Neuro-evolutionary systems for learning parametric fuzzy connectives from examples of behaviour [1]

Claudio Moraga

Dept. Artificial Intelligence, Technical University of Madrid, Spain
Dept. Computer Science and Computer Engineering, University of Dormtund, Germany
claudio@moraga.de

Abstract: Extraction of models for complex systems from numerical data of behavior is studied. In particular, systems representable as sets of fuzzy if-then rules where the premises are connected by t-norms or, by a parametric aggregation operator are discussed. A method is presented to extract this kind of fuzzy rules with support of neural networks and evolutionary algorithms.

1 Introduction

The extraction of models for complex systems from data of behaviour is a knowledge acquisition problem which may appear when working with real systems. The goal is to obtain models that are *both* understandable *and* accurate enough for a given application. The subject has received much attention from the research community in the last decade with the aim of combining the learning capability of neural networks with the expressiveness of fuzzy if-then rules using linguistic variables. Much of this work has been strongly influenced by the pioneering work of Jang [5], who introduced the ANFIS system. The important contribution of ANFIS is the idea of expressing as net architecture, the main components of a fuzzy inference system: fuzzification, implication and (if needed) defuzzification. Nodes of the first hidden layer realize the linguistic terms of the linguistic variable associated to a given external input. For this purpose they have a bell-shaped (or triangular/trapezoidal shaped) activation function, where the weight of the input and the value of the bias (from the neural net point of view) determine the width and center of the bell representing a linguistic term (from the fuzzy point of view), respectively. Figure 1 shows an ANFIS system with two input variables having three linguistic terms each, and giving rules of the Takagi-Sugeno type [11].

[1] Work leading to this paper has been partially supported by the German Ministry of Education and Research BMBF, Project RCH 99/023, by the Spanish State Secretary of Education and Universities of the Ministry of Education, Culture and Sports (Grant SAB2000-0048), and by the Social Fund of the European Community.

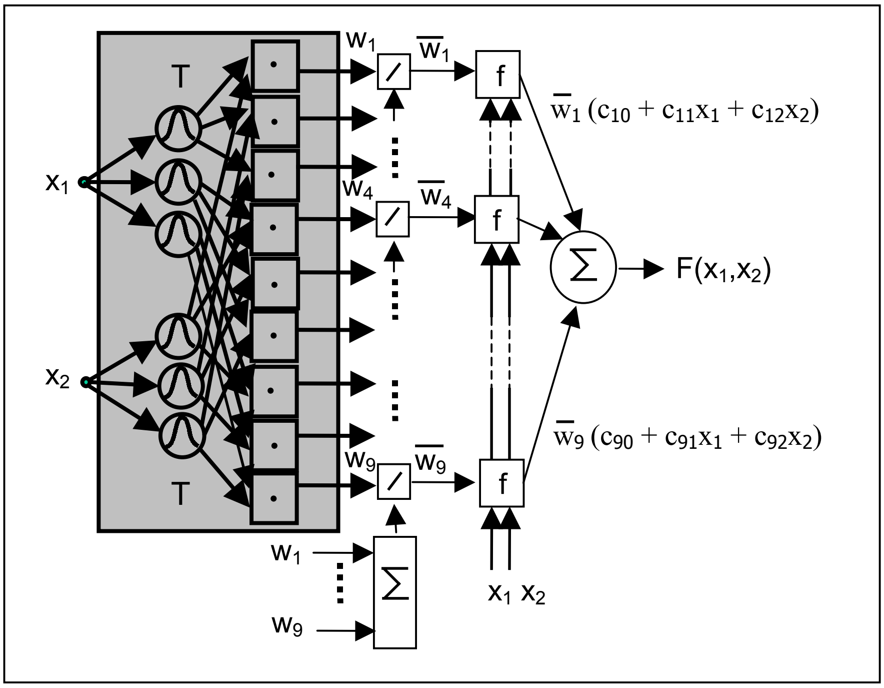

Fig. 1. : Structure of an ANFIS system to extract rules of type Takagi-Sugeno. Shaded in grey is shown the part of the network to learn the membership functions of the linguistic terms

