A LEARNING METHOD OF FUZZY REASONING BY GENETIC ALGORITHMS

J.L. Castro, M. Delgado and F. Herrera

Dept. of Computer Science and Artificial Intelligence
University of Granada
18071 Granada, Spain

1. Introduction

Fuzzy Rule Base Systems (FRBS) has been shown to be an important tool for problems where, due to the complexity or the imprecision, classical tools are unsuccessful. In [3,14] it has been proved that FRBS are universal approximators in the sense that for any continuous system it is possible to find a set of fuzzy rules able of approximating it with arbitrary accuracy. Now, the question is: How can we find this set of rules?.

Unfortunately, acquiring rules from experts is not an easy task. On the other hand, it is very difficult for a knowledge engineer to extract rules from static databases. Thus rule acquisition becomes a bottleneck in the knowledge engineering process.

In order to automatically obtain rules from an example database, various methods have been proposed in the literature. This methods include the learning methods employing Descent Method (Araki et al. [1]), or Neural Network (Yamaoka-Mukaidono [15], Lee [10]) as well as Belief or Uncertainty measures (Binaghi [2], Peng-Wang [13], Delgado-Gonzalez [4]).

However, from a practical point of view, these methods suffer from some of the following critical problems,

1) the number of rules is to be set in advance and so it must be derived by trial and error;

2) the search tends to falls into local optima;

3) a high degree of nonlinearity can appear in the search, which makes the method uneffective.

The result of the learning depends largely on the number of rules. When it is set under the rigth one, an inadequate FRBS to approximate the real system may results, while setting the set of rules over its correct size may sacrifice the generalization capability of the fuzzy rules. Therefore, the number of inference rules has to be determined from a standpoint of overall learning capability and generalization capability of fuzzy rules. The work to determine the number of rules requires to designers a huge number of experiment by trial and error.

Genetic Algorithms (GA) are a search technique drawing an increasing attention in the field of Artificial Intelligence. GA are algorithms that use operations found in natural genetics to guide the trek through a search space, and this enables them to escape from local optimum, being in this way (potentially) very powerfull optimization techniques for complex functions [6,7]. De Jong's work [5], in particular, establishes the GA as a robust search technique efficient across a

spectrum of problems, as compared to other traditional schemes. Subsequent application of GA's to other search and machine learning problems support the claim that GAs are broadly based.

The robustness and simple mechanics of GA make them inviting tools for learning the rules to be used in a FRBS. GA are also potentially useful for learning fuzzy rules due to some special traits that differentiate them from conventional techniques. In fact, recently, the use of Genetic Algorithms (GA) have been proposed for learning rules (Karr [7,8]), showing that AG are an appropriate tool for this problems. This papers present methods to be applied to special types of rules or to specific problems.

We present here a more general approach to this problem using GA. A very important advantage of our method is that the number of fuzzy rules is determined by the algorithm. Moreover, the method does not depend on the type of fuzzy reasoning neither on the shape of membership functions employed.

2. Formulation of the problem

As we have said above, in [3,14] it has been proved that fuzzy logic controllers are universal approximators, that is:

For each $a < b$, $a,b \in \mathbf{R}$ let $\mu(a,b):\mathbf{R} \longrightarrow \mathbf{R}$ be a membership function such that $\mu(a,b)(x) \neq 0$ iff $x \in (a,b)$. Let T and T' be some t-norms and I a fuzzy implication verifying $I(a,0) = 0$ if $a \neq 0$ (for example a R-implication or a t-norm implication $I(a,b) = T''(a,b)$). Let S be a t-conorm.

Assume $\mathcal{S} = \mathcal{S}(T',T,I,S,\mu(a,b))$ is the family of all Fuzzy Logic Controller (FLC) systems where:

i) The rules base is composed by a finite number of rules with the form

If x_1 is A_1 and x_2 is A_2 and ... x_n is A_n, then y is B.

ii) The membership function of each A_{ij} is $\mu(a^1_{ij}, a^2_{ij})$ for some $a^1_{ij} < a^2_{ij}$, $a^1_{ij}, a^2_{ij} \in \mathbf{R}$, i.e.

