Structure Determination in Fuzzy Modeling: A Fuzzy CART Approach

Jyh-Shing Roger Jang
The MathWorks, Inc., 24 Prime Park Way, Natick, Mass. 01760
jang@eecs.berkeley.edu or jang@mathworks.com

Abstract

This paper presents an innovative approach to the structure determination problem in fuzzy modeling. By using the well-known CART (classification and regression tree) algorithm as a quick preprocess, the proposed method can roughly estimate the structure (numbers of membership functions and number of fuzzy rules, etc.) of a fuzzy inference system; then the parameter identification is carried out by the hybrid learning scheme developed in our previous work [3, 2, 5]. Morevoer, the identified fuzzy inference system has the property that the total of firing strengths is always equal to one; this speeds up learning processes and reduces round-off errors.

1 Introduction

Fuzzy modeling [11, 10] is a new branch of system identification which concerns with the construction of a fuzzy inference system (or fuzzy model) that can predict and hopefully explain the behavior of an unknown system described by a set of sample data. Two primary tasks of fuzzy modeling are **structure determination** and **parameter identification**; the former determines the numbers of membership functions and fuzzy if-then rules while the latter identifies a feasible set of parameters under the given structure.

To tackle the problem of parameter identification, we have proposed the ANFIS (Adaptive-Network-based Fuzzy Inference System) architecture [3, 2, 5] that can identify a feasible set of parameters by a hybrid learning rule combining the backpropagation gradient descent and the least-squares method. Applications and properties of ANFIS were further investigated in [4, 6]. However, the problem of structure determination was not solved formally.

Based on the CART (classification and regression tree) algorithm, this paper proposes a quick method to solve the problem of structure determination. The proposed method generates a tree partition of the input space, which relieves the problem of "curse of dimensionality" (number of rules goes up exponentially with number of inputs) associated with grid partition. Moreover, the resulting ANFIS is more efficient because of its implicit weight normalization.

This paper is organized into five sections. In the next section, the basics of ANFIS is introduced. Section 3 explains briefly the CART algorithm. The proposed method of structure determination and the new ANFIS architecture are detailed in section 4. Section 5 gives a concluding remark.

2 ANFIS

This section introduces the basic architecture and the hybrid learning rule of ANFIS. For a detailed coverage, see [2, 5].

Considering a first-order TSK (Takagi, Sugeno and Kang) fuzzy inference system [11, 10] which contains two rules:

$$\text{Rule 1: If } X \text{ is } A_1 \text{ and } Y \text{ is } B_1, \text{ then } f_1 = p_1 x + q_1 y + r_1,$$
$$\text{Rule 2: If } X \text{ is } A_2 \text{ and } Y \text{ is } B_2, \text{ then } f_2 = p_2 x + q_2 y + r_2.$$

(If f_1 and f_2 are constants instead of linear equations, then we have zero-order TSK fuzzy model.) Figure 1(a) and (b) illustrate the fuzzy reasoning mechanism and the corresponding ANFIS architecture, respectively. Node functions in the same layer of ANFIS are of the same function family, as described below. (Note that O_i^j denotes the output of the i-th node in layer j.)

0-7803-1896-X/94 $4.00 ©1994 IEEE

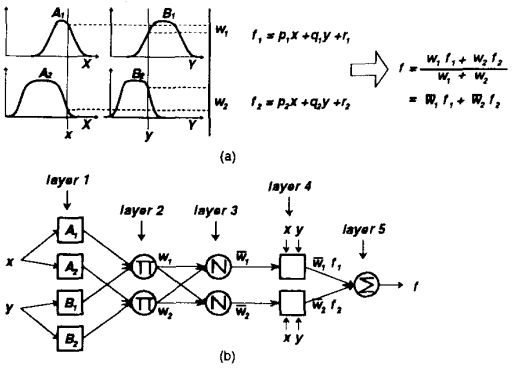

Figure 1: *(a) First-order TSK fuzzy model; (b) corresponding ANFIS architecture.*

Layer 1 Each node in this layer generates a membership grades of a linguistic label. For instance, the node function of the i-th node might be

$$O_i^1 = \mu_{A_i}(x) = \frac{1}{1 + [(\frac{x - c_i}{a_i})^2]^{b_i}}, \tag{1}$$

where x is the input to node i; A_i is the linguistic label (*small* , *large* , etc.) associated with this node; and $\{a_i, b_i, c_i\}$ is the parameter set that changes the shapes of the membership function. Parameters in this layer are referred to as the *premise parameters*.

