

# A Method for Compiling Neural Networks into Fuzzy Rules Using Genetic Algorithms and Hierarchical Approach

Vasile PALADE  
Severin BUMBARU  
Gheorghe NEGOITA

Department of Applied Informatics, University "Dunarea de Jos" Galati  
Str. Domneasca nr. 47, Galati 6200 Romania, Fax: (40) 36461353  
Email: {pvasile, sbumbaru, gnegoita}@alpha.ugal.ro

**Keywords:** neural networks, fuzzy rules, genetic algorithms, hierarchical systems.

## Abstract

Neural networks have been criticized for their lack of human comprehensibility, which make them to appear as black box structures to the user. This paper proposes a mechanism that compile a neural network into an equivalent set of fuzzy rules. Genetic algorithms are used to find the right structure of the fuzzy model equivalent with the neural network, and then to find the best shape of the membership functions. In order to reduce the number of fuzzy rules, we look for a hierarchical structure of the fuzzy system, considering the relations between the network inputs.

## 1. Introduction

Neural networks have been applied, with remarkable success, to a variety of real-world problems: control field, speech recognition, medical diagnosis, image computing etc. The major shortcoming of neural networks is represented by their low degree of human comprehensibility. Many authors have focused on solving this shortcoming of neural networks, by compiling the knowledge captured in the topology and weight matrix of a neural network, into a symbolic form; most of them into sets of ordinary if-then rules (Towell and Shavlik, 1993; Yoo, 1993; Craven and Shavlik, 1993; Thrun, 1994), others into formulas from propositional logic or from nonmonotonic logics (Pinkas, 1992), or into sets of fuzzy rules (Palade et al.,

1996; Lin and Lee, 1991). The fuzzy neural network are often used as an auto-tuning method for the determination and the adjustment of fuzzy rules.

The method proposed in the paper can be used to develop a set of fuzzy rules from experimental data, without any assistance from a human expert, in the field of control or pattern recognition. Our system can improve neural network based systems, such as neural controllers, with explanation facilities. For example, the decisions taken by a neural controller can be presented to the user in the form of fuzzy rules, leading to an increase of confidence in the controller action.

In the paper, we developed a mechanism for compiling the knowledge captured in the network's weight matrix into a set of fuzzy rules. In our previous works (Palade et al., 1996, Palade et al., 1995), we used genetic algorithms for tuning the shapes and the number of membership functions. We concluded that it would be difficult to apply genetic algorithms and to extract a complete set of fuzzy conjunctive rules, when the network has a greater number of inputs and outputs. We also concluded that we have to find the minimum set of fuzzy rules which approximates good enough the network. In this paper, we addressed the problem of compiling neural networks with more inputs and outputs, using a hierarchical approach and also genetic algorithms.

First, genetic algorithms are used for finding the best hierarchical structure of the fuzzy rules, and after that for tuning the shapes and the number of membership functions. The VIA method (Thrun, 1994) is used to extract a set of traditional rules. This set of rules is used to obtain the initial population of solutions, given a good point of start in the search of the right number and appropriate shapes of the membership functions. Our goal is to minimize the difference between the neural network and the fuzzy system obtained after finding the best hierarchical structure and membership function tuning.

The paper is divided into four sections. Section 2 summarizes the background concepts of the use of genetic algorithms to an arbitrary task. Section 3 presents the algorithm for tuning the shapes and the number of membership function, as well as some experimental results. The fourth section shows how to find a hierarchical structure of the fuzzy rules extracted from a neural network, using genetic algorithms as searching tools. The paper is ending with some conclusions.

## 2. Background of genetic algorithms

Genetic algorithms are general-purpose search algorithms that uses principles inspired from natural genetics. They have been used to a wide range of optimization and learning problems.

The first step in applying a genetic algorithms to a task is to find a representation for possible problem solutions. Typically, the genetic algorithms starts by randomly generating the initial population of likely problem solutions. Each element of the current solution space is called a chromosome, and its components are called genes.

At the begining of each generation, each chromosome in the current solution space is evaluated and assigned a measure of its fitness as a problem solution. Then the chromosomes are selected for replication based on their fitness value. The high performing chromosomes (high - performing problem solutions) will produce several copies of themselves, while the poorly performing chromosomes will not produce any copies.

In the next step, the chromosomes selected for replication are altered using genetic operators, in order to obtain new chromosomes which will further be evaluated. There are two basic genetic operators: crossover and mutation. The crossover operator swaps parts of two chromosomes to create two new chromosomes. The mutation operator alters one or

more components (genes) of a selected chromosome, in order to reinject information into the population and to ensure that any point in the search space can be reached. The offspring obtained after crossover, and eventually mutation, will replace older members in the population, and then are evaluated, selected for replication, and so on. In the following sections we describe the coding method, the fitness function and the genetic operations used in our task.