$$A_{ij}(x) = \mu(a^1_{ij}, a^2_{ij})(x).$$

iii) The membership function of each B_j is $\mu(b^1_j, b^2_j)$ for some $b^1_j < b^2_j$, $b^1_j, b^2_j \in \mathbf{R}$, i.e.

$$B_j(y) = \mu(b^1_j, b^2_j)(y).$$

iv) The fuzzy inference is made with T as fuzzy conjunction and the generalized modus ponens constructed from T' and I:

a) A rule

R_j: If x_1 is A_{1j} and x_2 is A_{2j} and ... x_n is A_{nj}, then y is B_j

will be applied if and only if the input $\underline{x}^o$ matches with the antecedent, i.e. iff $A_j(\underline{x}^o) \neq 0$, being $A_j(\underline{x}) = T(A_{1j}(x_1), A_{2j}(x_2), ..., A_{nj}(x_n))$.

b) If the input $\underline{x}^o$ matches with the antecedent then the inference is

$$\frac{\begin{array}{l}\text{If } x_1 \text{ is } A_1 \text{ and } x_2 \text{ is } A_2 \text{ and } \ldots x_n \text{ is } A_n, \text{ then } y \text{ is } B \\ \underline{x} \text{ is } A'\end{array}}{y \text{ is } B'}$$

$$B'(y) = \text{Sup } \{T'(A'(\underline{x}),\ I(A(\underline{x}),B(y))) \ /\ \underline{x} \in \mathbf{R}^n\ \}. \quad \text{(GMP)}$$

$$A(\underline{x}) = T(A_1(x_1),A_2(x_2),\ldots,A_n(x_n)),$$

Since the input is a point $\underline{x} = \underline{x}^o$

$$A'(\underline{x}) = \begin{cases} 1 & \text{if } \underline{x} = \underline{x}^o \\ 0 & \text{if } \underline{x} \neq \underline{x}^o \end{cases},$$

and thus the result is translated into

$$B'(y) = T'(1,\ I(A(\underline{x}^o),B(y))) = I(A(\underline{x}^o),B(y).$$

c) In general, we can express the the inference of the rule R_j when the input is $\underline{x} = \underline{x}^o$ by:

$$B_j'(y) = \begin{cases} 0 & \text{if } A\ (\underline{x}_j^o) = 0 \\ I(A_j(\underline{x}^o),B_j(y)) & \text{elsewhre.} \end{cases}$$

which in the case of a t-norm implication is the general expression

$$B_j'(y) = I(A_j(\underline{x}^o),B_j(y)).$$

v) The joint of all fuzzy rule is made by means of S: $B'(y) = T^*(\{B_j'(y)\})$.

vi) The defuzzification method is the center of the area:

$$y^o = S(\underline{x}^o) = \frac{\int B'(y).y\ dy}{\int B'(y)\ dy}$$

The only parameters being not prefixed in this class are the number of rules K, such that j=1,..,K; and the the real values a_{ij}^1, a_{ij}^2, b_j^1, b_j^2 which characterize the membership functions (i = 1..n, j = 1..k).

THEOREM. Let f: $U \subseteq \mathbf{R}^n \longrightarrow \mathbf{R}$ be a continuous function defined on a compact U. For each $\varepsilon > 0$ there exists a $S_\varepsilon \in \mathcal{S}$ such that

$$\sup\ \{|f(\underline{x}) - S_\varepsilon(\underline{x})|\ /\ x \in U\} \leq \varepsilon.$$

Basically, the problem considered in this paper consist on finding the finite set of fuzzy rules able to approximate to level ε the input-output behavior of a real system. That is, fixed the class $\mathcal{S} = \mathcal{S}(T',T,I,S,\mu(a,b))$ of FLC systems, where $\mu(a,b)$ is the triangular membership function with support = (a,b), to find the parameters k

and a^1_{ij}, a^2_{ij}, b^1_j, b^2_j (i=1..n, j=1..k) of a system of $\mathcal{S}$ which approximate the real system. The domain of every input variable X_i is a close real interval $[a_i,b_i]$. Similarly, the domain of the output variable Y is a close real interval $[c_j,d_j]$.

