graphs in Fig. 8 offers a sequence of partitions. The partitions are ordered by the inclusion and thus the partition sequence shows a hierarchical partitions. The hierarchical partitions of our example is shown by a partition tree in Fig. 9.

Therefore we have obtained hierarchical partitions of a system from a cover described by a hypergraph. Furthermore, the partition tree shows the relative logical distance between vertices of the system. For example, the edge $E_{5,0} = \{h, i\}$ is grouped into a macro-vertex at the first iteration, $E_{1,1} = \{a, b, c\}$ at the second iteration, etc. Then, the logical distance between a and b is greater than that between h and i.

V. Conclusion

We have developed a reduction method of hypergraph. In the reduction, an edge is merged into a macro-vertex. The reduction is realized by iterations, and the iterations provide a sequence of reductions.

In the reduction, an edge is merged into a macro-vertex and thus a macro-vertex in a reduced hypergraph represents an edge (subgraph). Therefore a reduced graph can give a partition of a system.

The sequence of reductions provides a sequence of partitions. The partitions are ordered by the inclusion of partitions. The sequence of partitions gives hierarchical partitions of the system. The proposed method allows to reduce the complexity of the system represented by hypergraphs.

References

[1] C. Berge, *Graphes et Hypergraphes.* Paris: Dunod, 1970.

[2] C. H. Papadimitrion and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity.* Englewood Cliffs: Prentice-Hall, 1982.

[3] J. E. Graver and M. E. Watkins, *Combinatorics with Emphasis on the Theory of Graphs.* Berlin: Springer-Verlag, 1977.

[4] B. Bollobas, *Graph Theory an Introductory Course.* New York: Springer-Verlag, 1979.

[5] H. Lee-Kwang and J. Favrel, "Hierarchical reduction and decomposition method for analysis and decomposition of Petri nets," *IEEE Trans. Syst. Man, Cybern.,* vol. SMC-15, pp. 272–280, 1985.

[6] H. Lee-Kwang, J. Favrel, and G. Oh "Hierarchical decomposition of Petri net languages," *IEEE Trans. Syst. Man, Cybern.,* vol. SMC-17, pp. 877–878, 1987.

[7] H. Lee-Kwang, J. Favrel, and P. Baptiste, "Generalized Petri net reduction method," *IEEE Trans. Syst. Man, Cybern.,* vol. SMC-17, no. 2, pp. 297–303, 1987.

[8] H. Lee-Kwang and J. Favrel, "Hierarchical reduction and decomposition of graphs for system analysis," in *IEEE Conf. Syst. Man, Cybern.,* Halifax, 1984, pp. 94–101.

[9] H. Lee-Kwang and K.-M. Lee, "Fuzzy hypergraph and fuzzy partition," *IEEE Trans. Syst. Man, Cybern.,* vol. 25, no. 1, pp. 196–201, Jan. 1995.

[10] S. C. Seth, Narayanaswamy, "A graph model for pattern-sensitive faults in random access memories," *IEEE Trans. Comput.,* vol. C–30, pp. 973–977, Dec. 1981.

[11] M. Franklin and K. K. Saluja, "Hypergraph coloring and reconfigured RAM testing," *IEEE Trans. Comput.,* vol. 43, no. 6, pp. 725–736, Jun. 1994.

[12] R. Swaminathan and D. Veeramani, "Decomposition of {0, 1} matrices," *IEEE Trans. Comput.,* vol. 43, no. 5, pp. 629–633, May 1994.

Simplification of Fuzzy-Neural Systems Using Similarity Analysis

C. T. Chao, Y. J. Chen, and C. C. Teng

***Abstract*—This paper presents a fuzzy neural network system (FNNS) for implementing fuzzy inference systems. In the FNNS, a *fuzzy similarity measure for fuzzy rules* is proposed to eliminate redundant fuzzy logical rules, so that the number of rules in the resulting fuzzy inference system will be reduced. Moreover, a *fuzzy similarity measure for fuzzy sets* that indicates the degree to which two fuzzy sets are equal is applied to combine similar input linguistic term nodes. Thus we obtain a method for reducing the complexity of a fuzzy neural network. We also design a new and efficient on-line initialization method for choosing the initial parameters of the FNNS. A computer simulation is presented to illustrate the performance and applicability of the proposed FNNS. The result indicates that the FNNS still has desirable performance under fewer fuzzy logical rules and adjustable parameters.**

I. Introduction

It is known that conventional approaches to system modeling that are based on mathematical tools, e.g., difference equations, perform poorly in dealing with complex and uncertain systems. The reason is that, in many cases, it is very difficult to find a global function or analytical structure for a nonlinear system. In contrast, fuzzy logic provides an inference morphology that enables approximate human reasoning capability to be applied in a fuzzy inference system. Therefore, a fuzzy inference system employing fuzzy logical rules can model the qualitative aspects of human knowledge and reasoning processes without employing precise quantitative analysis. In recent years, artificial neural networks have also played an important role in solving many engineering problems [6], [12]. Neural networks offer advantages such as learning, adaption, fault-tolerance, parallelism, and generalization. In view of the versatility of neural networks and fuzzy logic, a neural-network-based fuzzy inference system can be expected to exhibit many advantageous features.

The benefits of combing fuzzy logic and neural networks have been explored extensively in the literature, e.g., the fuzzy neural network in [3], [5], [8], the adaptive-network-based fuzzy inference system in [4], and the fuzzy logical system in [17]. The common advantages found in the above systems lie in that 1) they can automatically and simultaneously identify fuzzy logical rules and tune the membership functions, and 2) the parameters of their systems have clear physical meanings, which they do not have in general neural networks. Fuzzy systems utilizing the learning capability of neural networks can successfully construct the input-output mapping for many applications. However, no efficient process for reducing the complexity of a fuzzy neural network has been presented.

The concept of a measure of similarity in fuzzy sets has been applied in pattern recognition [13], fuzzy partitioning [1], pattern classification [14], and the compatibility relation between two fuzzy sets [11]. In these applications, the similarity between two elements or between an element and a fuzzy set are concerned in [1], [13]. Since the similarity in [2], [11], [14], is related to the relationship of two fuzzy sets, it meets the necessity of our research. In [9], Lin and Lee presented an algebraic and geometric derivation to provide

Manuscript received July 10, 1994; revised March 18, 1995.

The authors are with the National Chiao-Tung University, Institute of Control Engineering, Hsinchu 30050, Taiwan, R.O.C.

Publisher Item Identifier S 1083-4419(96)02311-4.

the FSM (Fuzzy Similarity Measure) in [2], [11], [14], with a clear mathematical and physical meaning. However, a fuzzy similarity measure has yet to be applied to reduce the complexity of a fuzzy neural network.

In this paper, we propose a fuzzy neural network system (FNNS) to implement fuzzy inference systems for system modeling. By using the fuzzy similarity measure, we derive simple approximate equations for calculating the degree of similarity of two fuzzy sets, both with bell-shaped membership functions. We present a fuzzy similarity measure for fuzzy rules to eliminate redundant fuzzy logical rules. We also apply the fuzzy similarity measure to combine similar linguistic terms into a single linguistic term to reduce the complexity of the FNNS. Thus we attempt to produce a simpler fuzzy inference system, with fewer fuzzy logical rules, which is more practical and useful in industrial applications. Fig. 1 shows a flow chart of the proposed FNNS.

This paper is organized as follows. Section II describes the structure and learning rules of the FNNS. The similarity measure for fuzzy sets and fuzzy rules is stated in Section III. In Section IV, a new on-line method of initializing the FNNS is presented. In Section V, an example is shown that demonstrates the capabilities of the proposed FNNS. Conclusions are summarized in the last section.

II. Fuzzy Neural Nework System (FNNS)

The initial network structure adopted in the proposed FNNS is shown in Fig. 2. The structure is distinguished by its direct construction of fuzzy rules without any other adjustment. For example, suppose we encounter the jth fuzzy rule described as follows:

$$\text{IF } x_1 \text{ is } A_1^j \text{ and } x_2 \text{ is } A_2^j \text{ and } \cdots, \text{ and } x_n \text{ is } A_n^j \text{ THEN } y \text{ is } \beta^j \tag{1}$$

where A_i^j and β^j are fuzzy sets in $U_i \subset R$ and $V \subset R$, respectively, and $\underline{x} = (x_1, \cdots, x_n)^T \in U_1 \times \cdots \times U_n$ and $y \in V$ are the input and output of the fuzzy inference system, respectively. A connectionist structure based on this fuzzy rule is illustrated in Fig. 3.

The main advantages of the network structure at the initial time are summarized as follows:

1) The network structure allows us to construct a fuzzy inference system *rule by rule.* In other words, we can implement each fuzzy rule without considering the other fuzzy rules.
2) We can directly incorporate human linguistic descriptions or prior expert knowledge (in the form of IF-THEN rules) into the network structure.
3) We do not take an ordinary fuzzy partition of the input space, so the number of rules does not increase exponentially with the number of inputs.
4) Elimination of redundant nodes (rule nodes or term nodes) is also *rule by rule.* This means that if we eliminate a rule node, then the associated term nodes are also removed.

