METHOD TO IMPROVE THE TRANSPARENCY OF NEUROFUZZY SYSTEMS

J. A. Domínguez-López
Centro de Investigación en Matemáticas (CIMAT)
Callejon de Jalisco s/n, Guanajuato CP36240, MEXICO

Keywords: Fuzzy control, neurofuzzy systems, transparency, learning, curse of dimensionality.

Abstract: Neurofuzzy systems have been widely applied to a diverse range of applications because their robust operation and network transparency. A neurofuzzy system is specified by a set of rules with confidences. However, as knowledge base systems, neurofuzzy systems suffer from the curse of dimensionality i.e., exponential increase in the demand of resources and in the number of rules. So, the interpretability of the final model can be lost. Thus, it is desired to have a simple rule-base to ensure transparency and implementation efficiency. After training, a rule can have several non-zero confidences. The more number of non-zero confidences, the less transparent the final model becomes. Therefore, it is elemental to reduce the number of non-zero confidences. To achieve this, the proposed algorithm search for (a maximum of) two non-zero confidences which give the same result. Thus, the system can keep its complexity with a better transparency. The proposed methodology is tested in a practical control problem to illustrate its effectiveness.

1 INTRODUCTION

Neurofuzzy systems have been successfully applied to a diverse range of applications because they combine the well-established modelling and learning capabilities of neural networks with the transparent knowledge representation of fuzzy systems. The fuzzy system is defined as a neural network type structure keeping its fundamental components. In this way, a fuzzy system can be derived from data or improved by learning from interaction with the environment. After training the neurofuzzy network, the final model is represented by the rule-base, which consists of N rules of IF-THEN form. A specific rule effectively describes the input-output relation of the system and at any time all the rules fire to a degree. The sum of all the firing rules gives the overall output.

The rule confidences are a degree of relationship between the input fuzzy set(s) and the output fuzzy set(s). When the rule confidence is zero, the input(s) is not related to the output(s) so the rule does not fire. Otherwise, the input(s) and output(s) have some relation and the rule partially fires when the membership degree of the rule antecedent is greater than zero. A rule can have several non-zero confidences. When there are many rules with several non-zero confidences, the network transparency is poor. Transparency is a desirable feature that allows us to understand the influence of each weight (or rule confidence) in the result network output. To improve the transparency of neurofuzzy systems, the proposed method searches for (a maximum of) two non-zero confidences which give the same result. The effectiveness of this proposed methodology is tested in a practical control problem.

2 NEUROFUZZY STRUCTURE

A neurofuzzy system consists of various components of a traditional fuzzy system, with the exception that each stage is performed by a layer of hidden neurons. Figure 1 shows the implementation of a neurofuzzy system. Each circle represents a neuron. These neurons are fuzzy rather than McCulloch-Pitts neurons. This means that they perform fuzzy operations (e.g., fuzzification, rule firing strength, defuzzification). A fuzzy production rules relates the networks inputs, $\mathbf{x}$, to its output, y. During training, this knowledge, which is represented as IF-THEN sentences, is stored. A fuzzy production rule is of the form:

$$r_{ij} : \text{IF } \mathbf{x} \text{ is } A^i \text{ THEN y is } B^j,\ c_{ij}$$

Domínguez-López J. (2005).
METHOD TO IMPROVE THE TRANSPARENCY OF NEUROFUZZY SYSTEMS.
In *Proceedings of the Second International Conference on Informatics in Control, Automation and Robotics*, pages 297-300

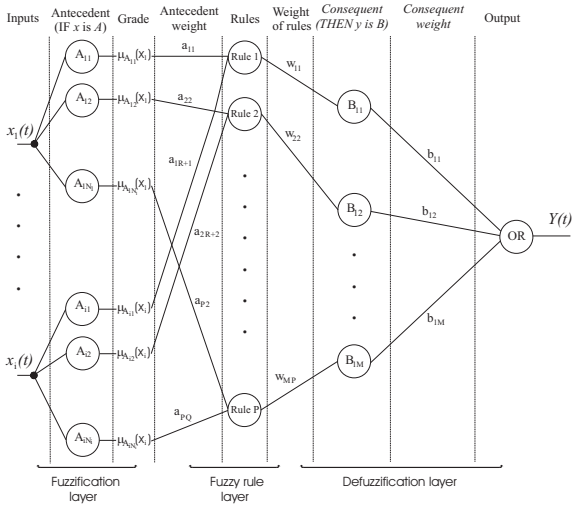

Figure 1: Architecture of a typical neurofuzzy system. The number of rules is given by $P = \prod_i N_i$, where i is the number of inputs and N_i is the number of antecedent fuzzy sets for input i while M is the number of consequent fuzzy sets. Also we have $Q = \sum_i N_i$ and $R = \sum_{k=0}^{i-1} N_k$.

where r_{ij} is the ijth rule, A^i represents the input fuzzy sets, B^j is the output fuzzy set and the rule confidence is $c_{ij} \in [0,\, 1]$. The value of c_{ij} indicates the degree of confidence in the relationship between A^i and B^j. When c_{ij} is zero the rule is inactive and does not affect the output. Otherwise, the rule is active and contributes to the output according to the degree of activation of the antecedent. Subsequently, the fuzzy rules can be learned by adapting the rule confidences, changing the strength with which a rule fires.

