Recent Advances on the Implementation of Fuzzy Systems Using Artificial Neural Networks

Athanasios K. Tsadiras and Konstantinos G. Margaritis

Department of Applied Informatics

University of Macedonia

Egnatias 156, P.O. Box 1591

GR-54006 Thessaloniki

GREECE

Basil G. Mertzios

Department of Electrical Engineering

Democritus University of Thrace

GR-67100 Xanthi

GREECE

Abstract: Various methods of implementing Fuzzy Systems using Artificial Neural Networks are presented. After a brief introduction to fuzzy systems, the approaches to their implementation are compared and their potentials discussed. Trends of the area are also given.

Athanasios K. Tsadiras received his BSc degree in Physics (1991) from the Aristotle University of Thessaloniki, Greece, and the MSc degree in Applied Artificial Intelligence (1992) from University of Aberdeen, Scotland. He is currently pursuing his Ph.D degree at the Department of Applied Informatics of the University of Macedonia, Greece, under a scholarship from the Greek National Scholarship Foundation (I.K.Y.). His research interests include fuzzy logic, fuzzy systems, neural networks, expert systems and artificial intelligence. He is a member of the IEEE/IEEE Computer Society, Association of Computer Machinery and Greek Physics Society.

Basil G. Mertzios received the Diploma Degree in Electrical and Mechanical Engineering from the Aristotle University of Thessaloniki, Hellas in 1979, and the Ph.D degree in Electrical Engineering from the Democritus University of Thrace, Hellas in 1982, both with honours. Part of his Ph.D work has been done at University of Toronto. From 1986 to 1992 he was an Associate Professor at the Democritus University of Thrace, Department of Electrical Engineering and Director of the Laboratory of Automatic Control Systems, and from 1992 a Professor at the same University. His research interests span the fields of automatic control systems, robotics, multidimensional digital signal processing and neural networks.

Konstantinos G. Margaritis received his Diploma degree in Electrical Engineering (1983) from the Aristotle University of Thessaloniki, Greece. He received his MSc. degree in Theory and Applications of Computation (1985) and the Ph.D degree in Computer Studies (1988), both from Loughborough University of Technology, U.K. He is currently an Associate Professor at the Department of Applied Informatics, University of Macedonia. His research interests include parallel and distributed processing, parallel systolic algorithms, neural networks, fuzzy logic and genetic algorithms. He is a member of IEEE/IEEE Computer Society, Association of Computer Machinery, Greek Technical Chamber and Greek Computer Society.

Introduction to Fuzzy Logic and Fuzzy Systems

Fuzzy logic was introduced in 1969 by L.A. Zadeh [1] and is an extension of Boolean Logic. In Boolean Logic a proposition can be either True or False. This is a serious weakness because it cannot represent expressions such as "Almost True" or "Certainly False" that are frequently used by people. There are several degrees of truth and Boolean Logic cannot represent them. Fuzzy Logic provides the mathematical framework for representing these degrees of truth. Furthermore in Fuzzy Logic imprecisely defined classes can be defined. Such a class is called Fuzzy Set and is defined by its membership function. This function returns the degree to which an object belongs to the fuzzy set. The membership function can take any value within the interval [0,1] (from 0, if the object does not belong to the set, to 1 if it completely does).

Fuzzy sets can be considered as linguistic values that can be assigned to linguistic (or fuzzy) variables. For example, the fuzzy set "High" can be the linguistic value of the linguistic variable "Temperature". If X and Y are fuzzy variables and $\overline{A}$, $\overline{B}$ are fuzzy sets then a simple fuzzy if-then rule is of the form "If X is $\overline{A}$ then Y is $\overline{B}$". The way to reason from such a rule is given by Generalized Modus Ponens. In classical Modus