Since the training of the neural network is done with a gradient descent algorithm, then all elementary functions along an input-output path have to be differentiable. Particularly, nodes at the second hidden layer (representing the conjunction of the degrees of satisfaction of the premises) must have a differentiable t-norm as transfer function. (With respect to t-norms see *e.g.* [13]). In ANFIS the product operation was chosen for this purpose. After proper training of the system as a neural network, weights and bias values identify the corresponding parameters of the associated fuzzy if-then-rule. Several ANFIS-like systems that follow the same strategy have been later reported in the literature. Even though this kind of system has proven to be able to deliver very accurate models, this is often achieved by giving up the understandability of the rules. Fuzzy sets that should represent linguistic terms exhibit after training strong and complex overlaps -(possibly due to an unfortunate initial choice with too few linguistic terms)- impairing a simple interpretation. A *constrained* gradient descent algorithm should be used to tune the neural networks with controlled overlap. This question has been of concern both in the early times (see *e.g.* [9]) as well as in recent days, as discussed in [11].

Even though ANFIS deserves recognition as a milestone in the development of neuro-fuzzy systems, its limitations must be also clearly stated. ANFIS requires from the user an *a priori* decision on the number of linguistic terms per variable, it only extracts rules where the premises are connected through *differentiable* t-norms and generates as many rules as the product of the number of linguistic terms per variable. For instance if a system has 5 input variables and the associated linguistic variables have only 3 terms each, ANFIS would generate $3^5 = 243$ rules. It becomes apparent that this amount of rules is a great burden on the understandability of the model. If however an adequate evolutionary design of the front end of ANFIS is made, a multiresolution effect may be obtained thus alleviating the problem of defining the number of linguistic terms per variable and reducing the number of rules, as discussed in [8]. The number of rules may be strongly reduced if a cluster-oriented neuro-fuzzy system is used, as analyzed in [1]. In this case the understandability of each rule may become more difficult, since a linear combination of input values is associated to a linguistic term; but this is compensated by the reduced number of rules needed to model a system. It is interesting to mention that this strategy may possibly be traced back to work done by Jang and Sun [6], who proved the functional equivalence between RBF neural networks and fuzzy inference systems.

2 New challenges

The following questions will be studied below:

i) Is it possible to design a neural network to learn appropriate fuzzy connectives to aggregate the premises of the if-then rules from behavioural data of a system?

ii) Is it possible to combine the neural network with an evolutionary algorithm to solve the problem?

Consider the following decision rule: *"**If** a Conference has a good name **and** the registration fee is convenient (from the point of view of a possible participant), **then** hurry up to submit a paper".* Assume that the t-norm minimum is used to realize the ***and***-conjunction. The consequence is that in case that the registration fee is right at the limit of the budget (*i.e.* $\mu_{convenient}(x_2) = 0.51$) it will make no difference whether the conference appears in the ISI-List ($\mu_{goodname}(x_1) = 0.996$) or is only known to a rather small group close to the organizers ($\mu_{goodname}(x_1) = 0.53$); since the recommendation to submit a paper will have in both cases the "strength" 0.51. (The assumption is made, that below 0.5 the recommendation begins to be "not to submit" a paper. This assumption is made in analogy to normalized linguistic variables. A degree of membership of an alement to a linguistic term between 0 and 0.5 implies a degree of membership greater than 0.5 to some neighbour term.) Since any other t-norm would produce values not larger than those produced by minimum, the choice of another t-norm would not avoid the problem illustrated above. This is obviously not the way a human being would use the rule. A straight t-norm connection of the premises is not adequate to express the willingness to compromise, which characterizes the above situation.