On the other hand, the disadvantage of the network structure is that it requires a large number of term nodes. As we shown in Fig. 2, we require $m \times n$ term nodes in layer two for n inputs and m fuzzy rules at the initial time. We will apply the *fuzzy similarity measure* in Section III to combine similar term nodes corresponding to a fixed input linguistic variable x_i and overcome this problem. Hence we must emphasize that the FNNS does not keep the initial structure after term node combination.

The class of the fuzzy inference system under consideration is a simplified type which uses a singleton to represent the output fuzzy set of each fuzzy logical rule. Thus β^j is the consequence singleton of the jth rule. Let m be the number of fuzzy IF-THEN rules, that is,

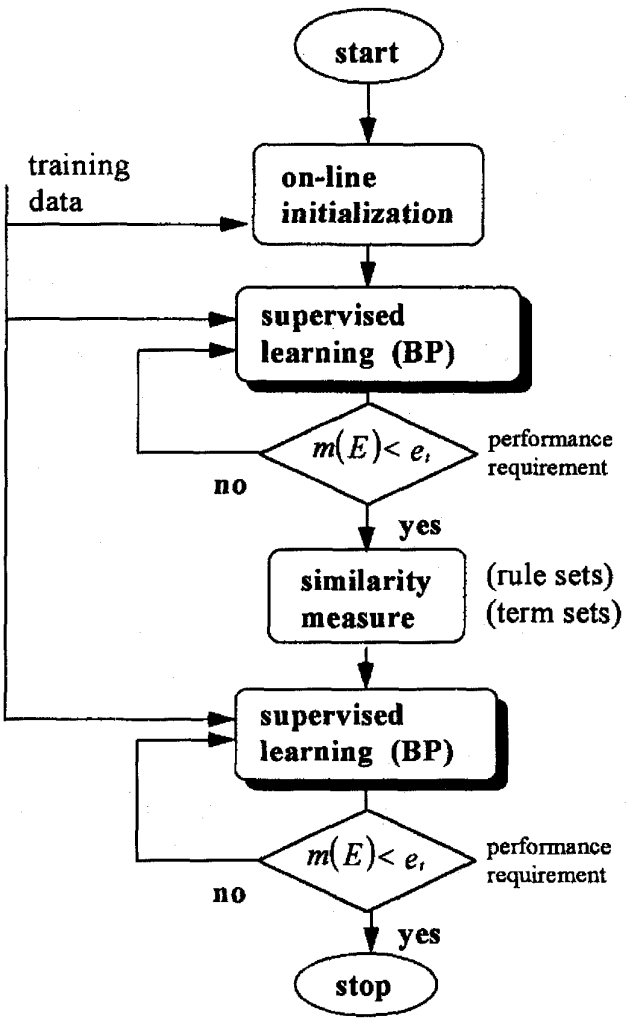

Fig. 1. Flow chart of the proposed FNNS.

$j = 1, 2, \cdots, m$ in (1). The numerical output of the fuzzy inference system with the *center average defuzzifier, product inference rule,* and *singleton fuzzifier* is of the following form:

$$y = \frac{\sum_{j=1}^{m} \beta^j \left[\prod_{i=1}^{n} \mu_{A_i^j}(x_i) \right]}{\sum_{j=1}^{m} \prod_{i=1}^{n} \mu_{A_i^j}(x_i)}, \tag{2}$$

where $\mu_{A_i^j}$ denotes the membership function of fuzzy set A_i^j. This simplified fuzzy inference system is proved to be a universal approximator [16] which is capable of approximating any real continuous function to any desired degree of accuracy, provided sufficiently many fuzzy logical rules are available.

A. Layered Operation of the FNNS

In this subsection, we shall describe the signal propagation and the basic function of every node in each layer. We use net_j^i and f_j^i to denote the summed net input and activation function of the jth node, respectively, in layer i. Moreover, x_j^i and y_j^i denote the input and output vector of the jth node in layer i, respectively.

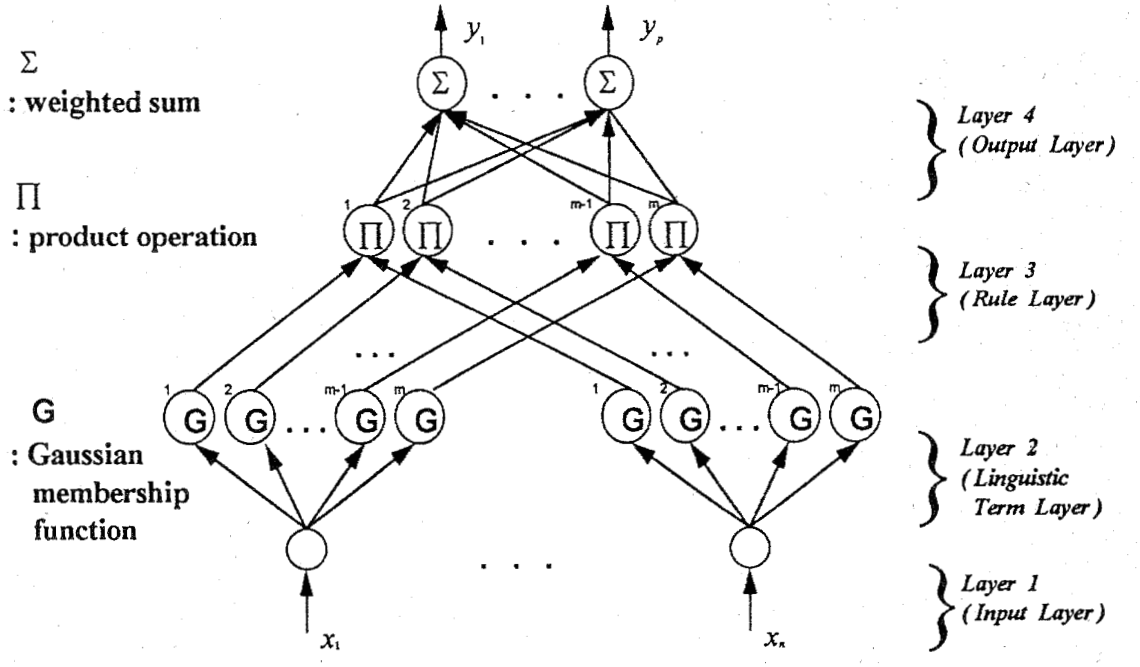

Fig. 2. Structure diagram of the FNNS.

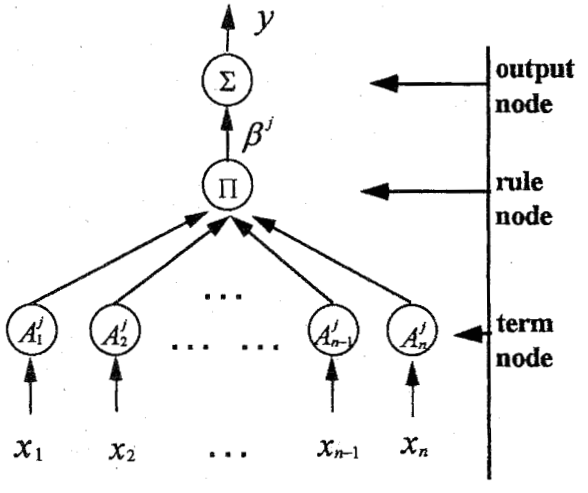

Fig. 3. Construction of the jth rule of the FNNS's network structure.

1) Layer 1: For the jth node of layer one, the net input and the net output are

$$net_j^1 = x_j^1 = x_j$$

and

$$y_j^1 = f_j^1(net_j^1) = net_j^1.$$

2) Layer 2: In this layer, each node performs a membership function. The Gaussian function, a particular example of radial basis functions, is adopted here as the membership function. Then

$$net_{ij}^2 = -\frac{(x_i^2 - m_{ij})^2}{(\sigma_{ij})^2}$$

and

$$y_{ij}^2 = f_{ij}^2(net_{ij}^2) = \exp(net_{ij}^2)$$

where m_{ij} and σ_{ij} are, respectively, the mean (or center) and the variance (or width) of the Gaussian function in the jth term node of the ith input linguistic variable x_i^2.

3) Layer 3: This layer implements the links between the term nodes and the rule nodes. Nodes in this layer perform the product operation. Thus, for the jth rule node

$$net_j^3 = \prod_i^n x_i^3$$

and

$$y_j^3 = f_j^3(net_j^3) = net_j^3.$$

4) Layer 4: This layer performs the COA (Center Of Area) defuzzification to obtain numerical outputs. The connection weight w_{ij}^4 between the ith rule node and the jth output node represents the consequence fuzzy singleton. The node operations are

$$net_j^4 = \sum_{i=1}^m w_{ij}^4 x_i^4$$

and

$$y_j^4 = \frac{f_j^4(net_j^4)}{\sum_{i=1}^m x_i^4} = \frac{net_j^4}{\sum_{i=1}^m x_i^4}$$

where the link weight w_{ij}^4 is the output action strength of the jth output associated with the ith rule.

B. Supervised Gradient Descent Learning of the FNNS

The adjusted parameters in the network structure of the FNNS can be divided into two categories based on the IF (premise) part and THEN (consequence) part of the fuzzy rules. In the premise part we are asked to fine tune the mean and variance of the Gaussian functions. In the consequence part, the adjusted parameters are the consequence weights.