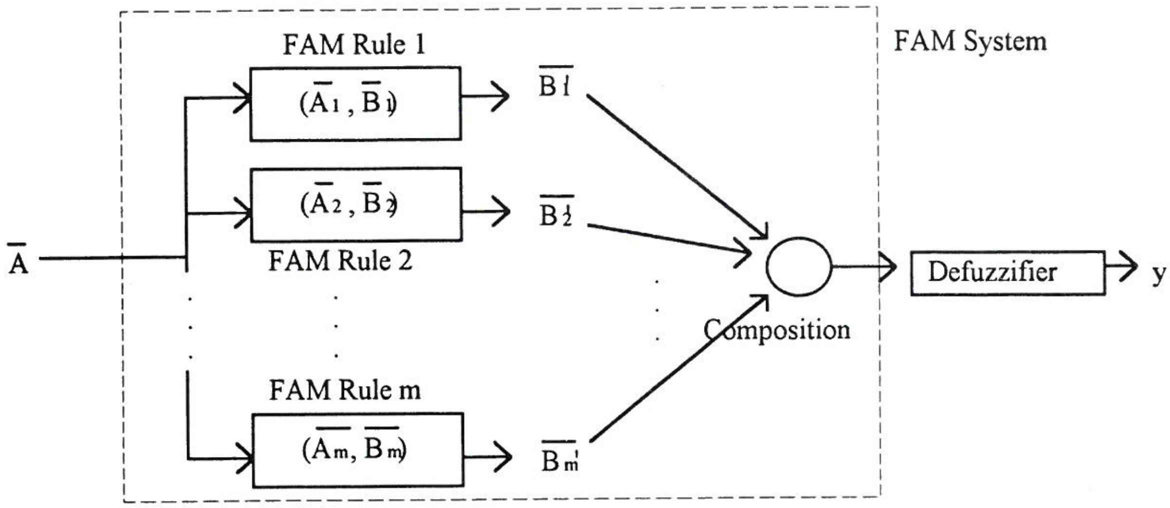

Figure 1. The Structure of a FAM System

Ponens the inference is done in the following way:

If x is a then y is b

x is a

We conclude that y is b

The weak point of this fundamental way of reasoning is that if x is slightly different from a then no conclusion can be drawn from the rule. In the extended Generalized Modus Ponens this weakness is removed. The form of this more powerful way of reasoning, follows:

If X is $\overline{A}$ then Y is $\overline{B}$

X is $\overline{A}'$

We conclude that Y is $\overline{B}'$

where $\overline{A}'$ and $\overline{B}'$ are slight modifications of $\overline{A}$ and $\overline{B}$ respectively. $\overline{B}'$ should be defined by $\overline{A}, \overline{B}$ and $\overline{A}'$. $\overline{B}'$ should resemble $\overline{B}'$ as much as $\overline{A}'$ resembles $\overline{A}$.

The fuzzy rule "If X is $\overline{A}$ then Y is $\overline{B}$" can be represented as $(\overline{A}, \overline{B})$. We can have m such rules $(\overline{A}_1, \overline{B}_1)$,..., $(\overline{A}_m, \overline{B}_m)$. A fuzzy system can be imagined as a Fuzzy Associative Memory (FAM) [2] (see Figure 1). Its behaviour is the following: Given an input fuzzy set $\overline{A}$, all m rules become active. Each rule R_i (i=1,...,m) depending on the resemblance between $\overline{A}$ and $\overline{A}_i$ will return a fuzzy set $\overline{B}_i'$ based on Generalized Modus Ponens. The desired behaviour is the more $\overline{A}$ resembles $\overline{A}_i$, the more $\overline{B}_i'$ resembles $\overline{B}_i$. The final output $\overline{B}$ is a composition of all $\overline{B}_i'$.

There are cases where the output should not be a fuzzy set but a crisp value that represents as good as possible the fuzzy set. This is done by a defuzzifier, which implements one of many defuzzification methods [3].

Implementation of Fuzzy Systems by Artificial Neural Networks

A fuzzy set is defined by its membership function and so every neural network that implements a fuzzy system has to have a method of representing the membership function. Various ways of representing a fuzzy set are given in Figure 2.