Layer 2 Each node in this layer calculates the firing strength of each rule via multiplication:

$$O_i^2 = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \ i = 1, 2. \tag{2}$$

Layer 3 The i-th node of this layer calculates the ratio of the i-th rule's firing strength to the sum of all rules' firing strengths:

$$O_i^3 = \overline{w}_i = \frac{w_i}{w_1 + w_2}, \ i = 1, 2. \tag{3}$$

Layer 4 Node i in this layer has the following node function

$$O_i^4 = \overline{w}_i f_i = \overline{w}_i(p_i x + q_i y + r_i), \tag{4}$$

where $\overline{w}_i$ is the output of layer 3, and $\{p_i, q_i, r_i\}$ is the parameter set. Parameters in this layer will be referred to as the *consequent parameters*.

Layer 5 The single node in this layer computes the overall output as the summation of all incoming signals:

$$O_1^5 = \textit{overall output} = \sum_i \overline{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{5}$$

Thus we have constructed an adaptive network in Figure 1(b) which is functionally equivalent to a fuzzy inference system in Figure 1(a). This adaptive network is called ANFIS, which stands for adaptive-network-based fuzzy inference systems.

The basic learning rule of ANFIS is the back-propagation gradient descent [12], which calculates error signals (defined as the derivative of the squared error with respect to each node's output) recursively from the output layer backward to the input nodes. This learning rule is exactly the same as the back-propagation learning rule used in the the common feedforward neural networks [9].

From the ANFIS architecture in Figure 1, it is observed that given the values of premise parameters, the overall output f can be expressed as a linear combinations of the consequent parameters:

$$\begin{aligned} f &= \overline{w}_1 f_1 + \overline{w}_2 f_2 \\ &= (\overline{w}_1 x)p_1 + (\overline{w}_1 y)q_1 + (\overline{w}_1)r_1 + (\overline{w}_2 x)p_2 + (\overline{w}_2 y)q_2 + (\overline{w}_2)r_2. \end{aligned} \tag{6}$$

Based on this observation, we have proposed a hybrid learning algorithm [2, 5] which combines the gradient descent and the least-squares method to find a feasible set of parameters. Both on-line and off-line learning paradigms are supported, see [5] for details.

3 Decision Tree and CART Algorithm

A. Decision Tree

A binary decision tree is a tree structure that consists of internal nodes (with two children) and terminal nodes (without children). Each internal node is associated with a decision function to indicate which node to visit next; while each terminal node shows the output of a given input vector that leads the visit to this node. For classification problems, each terminal node contains an alphabet that indicates the predicted class of a given feature vector. In contract, for regression problems, the terminal nodes usually contain a constant that is the output value of the given input vector. Figure 2 (a) is a typical binary decision tree for regression purpose, where the inputs are x and y and the output is z. Obviously the decision tree partitions the input space into four nonoverlapping rectangular regions (see Figure 2 (b)), each is assigned a constant f_i as the output value. Figure 3 (a) is the surface plot of the overall input-output behavior, with $a = 6$, $b = 7$, $c = 3$, $f_1 = 9$, $f_2 = 5$, $f_3 = 3$ and $f_4 = 1$.

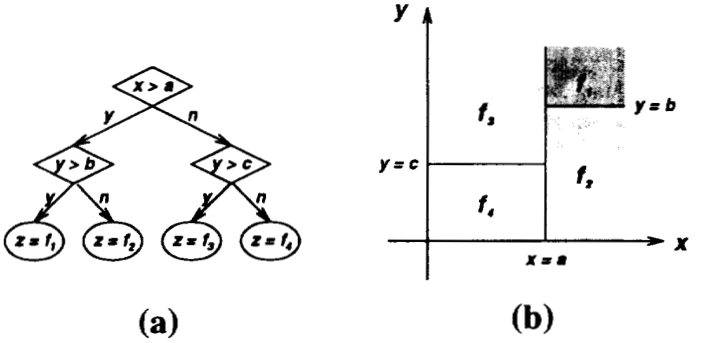

Figure 2: *(a) A binary decision tree and (b) its input space partitioning.*

If we assign a linear function of the input variables to each terminal node, then the resulting surface will be piecewise linear as shown in Figure 3 (b), where $f_1 = 3x + 4y + 20$, $f_2 = 6x - y + 5$, $f_3 = -2x + 2y + 10$, and $f_4 = 2x - y - 20$.

