ON MODELLING OF A FUZZY CONTROLLER BY MEANS OF TWO VERSIONS OF MULTILAYER FEEDFORWARD NEURAL NETWORKS

U. Brunsmann, E. Czogala[*], and H. von Koch
Fachhochschule Wurzburg-Schweinfurt
Germany

Abstract

Selected problems on modelling a fuzzy controller by means of multilayer feedforward neural networks have been analysed in this paper. Based on simulation results, the performance of two networks with different structures of input layers have been shown. The advantages and disadvantages of the above presented modelling have been pointed out.

Keywords: fuzzy controller, neural network

[*] On leave from the Technical University of Silesia

1. Introduction

Just as in [4], we will consider here a fuzzy controller as a system that processes fuzzy information. Let us assume that our fuzzy controller contains a collection of fuzzy control rules of the form

$$R_r: \text{ If Error} = A_j \text{ and Change in Error} = B_k \text{ then Control Action} = C_p \qquad (1.1)$$

for $1 \le r \le K$.

A_j, B_k, C_p are linguistic values (fuzzy sets) for variables Error, Change in Error and Control Action defined in universes of discourse $\mathbf{X}$, $\mathbf{Y}$, and $\mathbf{Z}$, respectively.

Let as point out here an explicite connective 'and' between variables Error and Change in Error and an implicit sentence connective 'also' which links all the rules into a knowledge base.

Because our further considerations are based on a discretization of the rules, let us choose the same as in [4], discretized intervals $[a_1,b_1]$, $[a_2,b_2]$ and $[a_3,b_3]$ for supports of all the A_j, B_k and C_p in $\mathbf{X}$, $\mathbf{Y}$ and $\mathbf{Z}$, respectively. So we have

1. $x_i = a_1 + i\,(b_1 - a_1)/N_1 \quad (0 < i < N_1)$ for a positive integer N_1

2. $y_i = a_2 + i\,(b_2 - a_2)/N_2 \quad (0 < i < N_2)$ for a positive integer N_2

3. $z_i = a_3 + i\,(b_3 - a_3)/N_3 \quad (0 < i < N_3)$ for a positive integer N_3

A fuzzy control rule is usually implemented by a fuzzy implication (a fuzzy relation in $\mathbf{X} \times \mathbf{Y} \times \mathbf{Z}$):

$$R_r = (A_j \text{ and } B_k) \dashrightarrow C_p \qquad (1.2)$$

where (A_j and B_k) may be interpreted as a fuzzy set $A_j \times B_k$ in $\mathbb{X} \times \mathbb{Y}$.

Given input information: A'(error) and B'(change in error), the control action C' can be deduced employing the compositional rule of inference, the definitions of fuzzy implication and connectives 'and'and 'also'. Even if we choose a particular compositional rule of inference, fuzzy implication and both connectives 'and' and 'also', the inference process can still be realized in different ways. Namely, if we consider input information (error and change in error) as vectors, we shall write the compositional rule of inference in the form:

$$C' = B' \circ (A' \circ R) \tag{1.3}$$

where R is the global relation obtained by connection of all rules (e.g. $R = \bigcup_{r=1}^{K} R_r$)
Alternatively we can use the cartesian product matrix as input information and apply the following formula:

$$C' = (B' \times A') \circ R \tag{1.4}$$

Taking into account for example sup-min as composition operations, min for implication, min for 'and' and max for 'also' connectives, we get the same inference result from both formulas (1.3) and (1.4). A different selection of the operations may produce different inference results.

The motivation for this paper is to show that the above mentioned two approaches to the inference process in a fuzzy controller lead to the construction of two versions of multilayer feedforward neural networks which can be trained to model such a fuzzy controller.

2. The structure of two versions of neural networks modelling a fuzzy controller

Considering the discretization presented in the first section and two versions of input information for a fuzzy controller we will construct now two respective versions of multilayer feedforward neural networks. In order to compare the performance of the networks we will use the same structure of all layers with the exception of the input layers. The structure of the input layer is considered to be linear for the vector version and rectangular for the matrix version of neural network (Fig.1). Let us now then consider each version separately.

Vector version

According to formula (1.3) and the discretization from the first section, this version will have $m_1 = N_1 + N_2 + 2$ input neurons. The number of output neurons is given by $n = N_3 + 1$. Taking into account the number of hidden layers as well as the numbers of neurons in them considered as $h_1, h_2, \dots$, we can annotate the structure of the whole network as

$$m_1 - h_1 - h_2 - \dots - n$$

The training input-output pairs (v_r, t_r) are of the form

$$v_r = (A_j(x_0), \dots, A_j(x_{N_1}), B_k(y_0), \dots, B_k(y_{N_2}))$$

$$t_r = (C_p(z_0), \dots, C_p(z_{N_3}))$$

for $1 \leq r \leq K$.