## 3. Tuning the shape and the number of membership functions

In this section, we shortly present the algorithm for tuning the shape and the number of membership functions, using genetic algorithms. Some simulation results are also presented. In the algorithm, we use the VIA method, developed by Thrun (Thrun, 1994), in order to extract symbolic rules from arbitrary Backpropagation neural networks.

The rules extracted by VIA algorithm have the following form:

$$\begin{aligned} &\text{if } (a_1 \leq x_1 \leq b_1) \text{ and } (a_2 \leq x_2 \leq b_2) \dots \\ &\quad \dots \text{ and } (a_m \leq x_m \leq b_m) \\ &\text{then } c_j \leq y_j \leq d_j \end{aligned}$$

where  $x_i, i=1\dots m$ , are the inputs of the networks, and  $y_j$  is the  $j$  output of the network.

More compact, we can write the rules in the form given bellow:

$$\begin{aligned} &\text{if input } \in \text{ some hypercube } \tau = [a_i, b_i]^m \\ &\text{then output } j \in [c_j, d_j] \end{aligned}$$

Each unit in the network has assigned an interval  $[a_i, b_i] \in [0,1]$ . The interval  $[0,1]$  assigned to a unit corresponds to the state of maximum ignorance about the activation of the unit. The refining of the intervals of all units in the network is done by techniques of linear programming, such as Simplex algorithm, propagating the constraints in both directions.

The algorithm for mapping a neural network into a fuzzy model is summarized bellow:

- (1) Set a number of certain membership functions, so that they divide the input space equally.
- (2) For the upper bases of the membership functions of the inputs, calculates, by VIA

method, the corresponding intervals of the output.

(3) Based on the crisp rules obtained in the previous step, construct an initial population of solutions, initializing randomly the parameters from the lower bases of trapezoidal membership functions.

(4) Tune the shape of membership functions, until the best fitness value becomes less than the target one, or searching time reaches the limited generations. If the limited number of generations is reached, go to the next step( we have to find another structure of the fuzzy model). Otherwise, the minimal and optimal fuzzy model is obtained.

(5) Let the mutation operator to prune a membership function in one of the inputs or in the output. Go to the step number 2.

*Experimental results:*

The ability of the algorithm to adjust the shape and the number of membership functions was proved on a neural network with two inputs and one output. These simulations are made considering a complete set of conjunctive fuzzy rules. The fourth section presents how to find a hierarchical structure of the fuzzy rule set. We have taken 7 equidistant trapezoidal membership functions in each input space, as shown in figure 1. We could take a greater number of membership functions in each input space, for a good approximation, but the number of membership functions for the output grows exponentially, leading to a loss of comprehensibility of fuzzy rules.

Let's say we want to derive the fuzzy rule when  $x_1$  is "Relative Low" and  $x_2$  is "High". So, we will take the upper bases of the corresponding trapezoidal membership functions in the premise and we will deduce, by VIA method, the crisp rule: if  $x_1 \in [0.305 ; 0.355]$  and  $x_2 \in [0.815 ; 0.865]$  then  $y \in [0.46 ; 0.56]$ , which correspond to a fuzzy rule with a trapezoidal membership function with the upper base  $[0.46 ; 0.56]$ , in the rule consequence. We have taken all possible combinations and we obtained a 7 x 7 matrix of fuzzy rules. Only conjunctive rules are considered in our experiment. The number of resulting membership functions for the output was reduced, but with the price of worsening of the approximation error.

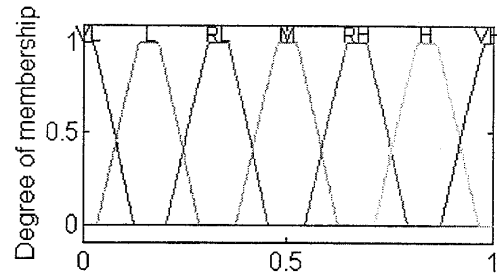


Figure 1. Input membership functions

Every solution in the initial solution population is obtained by VIA method, as shown above. The population was of 80-100 members. Each chromosome in the population contains 72 real numbers between 0 and 1. The first 7 x 4 numbers represent the parameters of trapezoidal membership functions of one input, and the last 11 x 4 numbers represent the parameters of trapezoidal membership functions of the output. We represented only one input along the length of the chromosome, as the two inputs are interchangeable, in our experiment. A gene, which can be changed between two chromosomes, is composed by four consecutive numbers.

The fitness function, used in evaluation, is the sum of squared errors between the output value of the neural network, and the output value of the fuzzy system with the parameters taken from the current chromosome in the population.