Once the FNNS has been initialized, a gradient descent-based backpropagation algorithm (BP) [15] is employed to adjust the parameters of the fuzzy neural network by using the training patterns. The main

goal of supervised learning is to minimize the error function

$$E = \sum_{j=1}^{p} \frac{1}{2}(d_j^4 - y_j^4)^2$$

where y_j^4 is the current output of the jth output node and d_j^4 is the desired output. If w_{ij} is the adjusted parameter, then the learning rule used is

$$w_{ij}(t+1) = w_{ij}(t) - \eta \frac{\partial E}{\partial w_{ij}} + \alpha \Delta w_{ij}(t)$$

and

$$\Delta w_{ij}(t) = w_{ij}(t) - w_{ij}(t-1) \tag{3}$$

where η is the learning rate and α, between 0 and 1, is the momentum parameter. By recursive applications of the chain rule, the error term for each layer is first calculated. The adaptation of weights of the corresponding layer is then given. Next, we will begin to derive the learning law for each layer in the feedbackward direction.

1) Layer 4: The error term to be propagated is given by the jth output node, as follows:

$$\delta_j^4 = \frac{-\partial E}{\partial net_j^4} = \frac{-\partial E}{\partial y_j^4}\frac{\partial y_j^4}{\partial net_j^4} = \frac{d_j^4 - y_j^4}{\sum_{i=1}^{m} y_i^3}.$$

Then we can derive

$$-\frac{\partial E}{\partial w_{ij}^4} = -\frac{\partial E}{\partial y_j^4}\frac{\partial y_j^4}{\partial net_j^4}\frac{\partial net_j^4}{\partial w_{ij}^4} = \delta_j^4 \cdot y_i^3.$$

Hence, by (3), the consequence weights are updated by

$$w_{ij}^4(t+1) = w_{ij}^4(t) + \eta \delta_j^4(t) y_i^3(t) + \alpha \Delta w_{ij}^4(t).$$

2) Layer 3: Only the error signals δ_i^3 need to be computed and propagated since there is no weight adjustment in this layer. The error term δ_i^3 is derived as follows:

$$\delta_j^3 = \frac{-\partial E}{\partial net_j^3} = \frac{-\partial E}{\partial y_j^3}\frac{\partial y_j^3}{\partial net_j^3} = \frac{-\partial E}{\partial y_j^3} = -\sum_{k=1}^{p}\frac{\partial E}{\partial y_k^4}\frac{\partial y_k^4}{\partial y_j^3} = \sum_{k=1}^{p}\delta_k^4(w_{jk}^4 - y_k^4)$$

where p is the number of output nodes.

3) Layer 2: The multiplication operation is done in this layer. First, the error term is computed:

$$\begin{aligned}\delta_{ij}^2 &= \frac{-\partial E}{\partial net_{ij}^2} = \frac{-\partial E}{\partial y_{ij}^2}\frac{\partial y_{ij}^2}{\partial net_{ij}^2} \\ &= \frac{-\partial E}{\partial y_{ij}^2}\cdot \exp(net_j^2) \\ &= \left(\sum_k \frac{-\partial E}{\partial y_k^3}\frac{\partial y_k^3}{\partial y_{ij}^2}\right)\cdot y_{ij}^2 \\ &= \left(\sum_k \delta_k^3 \cdot \frac{y_k^3}{y_{ij}^2}\right)\cdot y_{ij}^2 \\ &= \sum_k \delta_k^3 \cdot y_k^3 \end{aligned} \tag{4}$$

where the subscript k denotes all the rule nodes connected to the jth term node of input variable x_i. In fact, before we use the similarity measure for term node combination, (4) can be simplified as

$$\delta_{ij}^2 = \delta_j^3 \cdot y_j^3$$

for the initial network structure of the FNNS. We can continue to derive

$$-\frac{\partial E}{\partial m_{ij}} = -\frac{\partial E}{\partial y_{ij}^2}\frac{\partial y_{ij}^2}{\partial m_{ij}} = \delta_{ij}^2 \cdot \left[\frac{2(x_i^2 - m_{ij})}{\sigma_{ij}^2}\right].$$

Similarly, the adaptive rule of σ_{ij} is derived as follows:

$$-\frac{\partial E}{\partial \sigma_{ij}} = -\frac{\partial E}{\partial y_{ij}^2}\frac{\partial y_{ij}^2}{\partial \sigma_{ij}} = \delta_{ij}^2\left[\frac{2(x_i^2 - m_{ij})^2}{\sigma_{ij}^3}\right].$$

Thus the update rules for m_{ij} and σ_{ij} are

$$m_{ij}(t+1) = m_{ij}(t) + \eta\delta_{ij}^2\left[\frac{2(x_i^2 - m_{ij})}{\sigma_{ij}^2}\right] + \alpha\Delta m_{ij}(t)$$

and

$$\sigma_{ij}(t+1) = \sigma_{ij}(t) + \eta\delta_{ij}^2\left[\frac{2(x_i^2 - m_{ij})^2}{\sigma_{ij}^3}\right] + \alpha\Delta\sigma_{ij}(t).$$

III. Similarity Measure for Fuzzy Sets and Fuzzy Rules

Once supervised learning using the BP algorithm is finished, we may find that some of the fuzzy sets in layer 2 are almost the same. In other words, some term sets of the corresponding universe of discourse have a high degree of similarity. Term sets with a high degree of similarity can be combined into a single term set, that is, they can share a common term node. We can use the following *fuzzy similarity measure* [2], [9], to check the degree of similarity of two fuzzy sets:

$$Degree(A_1 = A_2) = E(A_1, A_2) = \frac{M(A_1 \cap A_2)}{M(A_1 \cup A_2)} \tag{5}$$

where $\cap$ and $\cup$ denote the intersection and union of fuzzy sets A_1 and A_2, respectively. $M(\cdot)$ is the size of a fuzzy set and $0 \le E(A_1, A_2) \le 1$.

From (5), we see that the computation of the similarity of two fuzzy sets requires calculating the size of intersection and union of two Gaussian membership functions. For any two fuzzy sets A_1 and A_2, $M(A_1 \cup A_2)$ can be easily derived as follows:

$$M(A_1 \cup A_2) = M(A_1) + M(A_2) - M(A_1 \cap A_2). \tag{6}$$

Calculating the size of the intersection of two Gaussian membership functions, however, is very complex because of the nonlinear shape of Gaussian functions. To make the computation of (5) feasible, we can use a tent function to approximate a Gaussian function. A Gaussian membership function with center m and width σ can be approximated by using a triangular membership function with center m and width $\sigma\sqrt{\pi}$ [9], that is

$$\exp\left[-\frac{(x-m)^2}{\sigma^2}\right] \longrightarrow \max\left[0, \frac{\sigma\sqrt{\pi} - |x-m|}{\sigma\sqrt{\pi}}\right]. \tag{7}$$

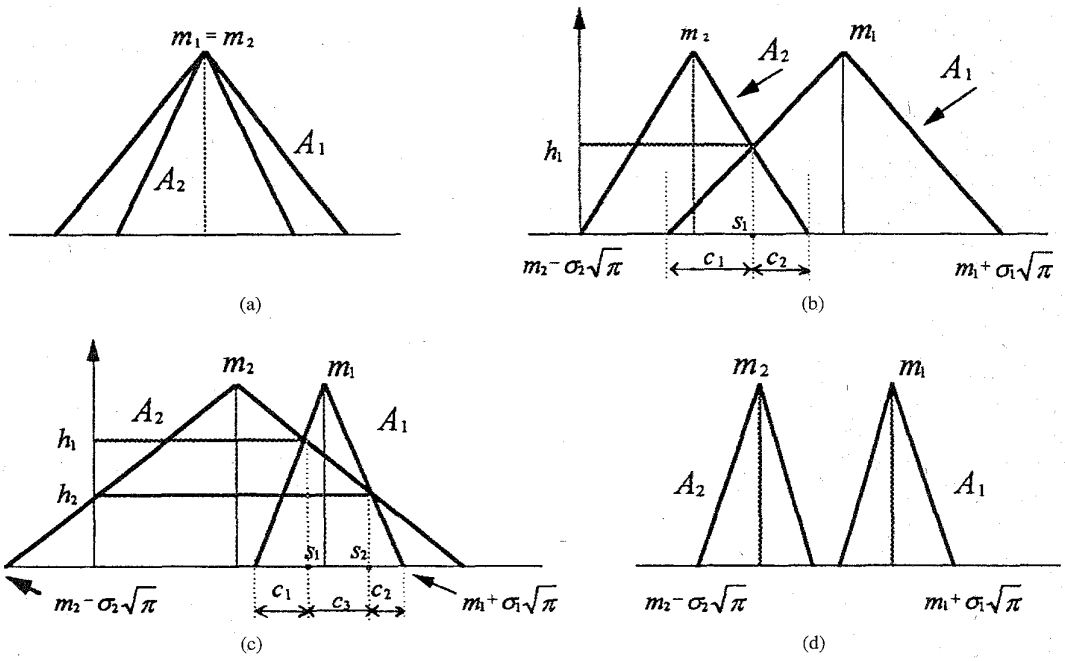

Fig. 4. Similarity measure of two triangular fuzzy sets, A_1 and A_2: (a) $A_2 \subseteq A_1$; (b) two membership functions have an intersection point, (s_1, h_1); (c) two membership functions have two intersection points, (s_1, h_1), and $(s_2, h2)$; (d) $A_1 \cap A_2 = \emptyset$.