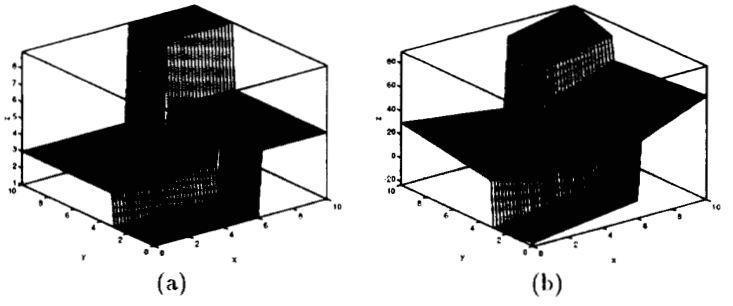

Figure 3: *Input-output behaviors of decision trees with terminal nodes characterized by constants and linear equations, respectively.*

Apparently the decision tree is a very easy-to-interpret representation of a nonlinear input-output mapping. However, the discontinuity at the decision boundaries (say, $x = 6$ in Figure 3 (a) and (b)) is unnatural and it brings undesired effects to the overall regression and generalization.

B. Classification and Regression Trees (CART) Algorithm

The use of tree-based regression goes back to the AID (Automatic Interaction Detection) program of Morgan and Sonquist [8]. A complete treatment of this methodology was developed by Breiman et al. [1] in their book entitled **Classification and Regression Trees**; thus the methodology is often referred to as

the CART algorithm. We will briefly summarize our use of the CART procedure to build a regression tree; the reader is directed to the cited reference for the complete CART methodology.

To construct an appropriate regression tree, CART first grows the tree extensively based on the training data set, and then prunes the tree back based on a minimum cost-complexity principle [1]. The result is a sequence of trees of various sizes; the final tree is picked up as the tree that performs best when the test data set is presented.

More specifically, CART grow a regression tree by determining a succession of splits (decision boundaries) of a sample of training data. Starting from the root node (which contains all the training data), an exhaustive search is make for the best split that can reduce a error measure (usually squared error) most. Once the best split is determined, the procedure is repeated at the two child nodes that are subsequently formed. This recursive procedure terminates either when the error measure associate with a node is below a certain tolerance, or when the error reduction of further splitting will not exceed a certain threshold.

The tree obtained by the above growing procedure is often too large, and it is biased toward the training data set, yielding an untrustworthy high accuracy on reproducing desired outputs of the training data. Based on the principle of minimum cost-complexity, a sequence of candidate trees can be obtained by pruning terminal nodes sequentially, where each pruning should result a minimal increase in the error measure. Then from these candidate trees, we pick the one that can generate minimal error measure when the test (validating) data set is presented.

For terminal nodes with constant output values (see Figure 3 (a)), CART can always construct an appropriate tree with a right size and, at the same time, find which inputs are irrelevant and thus not used in the tree. On the other hand, if the terminal nodes are characterized by linear equations (Figure 3 (b)), then the irrelevant inputs are harder to find unless more computation are involved. Also for terminal nodes with linear equations, the search of the best split can be more efficient if we employ the sequential least-squares method (see, for example, [7]) to identify the linear coefficients.

4 Combining CART and ANFIS

Obviously the decision tree in Figure 2 is equivalent to a set of crisp rules:

$$\begin{cases} \text{if } x > a \text{ and } y > b, \text{ then } z = f_1. \\ \text{if } x > a \text{ and } y < b, \text{ then } z = f_2. \\ \text{if } x < a \text{ and } y > c, \text{ then } z = f_3. \\ \text{if } x < a \text{ and } y < c, \text{ then } z = f_4. \end{cases} \tag{7}$$