The above given training pairs represent knowledge contained in fuzzy rules of a fuzzy controller.

Matrix version

According to formula (1.4) and using the same discretization as in the vector version, a matrix version will have $m_2 = (N_1 + 1)(N_2 + 1)$ input neurons. Assuming the same number of output neurons i.e. $n = N_3 + 1$ and taking into account the number of hidden layers as well as the numbers of neurons in them considered as $h_1, h_2, \dots$, we can also annotate the structure of the whole network as

$$m_2 - h_1 - h_2 - \dots - n$$

The training input-output pairs (q_r, t_r) are of the form

$$q_r = \begin{bmatrix} A_j(x_0) \wedge B_k(y_0), \dots, A_j(x_{N_1}) \wedge B_k(y_0) \\ \vdots \qquad\qquad \vdots \\ A_j(x_0) \wedge B_k(y_{N_2}), \dots, A_j(x_{N_1}) \wedge B_k(y_{N_2}) \end{bmatrix}$$

$$t_r = (C_p(z_0), \dots, C_p(z_{N_3}))$$

for $1 \leq r \leq K$ (symbol ^ stands for min operation). Analogically as in the previous version, the pairs (q_r, t_r) represent knowledge contained in the fuzzy rules of a fuzzy controller.

3. Simulation results

We now apply our two versions of multilayer feedforward neural networks to model a fuzzy controller which successfully controls a return fine hopper level at the sintering plant [2,4]. The fuzzy controller is specified by the rule table presented in Table 1 and the definition of the respective linguistic values is given in Fig. 2.

Table 1

Rule	R1	R2	R3	R4	R5	R6	R7	R8	R9
Error	vl	sl	sl	me	me	me	sh	sh	vh
Change in Error	sn	sn	sp	ne	ze	po	sn	sp	sp
Control Action	ne	ne	sn	sn	ze	sp	sp	po	po

The following abbreviations have been used in Table 1

vl - very low, sl - small low, me - medium, sh - small high, vh - very high, ne - negative, sn - small negative, ze - zero, sp - small positive, po - positive.

Just as in [1] and [4] we have chosen here 29 numbers x_i in [0.20,0.90] for Error and 17 numbers y_i in [-4,4] for Change in Error. The x_i and y_i numbers are equally spaced at their respective intervals. This means that we have for the vector version $m_1 = 46$ input neurons and for the matrix version $m_2 = 493$ input neurons. For the Control Action, 25 numbers z_i in [-3,3] have been taken. It means that the output layers of both networks have also 25 neurons. Additionally one hidden layer with 25 neurons has been introduced.

In order to train and test both versions of networks, California Scientific Software's BrainMaker Professional v 1.50 has been used. The training process was carried out in two stages.

First, both networks were trained using only rules as input information. Taking into account the values of the training process parameters : training tolerance - 0.01, testing tolerance - 0.01, backpropagation learning rate - 0.6 and backpropagation smoothing factor - 0.3, the learning process has been finished for the vector version by 1314 rounds (9 patterns for each round) and for the matrix version by 658 rounds. Both trained networks do not react properly on singleton pattern, however the generalization property allows both of them correctly to react on input

information noticably changed in comparison with trained patterns. At the second stage additional information originating from singletons was introduced. This information was limited to 181 rules where in each of them Error, Change in Error and Control Action are singletons. So both versions of our networks were trained using 9 + 181 = 190 rules. The training and testing tolerance parameters were the same as in the previous stage i.e. 0.01. However the backpropagation learning rate and smoothing factor at the begining were taken 1 and 0.9 respectively and after getting halted in a local minimum, they were changed to 0.6 and 0.3 respectively. The learning process has been finished for the vector version by 1567 rounds (190 patterns for each round) and for the matrix version by 639 rounds. In this case both networks react properly on respective singletons.
Summerizing, let us point out a significant difference in numbers of rounds learned by each version. An explanation of that phenomena needs deeper treatment of problems connected with fuzzy logic controller and neural networks.

4. Concluding remarks

The above presented considerations show that the structure of multilayer feedforward neural network modelling a fuzzy controller is not unique. Two versions of such neural networks have been described. The simulation results also show that the number of rounds is smaller for the vector version than for the matrix version; that can be explained by the fact that for the matrix version (more weights) it is easier to distribute respective information. Both presented versions do not react properly on singletons patterns. In order considerably to improve performance such networks should be trained with additional rules

in which conditions and conclusions are singletons.
It should also be pointed out here that however the equivalence theorem of neural networks and fuzzy controllers exists [7], the problem of practical realization of such neural network learned only using fuzzy rules and reacting correctly on singletons (in the similar way as a fuzzy controller does) remains still open.