Thus, the similarity measure of two fuzzy sets in the FNNS can be directly applied by using the approximation equations described in the next section.

A. Approximation Equations for the Similarity Measure

We consider the similarity measure in four different cases based on the triangular membership functions. Fig. 4(a)–(d) show the four cases under consideration. The fuzzy sets are denoted by A_1 and A_2, with the corresponding centers, m_1 and m_2, and widths σ_1 and σ_2, respectively. We will derive the similarity of these two fuzzy sets case by case. It is noted that we consider $m_1 > m_2$ in cases (ii)–(iv). If $m_1 < m_2$, then switch m_1 and m_2, and σ_1 and σ_2.

Case (i): $m_1 = m_2$ and $\sigma_1 \geq \sigma_2$: In this case, these two membership functions have the same center and no intersection point [see Fig. 4(a)]. Using (5) and (6), we can derive the similarity measure as follows:

$$\begin{aligned} M(A_1 \cap A_2) &= M(A_2), \\ M(A_1 \cup A_2) &= M(A_1) + M(A_2) \\ &\quad - M(A_1 \cap A_2) \\ &= M(A_1), \end{aligned}$$

thus

$$E(A_1, A_2) = \frac{M(A_2)}{M(A_2)} = \frac{\sigma_2}{\sigma_1}. \tag{8}$$

From (8), we can see that the degree of similarity of A_1 and A_2 is just the ratio of σ_2 to σ_1. In a particular case, if $\sigma_1 = \sigma_2$ then $E(A_1, A_2) = 1$, i.e., $A_1 = A_2$.

Case (ii): $|\sigma_1 - \sigma_2|\sqrt{\pi} \leq m_1 - m_2 \leq \sigma_1 + \sigma_2)\sqrt{\pi}$ and $m_1 > m_2$: In this case, these two membership functions have an intersection point at (s_1, h_1) [see Fig. 4(b)]. The size of $A_1 \cap A_2$ is derived as follows:

$$M(A_1 \cap A_2) = \tfrac{1}{2}(c_1 + c_2)h_1, \tag{9}$$

where

$$c_1 = \frac{\sigma_1(m_2 - m_1) + \sigma_1(\sigma_1 + \sigma_2)\sqrt{\pi}}{\sigma_1 + \sigma_2}$$
$$c_2 = \frac{\sigma_2(m_2 - m_1) + \sigma_2(\sigma_1 + \sigma_2)\sqrt{\pi}}{\sigma_1 + \sigma_2}$$

and

$$h_1 = \frac{(m_2 - m_1) + (\sigma_1 + \sigma_2)\sqrt{\pi}}{(\sigma_1 + \sigma_2)\sqrt{\pi}}$$

Substituting (6) and (9) into (5), we obtain

$$E(A_1, A_2) = \frac{(c_1 + c_2)h_1}{2(\sigma_1 + \sigma_2)\sqrt{\pi} - (c_1 + c_2)h_1}. \tag{10}$$

Case (iii): $m_1 - m_2 \leq |\sigma_2 - \sigma_1|\sqrt{\pi}$ and $m_1 > m_2$: There are two situations $\sigma_1 \leq \sigma_2$ and $\sigma_1 > \sigma_2$ in this case. For brevity, we only consider $\sigma_1 \leq \sigma_2$ in the following derivation. As we see in Fig. 4(c), these two membership functions have two intersection points at (s_1, h_1) and (s_2, h_2). The size of $A_1 \cap A_2$ is derived as follows:

$$M(A_1 \cap A_2) = \tfrac{1}{2}[c_1h_1 + c_2h_2 + c_3h_3] \tag{11}$$

where

$$c_1 = \frac{\sigma_1(m_2 - m_1) + \sigma_1(\sigma_2 + \sigma_1)\sqrt{\pi}}{\sigma_1 + \sigma_2}$$
$$c_2 = \frac{\sigma_1(m_2 - m_1) + \sigma_1(\sigma_2 - \sigma_1)\sqrt{\pi}}{\sigma_2 - \sigma_1}$$
$$c_3 = 2\sigma_1\sqrt{\pi} - (c_1 + c_2),$$
$$h_1 = \frac{(m_2 - m_1) + (\sigma_2 + \sigma_1)\sqrt{\pi}}{(\sigma_2 + \sigma_1)\sqrt{\pi}}$$
$$h_2 = \frac{(m_2 - m_1) + (\sigma_2 - \sigma_1)\sqrt{\pi}}{(\sigma_2 - \sigma_1)\sqrt{\pi}}$$

and

$$h_3 = h_1 + h_2.$$

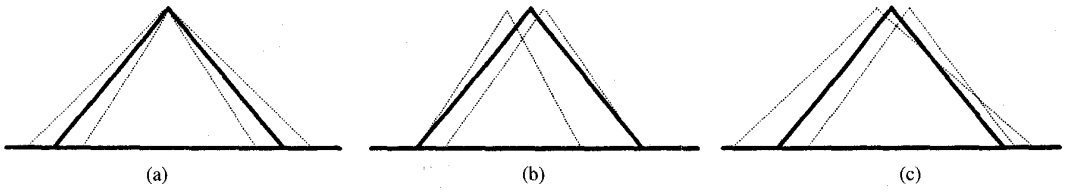

Fig. 5. Combination (solid line) of two fuzzy sets (dotted line) with high degree of similarity measure: (a) one is the subset of the other; (b) two fuzzy sets have single intersection point; and (c) two fuzzy sets have two intersection points.

Substituting (6) and (11) into (5), we obtain

$$E(A_1, A_2) = \frac{c_1h_1 + c_2h_2 + c_3h_3}{2(\sigma_1 + \sigma_2)\sqrt{\pi} - (c_1h_1 + c_2h_2 + c_3h_3)}. \tag{12}$$

We note that only slight modification of these above equations is needed to derive the equations when $\sigma_1 > \sigma_2$.

Case (iv). $m_1 - m_2 > (\sigma_1 + \sigma_2)\sqrt{\pi}$ *and* $m_1 > m_2$*:* In this case, these two membership functions have no intersection as shown in Fig. 4(d). Thus

$$M(A_1 \cap A_2) = 0 \tag{13}$$

and

$$E(A_1, A_2) = 0. \tag{14}$$

On the basis of the above discussion, we can easily calculate the degree of similarity of two fuzzy sets. Whether the degree of similarity of two fuzzy sets is high enough will depend on a reference value provided by the user. With a given reference value γ_s, $0 < \gamma_s \leq 1$, if $E(A_1, A_2) \geq \gamma_s$, then we can combine A_1 and A_2 into a new fuzzy set A_{new}. There is no standard method for determining A_{new}. We try to determine A_{new} in the following. We note that since no combination will occur in case (d) for zero degree of similarity, thus only three cases are considered below.

Case (i).

$$m_{new} = m_1 \quad \text{or} \quad m_2 \tag{15}$$

and

$$\sigma_{new} = \frac{\sigma_1 + \sigma_2}{2}. \tag{16}$$

Case (ii).

$$m_{new} = \frac{(m_1 + m_2) + (\sigma_1 - \sigma_2)\sqrt{\pi}}{2} \tag{17}$$

and

$$\sigma_{new} = \frac{(m_1 - m_2) + (\sigma_1 + \sigma_2)\sqrt{\pi}}{2\sqrt{\pi}}. \tag{18}$$

Case (iii).

$$m_{new} = \frac{m_1 + m_2}{2} \tag{19}$$

and

$$\sigma_{new} = \frac{\sigma_1 + \sigma_2}{2}. \tag{20}$$

Fig. 5 illustrates the combination of these three cases.

B. Similarity Measure of Fuzzy Rules

In Section III-A, the similarity measure for fuzzy sets can reduce the number of term nodes. However, we also wish to reduce the number of rules. To reduce the number of rule nodes, we must eliminate fuzzy rules of little influence and combine similar fuzzy rules into an equivalent fuzzy rule. The former is referred to as rule elimination; the latter is rule combination. In this subsection, we attempt to combine rules by using a similarity measure.

To determine whether two fuzzy rules are similar, we must evaluate the degree of similarity of the fuzzy rules. With the proposed FNNS, more specifically, we need to calculate the degree of similarity of both the consequences and preconditions. For simplicity, we consider the FNNS to be used for an MISO case. We will describe the similarity measure for the fuzzy rules described below:

$\mathbf{R^k}$: IF x_1 is A_1^k and x_2 is A_2^k and $\cdots$ and x_n is A_n^k THEN y is β^k

$\mathbf{R^l}$: IF x_1 is A_1^l and x_2 is A_2^l and $\cdots$ and x_n is A_n^l THEN y is β^l

where R^k and R^l represent the kth and the lth fuzzy rules, respectively.