Given any input vector (x, y), only one rule out of four will be fired at full strength while the other three rules are not activated at all. This crispness reduce the computation burden in constructing the tree using CART, but it also gives undesired discontinuous boundaries. To smooth out the discontinuity at each split, we propose the use of fuzzy sets to represent the premise parts of the rule set in equation (7) and thus converting equation (7) into a set of fuzzy if-then rules of either zero-order (when f_i's are constants) or first-order (when f_i's are linear equations) TSK fuzzy model [11, 10]. For instance, the statement $x > c$ can be represented as a fuzzy set characterized by either the sigmoidal membership function (MF) with one parameters α:

$$\mu_{x>c}(x; \alpha) = sig(x; \alpha, c) = \frac{1}{1 + exp[-\alpha(x - c)]}, \tag{8}$$

or the extended S MF with two parameters α, and γ:

$$\mu_{x>c}(x; \alpha, \gamma) = s(x; \alpha, c, \gamma) = \begin{cases} 0 & \text{if } x \le c - \alpha, \\ \frac{1}{2}\left[\frac{x-(c-\alpha)}{\alpha}\right]^{2\gamma} & \text{if } c - \alpha < x \le c, \\ 1 - \frac{1}{2}\left[\frac{c+\alpha-x}{\alpha}\right]^{2\gamma} & \text{if } c < x \le c + \alpha, \\ 1 & \text{if } c + \alpha < x. \end{cases} \tag{9}$$

(Note that when $\gamma = 0.5$, the above S MF becomes a ramp function.) Figure 4 shows the sigmoidal and the S MF's for the linguistic term $x > c$; Figure 5 is the resulting surface plots of Figure 3 when the S MF is used (with $\alpha = 1$ and $\gamma = 1$). Remember that when $\alpha \to \infty$ in the sigmoidal MF or when $\gamma \to \infty$ in the S MF, both MF's reduce to the step function and the fuzzy rules reduce to the original crisp rules.

Based on the fuzzy version of the rules in equation (7), we can derive another class of adaptive network to identify the premise and consequent parameters for the underlying fuzzy inference system. This innovative ANFIS architecture is shown in Figure 6, where layer 1 calculates the membership grades of given input variables (INV nodes represent negation operator); layer 2 multiples the given membership grades to find the firing strength of each rule; layer 3 computes the contribution of each rule based on given firing strengths; and layer 4 find the summation of incoming signals, which is equal to the overall output of this fuzzy inference

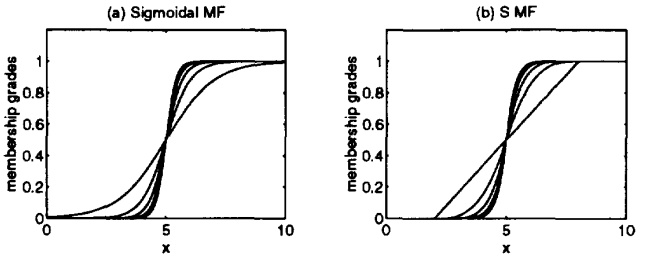

Figure 4: *Two types of MF's for $x > c$ (where $c = 5$): (a) sigmoidal MF with different α's; (b) extended S MF with different γ's.*

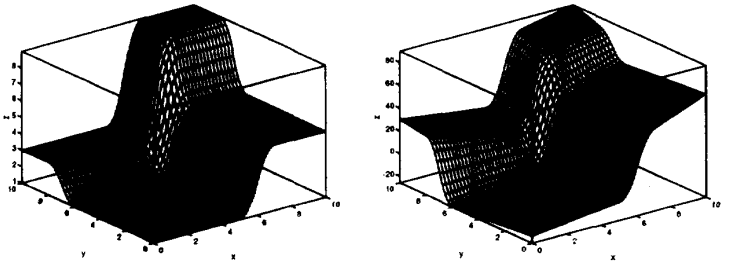

Figure 5: *Input-output behaviors of decision trees with terminal nodes characterized by constants and linear equations, respectively.*

system. Premise and consequent parameters are contained in layer 1 and 3, respectively; these parameters are fine-tuned according to the fast hybrid learning rules developed in [2, 5]. Note that the normalization layer (layer 3) in Figure 1 is missing in Figure 6. This is attributed to the following theorem.

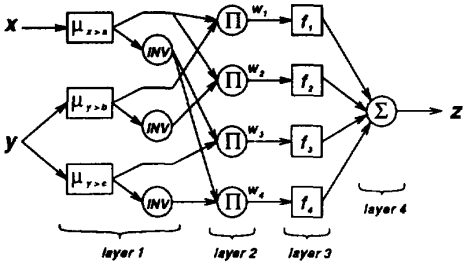

Figure 6: *ANFIS architecture corresponding to the fuzzy version of the rule set in equation (7).*

Theorem 4.1 *In converting a decision tree to a fuzzy inference system, if (1) $\mu_{x>a}(x)+\mu_{x\leq a}(x) = 1$, where x is any of the input variables and a is any of the splitting points of x; and (2) multiplication is used as the T-norm operator to calculate each rule's firing strength, then the summation over each rule's firing strength is always equal to one.*

Proof: This theorem can be prove by induction. Let n be the number of rules and w_i, $i = 1, \ldots n$ be the firing strength of the ith rule. For $n = 2$, we have $w_1 + w_2 = 1$ since w_1 and w_2 are the membership grades for $\mu_{x>a}(x)$ and $\mu_{x\leq a}(x)$ for certain input X and certain split point a.