2.1 Learning uninorms with a neural network

Uninorms were introduced by Yager and Rybalov [14], as a generalization of t-norms and t-conorms. Uninorms belong to the class of aggregation operators. (For more information on aggregations, see *e.g.* [3])

Let u: $[0,1]^2 \rightarrow [0,1]$ be associative, commutative and monotone in both arguments. Moreover for all $x \in [0,1]$ and for a given $e \in [0,1]$, $u(e,x) = u(x,e) = x$. Then u ist a uninorm. It is fairly obviuos that if $e = 1$, u turns into a t-norm and if $e = 0$, into a t-conorm. The best known pairs of uninorms are (R_*, R^*) [14] and (R_e, R^e) [4] defined as follows:

$R_*(x_1,x_2) =$ $\max(x_1,x_2)$ if $\min(x_1,x_2) \geq e$ and $\min(x_1, x_2)$ otherwise

$R^*(x_1,x_2) =$ $\min(x_1,x_2)$ if $\max(x_1,x_2) \leq e$ and $\max(x_1,x_2)$ otherwise

$R_e(x_1,x_2) =$ $\max(x_1,x_2)$ if $\min(x_1,x_2) \geq e$; 0 if $\max(x_1,x_2) \leq e$ and $\min(x_1,x_2)$ otherwise

$R^e(x_1,x_2) =$ $\min(x_1,x_2)$ if $\max(x_1,x_2) \leq e$; 1 if $\min(x_1,x_2) \geq e$ and $\max(x_1,x_2)$ otherwise

R_e and R^e use $e \in (0,1)$ and are known to be the smallest and the largest uninorms, respectively.

For the former example consider $R_*(\mu_{goodname}(x_1), \mu_{convenient}(x_2))$ and let $e = 0.5$. $R_*(0.996, 0.51) = \max(0.996, 0.51) = 0.996$ meanwhile $R_*(0.53, 0.51) = \max(0.53, 0.51) = 0.53$. It becomes apparent that in this case the uninorm R_* behaves as the t-conorm maximum and leads to a clear distinction between both situations. The same result would be obtained *in this case* with R^* and R_e.

Keep $e = 0.5$ and use $R^e(\mu_{goodname}(x_1), \mu_{convenient}(x_2))$. Since for both conferences $\min(\mu_{goodname}(x_1), \mu_{convenient}(x_2)) = 0.51 > e$, then $R^e(\mu_{goodname}(x_1), \mu_{convenient}(x_2)) = 1$ and there is no distinction between the conferences. Let however $e = 0.55$, then $R^e(0.996, 0.51) = \max(0.996, 0.51) = 0.996$ meanwhile $R^e(0{,}53, 0.51) = \min(0.53, 0.51) = 0.51$. Obviously R^e behaves as a t-conorm for the first conference but as a t-norm for the second one. This simple example illustrates the importance of the right choice of *e*. In what follows we will show a possible way to learn *e* from data of behaviour.

Figures 2a and 2b illustrate an ANFIS-like front-end to learn the parameters (position and width) of the Gaussian, Bernoullian or Splines-bell activation functions used for the processors of the first layer, which will define the linguistic terms associated to the corresponding crisp inputs. Furthermore, the parameter *e* may be adjusted according to the training data. All weights are 1 unless otherwise specified. It becomes apparent, that even though this neural network is very simple and only a few parameters need to be adjusted, this cannot be done with a gradient descent algorithm, since maximum, minimum and the Heaviside function used to determine *e* are not overall differentiable. The problem is however simple enough to be solved by using an evolutionary strategy [10] or a real-coded genetic algorithm (see *e.g.* [2]).