1) Similarity Measure for Consequences: In the proposed FNNS, the consequences of the fuzzy rules are represented as connection weights. To calculate the degree of similarity of two consequence weights β^k and β^l, we should define a fuzzy set A_c. Then the similarity measure for β^k and β^l can be characterized as follows:

$$E_c(\beta^k, \beta^l) = \begin{cases} 1, & \text{if } \mu_{A_c}(\beta^k - \beta^l) \geq \gamma_c \\ 0, & \text{if } \mu_{A_c}(\beta^k - \beta^l) < \gamma_c \end{cases} \tag{21}$$

where $\mu_{A_c}(\beta^k - \beta^l)$ is the degree of similarity of β^k and β^l and γ_c, $0 < \gamma_c \leq 1$, is a reference value determined by the user.

In the FNNS, the membership function of A_c is a triangular function as shown below:

$$\mu_{A_c}(x) = \max\left[0, \frac{\beta_{\max} - \beta_{\min} - |x|}{\beta_{\max} - \beta_{\min}}\right] \tag{22}$$

where $\beta_{\max}$ and $\beta_{\min}$ are the maximum and minimum consequence weights in the training result of the FNNS, respectively.

2) Similarity Measure for Preconditions: With the kth fuzzy rule, the corresponding preconditions are $A_1^k, A_2^k, \cdots$, and A_n^k. Similarly, the corresponding preconditions of the lth fuzzy rule are $A_1^l, A_2^l, \cdots$, and A_n^l. To calculate the degree of similarity of the preconditions of these two fuzzy rules, we must check the degree of similarity of every fuzzy set pair, i.e., we have to check $E(A_i^k, A_i^l)$, for $i = 1, \cdots, n$. Thus, the similarity measure of the preconditions (E_p) can be characterized as follows:

$$E_p(\mathbf{A^k}, \mathbf{A^l}) = \min\{E(A_1^k, A_1^l), E(A_2^k, A_2^l), \cdots, E(A_n^k, A_n^l)\} \tag{23}$$

where

$$\mathbf{A^k} = [A_1^k, A_2^k, \cdots, A_n^k]$$

and

$$\mathbf{A^l} = [A_1^l, A_2^l, \cdots, A_n^l].$$

Once $E_p(\mathbf{A^k}, \mathbf{A^l})$ reaches a reference value γ_p, $0 < \gamma_p \leq 1$, then all of these fuzzy set pairs are considered to be very similar. This is the reason why we employ the min operation on (23).

Based on the discussion in (1) and (2), the similarity measure of the fuzzy rules is defined as

$$E_r(R^k, R^l) = E_c(\beta^k, \beta^l) \cdot E_p(\mathbf{A^k}, \mathbf{A^l}). \tag{24}$$

The user is asked to set a reference value γ_r, $0 < \gamma_r \leq 1$, in the FNNS. Then any two fuzzy rules R^k and R^l with $E_r(R^k, R^l) \geq \gamma_r$ can be combined into a new fuzzy rule R^{new}. In other words, the rule combination can be applied for any two fuzzy rules if the following conditions are satisfied.

1) $E_c = 1$, i.e., the two fuzzy rules have almost the same consequence weight.
2) $E_p \geq \gamma_r$, i.e., the degree of similarity of the preconditions must be high enough.

Once R^{new} is taken to replace both R^k and R^l, then the corresponding term nodes R^k and R^l will be eliminated. The term nodes of R^{new} can be directly obtained by using the combination method of fuzzy sets presented in Section III-A, i.e., A_i^{new} is the fuzzy set combination of A_i^k and A_i^l for $i = 1, \cdots, n$. On the other hand, if γ_c is high enough, then the consequence weight β^{new} of R^{new} can be simply chosen as $\beta^{new} = (\beta^l + \beta^k)/2$.

If the FNNS is used in an n-input/p-output case, the combination of two fuzzy rules is individually considered for each output. First, we check condition (1) for each output. Assume condition (1) is satisfied for all of the outputs, i.e.,

$$E_c(\beta_j^k, \beta_j^l) = 1, \quad \forall j, \quad j = 1, \cdots, p,$$

where β_j^k and β_j^l represent the kth and the lth consequences of the jth output node, respectively. Then, we calculate $E_p(\mathbf{A^k}, \mathbf{A^l})$ for the fuzzy rules. If E_p is greater than the reference value γ_r, then R^k and R^l are combined with the new consequence

$$\beta_j^{new} = \frac{\beta_j^k + \beta_j^l}{2}, \quad \forall j, \quad j = 1, \cdots, p.$$

With the combination of preconditions, the methods used in the MISO case can be directly applied here.

IV. A New On-Line Initialization Method

As mentioned in Section II, the initial structure of FNNS does not take an ordinary fuzzy partition of the input space. Therefore, how to choose the initial parameters of the FNNS becomes an important problem. We find in practical simulations that the determination of initial parameters will seriously affect the FNNS's performance of learning convergence. In this section, we try to develop a new on-line initialization method to improve the performance of the FNNS.

Since the parameters of the FNNS have a clear relationship with the input-output data, the initial FNNS can be constructed as a good approximation of an unknown function based on the input-output data. In the on-line initialization method, the initialization takes place immediately after each training pattern is presented. We do not start the back-propagation training algorithm in Section II-B for the first m, the default fuzzy rule number, time points. Suppose, at instant j, $1 \leq j \leq m$, a training pattern $[x_1(j), \cdots, x_n(j); y(j)]$ is presented. We can directly set the parameters

$$\beta^j = y(j) \tag{25}$$

and

$$m_{ij} = x_i(j), \quad 1 \leq i \leq n. \tag{26}$$

In this way, when m training patterns are presented, we can obtain m consequence weights (β^j, $j = 1, \cdots, m$) and the centers for the input fuzzy sets (A_i^j, $j = 1, \cdots, m$).

The remaining problem is how to determine the corresponding width (σ_{ij}) for A_i^j, this is also the main problem in the on-line initialization method. Though we can match the first m training pairs quite well by choosing σ_{ij} to be sufficiently small, we will have large approximation errors for other input-output pairs [16]. Therefore, the reasonable choice of σ_{ij} should make the input membership functions cover the input range in a good way. Moreover, the method in [16] results in a fixed value of σ_{ij} once the first m training pairs are fed into the fuzzy neural network. We expect to obtain a more flexible result to satisfy our requirements.

In the fuzzy neural network systems [3], [4], [17], the initial value of parameters can be easily set in such a way that the membership functions are equally spaced along the operating range of each input variable. Then these membership functions will satisfy ϵ-*completeness* [7], which means that given a value x of one of the inputs in the operating range, we can always find a linguistic label A such that $\mu_A(x) \geq \epsilon$. In this manner, the fuzzy inference system can provide smooth transition and sufficient overlapping from one linguistic label to another. It is especially mentioned that if the ϵ-completeness condition is not satisfied, there may be no fuzzy rules fired when the input data is fed into the fuzzy neural network. Thus we expect to present a flexible method to properly choose σ_{ij} such that the input membership functions can satisfy ϵ-completeness.

Before going further to show the choice and characteristic of σ_{ij}, we introduce the following notation. We note that the following notation is based on a fixed k or A_i^k, $1 \leq k \leq m$.

1) A_i^R: the fuzzy set which is on the **right** side of A_i^k and is closest to A_i^k.
2) A_i^L: the fuzzy set which is on the **left** side of A_i^k and is closest to A_i^k.
3) m_{iR}: the corresponding center of A_i^R.
4) m_{iL}: the corresponding center of A_i^L.
5) $A_{i,r}$: the **rightest** fuzzy set in A_i^j, $j = 1, \cdots, m$.
6) $A_{i,l}$: the **leftest** fuzzy set in A_i^j, $j = 1, \cdots, m$.
7) $m_{i,r}$: the corresponding center of $A_{i,r}$.
8) $m_{i,l}$: the corresponding center of $A_{i,l}$.

Let X_i denote the universe of discourse of the input x_i, we can also treat fuzzy sets A_i^j, for $j = 1$ to m, as fuzzy numbers defined in X_i. In the FNNS, we let $\mathbf{A_i} = (A_i^1, A_i^2, \cdots, A_i^m)$ be a *semi-closed* fuzzy set [10], i.e., a fuzzy set with

$$\mu_{A_{i,r}}(x_i) = 1, \quad x_i \in X_i \quad \text{and} \quad x_i \geq m_{i,r} \tag{27}$$

and

$$\mu_{A_{i,l}}(x_i) = 1, \quad x_i \in X_i \quad \text{and} \quad x_i \leq m_{i,l}. \tag{28}$$

The special choice for σ_{ij} is

$$\sigma_{ij} = \frac{\max\{|m_{ij} - m_{iR}|, |m_{ij} - m_{iL}|\}}{\sqrt{|\ln \lambda_i|}} \tag{29}$$

where λ_i is the overlapping factor, $0 < \lambda_i < 1$. We now show by choosing σ_{ij} this way, the membership functions of the linguistic labels A_i^j, $j = 1, \cdots, m$ will cover X_i with a good property.

Theorem 4.1: The semi-closed fuzzy set $\mathbf{A_i} = (A_i^1, A_i^2, \cdots, A_i^m)$, where each linguistic label A_i^j has a Gaussian membership function constructed by the preceding initial m_{ij} [see (26)] and σ_{ij} [see (29)], will satisfy ϵ-completeness. That is,

$$\begin{aligned} &\text{for all} \quad x_i \in X_i \\ &\text{there exists} \quad k \in 1, 2, \cdots, m \\ &\text{such that} \quad \mu_{A_i^k}(x_i) \geq \epsilon = \lambda_i \end{aligned}$$

where λ_i, $0 < \lambda_i < 1$, is the overlapping factor.