Acknowledgements

The second named author was found his guest lectureship at the Fachhochschule Wurzburg-Schweinfurt stimulating and would like to express his gratitude to the President Prof. Dr. W. Fechner for making it all possible.

References

1. J. Buckley, E. Czogala, Linguistic Approximate Reasoning Inference Engine for Expert Systems (LARIEES), Proceedings of the World Congress on Expert Systems, Orlando, Florida, December 16-19, 1991, 108-115.

2. J. Buckley, E. Czogala, Fuzzy models, fuzzy controllers, and neural nets, in: McAllister (ed.), Frontiers of Applied Mathematics, SIAM, Philadelphia, (to appear).

3. Y. Hayashi, J. Buckley, E. Czogala, Fuzzy expert systems versus neural networks, IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'92, March 8-12, 1992, San Diego, California.

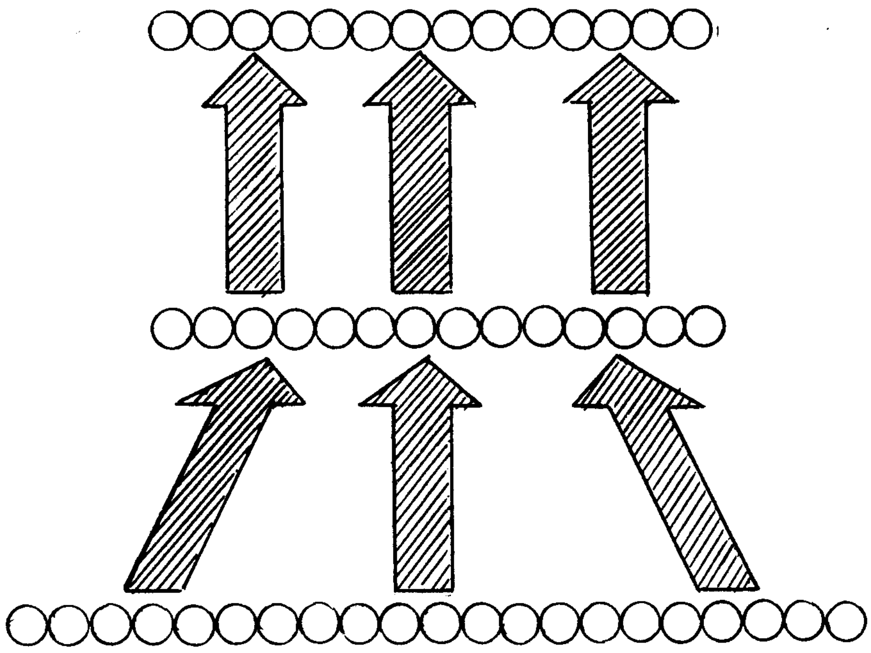
4. Y. Hayashi, E. Czogala, J. Buckley, Fuzzy neural controller, IEEE International Conference on Fuzzy Systems, FUZZ-IEEE'92, March 8-12, 1992, San Diego, California.

5. Y. Hayashi, J. Buckley, E. Czogala, Applications of fuzzy neural networks, IEEE International Conference on Systems Engineering, September 17-19, Kobe (Japan).

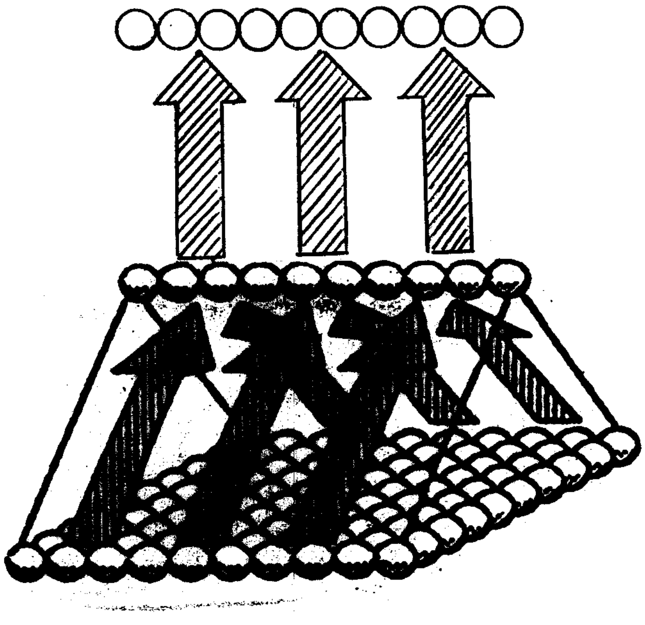
6. Y. Hayashi, J. Buckley, E. Czogala, Fuzzy neural network, 2nd International Conference on Fuzzy Logic and Neural Networks (IIZUKA'92), July 17-22, 1992.