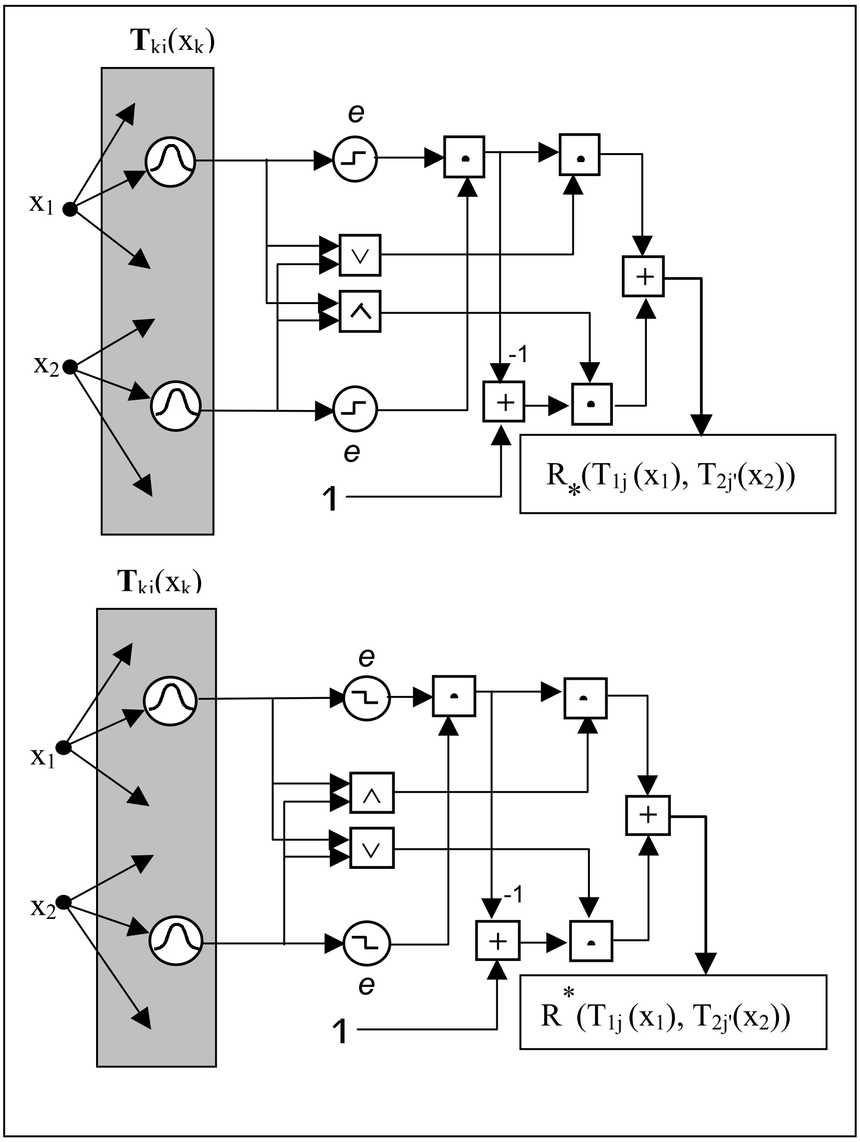

Fig. 2. ANFIS-like front-end to learn the linguistic terms and the *e*-parameter for R_* (top) and R^* (bottom) for one rule

By comparing the definitions of R_* and R_e it is simple to see that the network for R_* may be slightly modified to realize R_e. Similarly in the case of R^* and R^e. This is illustrated in figure 3. The parameters may be adjusted by using again an evolutionary strategy or a real-coded genetic algorithm.