Once, the neurofuzzy controller has been designed and constructed, the objective of the selected learning algorithm is to determine the appropriate values for the parameters of the membership functions and the linking weights (Chen and Peng, 1999). The weights of the antecedent and consequent require as many parameters as modifiable parameters of the membership functions. So, it is common that instead of a weight vector, $\overline{w}$, it is a weight matrix, $\mathbf{w}$. For instance, the triangular membership functions have three parameters that can be updated. This leads to have several free parameters to update, slowing the learning process. In addition, the resulted membership distribution may not be as transparent as with the designer's distribution. For example, in (Berenji and Khedkar, 1992), before learning, the membership 'positive small' is in the positive region of the universe of discourse but, after learning, it is in the negative region, losing its meaning. This can be corrected if the system is able to correct inappropriate definitions of the labels. When the neurofuzzy system has only one modifiable weight vector (i.e., the rule confidence vector), leaving the other vectors and the fuzzy memberships fixed, the system can still describe completely the input-output mapping for a The use of rule confidences rather than a weight vector allows the model to be represented as a set of transparent fuzzy rules (Brown and Harris, 1994). However, using a rule weight vector reduces considerable the storage requirements and the computational cost (Harris et al., 2002, p. 92). Nevertheless, it is possible to alternate between the rule weight vector and the rule confidence without losing any information.

The transformation from the weight vector, w_i, to the vector of rule confidence, $\mathbf{c}_i$, is a one-to-many mapping. The weight vector can be converted into confidences by measuring its grade of membership to the various fuzzy output sets, $\mu_{B^j}(\cdot)$:

$$\mathbf{c}_{ij} = \mu_{B^j}(w_i)$$

The inverse transformation, from $\mathbf{c}_i$ to w_i, is given by:

$$w_i = \sum_j \mathbf{c}_{ij}\, y_j^c$$

where y_j^c is the centre of the jth output set $\mu_{B^j}(u)$ which has bounded and symmetric membership functions.

The maximum number of rule confidences, p_c, depends on the number of inputs, n, and the number of fuzzy sets in each input, p_i, and in the output, q:

$$p_c = q \prod_{i=1}^{n} p_i \tag{1}$$

Accordingly, if the number of rules is large because there are many inputs and/or many fuzzy sets per input, transparency can get lost. Consequently, it is important to keep relatively low the number of rules, avoiding redundant ones. In addition, as the input dimension increases, the requirement of resources (data, memory, processing time, ...) increases exponentially (Bossley, 1997). Therefore, fuzzy systems suffer from the *curse of dimensionality* (Bellman, 1961). Consequently, practical fuzzy and neurofuzzy systems are reduced to problems with input dimension typically less than four. However, if some form of model complexity reduction is applied, fuzzy and neurofuzzy systems can be used to solve high dimensional problems (Harris et al., 2002) and still be *transparent*.

3 FUZZY INFERENCE ENGINE

The fuzzy inference engine evaluates the control rules stored in the rule-base. It performs four main tasks: rule firing, strength calculation, fuzzy implication and rule aggregation. The current fuzzy input set is

matched with the antecedent of all rule to produces the fuzzy output set. Then, the activated output fuzzy sets are defuzzified to obtain the control action. The output represents the degree of relationship between the input and each output fuzzy set (Harris et al., 2002). The degree of relationship between the system inputs $\mathbf{x}$ and the system output y is given by:

$$\mu_{r_{ij}}(\mathbf{x},\, y) = \mu_{A^i}(\mathbf{x}) \hat{*} \mathbf{c}_{ij} \hat{*} \mu_{B^i}(y)$$

where $\hat{*}$ represents T-norm operation (e.g., min and algebraic product functions), $\mu_{B^i}(y)$ is the fuzzy output set and the fuzzy input set is $\mu_{A^i}(\mathbf{x})$, which is obtained from the fuzzy intersection (AND) of n-univariate fuzzy sets:

$$\mu_{A^i}(\mathbf{x}) = \mu_{A_1^i}(x_1) \hat{*} \cdots \hat{*} \mu_{A_n^i}(x_n)$$