Proof: According to the location of x_i in X_i, we can prove this theorem under several different cases as shown below.

1) If $x_i \geq m_{i,r}$, then by applying (27), we have
$$\mu_{A_{i,r}}(x_i) = 1 \geq \lambda_i.$$
2) If $x_i \leq m_{i,l}$, then by applying (28), we have
$$\mu_{A_{i,l}}(x_i) = 1 \geq \lambda_i.$$
3) If there exists $k \in 1, 2, \cdots, m$, such that $m_{ik} \leq x_i \leq m_{iR}$ and $|m_{ik} - m_{iR}| \geq |m_{ik} - m_{iL}|$, then we have
$$\sigma_{ik} = \frac{|m_{ik} - m_{iR}|}{\sqrt{\ln \lambda_i}}$$
in this case. By using the Gaussian membership function, we can continue to obtain
$$\begin{aligned} \mu_{A_i^k}(x_i) &\geq \mu_{A_i^k}(m_{iR}) \\ &= \exp\left[-|\ln \lambda_i|\left(\frac{m_{iR} - m_{ik}}{m_{ik} - m_{iR}}\right)^2\right] \\ &= \lambda_i. \end{aligned}$$
4) If there exists $k \in 1, 2, \cdots, m$, such that $m_{iL} \leq x_i \leq m_{ik}$ and $|m_{ik} - m_{iR}| \geq |m_{ik} - m_{iL}|$, then σ_{ik} is the same as shown in case (3). Thus we have
$$\begin{aligned} \mu_{A_i^k}(x_i) &\geq \mu_{A_i^k}(m_{iL}) \\ &= \exp\left[-|\ln \lambda_i|\left(\frac{m_{iL} - m_{ik}}{m_{ik} - m_{iR}}\right)^2\right] \\ &\geq \exp\left[-|\ln \lambda_i|\left(\frac{m_{iR} - m_{ik}}{m_{ik} - m_{iR}}\right)^2\right] \\ &= \lambda_i. \end{aligned}$$
The proof for the other cases induced by $|m_{ik} - m_{iR}| < |m_{ik} - m_{iL}|$ is very similar to case (3) or case (4). So, we omit the process and complete this proof. □

Though we can incorporate prior expert information to do better initial parameter-choosing of the FNNS, we finally gave up this attempt, because we believed that the proposed on-line initialization method is efficient and sufficient in practical applications. In fact, based on our simulation results in the next section, this is indeed true.

V. An Illustrative Example

In this example [12], the plant to be identified is described by the second-order difference equation

$$y(k+1) = f[y(k), y(k-1)] + u(k)$$

where the unknown function f has the form

$$f[y(k), y(k-1)] = \frac{y(k)y(k-1)[y(k)+2.5]}{1 + y^2(k) + y^2(k-1)}.$$

A series-parallel type of identifier [12], implemented by the FNNS, is described by the equation

$$\hat{y}(k+1) = \hat{f}[y(k), y(k-1)] + u(k),$$

where $\hat{f}[y(k), y(k-1)]$ is in the form of (2) with $m = 40$ (number of initial rules) and $n = 2$. The input $u(k)$ was assumed to be a random signal uniformly distributed in the interval $[-2, 2]$.

First, the initial parameters of the FNNS are obtained by applying the on-line initialization method with the first 40 data points. The overlapping factors, $\lambda_i = 0.7$ for $i = 1$–2, are chosen to set up the corresponding variance of the Gaussian membership functions. The parameters of the initial rules are illustrated in Table I, where $G(a, b)$ is a Gaussian fuzzy set with mean a and variance b. For $k > 40$, the parameters in the FNNS are adjusted at every instant of time using a learning rate of $\eta = 0.1$ and a momentum parameter of $\alpha = 0.75$. Suppose one epoch of learning takes 200 time points, the supervised learning is continued for 200 epochs of training and the mean square errors are computed over 200 training steps, i.e.,

$$\begin{aligned} m(E_i) = \frac{1}{200}\sum_{k=200i+41}^{200i+240}[y(k) - \hat{y}(k)]^2 \\ \text{for} \quad i = 0, 1, \cdots, 199. \end{aligned}$$

The rules obtained after 200 epochs of learning are listed in Table II. Since we think that the parallel model [12] is more reasonable for testing the performance of an identifier, we use the parallel model to test the FNNS. In Fig. 6, the outputs of the plant as well as the FNNS for an input $u(k) = \sin(2\pi k/25)$ are shown and are seen to be indistinguishable.

The similarity measure was applied to the input fuzzy sets obtained in the preceding process to combine the similar fuzzy rules and term nodes. Table III shows the number of rules after rule combination under reference values $\gamma_c = 0.9$ and $\gamma_r = 0.9, \cdots, 0.1$. As shown in Table III, for a fixed value of γ_c, a smaller value of γ_r will generally combine more fuzzy rules. Especially, we found that the number of rules was greatly reduced from 40–22 when $\gamma_r = 0.1$. To show the feasibility of doing rule combination by the proposed similarity measure for fuzzy rules, we take $\gamma_r = 0.1$ as an example in the following process. The rules after rule combination are listed in Table IV. We find that the rules marked with the symbol "♣" (△, ◇, etc.) in Table II have been combined into a single rule shown in Table IV. Furthermore, the reference value of the similarity degree γ_s was set for term node combination. Table V shows the number of term nodes after term node combination under different values of γ_s. For brevity, let us only consider $\gamma_s = 0.4$ in the following simulation and comparison.

The final rules are shown in Table VI after supervised learning is applied again for 100 epochs of learning. In fact, the time consumed for training the FNNS again can be greatly reduced if we choose larger values of γ_c, γ_r, and γ_s. In Table VI, the term sets marked with the symbol "♣" (△, ◇, etc.) have been combined into a single term node that is not labeled as "— —." Also, in Table VI, some term sets with extremely large width (variance) will be eliminated and are labeled as "**." Since the membership grades of the corresponding input according to those term sets always approach to unity. Thus, those term sets are referred to as redundant term sets, which can hardly affect the results of rule reasoning. The outputs of the system and the FNNS with the final fuzzy rules are shown in Fig. 7. As shown in Fig. 7, the result is still desirable.

The neural network identifier $\mathcal{N}^3_{2,20,10,1}$ in [12] had two hidden layers with 20 and 10 neurons in each layer respectively; hence, the neural identifier had 250 $(= 2 \times 20 + 20 \times 10 + 10)$ adjustable parameters. However, the number of adjustable parameters in the FNNS is only 200 $(= 40 \times 2 \times 2 + 40)$ at the initial time and 84

TABLE I
INITIAL FUZZY RULES

No.	y(k)	y(k-1)	y(k+1)
1	$G(0.00, 0.05)$	$G(0.00, 0.05)$	0.38
2	$G(0.38, 0.04)$	$G(0.00, 0.05)$	0.92
3	$G(0.92, 0.26)$	$G(0.38, 0.04)$	0.32
4	$G(0.32, 0.04)$	$G(0.92, 0.26)$	0.29
5	$G(0.29, 0.02)$	$G(0.32, 0.04)$	−1.70
6	$G(-1.70, 0.28)$	$G(0.29, 0.02)$	−0.14
7	$G(-0.14, 0.32)$	$G(-1.70, 0.28)$	−1.22
8	$G(-1.22, 0.08)$	$G(-0.14, 0.32)$	1.82
9	$G(1.82, 0.12)$	$G(-1.22, 0.08)$	−2.37
10	$G(-2.37, 0.11)$	$G(1.82, 0.12)$	−0.69
11	$G(-0.69, 0.04)$	$G(-2.37, 0.11)$	2.04
12	$G(2.04, 0.35)$	$G(-0.69, 0.04)$	−1.92
13	$G(-1.92, 0.18)$	$G(2.04, 0.35)$	−0.64
14	$G(-0.64, 0.32)$	$G(-1.92, 0.18)$	2.02
15	$G(2.02, 0.12)$	$G(-0.64, 0.32)$	0.50
16	$G(0.50, 0.08)$	$G(2.02, 0.12)$	0.27
17	$G(0.27, 0.12)$	$G(0.50, 0.08)$	1.42
18	$G(1.42, 0.07)$	$G(0.27, 0.12)$	0.65
19	$G(0.65, 0.07)$	$G(1.42, 0.07)$	−0.79
20	$G(-0.79, 0.04)$	$G(0.65, 0.07)$	0.46
21	$G(0.46, 0.04)$	$G(-0.79, 0.04)$	−1.09
22	$G(-1.09, 0.14)$	$G(0.46, 0.04)$	−1.27
23	$G(-1.27, 0.28)$	$G(-1.09, 0.14)$	−0.07
24	$G(-0.07, 0.04)$	$G(-1.27, 0.28)$	1.54
25	$G(1.54, 0.07)$	$G(-0.07, 0.04)$	0.64
26	$G(0.64, 0.08)$	$G(1.54, 0.07)$	−0.87
27	$G(-0.87, 0.14)$	$G(0.64, 0.08)$	−2.20
28	$G(-2.20, 0.18)$	$G(-0.87, 0.14)$	1.81
29	$G(1.81, 0.15)$	$G(-2.20, 0.18)$	0.08
30	$G(0.08, 0.12)$	$G(1.81, 0.15)$	−1.26
31	$G(-1.26, 0.02)$	$G(0.08, 0.12)$	−0.09
32	$G(-0.09, 0.03)$	$G(-1.26, 0.02)$	−0.76
33	$G(-0.76, 0.04)$	$G(-0.09, 0.03)$	−0.78
34	$G(-0.78, 0.01)$	$G(-0.76, 0.04)$	1.32
35	$G(1.32, 0.26)$	$G(-0.78, 0.01)$	0.76
36	$G(0.76, 0.10)$	$G(1.32, 0.26)$	1.58
37	$G(1.58, 0.15)$	$G(0.76, 0.10)$	2.03
38	$G(2.03, 0.01)$	$G(1.58, 0.15)$	2.57
39	$G(2.57, 0.44)$	$G(2.03, 0.01)$	3.25
40	$G(3.25, 0.44)$	$G(2.57, 0.35)$	3.03