As crossover operator, we used a two-point crossover, one point along the first 7 genes, and the other point along the genes corresponding to the output. The crossover points are chosen randomly and must be divisible with 4. The mutation operator can alter any parameters of a gene. When the mutation operator alters the upper bases of a trapezoidal membership functions of the input, we automatically calculate, by VIA method, the upper bases of the corresponding membership functions of the output. The mutation rate was 3 percent of the total numbers of genes in the population.

We tried to find a solution in around 300 - 400 generations. In figure 3 is shown the fitness of the best solution in the population along the generations, for two different runs. Figure 2 presents the shapes of membership functions obtained after tuning.

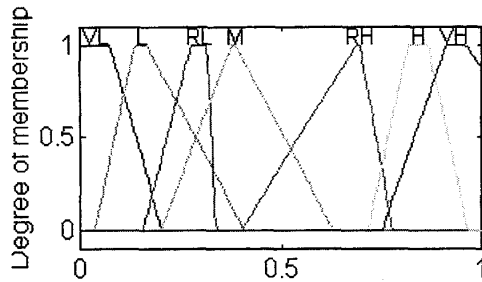


Figure 2. Obtained membership functions

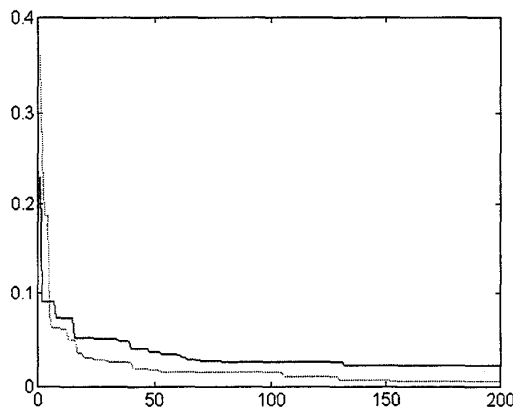


Figure 3. The fitness of the best solution

#### 4. Hierarchical approach

The main problem encountered by our method is the exponential increasing of the number of fuzzy rules with the increase of the dimension of the input space. The number of rules is the product of the number of membership functions for each input. In order to reduce the number of membership functions and consequently the number of fuzzy rules, we applied GAs to find a hierarchical structure of the fuzzy system, considering the relations between the input variables, in a similar manner as Shimojima et al. used in their work (Shimojima et al., 1995).

The key idea is to collect the inputs which have closer relationships in one fuzzy unit. For example, let's take a network with 4 inputs and one output, and let's consider three equidistant membership functions in each input space. The total number of fuzzy rules which we could obtain is  $81 = 3 \times 3 \times 3 \times 3$ , considering all the

combinations. Now, let's suppose there are some relationships between the inputs 1 and 2, one one hand, and between the inputs 3 and 4, on the other hand. In this case, the neural network can be described by the hierarchical fuzzy system shown in figure 5. Every fuzzy unit is described by  $3 \times 3$  rules. Then, the total number of fuzzy rules which describe the network is drastic reduced.

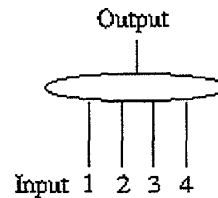


Figure 4. Black-box neural network

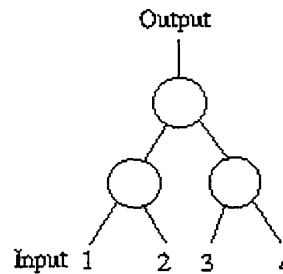


Figure 5. Hierarchical structure

*The coding method:*

If the network has  $n$  inputs, we arrange  $2^{n-2}$  units in a tree with 1, 2, 4, 8, ... units on each level. Every input is assigned to a fuzzy unit in the tree. If a fuzzy unit has only one input, the input is passed to the upper fuzzy unit in the tree. For example, if  $n = 5$ , we arrange the units in the tree shown in figure 6. With the assigning given in figure 6, and after eliminating unnecessary fuzzy units, we obtain the structure given in figure 7.

The structure of a chromosome is shown in figure 6. A gene corresponds to a network input and contain the number of the fuzzy unit which is assigned to that input. As crossover operator we used a common one point crossover operator. The fitness function used for this part is:

$$f = E + \alpha * N$$

where  $E$  is the sum of the squared error between the network output and the output of the fuzzy model,  $N$  is the number of fuzzy rules, and  $\alpha$  a coefficient.