In order to form a fuzzy rule $\mathbf{R}$, all the individual relations of input-output sets, $\mu_{r_{ij}}$, have to be connected using a multivariable S-norm operator, $\widehat{\sum}$:

$$\mu_{\mathbf{R}}(\mathbf{x},\, y) = \widehat{\sum_{i,\,j}} \mu_{r_{ij}}(\mathbf{x},\, y)$$

To produce a single fuzzy output set, $\mu_B(y)$, the fuzzy inference machine matches the current inputs with the antecedents of all rules. This inference is given by:

$$\mu_B(y) = \widehat{\sum_{\mathbf{x}}} (\mu_{A^i}(\mathbf{x}) \hat{*} \mu_{\mathbf{R}}(\mathbf{x},\, y)) \qquad (2)$$

Finally, this single fuzzy output has to be converted back into a real-value output. This is done performing a defuzzification operation. The most widely used method is the centre of gravity (CoG) because it consistently produces better results than the other methods (Bossley, 1997). The output of a fuzzy system that uses CoG defuzzification method is given by:

$$y(\mathbf{x}) = \frac{\int_Y \mu_B(y) y \,\mathrm{d}y}{\int_Y \mu_B(y) \,\mathrm{d}y} \qquad (3)$$

Using Equation 2 in Equation 3, the real-valued output becomes:

$$y(\mathbf{x}) = \frac{\int_Y \int_X \mu_A(\mathbf{x}) \hat{*} \widehat{\sum}_{ij} \mu_{r_{ij}}(\mathbf{x},\, y)\, y \,\mathrm{d}\mathbf{x}\mathrm{d}y}{\int_Y \int_X \mu_A(\mathbf{x}) \hat{*} \widehat{\sum}_{ij} \mu_{r_{ij}}(\mathbf{x},\, y) \,\mathrm{d}\mathbf{x}\mathrm{d}y} \qquad (4)$$

When the fuzzy output sets are bounded and symmetric, the i^{th} rule confidence vector $\mathbf{c}_i$ is normalised (i.e., $\sum_j \mathbf{c}_i = 1$), and the fuzzy inputs sets form a partition of unity, Equation 4 reduces to:

$$y(\mathbf{x}) = \frac{\int_X \mu_A(\mathbf{x}) \,\hat{*} \widehat{\sum}_i \mu_{A^i}(\mathbf{x}) \hat{*} \widehat{\sum}_j \mathbf{c}_{ij}\, y_j^c \,\mathrm{d}\mathbf{x}}{\int_X \mu_A(\mathbf{x}) \,\mathrm{d}\mathbf{x}} \qquad (5)$$

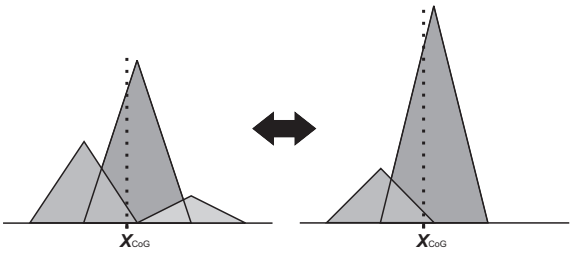

Figure 2: Two non-zero rule confidences can represent three or more non-zero rule confidences.

Accordingly, the prosed method searches a $\mathbf{c}'_{ij}$ that satisfies:

$$\widehat{\sum_j} \mathbf{c}_{ij}\, y_j^c = \widehat{\sum_j} \mathbf{c}'_{ij}\, y_j^c$$

where $\mathbf{c}'_j$ has only two non-zero values and $\mathbf{c}_j$ has three or more non-zero values.

In order to find $\mathbf{c}'_j$, the method searches for two non-zero rule confidences that have the same centre of gravity as the original $\mathbf{c}_j$. So, both rule confidences have identical contribution in the obtention of the real-valued output. This is illustrated in Figure 2, where two non-zero rule confidences represent three.

4 EXPERIMENTAL EXAMPLE: THE CART-POLE BALANCING PROBLEM

To illustrate the proposed method and to show its effectiveness, the neurofuzzy system described in Section 2 is used in a very popular problem to test controllers: the cart-pole balancing problem. The controller has to apply a sequence of right and left forces to the cart such the pole remains balanced. The controller fails if either the pole falls or the cart hits the track end. The implemented neurofuzzy controller has two inputs, the pole angle and pole angular velocity, and one output, the applied force. The linguistic variables used for the term sets are simply value magnitude components: Negative Large (*NL*), Negative Medium (NM), Negative Small (*NS*), Zero (*Z*), Small (*S*), Medium (*M*) and Large (*L*) for the fuzzy sets *pole angle* and *applied force*, while for the fuzzy set *pole angular velocity* they are *NL*, *NS*, *Z*, *S* and *L*. The neurofuzzy controller was trained with reinforcement learning, using the training framework described in (Domínguez-López et al., 2004). After training, the total number of non-zero rule confidences was 92. Then, the proposed method is used to improve the transparency. The method gives a total of 65 non-zero rule confidences. This final model is

Table 1: Rule-base and rule confidences (in brackets) obtained after applying the proposed method. There are 65 non-zero rule confidences.