In Figure 2(a) the fuzzy set has been sampled at various points. Fuzzy set $\overline{A}$ is represented as a series of couples $(x_i, \mu_{\overline{A}}(x_i))$, where $\mu_{\overline{A}}$ is the membership function of fuzzy set $\overline{A}$. The rate of sampling can be higher in areas of the membership function where more detailed representation is required and low in other areas. This sampling method has the advantage of not requiring any convexity [4] of the membership function .

In Figure 2(b) the interval method is used. Now the membership function is represented as a series of α-cuts sets of the fuzzy set, where α-cuts are crisp sets defined as $A_\alpha = \left\{x \in X | \mu_{\bar{A}}(x) \geq \alpha\right\}$ [4]. Clearly, if we know an adequate number of á-cuts of the fuzzy sets at discrete points of the interval [0,1] we can restore the fuzzy set. Therefore a fuzzy set can be considered as a series of crisp sets and common set theory can be used. In case the fuzzy set is convex, every α-cut is a single interval. This feature simplifies the representing method.

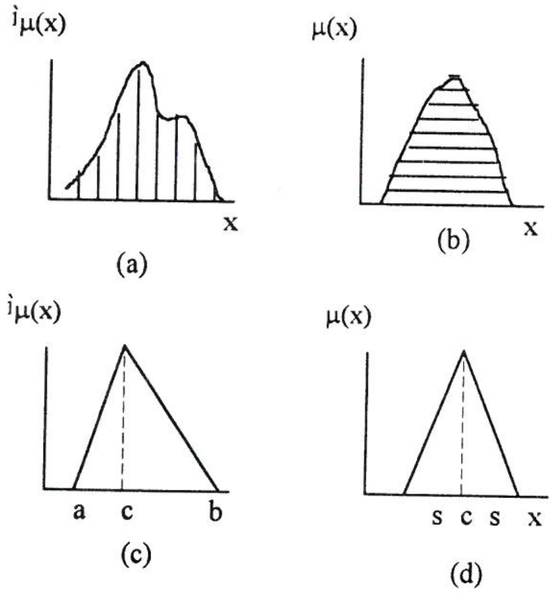

Figure 2. Methods of Representing Fuzzy Sets

The method in Figures 2(c), 2(d) is used only when the fuzzy set is normal (that is, at least at one point the membership function will get the value 1), and the shape of the membership function is triangular. In this case just the triplet (a,c,b) (see Figure 2c) is adequate. If the fuzzy system is also symmetric as in Figure 2(d) then just the couple (c,s) is enough to represent the fuzzy set.

Keller et al [5][6] use the first method of sampling the membership function to represent the fuzzy sets and propose a method of implementing fuzzy if-then rules using neural networks. Fuzzy sets as "Low", "Medium", "High", "Unknown" and most of their modifiers "Very", "Very-very"-(Very^2), "More-or-less" have been defined and sampled at 11 different points. A three-layer feedforward neural network has been used where the input layer is not fully connected to the hidden one, but the hidden one is fully connected to the output layer (Figure 3). A fuzzy rule of the form "If X is Low then Y is High" has been tried to be encoded. The standard backpropagation algorithm [7] has been used with the training set of Table I(a).