TABLE II
THE RULES AFTER SUPERVISED LEARNING

No.		y(k)	y(k-1)	y(k+1)
△	1	$G(0.65, 0.29)$	$G(0.03, 0.13)$	0.26
	2	$G(0.19, 0.08)$	$G(0.06, 0.10)$	0.74
△	3	$G(1.12, 0.34)$	$G(0.24, 0.27)$	1.02
	4	$G(0.28, 0.03)$	$G(0.88, 0.20)$	0.42
	5	$G(0.09, 0.02)$	$G(0.38, 0.04)$	−1.52
△	6	$G(-1.69, 0.27)$	$G(0.33, 0.00)$	−0.13
△	7	$G(-0.33, 0.72)$	$G(-1.84, 0.53)$	−0.85
♡	8	$G(-1.37, 1.94)$	$G(-0.41, 0.23)$	0.41
◇	9	$G(1.08, 0.71)$	$G(-0.80, 0.77)$	−3.03
	10	$G(-2.81, 1.07)$	$G(1.53, 22.14)$	0.06
♮	11	$G(-1.72, 20.60)$	$G(-2.07, 0.99)$	0.59
	12	$G(2.15, 1.46)$	$G(-1.85, 1.67)$	−9.67
△	13	$G(-1.76, 0.38)$	$G(2.13, 0.41)$	−0.57
♣	14	$G(-0.56, 0.45)$	$G(-1.65, 0.91)$	1.19
◇	15	$G(2.87, 5.07)$	$G(-0.72, 1.35)$	−3.72
	16	$G(0.43, 0.16)$	$G(2.10, 0.19)$	0.32
△	17	$G(0.67, 0.63)$	$G(0.57, 0.49)$	2.30
	18	$G(13.14, 1626.50)$	$G(19.88, 1661.62)$	−0.29
△	19	$G(0.46, 3.03)$	$G(1.34, 2.50)$	−0.05
△	20	$G(-0.81, 0.19)$	$G(0.20, 1.99)$	−0.17
	21	$G(0.49, 0.08)$	$G(-0.79, 0.08)$	−1.06
	22	$G(-1.15, 0.27)$	$G(0.55, 0.12)$	−0.29
♡	23	$G(-1.39, 0.22)$	$G(-0.74, 0.25)$	0.29
♠	24	$G(-0.78, 1.21)$	$G(-1.00, 1.32)$	5.10
△	25	$G(1.34, 0.59)$	$G(0.07, 1.44)$	0.21
‡	26	$G(2.18, 1.14)$	$G(1.74, 1.71)$	4.92
◇	27	$G(-0.96, 1.43)$	$G(1.50, 1.36)$	−5.12
♡	28	$G(-2.32, 0.17)$	$G(-0.86, 0.16)$	1.23
	29	$G(1.51, 0.20)$	$G(-2.58, 0.01)$	−1.25
	30	$G(-0.25, 18.02)$	$G(0.78, 57.50)$	0.56
△	31	$G(-0.64, 0.94)$	$G(0.60, 0.56)$	−1.76
	32	$G(-0.11, 0.02)$	$G(-1.19, 0.03)$	−0.68
	33	$G(-0.79, 0.01)$	$G(-0.23, 0.03)$	−0.65
♣	34	$G(-0.75, 0.16)$	$G(-0.83, 0.13)$	1.25
△	35	$G(1.36, 4.00)$	$G(-0.41, 1.08)$	−0.06
♮	36	$G(-1.97, 26.91)$	$G(0.20, 2.25)$	−1.12
♠	37	$G(1.27, 0.99)$	$G(1.01, 1.02)$	5.25
	38	$G(-0.57, 1323.60)$	$G(1.10, 1.71)$	2.61
‡	39	$G(3.37, 0.43)$	$G(2.76, 14.13)$	2.94
	40	$G(4.63, 2.67)$	$G(3.10, 3.36)$	10.56

$[= (15+16)\times 2+22]$ at the final time. Consequently, from a system complexity point of view (in the sense of number of free parameters), the FNNS model is much simpler than the neural network in [12]. Table VII summarizes these comparisons.

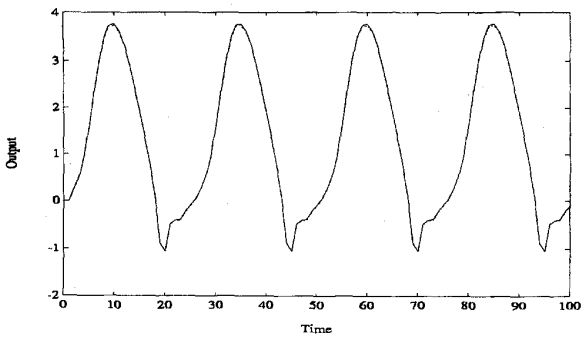

Fig. 6. Outputs of the original system and the FNNS (under fuzzy rules after 200 epochs of learning).

TABLE III
NUMBER OF RULES AFTER RULE COMBINATION (γ_c = 0.9)

γ_r	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
Number of rules	40	40	40	39	39	38	34	29	22

TABLE IV
THE RULES AFTER RULE COMBINATION

No.		y(k)	y(k-1)	y(k+1)
△	1	$G(-0.27, 1.02)$	$G(-0.55, 0.92)$	0.69
	2	$G(0.19, 0.08)$	$G(0.06, 0.10)$	0.74
	3	$G(0.28, 0.03)$	$G(0.88, 0.20)$	0.42
	4	$G(0.09, 0.02)$	$G(0.38, 0.04)$	−1.52
♡	5	$G(-1.69, 0.27)$	$G(0.33, 0.00)$	−0.13
◇	6	$G(1.79, 2.99)$	$G(0.63, 1.85)$	−4.25
	7	$G(-2.81, 1.07)$	$G(1.53, 22.14)$	0.06
♮	8	$G(-1.85, 23.76)$	$G(0.18, 2.26)$	−0.26
	9	$G(2.15, 1.46)$	$G(-1.85, 1.67)$	−9.67
♣	10	$G(-1.76, 0.38)$	$G(2.13, 0.41)$	−0.57
	11	$G(0.43, 0.16)$	$G(2.10, 0.19)$	0.32
	12	$G(13.14, 1626.50)$	$G(19.88, 1661.62)$	−0.29
	13	$G(0.49, 0.08)$	$G(-0.79, 0.08)$	−1.06
	14	$G(-1.15, 0.27)$	$G(0.55, 0.12)$	−0.29
♠	15	$G(0.04, 1.68)$	$G(-0.26, 1.74)$	5.17
‡	16	$G(2.77, 0.79)$	$G(2.25, 7.92)$	3.93
	17	$G(1.51, 0.20)$	$G(-2.58, 0.01)$	−1.25
	18	$G(-0.25, 18.02)$	$G(0.78, 57.50)$	0.56
	19	$G(-0.11, 0.02)$	$G(-1.19, 0.03)$	−0.68
	20	$G(-0.79, 0.01)$	$G(-0.23, 0.03)$	−0.65
	21	$G(-0.57, 1323.60)$	$G(1.10, 1.71)$	2.61
	22	$G(4.63, 2.67)$	$G(3.10, 3.36)$	10.56

TABLE V
NUMBER OF TERM NODES AFTER TERM NODE COMBINATION (γ_c = 0.9 AND γ_r = 0.1)

γ_s	0.9	0.8	0.7	0.6	0.5	0.4	0.3	0.2	0.1
Number of term nodes for $y(k)$	20	20	19	18	17	15	13	11	8
Number of term nodes for $y(k-1)$	21	21	20	18	18	16	13	12	11