7. J. Buckley, Y. Hayashi, E. Czogala, On the equivalence of neural nets and fuzzy expert systems, International Joint Conference on Neural Networks, IJCNN'92, Baltimore (USA).

8. E. Czogala, W. Cholewa, Uncertainty treatment in Fuzzy Production System of CC_SHELL, Bulletin for Studies and Exchanges on Fuzziness and its Applications (BUSEFAL) No. 48, October 1991.

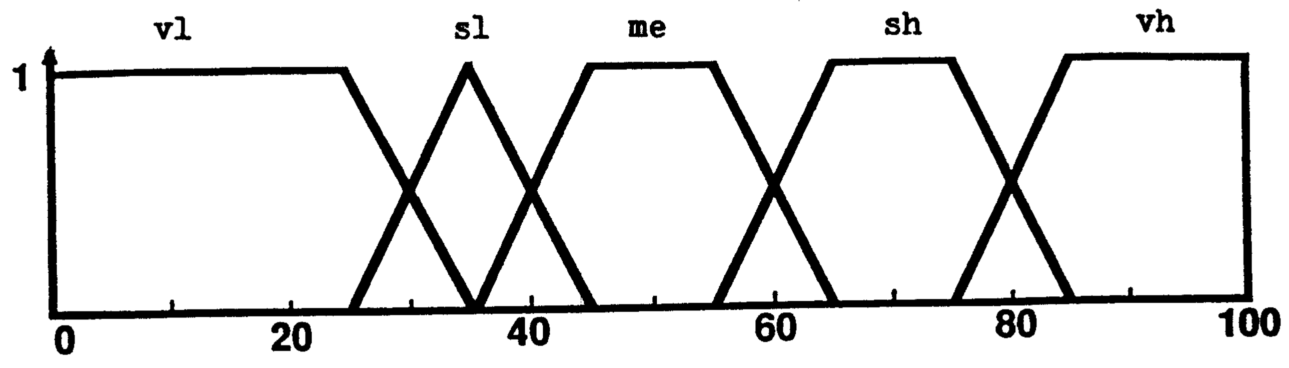


Vector structure of neural network

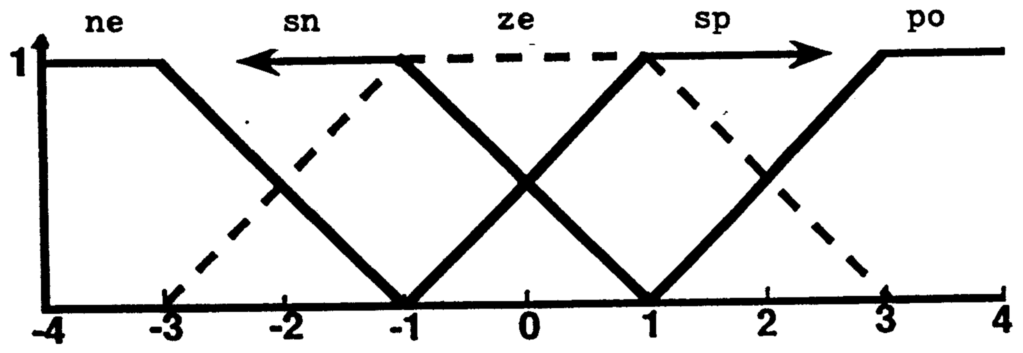


Matrix structure of neural network

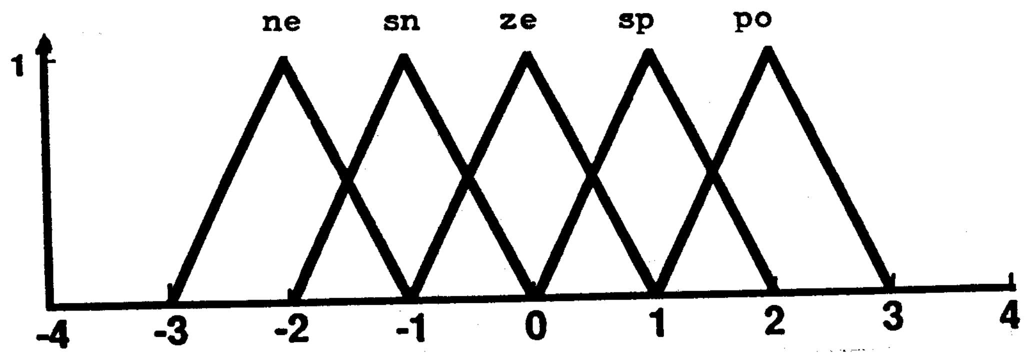
Fig. 1
Two structures of multilayer feedforward neural network
modelling a fuzzy controller



Fuzzy Sets for Error



Fuzzy Sets for Change in Error



Fuzzy Sets for Control Action

Fig. 2

Fuzzy Sets for Error, Change in Error and Control Action