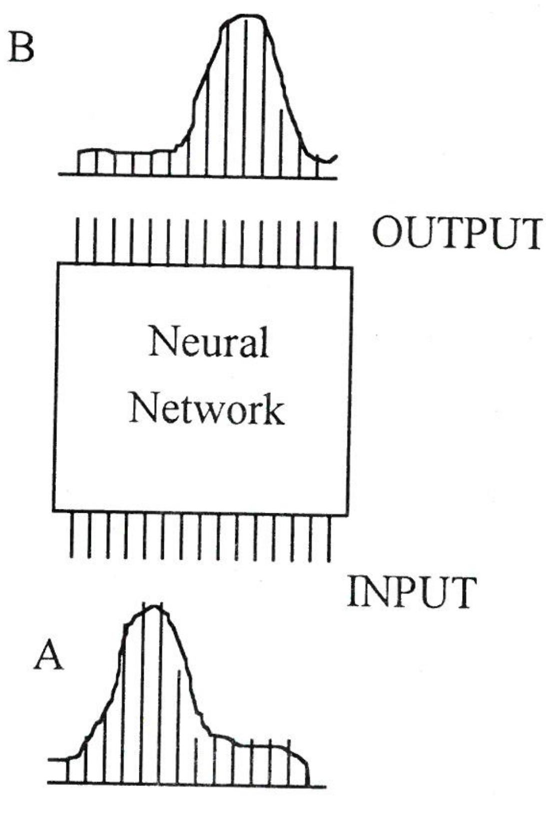

Figure 3. Implementation of Fuzzy If-Then Rules Sampling Fuzzy Sets

Table I(a)		Table I(b)	
Training Set		Testing Set	
Input	Output	Input	Expected Output
Low	High	Very^2 Low	Very^2 High
Very Low	Very High	$(\text{More-or-less})^2$ Low	$(\text{More-or-less})^2$ High
More-or-less Low	More-or-less High	Medium	Unknown
Not Low	Unknown	Noisy Low	High

The structure has been tested (a part of the testing set is shown at Table I(b)) and the system returned the expected outputs. It should be noticed that for disjunctive rules of the form "If X is $\overline{A}$ OR Y is $\overline{B}$ then Z is $\overline{C}$", a neural network with two hidden layers should be used, due to the increased complexity of the rule.

The interval method of a fuzzy set representation is current with Ishibuchi et al [8][9]. Interval arithmetic is used, that means a generalization of ordinary arithmetic for closed intervals. If X and Y are closed intervals then $X=[x^L,x^U]$ and $Y=[y^L,y^U]$.

Addition and multiplication are defined as $X+Y=[x^L,x^U]+[y^L,y^U]=[x^L+y^L,x^U+y^U]$ and

$$kX = k[x^L, x^U] = \begin{cases} [kx^L, kx^U] & \text{if } k \geq 0 \\ [kx^U, kx^L] & \text{if } k < 0 \end{cases}$$

(k, real number). In neural networks the logistic function $f(x) = \frac{1}{1+e^{-x}}$ is proper and for intervals

we have $f(X)=f([x^L,x^U])=[f(x^L),f(x^U)]$. For fuzzy rules of the form "If X is $\overline{A}$ AND Y is $\overline{B}$ then Z is $\overline{C}$" the interval arithmetic is suggested and the rule is implemented as shown in Figure 4. A three -layer feedforward neural network is used and it is trained with the standard backpropagation algorithm. At each time step, an interval of the fuzzy sets $\overline{A}$ and $\overline{B}$ is input into the neural network, producing an interval of the output fuzzy set $\overline{C}$. It should be mentioned that a real number x can be represented as the interval [x,x].

This means that both real number input-output pairs and fuzzy rules can be applied during the training process of the neural network. In other words, the conventional neural network that maps input-output pairs can be enhanced by fuzzy if-then rules, deduced from experience. The fuzzy rules express that if the input vector belongs to a specific area of the input space then the output vector should belong to a specific area of the output space.

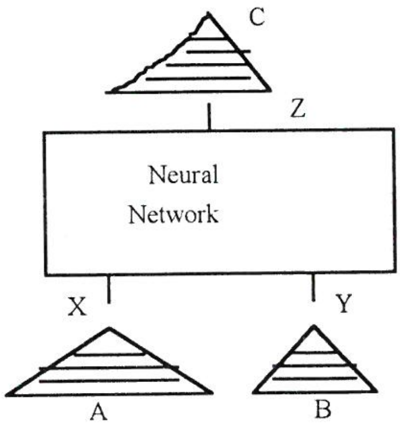

Figure 4. Using the Interval Method for Implementing Fuzzy If-Then Rules

Something that should be stressed is that the above methods are based on classical artificial neural networks , where weights and biases are real numbers. Another technique will consist in fuzzy neural networks [9][10] where weights and biases are fuzzy numbers (convex, normal fuzzy sets). Hayashi et al [10] opted for fuzzy neural networks and fuzzified the delta rule of the backpropagation learning algorithm. The drawback of this approach is that the derivatives required by the learning procedure are evaluated in a simplified way. During the differentiation fuzzy numbers are assumed to be real numbers. The Extension Principle [11] was used which determines how a function of real numbers can be extended to fuzzy numbers.