Let us suppose we only know a set of training examples of the system consisting in the values that the variables take during an experiment in which the system is controlled by a human:

"in the time t = i, the value of the variables X_j and Y_j are x^i_j and y^i_j respectively"

and possibly a number s (it is possible s=0) of linguistic rules given by the domain expert

$$R_h\text{: If } X_1 \text{ is } L^h_1 \text{ and ... and } X_n \text{ is } L^h_n \text{ then } Y \text{ is } T^h,$$

h = 1..s, where L^h_i and T^h_i are linguistic labels.

3. The method

We have subdivided the solution of this problem in the following steps:

1.- We obtain "good" rules from the examples;

2.- We obtain optimal membership functions for any linguistic rule given by the domain expert.

3.- We get from the rules candidates (those obtained in first and second step) an optimal subset of rules, which will be the output of our algorithm.

The set of rules candidates must be exhaustive, in the sense that for any input must have a rule to apply. In first step we reduce the rule candidates in order to avoid the exponential complexity of the search space in the third step.

3.1. Learning new "good" fuzzy rules

We apply a three-operator GA in order to obtain a set of fuzzy rules, which will be exhaustive and appropriate. The term exhaustive means that for any possible input of the system point $(x_1,...,x_n)$ must have in the set a fuzzy rule to apply. As the set of possible inputs is continuous, in order to check exhaustively we divide the domain of every variable X_i, in d+1 points,

$$x_i[l] = a_i + l(b_i - a_i)/d,\ l = 0...d.$$

When we apply a three-operator GA to a search problem, three decisions must be made: how to code the possible solutions to the problem as finite bit strings, how to evaluate the merit of each string, and which will be the stop condition.

A fuzzy rule

$$\text{If } X_1 \text{ is } A_1 \text{ and ... and } X_n \text{ is } A_n, \text{ then } Y \text{ is } B$$

is determined by the parameters of the membership function of every A_i and B. The study of sensibility of fuzzy rules has shown that the type of the membership function is not so important as its support. Thus, we will restrict to the standard type, simmetrical triangular membership functions.

For coding, a common method called a "concatenated, mapped, unsigned binary coding" will be used. The domain of each variable is discretized by mapping from a

minimum value C_{min} to a maximum value C_{max} using a k-bit, unsigned binary integer according to the equation:

$$C = C_{min} + \frac{binrep}{(2^k-1)} (C_{max} - C_{min})$$

where binrep is the integer value represented by an k-bit string, and C is the value coded. Thus each triangle is coded as two k-bit (left limit, and right limit of the triangle), and the rule is coded by a $h = \sum_{i=1}^{n+p} 2*k_i$ (k_i is the length of the string associated to the discretization of the domain of each variable, x_i, $1 \le i \le n$, y_i, $n+1 \le i \le n+p$)

The evaluation function of the string coding a rule R will be

$$f_e(R) = \begin{cases} (\varepsilon - App(R))*b & \text{if } App(R) \le \varepsilon \\ 0 & \text{otherwise} \end{cases}$$

where

$$App(R) = \sup \{A(\underline{x}^l)*|y^o(\underline{x}^l)-y^l| \ / \ l=1..r \text{ and R can be applied to } \underline{x}^l\},$$

being y^o is the output of the rule R when the input is $\underline{x}^l$ obtained by the inference process, and b is the product of the support wide of each triangle. R can be applied to $\underline{x}^l$ if $\underline{x}^l_i$ belongs to the support of the variables x_i for all i.

The stop condition will be:

a) Approximation: Each rule of the population verifies $App(R) \le \varepsilon$, and

b) Exhaustivity: for any $(x_1,..,x_n) \in D$, there exists a rule R in the population which can be applied to $(x_1,..,x_n)$. This condition is checked by look at the base of every triangle in the rules of the population.

3.2. Learning high-performance membership functions

In this subsection we present a three-operator GA to select optimal membership functions of a linguistic rule. Specifically, given a rule R,

If X_1 is L_1 and ... and X_n is L_n, then Y is T

where L_i and T_i are linguistic labels, we must determine the membership function of each L_i and T_i in order to obtain an optimal rule, that as we have said above, we will restrict to triangular membership functions.

In a first step we assign a prototype example to the rule. The value of any variable in the example will be used to be the mode of the membership function of the linguistic label qualifying this variable. Then, the GA is used to translate, and expand or shrink the base of each triangle.

