technologies (EUFIT94)*, pages 552–557, Aachen.

D. Nauck and R. Kruse (1994b). *NEFCON–I: An X–Window based Simulator for Neural Fuzzy Controllers.* In *Proc. IEEE Int. Conf. Neural Networks 1994 at IEEE WCCI'94*, pages 1638–1643, Orlando.

D. Nauck and R. Kruse (1995a). *NEFCLASS – A Neuro–Fuzzy Approach for the Classification of Data.* In K. George, J. H. Carrol, E. Deaton, D. Oppenheim and J. Hightower, eds.: *Applied Computing 1995. Proc. of the 1995 ACM Symposium on Applied Computing, Nashville, Feb. 26–28*, pages 461–465. ACM Press, New York.

D. Nauck and R. Kruse (1995b). *Neuro–Fuzzy Systems Research and Applications outside of Japan (in Japanese).* In M. Umano, I. Hayashi and T. Furuhashi, eds.: *Fuzzy–Neural Networks (in Japanese)*, Soft Computing Series. Asakura Publ., Tokyo. To be published.

N. Tschichold-Gürman (1995). *Generation and Improvement of Fuzzy Classifiers with Incremental Learning using Fuzzy RuleNet*. In K. George, J. H. Carrol, E. Deaton, D. Oppenheim and J. Hightower, eds.: *Applied Computing 1995. Proc. of the 1995 ACM Symposium on Applied Computing, Nashville, Feb. 26–28*, pages 466–470. ACM Press, New York.