On the other hand, Ishibuchi et al [9] succeeded in overcoming this problem. With neural networks with fuzzy weights and biases still present , the interval method of representation was mainly applied. The Extension Principle was also involved and learning was performed by the backpropagation algorithm. The derivatives are now evaluated in a much more sophisticated way. A differentiation operated in the intervals used and the results of computer simulations were the following. If compared with neural networks with crisp weights and biases, fuzzy neural networks are better when the fuzziness of the fuzzy target output is greater than that of the input. Crisp neural networks are better when the fuzziness of the output is smaller than that of the input. The above behaviour can be explained if we notice that the existence of fuzzy numbers for weights and biases in the neural network increases the fuzziness of the inputs and leads to fuzzier outputs.

A different approach is made by Horikawa et al [12][13]. The structure of a neural network is designed in such a way that rule extraction and membership function tuning are performed. Depending on the type of output, three types of neural networks are defined. In the first type the output is a constant (e.g. class 1, class 2, ... for classification problems). In the second type the output is a first order linear equation function of the input values
$f(x_1, x_2, \ldots, x_n) = a_0 + a_1x_1 + a_2x_2 + \ldots + a_nx_n$

In the third type the output is a fuzzy set.

The success of this approach comes from the fact that the logistic function that neurons of artificial neural networks use, can be considered as the membership function of a fuzzy set (see Figure 5a). If f is the logistic function then it can be the membership function of the fuzzy set "Big". Moreover, 1-f can be the membership function of the fuzzy set "Small". Fuzzy set "Medium" can be produced by a composition of f and 1-f. The connection weights of the neural network are parameters that determine the characteristics of the membership function (the central point and the gradient). The backpropagation algorithm is used. During the learning stage the weights are modified, while changing the characteristics of the fuzzy sets. At the final stage, the membership functions of the fuzzy sets are tuned (Figure 5b).

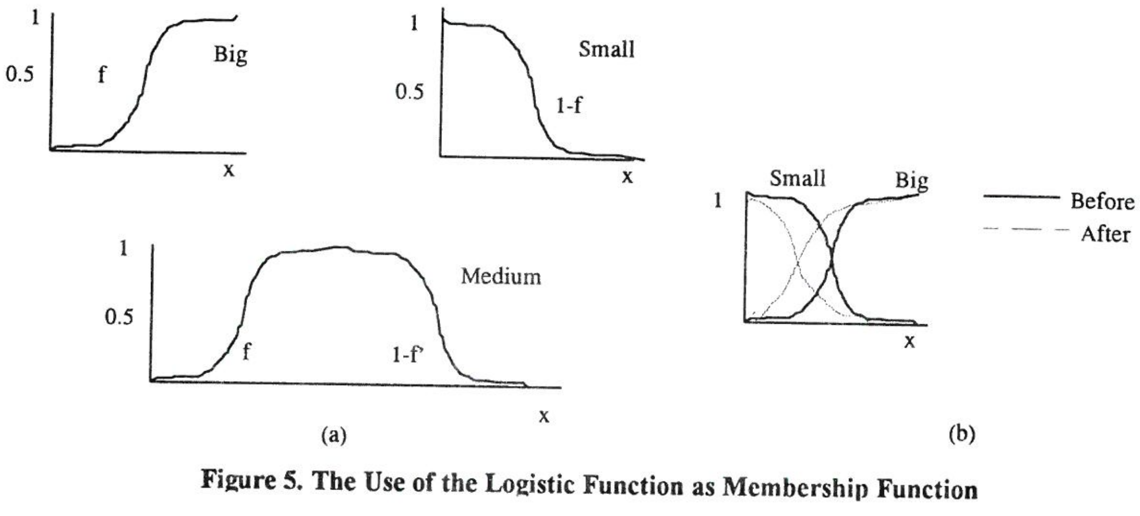

Figure 5. The Use of the Logistic Function as Membership Function

What really happens is that in the beginning such fuzzy sets as Small, Medium, Big are proposed and after the training, the shapes of these fuzzy sets have been changed (tuned) in a way such that the consequences of the input should be the expected output. This means that extraction of fuzzy rules is performed at the same time.