Suppose that $\sum_{i=1}^{n} w_i = 1$ holds when $n = k$. When $n = k + 1$, we want to show that $\sum_{i=1}^{k+1} w_i = 1$ still holds. Without loss of generality, we can assume the newly generated rules are rules k and $k + 1$, which are

the result of the splitting at the previously terminal node k (or rule k). Consequently, we have

$$\begin{aligned} \sum_{i=1}^{k+1} w_i &= \sum_{i=1}^{k-1} w_i + w_k + w_{k+1} \\ &= \sum_{i=1}^{k-1} w_i + \hat{w}_k(\mu_{x>a}(x) + \mu_{x\leq a}(x)) \\ &= \sum_{i=1}^{k-1} w_i + \hat{w}_k \\ &= 1, \end{aligned}$$

where $\hat{w}_k$ is the firing strength of rule k before splitting. This concludes the proof.

□

These two constrains are satisfied through out the learning process of the ANFIS architecture in Figure 6; this eliminates the need for another normalization layer and thus reducing the computation burden and round-off errors.

However, due to the time constraint, we do not have simulation results at this moment. Extensive simulation results will be presented along with this paper at the conference.

5 Concluding Remarks: Advantages and Problems Solved

We have proposed a CART plus ANFIS approach to complete the two tasks of fuzzy modeling, that is, structure determination and parameter identification. The major advantages offered by this approach is that we can now quickly determine the roughly correct structure of a fuzzy inference through CART, and then refine the MF's and output functions via an efficient ANFIS architecture without normalization layer. Note CART can select relevant inputs and do tree partition (instead of grid partition which causes the problem of "curse of dimensionality") of the input space; while ANFIS refine the regression and make it smooth and continuous everywhere. Thus it can be seen that CART and ANFIS are complementary and their combination makes a solid approach to fuzzy modeling.

References

[1] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, Inc., Belmont, California, 1984.

[2] J.-S. Roger Jang. Fuzzy modeling using generalized neural networks and Kalman filter algorithm. In *Proc. of the Ninth National Conference on Artificial Intelligence (AAAI-91)*, pages 762–767, July 1991.

[3] J.-S. Roger Jang. Rule extraction using generalized neural networks. In *Proc. of the 4th IFSA World Congress*, pages 82–86 (in the Volume for Artificial Intelligence), July 1991.

[4] J.-S. Roger Jang. Self-learning fuzzy controller based on temporal back-propagation. *IEEE Trans. on Neural Networks*, 3(5):714–723, September 1992.

[5] J.-S. Roger Jang. ANFIS: Adaptive-network-based fuzzy inference systems. *IEEE Trans. on Systems, Man, and Cybernetics*, 23(03), May 1993.

[6] J.-S. Roger Jang and C.-T. Sun. Functional equivalence between radial basis function networks and fuzzy inference systems. *IEEE Trans. on Neural Networks*, 4(1):156–159, January 1993.

[7] L. Ljung. *System identification: theory for the user*. Prentice-Hall, Englewood Cliffs, N.J., 1987.

[8] J. N. Morgan and J. A. Sonquist. Problems in the analysis of survey data, and a proposal. *Journal of American Statistics Association*, 58:415–434, 1963.

[9] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and James L. McClelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, volum 1*, chapter 8, pages 318–362. The MIT Press, 1986.

[10] M. Sugeno and G. T. Kang. Structure identification of fuzzy model. *Fuzzy Sets and Systems*, 28:15–33, 1988.

[11] T. Takagi and M. Sugeno. Fuzzy identification of systems and its applications to modeling and control. *IEEE Trans. on Systems, Man, and Cybernetics*, 15:116–132, 1985.

[12] P. Werbos. *Beyond regression: New tools for prediction and analysis in the behavioral sciences*. PhD thesis, Harvard University, 1974.