Applied force		Pole angular velocity				
		NL	*NS*	*Z*	*S*	*L*
Pole angle	*NL*	L (0.9) M (0.1)	L (0.7) M (0.3)	L (0.6) M (0.4)	L (0.3) M (0.7)	L (0.2) M (0.8)
	NM	L (0.8) M (0.2)	L (0.7) M (0.3)	L (0.2) M (0.8)	M (1.0)	L (0.3) M (0.7)
	NS	M (1.0)	M (0.8) S (0.3)	M (0.3) S (0.7)	S (0.2) Z (0.8)	S (0.1) Z (0.9)
	Z	S (1.0)	M (0.2) S (0.8)	S (0.3) Z (0.7)	Z (0.3) NS (0.7)	Z (0.2) NS (0.8)
	S	Z (1.0)	Z (0.8) NS (0.2)	Z (0.3) NS (0.7)	NM (0.7) NL (0.3)	NS (0.2) NM (0.8)
	M	Z (0.2) NS (0.8)	NM (0.7) NL (0.3)	NM (0.8) NL (0.2)	NM (0.3) NL (0.7)	NM (0.1) NL (0.9)
	L	NM (1.0)	NM (0.7) NL (0.3)	NM (0.4) NL (0.6)	NM (0.3) NL (0.7)	NM (0.1) NL (0.9)

shown in Table 1. This is a reduction of around 40% in the number of rule confidences. So, the proposed method has improved the transparency. Although not shown explicitly here, the performance of both rule-bases was identical: Both rule-bases were tested 30 times with no failure in each run of 10 min.

5 CONCLUSION

One of the major advantages of neurofuzzy systems is their transparency, so even a non-specialist in control can understand and manipulate the rule-base. However, the interpretability of the final model can be lost if after training each rule has a large number of non-zero confidences. Nevertheless, the transparency can be recovered without affecting the system complexity and performance. To improved the transparency, the proposed methodology searched for a maximum of two non-zero confidences per rule. These two non-zero confidences have the same contribution as the original ones, thus the system performance is not affected. In the illustrative example, the transparency was improved, with a reduction of 40% in the number of non-zero rule confidences. The improvement depends on the transparency level of the original rule-base. When the original transparency level is *good*, the improvement will be *low*, and if the level is *poor*, the improvement will be *high*. Using Equation 1 we can obtain the maximum and minimum improvements: $p_c(q=1)/p_c$ and $p_c(q=2)/p_c$, respectively. Notice that here q is used as number of non-zero rule confidences per rule rather than number of fuzzy sets in the output.

Finally, the method described here does not remedy the curse of dimensionally. It only heals the redundancy in the rule confidences. Thus, the proposed methodology should be used in conjunction with some form of model complexity reduction (e.g., parsimonious modelling) in order to solve high dimensional problems. In this way, it is possible to guarantee that the final model would have the best accuracy and transparency trade-off.

REFERENCES

Bellman, R. (1961). *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton, NJ.

Berenji, H. and Khedkar, P. (1992). Learning and tuning fuzzy logic controllers through reinforcements. *IEEE Transactions on Neural Networks*, 3(5):724–740.

Bossley, K. M. (1997). *Neurofuzzy Modelling Approaches in System Identification*. PhD thesis, University of Southampton, Southampton, UK.

Brown, M. and Harris, C. J. (1994). *Neurofuzzy Adaptive Modelling and Control*. Prentice Hall International, New York, NY.

Chen, C. T. and Peng, S. T. (1999). Intelligent process control using neural fuzzy techniques. *Journal of Process Control*, 9(6):493–503.

Domínguez-López, J. A., Damper, R. I., Crowder, R. M., and Harris, C. J. (2004). Adaptive neurofuzzy control of a robotic gripper with on-line machine learning. *Robotics and Autonomous Systems*, 48(2-3):93–110.

Harris, C. J., Hong, X., and Gan, Q. (2002). *Adaptive Modelling, Estimation and Fusion from Data: A Neurofuzzy Approach*. Springer-Verlag, Berlin and Heidelberg, Germany.