Conclusions - Trends

Various techniques for implementing fuzzy systems by artificial neural networks have been presented. The interval approach seems to be the most promising. Membership function tuning and rule extraction are rather interesting ideas and a lot of work still has to be done. Furthermore productive neural networks have been proposed for the fuzzy logic inference [14] and Buckley et al [15] have shown that artificial neural networks and fuzzy expert systems are equivalent. The co-operation of fuzzy logic and neural networks seems to be a successful one. Future is getting fuzzier.

This work was supported by the Greek National Scholarship Foundation.

REFERENCES

1. ZADEH, L., **Fuzzy Sets,** INFORMATION AND CONTROL, Vol. 8, 1965, pp. 338-353.
2. KOSKO, B., **Neural Networks and Fuzzy Systems,** PRENTICE-HALL, 1992.
3. YAGER, R., **Implementing Fuzzy Logic Controllers Using a Neural Network Framework,** FUZZY SETS AND SYSTEMS, 48, 1992, pp. 53-64.
4. DUBOIS, D. and PRADE, H., **Fuzzy Sets and Systems- Theory and Applications,** ACADEMIC PRESS , 1980.
5. KELLER J. and TAHANI, H., **Implementation of Conjunctive and Disjunctive Fuzzy Logic Rules with Neural Networks,** INTER. J. OF APPROXIMATE REASONING, 6 , 1992, pp. 221-240.
6. KELLER, J. and TAHANI, H., **Backpropagation Neural Networks for Fuzzy Logic,** INFORMATION SCIENCES, 62, 1992, pp. 205-221.
7. RUMELHART, D., MCCLELLAND, J. and the PDP Research Group, **Parallel Distributed Processing,** MIT PRESS, Cambridge, MA, 1986.
8. ISHIBUCHI, H., FUJIOKA, R. and TANAKA, H., **Neural Networks that Learn from Fuzzy If-Then Rules,** IEEE TRANS. ON FUZZY SYSTEMS, Vol. 1, No. 2, 1993, pp. 85-97.
9. ISHIBUCHI, H., TANAKA, H. and OKADA, H., **Fuzzy Neural Networks with Fuzzy Weights and Fuzzy Biases,** Proceedings of 1993 IEEE International Conference on Neural Networks, Volume III, pp. 1650-1655.
10. HAYASHI, Y., BUCKLEY, J. and CZOGALA, E. **Fuzzy Neural Networks with Fuzzy Signals and Weights,** INTER. J. OF INTELLIGENT SYSTEMS, Vol. 8, No. 4, 1993, pp. 527-537.
11. ZADEH, L. A., **The Concept of a Linguistic Variable and its Application to Approximate Reasoning: Part 1, 2 and 3,** INFORMATION SCIENCES, 1975, Vol. 8, pp. 199-249, Vol. 9, pp. 43-80.
12. HORIKAWA, S., FURUHASHI, T. and UCHIKAWA, Y., **On Fuzzy Modelling Using Fuzzy Neural Networks with the Back-propagation Algorithm,** IEEE TRANS. ON NEURAL NETWORKS, Vol. 3, No. 5, 1992, pp. 801-806.
13. HORIKAWA, S., FURUHASHI, T. and UCHIKAWA, Y., **On the Identification of Structures in Premises of a Fuzzy Model Using a Fuzzy Neural Network,** Proceedings of the 2nd IEEE Int. Conf. on Fuzzy Systems ,1993.
14. BULSARI, A., **Training Artificial Neural Networks for Fuzzy Logic,** COMPLEX SYSTEMS, 6, 1992, pp. 443-457.
15. BUCKLEY J., HAYASHI, Y. and CZOGALA, E., **On the Equivalence of Neural Nets and Fuzzy Expert Systems,** FUZZY SETS AND SYSTEMS, 53, 1993, pp. 129-134.