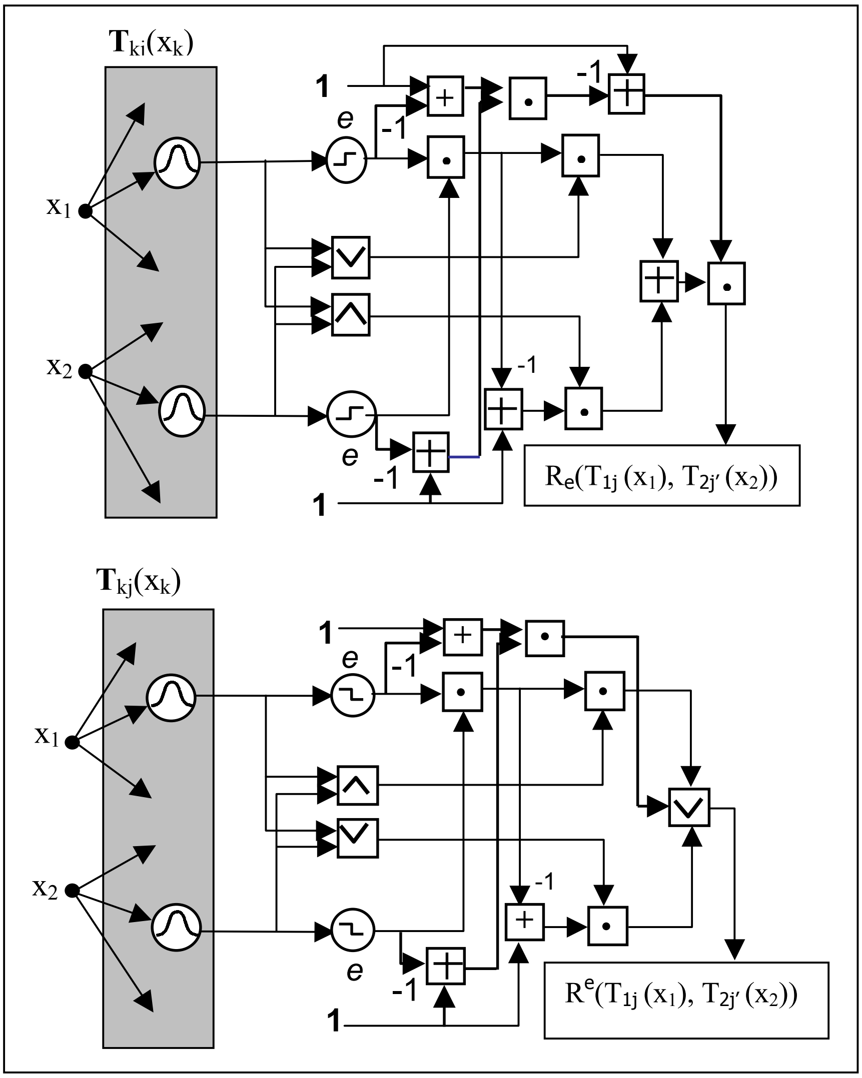

Fig. 3. Network to learn the linguistic terms and the *e*-parameters of R_e based on R_* (top) and of R^e based on R^* (bottom) for one rule, respectively

2.2 Learning a generalized λ-mean

Generalized λ-means [7] represent another kind of parametric combination of t-norms and conorms.

Let T be a t-norm and S be its t-conorm. Then for some $\lambda \in (0,1)$ and all $x_1, x_2 \in [0,1]$ the λ-mean g_λ is defined as follows :

$$g_\lambda(x_1,x_2) = \begin{cases} \min(\lambda, S(x_1,x_2)) & \text{if} \quad x_1, x_2 \in [0,\lambda] \\ \max(\lambda, T(x_1,x_2)) & \text{if} \quad x_1, x_2 \in [\lambda,1] \\ \lambda & \text{otherwise} \end{cases}$$

Let $T(x_1,x_2) = x_1 \cdot x_2$ and $S(x_1,x_2) = x_1 + x_2 - (x_1 \cdot x_2)$. The network shown in Fig. 4, trained with an evolutionary strategy or a real-coded genetic algorithm can learn the parameters of the linguistic terms as well as λ. The dashed line frames the part of the net where the values are calculated. The rest of the net makes the selection according to the values of the inputs.