TABLE VI
FINAL FUZZY RULES

No.	y(k)		y(k-1)		y(k+1)
1	♣	$G(-0.84, 1.22)$	♣	$G(-1.55, 1.48)$	1.37
2		$G(1.02, 0.86)$		$G(0.83, 0.83)$	2.96
3		$G(0.01, 6.35)$		$G(-0.73, 26.07)$	0.92
4		$G(-0.92, 1.46)$		$G(2.21, 2.52)$	−4.57
5	△	$G(-1.77, 0.99)$		$G(0.26, 1.45)$	−0.57
6	♠	$G(2.49, 1.79)$	♠	$G(-1.41, 1.57)$	−6.79
7		$G(-3.74, 0.22)$		$G(1.54, 22.14)$	1.06
8	◇	$G(-1.50, 20.78)$	♠	−−	−2.52
9	♠	−−		$G(-2.47, 1.81)$	−12.68
10	△	−−	△	$G(4.12, 1.43)$	−0.49
11	♡	$G(1.15, 1.17)$	△	−−	−3.17
12	□	**		**	−0.00
13	♡	−−		$G(-0.92, 0.89)$	−4.62
14		$G(-0.63, 0.72)$		$G(0.73, 0.80)$	−2.19
15	♣	−−	♣	−−	5.86
16		$G(1.92, 1.18)$	♡	$G(3.76, 4.02)$	4.97
17		$G(0.60, 67.36)$		$G(-3.08, 0.78)$	−0.56
18	◇	−−		$G(0.77, 57.49)$	0.56
19		$G(0.13, 6.22)$		$G(-1.91, 0.71)$	−0.98
20		$G(-0.77, 21.48)$		$G(-0.75, 32.93)$	0.16
21	□	−−	♠	−−	0.28
22		$G(5.35, 3.59)$	♡	−−	14.32

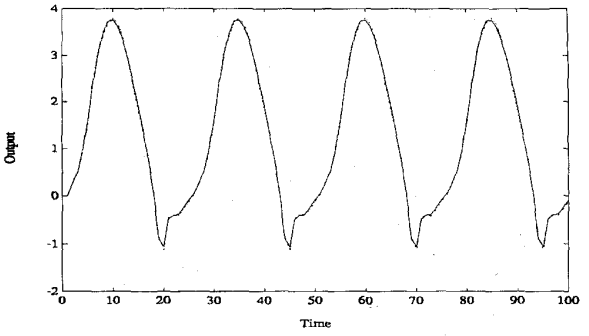

Fig. 7. Outputs of the original system and the **FNSS** (under final fuzzy rules).

VI. CONCLUSION

In this paper, a fuzzy neural network system called the FNNS has been presented for implementing fuzzy inference systems. The main purpose of the FNNS is to produce a simpler fuzzy inference system, with fewer fuzzy logical rules and adjustable parameters, which will be more efficient and useful in practical applications. In order to accomplish this purpose, we propose a fuzzy similarity measure for fuzzy rules to eliminate redundant fuzzy logical rules. Hence the complexity of a fuzzy neural network or a fuzzy inference system

TABLE VII
COMPARISONS OF THE PROPOSED SYSTEM WITH OTHER IDENTIFIERS

Model Name	Final Number of Rules	Final Number of Parameters
Narendra etc.[12]	$\mathcal{N}^3_{2,20,10,1}$	250
L.X.Wang[16]	40	200
proposed FNNS	22	84

can be reduced. A measure of the similarity for fuzzy sets, which indicates the degree to which two fuzzy sets are equal, is also applied to combine similar input linguistic term nodes of a fuzzy neural network. This greatly reduces the number of adjustable parameters. We also derive a new and effective on-line initialization method for choosing the initial parameters of the FNNS. A computer simulation has been presented to illustrate the procedure of the proposed FNNS. The simulation shows that the FNNS indeed yields simpler and more efficient results.

REFERENCES

[1] E. Backer, *Cluster Analysis by Optimal Decomposition of Induced Fuzzy Sets.* Amsterdam: Delft Univ. Press, 1978.

[2] D. Dubois and H. Prade, "A unifying view of comparison indices in a fuzzy set theoretic framework," in *Fuzzy Sets and Possibility Theory: Recent Developments,* R. R. Yager, Ed. New York: Pergamon, 1982.

[3] S. Horikawa, T. Furuhashi, and Y. Uchikawa, "On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm," *IEEE Trans. Neural Networks,* vol. 3, no. 5, pp. 801–806, 1992.

[4] J. S. Jang, "ANFIS: Adaptive-network-based fuzzy inference system," *IEEE Trans. Syst., Man, Cybern.,* vol. 23, no. 3, pp. 665–684, May/June 1993.

[5] J. M. Keller, R. R. Yager, and H. Tahani, "Neural network implementation of fuzzy logic," *Fuzzy Sets Syst.,* vol. 45, pp. 1–12, 1992.

[6] C. C. Ku and K. Y. Lee, "Diagonal recurrent neural networks for dynamic systems control," *IEEE Trans. Neural Networks,* vol. 6, pp. 144–156, Jan. 1995.

[7] C. C. Lee, "Fuzzy logic in control systems: Fuzzy logic controller—Part I & II," *IEEE Trans. Syst., Man, Cybern.,* vol. 20, no. 2, pp. 404–435, 1990.

[8] C. T. Lin and C. S. G. Lee, "Neural-network-based fuzzy logic control and decision system," *IEEE Trans. Comput.,* vol. C-40, no. 12, pp. 1320–1336, Dec. 1991.

[9] ——, "A neural fuzzy control system with structure and parameter learning," *Fuzzy Sets Syst.*, vol. 70, pp. 183–212, 1995.

[10] K. Liu and F. L. Lewis, "Some issues about fuzzy logical control," in *Proc. 32nd Conf. Decision and Control,* San Antonio, TX, Dec. 1993, pp. 1743–1748.

[11] A. Nafarieh and J. M. Keller, "A new approach to inference in approximate reasoning," *Fuzzy Sets Syst.,* vol. 41, pp. 17–37, May 1991.

[12] K. S. Narendra and K. Parthasarathy, "Identification and control of dynamical systems using neural networks," *IEEE Trans. Neural Networks,* vol. 1, no. 1, pp. 4–27, Mar. 1990.

[13] S. K. Pal and D. K. Dutta Majumder, *Fuzzy Mathematical Approach to Pattern Recognition.* New York: Wiley, 1986.

[14] M. Roubens, "Pattern classification problems and fuzzy sets," *Fuzzy Sets Syst.,* vol. 1, pp. 239–253, 1978.

[15] *Parallel Distributed Processing,* D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA: MIT Press, 1986, vol. 1.

[16] L. X. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis.* Englewood Cliff, NJ: Prentice Hall, 1994.

[17] C. W. Xu and Y. Z. Lu, "Fuzzy modeling identification and self-learning for dynamical systems," *IEEE Trans. Syst., Man, Cybern.,* vol. SMC-17, no. 4, pp. 683–689, July/Aug. 1987.

A Dynamic Gesture Recognition System for the Korean Sign Language (KSL)

Jong-Sung Kim, Won Jang, and Zeungnam Bien

***Abstract*— The sign language is a method of communication for the deaf-mute. Articulated gestures and postures of hands and fingers are commonly used for the sign language. This paper presents a system which recognizes the Korean Sign Language (KSL) and translates into a normal Korean text. A pair of Data-Gloves are used as the sensing device for detecting motions of hands and fingers. For efficient recognition of gestures and postures, a technique of efficient classification of motions is proposed and a fuzzy min–max neural network [4] is adopted for on-line pattern recognition.**

I. INTRODUCTION

Gestures and postures have been used as a means of communication among people for a long time, being interpreted as streams of tokens for a language [1]. They may vary from the stylized lexicon of a traffic cop to the highly developed syntax of a natural language such as the sign language.

The sign language is a method of communication for the deaf-mute. It is understood by means of gestures of both hands and fingers [2].

This paper deals with a system which recognizes the Korean Sign Language (KSL) and translates it into a normal Korean text.

According to a standard KSL dictionary, the 45-year-old Korean Sign Language contains about 6000 vocabulary words. However, they are formed by combining a relatively small number of basic gestures. Moreover, two types of gestures of hands and fingers are used: one type consists of static postures and the other is dynamic gestures. The former consists of 31 distinct postures expressing the dactylology while the latter is made up with changing patterns, constituting the main body of the KSL and expressing different meanings of vocabulary words.

One may extract features of static postures of 10 fingers by identifying and recognizing the dactylology in the space domain. On the other hand, the recognition of changing patterns of dynamic gestures in the time domain is essential to understand any KSL-based sentences. This means that the recognition of the KSL should be conducted in real-time. For our system, an electronic device, called Data Glove [3], is adopted as an input device in consideration of cost effectiveness of hardware versus real-time processing capability. It is remarked that, in case that an 8-bit gray level vision system is adopted as an input sensing device, the system is required to handle at least 8 Mb/s while, in case of Data Glove, the device needs to handle about 600 b/s. It is also known that the pattern classes of KSL gestures are not linearly separable and that patterns tend to overlap with each other. Therefore, it is desirable to design a pattern classifier in such a way that the amount of mis-classification for those overlapping classes is minimum. Also, the system needs some form of learning capability due to the varying nature of the patterns to handle.

It is remarked that in [6] and [7], neural network based methods were presented for recognition of the American Sign Language (ASL). In the work by Fel [6] were used the back-propagation

Manuscript received August 5, 1994; revised February 26, 1995.

J.-S. Kim and Z. Bien are with the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Taejon 305-701, Korea.

W. Jang is with the Agency for Defense Development, Taejon 305-600, Korea.

Publisher Item Identifier S 1083-4419(96)02312-6.