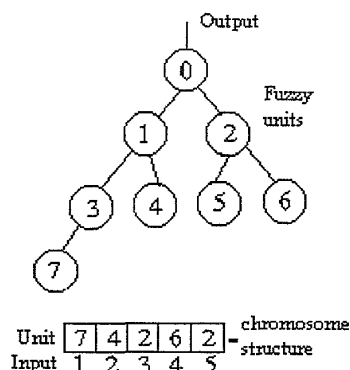


Figure 6. The coding method

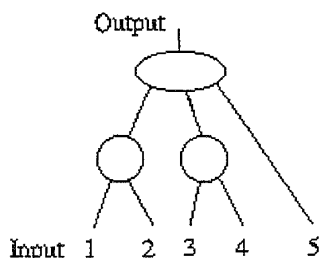


Figure 7. The obtained hierarchical structure

In fact, the coding method is the same used by Shimojima et al. (Shimojima et al., 1995), instead we used another fitness function and genetic operators.

As an example, we tried to find a hierarchical structure for a neural network which implements the function:

$$f(x_1, x_2, x_3, x_4) = (\min(x_1, x_2) + \min(x_3, x_4)) / 4$$

The initial population contained 15 solutions. Meeting our expectations, the best hierarchical structure which we could obtain is shown in figure 5. This was happened in around 15 generations.

Following the considerations introduced in this section, the algorithm presented in section 3 can be developed,

in order to address the neural networks of big sizes. So, the algorithm is divided into two main subprocedures:

1. find the best hierarchical structure of the fuzzy model equivalent with the network.
2. tune the shapes and the number of membership functions of the hierarchical structure found in the previous step.

The first subprocedure starts with the set-up of a number of equidistant membership functions in each input space, and the calculating of the corresponding membership functions of the output, in a similar manner used in the first two steps of the algorithm presented in section 3. Using the coding method and the genetic operators introduced before, let the GAs to find the best structure of the fuzzy model. The second subprocedures of the improved algorithm, given above, is in fact the algorithm presented in section 3.

## 5. Conclusion

We improved our algorithm for compiling neural networks into an equivalent set of rules, based on a hierarchical approach, in order to address the networks with more inputs. Our previous algorithm (Palade et al., 1996) had a serious limitation, regarding its application to neural networks of big sizes. We proved that GAs can tune not only the shapes of membership functions, but also the number of membership functions and fuzzy rules.

## References

Craven M., J.W.Shavlik (1993). Learning Symbolic Rules Using Artificial Neural Networks. *Proceeding of the 10-th International Conference on Machine Learning*, Amherst.

Gonzales A., R. Perez and J.L. Verdegay (1993). Learning the Structure of a Fuzzy Rule: A Genetic Approach. *Proceedings of the First European Congress on Fuzzy and Intelligent Technologies*, Aachen - Germany.

Ishigami H., T. Fukuda, T. Shibata and F. Arai (1995). Structure Optimization of Fuzzy Neural Network by Genetic Algorithm. *Fuzzy Sets and System* **71**.

Janson D.J., J. F. Frenzel (1993). Training Product Unit Neural Networks with Genetic Algorithms. *IEEE Expert Intelligent Systems and Their Applications*, vol. **8** nr. 5.

Kosko B. (1992). *Neural Networks and Fuzzy Systems*. Prentice - Hall International Inc.

Lin C.T. and C.S. George Lee (1991). Neural - Network Based Fuzzy Logic Control and Decision System. *IEEE Transactions on Computers*, vol. **40** No. **12**, 1991.

- Nomura H., Hayashi I. and Wakami N. (1992). A Learning Method of Fuzzy Inference Rules by Descent Method. *Proceedings of the International Conference on Fuzzy Logic and Neural Networks. Iizuka'92*.
- Palade V., V. Mazilescu and S. Bumbaru (1992). Special Issues on the Use of Neural Networks in the Intelligent Control Systems. *The 1992 Annals of "Dunarea de Jos" University of Galati*, Fascicle nr. 3, Galati.
- Palade V., S. Bumbaru and V. Arition (1995). Mapping Neural Networks into Fuzzy Rules. *The 1995 Annals of the University "Dunarea de Jos" of Galati*, Fascicle nr. 3, Galati.
- Palade V., V. Arition and C. Segal (1996). Use of Genetic Algorithms for Tuning Fuzzy Models. *The 1996 Annals of the University "Dunarea de Jos" of Galati*, Fascicle nr. 3, Galati.
- Shimojima K., T. Fukuda and Y. Hasegawa (1995). Self-tuning Fuzzy Modelling with Adaptive Membership Function, Rules, and Hierarchical Structure Based on Genetic Algorithm. *Fuzzy Sets and Systems* 71.
- Thrun S.B. (1994). *Extracting Symbolic Knowledge from Artificial Neural Networks*. Revised Version of Technical Research Report TR-IAI-93-5, Institut für Informatik III - Universität Bonn.
- Towell G., J.W. Shavlik (1993). The Extraction of Refined Rules from Knowledge - Based Neural Networks. *Machine-Learning*, vol.13.
- Yoo J.H. (1993). *Symbolic Rule Extraction from Artificial Neural Networks*. PhD thesis, Wayne State University.