Thus each triangle is coded as two k-bit (left limit and right limit of the triangle), and the rule is coded by the code of its membership functions as a h-bit string, ($h = \sum_{i=1}^{n+p} 2*k_i*(n+p)$).

The evaluation function of the string associated to the string coding a rule R is again

$$f_e(R)= \begin{cases} (\varepsilon\text{-App}(R))*b & \text{if } App(R) \le \varepsilon \\ 0 & \text{otherwise.} \end{cases}$$

Finally the stop condition will be if there exists a string such that App(R) ≤ ε for all rules given by the expert or the number of iterations is greater than a prefixed number.

3.3. Obtaining an optimal set of rules

Now, we must obtain the final set of rules. The set of rules obtained in first step together the rules considered in second step are the set of cadidates rules. This set will be ordered in a arbitrary way from 1 until the number m of candidates rules. Then, a m-bit string represents a subset of candidates rules and we apply a GA to obtain an optimal subset of rules.

The evaluation function of a subset S will be

$$f_e = \sup \{ | Inf(S(x_1^l,..,x_n^l)-y^l | / \ l = 1..r\},$$

where $Inf(S(x_1^l,..,x_n^l))$ is ∞ if none rule in S can be applied to $x_1^l,..,x_n^l$, and the stop condition will be $f_e \le \varepsilon$, for a prefixed $\varepsilon > 0$.

References

[1] Araki, S., H. Nomura, I. Hayashi and N. Wakami, A self-generating Method of Fuzzy Inference Rules, *Proceedings of IFES'91,* 1047-1058, 1991.

[2] Binaghi, E., Learning of uncertainty classification rules in medical diagnosis in Symbolic and Quantitative Approaches to Uncertainty, *Lectures Notes in Computer Science 548 (R. Kruse, P. Siegel, Eds.),* 115-119, 1991.

[3] J.L. Castro, Fuzzy Logic Controllers are Universal Approximators. *Submitted to IEEE Transactions on Systems, Man, and Cybernetics,* 1992.

[4] Delgado, M. and A. González, A frequency model to assess Dempster/Shafer's evidences, *Submitted to International Journal of Approximate Reasoning,* 1992.

[5] De Jong, K.A., An Analysis of the Behavior of a >Class of Genetic Adaptive Systems, *Doctoral Thesis,* Department of Computer and Communication Sci*ences,* University of Michiga, Ann Arbor, 1975.

[6] Goldberg, D.E., *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.

[7] J.H. Holland, *Adaptation in Natural and Artificial Systems.* The University of Michigan Press, Ann Arbor, 1975.

[8] Karr, C., Genetic Algorithms for Fuzzy Controllers, *AI EXPERT,* 26-33, Feb. 1991.

[9] Karr, C., Applying Genetics Algorithms to Fuzzy Logic, *AI EXPERT,* 38-43, March 1991.

[10] Lee, C.C., A self-learning rule-based controller employing approximate reasoning and neural net concepts, *International Journal of Intelligent Systems* 6, 71-93, 1991.

[11] K.S. Leung, Y. Leung, L. So and K.F. Yam, Rule Learning in Expert Systems using Genetic Algorithm. *Proceedingns of IIZUKA'92,* 201-204, 1992.

[12] H. Nomura, I. Hayashi and N. Wakami, A Learning Method of Simplified Fuzzy Reasoning by Genetic Algorithm. *Proc. of the Int. Fuzzy Systems and Intel*ligent *Control Conference,* 236-245, Lousville, KY, 1992.

[13] Peng, X., Wang, P., On generating linguistic rules for fuzzy models, Lectures Notes in Computer Science 313, Uncertainty and Intelligent Sustems (B. Bouchon, L. Saitta, R.R. Yager (Eds.), 185-192 (1988).

[14] Wang, L.X. Fuzzy Systems are Universal Approximators", *IEEE Trans. Sys. Man Cyber.*, Vol. SMC-7, No. 10, pp. 1163-1170, 1992.

[15] Yamaoka, M. and M. Mukaidono, A learning method of fuzzy inference rules with a neural network. Fourth IFSA Congress, Engineering 222-225, 1991.