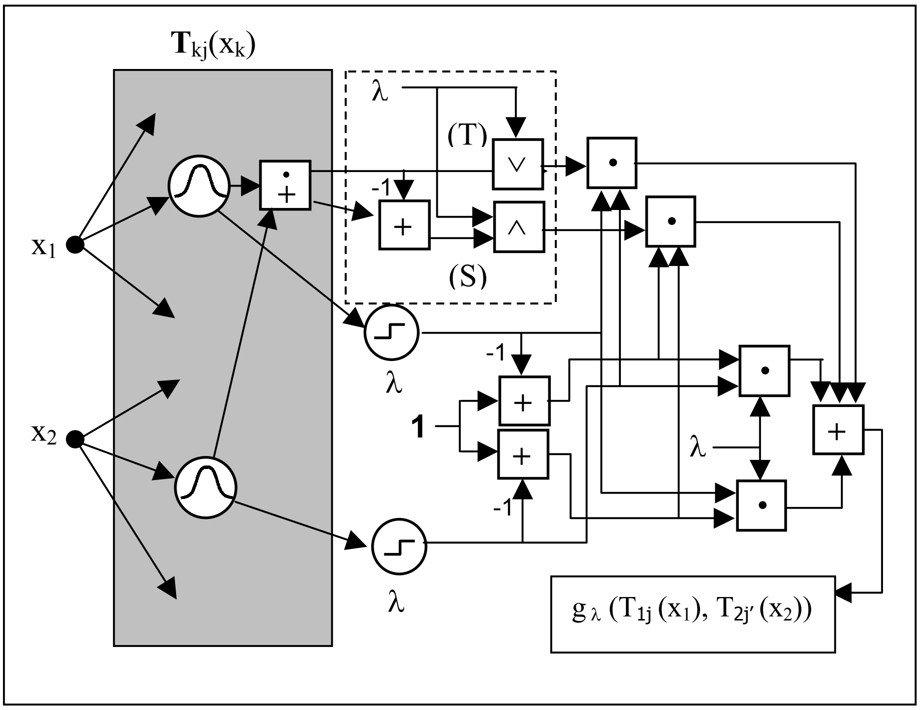

Fig. 4. Network to learn the linguistic terms and the λ-parameter of a generalized λ-mean for one rule

In this as well as in the former cases, the trained network becomes a piece of dedicated hardware for the fast evaluation of the rules. It is fair to say that if there is no *a priori* information on the kind of connectives to be used among the premises, a hypothesis has to be stated. The method explained above finds the proper parameters if the hypothesis is correct, otherwise it rejects the hypothesis by converging to a non-acceptable level of error when trying to fit the available performance data.

References

1. Bersini, H., Bontempi, G.: Now comes the time to defuzzify neuro-fuzzy models. *Fuzzy Sets and Systems* **90** (1997) 161-169
2. Deb, K.: Genetic algorithms for function optimization. In: *Genetic Algorithms and Soft Computing* (F. Herrera, J.L. Verdegay, Eds.). Physica Verlag, Heidelberg (1996)
3. Dubois, D., Prade, H.: A Review of Fuzzy Set Aggregation Connectives, *Information Sciences* **36** (1985) 85-121
4. Fodor, J., Yager, R., Rybalov, A.: Structure of uninorms. *International Jr. of Uncertainty, Fuzziness and Knowledge-based Systems* **5** (4) (1997)411-427
5. Jang, J.S.R.: Self-learning fuzzy controllers based on temporal back propagation, *IEEE Trans. Neural Networks* **3** (5) (1992) 714-721
6. Jang, J.S.R., Sun, C.T.: Functional equivalence between radial basis function networks and fuzzy inference systems, *IEEE Trans. Neural Networks* **4** (1) (1993) 156-159
7. Klir, G.J., Yuan, B.: *Fuzzy Sets and Fuzzy Logic: Theory and Applications.* Prentice Hall, N.J. (1995)
8. Moraga, C.: Neuro-fuzzy modeling of compensating systems. In: *Quo vadis Computational Intelligence?*, (P. Sinčák, J. Vaščák, Eds.), 385-398. Physica-Verlag, Heidelberg (2000)
9. Pal, S.K., Mitra, S: Multilayer perceptron, fuzzy sets, and classification, *IEEE Trans. Neural Networks* **3** (5) (1992) 683-697
10. Rechenberg I.: *Evolutionstrategie.* Friedrich Frommann Verlag, Stuttgart (1972)
11. Su, M.-C., Chang, H.-T.: Application of neural networks incorporated with real-valued genetic algorithms in knowledge acquisition, *Fuzzy Sets and Systems* **112** (1) (2000) 85-98
12. Takagi, T., Sugeno, M.: Fuzzy identification of systems and its application to modeling and control, *IEEE Trans. Man, Systems, and Cybernetics* **15** (1) (1985) 116-132
13. Weber, S.: A general concept of fuzzy connectives, negations and implications based on t–norms and t–conorms", *Fuzzy Sets and Systems* **11** (1983) 115-134
14. Yager, R., Rybalov, A.: Uninorm aggregation operators. *Fuzzy Sets and Systems* **80** (1996) 111-120