Design of Self-Learning Fuzzy System by GA Approach

Shian-Tang Tzeng
Department of Electronic Engineering, Kao Yuan University
smtstn@cc.kyu.edu.tw

Abstract

In this paper, an effective genetic algorithm (GA) approach is proposed for tuning the parameters of membership functions based on input-output pairs. By minimizing a quadratic measure of the error in the least-squares sense, the real-valued chromosomes of a population are evolved to get the best coefficients. Comparison to the well-known back-propagation algorithm for fuzzy logic system shows that both are powerful training algorithms, but much better performance is obtained with the proposed technique. Several numerical design examples are presented to demonstrate the efficiency and effectiveness of this proposed approach.

1. Introduction

With the rapid industrial development, the applications of fuzzy systems are getting more and more important. They have demonstrated their ability to deal with a wide variety of different problems [1]-[2]. Conventionally, the design of fuzzy system is the fact that the number of fuzzy sets must be defined in advance. However, it will take a lot of computation effort for a large number of fuzzy rules. On the other hand, the well-known back-propagation algorithm is used to design the fuzzy logic systems [3]-[8]. By observing these methods, the fuzzy inference systems can be represented as feed-forward neural networks. It is shown that the fuzzy inference systems with neural-fuzzy networks have the learning abilities from the input-output pairs. In general, the back-propagation algorithm is the most popular training algorithm developed for feed-forward neural networks [9]. However, it suffers from the disadvantages such as slow convergence and convergence to a local minimum. Therefore, in order to deal with the problem, the GA approach is proposed to tune the parameters of membership functions. Moreover, this method is expected to have good performance without requiring any derivatives or other auxiliary knowledge [10]. At the same time, it also prevents the time-consuming and trial-and-error parameter selection method.

Moreover, the GA method is powerful, broadly applicable stochastic search, and optimization techniques based on principles from evolutionary theory. It has been successfully used to design fuzzy systems and FIR digital filters [11-17]. Therefore, we attempt to take full advantage of the GA approach to determine the coefficients of membership functions.

2. GA approach

The usual form of GA was described by Goldberg [18]. The binary representation traditionally used in genetic algorithms has some shortcomings when used to solve multidimensional and high-precision numerical problems [19]. Extensive computation is required. Genetic algorithms perform poorly at solving such problems. Therefore, herein, these chromosomes are set to real-values rather than binary bit strings.

2.1. Crossover operation

Crossover is the main genetic operator. For direction-based operators [20], problem-specific knowledge is introduced into genetic operation in order to produce improved offspring. The operator generates a single offspring $B'$ from two parents $B_1$ and $B_2$ according to the following rule:

$$B' = r \cdot (B_1 - B_2) + B_1$$

where $r$ is a random number between 0 and 1. It also assumes that the parent $B_1$ is not worse than $B_2$; that is, $\text{fitness}(B_1) \leq \text{fitness}(B_2)$ for minimization problems.

2.2. Mutation operation

Mutation is a background operator which produces spontaneous random changes in various chromosomes. A simple way to achieve mutation would be randomly generated.
2.3. Selection operation

According to the fitness values, a new generation is formed by selecting the better chromosomes from the parents and offspring, and rejecting others so as to keep the population size constant.

The GA design procedure is described as follows:

STEP 1: Initial real-valued chromosomes of the population size (PS) are randomly generated.

STEP 2: Crossover rules combine two parents to produce the children in the next generation. It produces PS/2 offspring.

STEP 3: Mutation rules randomly modify individual parents to produce children. The mutation rate is set to be 1%.

STEP 4: Evaluate these chromosomes (PS+PS/2+1%PS).

STEP 5: The best PS chromosomes are selected among parents and offspring to form a new population for the next generation.

STEP 6: Does the number of generations reach the specified value? If the answer is YES, stop the process; otherwise, go to STEP 2.

After numerous generations, the algorithm converges on the best chromosome, which represents the best solution to the problem. In this work, the proper population size is set to be 100.

3. Fuzzy basis function

The fuzzy systems can be represented as a linear combination of fuzzy basis functions described by Wang and Mendel [8]. It is represented in a simple structure similar to that of radial basis function. A multi-input and single-output (MISO) fuzzy inference system: $X \subset R^n \rightarrow Y \subset R$ is taken into account in this paper. Now suppose that there are M fuzzy IF-THEN rules in the following form:

$R^i: IF \ x_i \ is \ A^i_j \ and \ \cdots \ and \ x_n \ is \ A^i_n, \ THEN \ y \ is \ \theta^i_j \ (2)$

where $A^i_j$ is a linguistic term characterized by a fuzzy membership function $\mu_{A^i_j}(x_i)$. $\theta^i_j$ is a singleton, and $j = 1, \ldots, M$. The fuzzy basis function form with center average defuzzifier, product-inference rule, singleton fuzzifier, and Gaussian membership function is used here. The Gaussian membership function is a popular method for specifying fuzzy sets. Its curve has the advantage of being smooth and nonzero at all points. Therefore, for a given crisp input vector $x = (x_1, \ldots, x_n)^T \in X$ where the superscript T denotes the vector transpose operation, the defuzzified inferred output is given as follows:

$$y = f(x) = \frac{\sum_{j=1}^{M} \theta^i_j \left( \prod_{i=1}^{n} \mu_{A^i_j}(x_i) \right)}{\sum_{j=1}^{M} \left( \prod_{i=1}^{n} \mu_{A^i_j}(x_i) \right)} \quad (3)$$

where $f : X \subset R^n \rightarrow Y \subset R$ and $\mu_{A^i_j}(x_i)$ is the Gaussian membership function defined by

$$\mu_{A^i_j}(x_i) = \exp \left[ -\frac{1}{2} \left( \frac{x_i - c^i_j}{\sigma^i_j} \right)^2 \right] \quad (4)$$

where $c^i_j$ and $\sigma^i_j$ stand for the center and width parameters. Moreover, the fuzzy basis function can be represented as

$$p_j(x) = \frac{\prod_{i=1}^{n} \mu_{A^i_j}(x_i)}{\sum_{j=1}^{M} \left( \prod_{i=1}^{n} \mu_{A^i_j}(x_i) \right)} \quad (5)$$

Then the fuzzy logic system is equivalent to a fuzzy basis function expansion:

$$f(x) = \sum_{j=1}^{M} p_j(x) \theta^i_j$$

where $\theta^i_j \in R$ are constants. Assume that there are N input-output pairs: $(x(l), d(l))$, $l = 1, \ldots, N$. In this paper, the task is to design the fuzzy basis function expansion such that the error between $f(x(l))$ and $d(l)$ is minimized. Therefore, the error function, fitness, with $k$th chromosome is defined as follows:

$$E_k = \sum_{l=1}^{N} \left[ f_k(x(l)) - d(l) \right]^2 \quad (7)$$

where

$$f_k(x(l)) = \frac{\sum_{j=1}^{M} \theta^i_k \prod_{i=1}^{n} \exp \left( -\frac{1}{2} \left( \frac{x_i(l) - c^i_j}{\sigma^i_j} \right)^2 \right)}{\sum_{j=1}^{M} \prod_{i=1}^{n} \exp \left( -\frac{1}{2} \left( \frac{x_i(l) - c^i_j}{\sigma^i_j} \right)^2 \right)} \quad (8)$$

The subscript $k$ denotes the $k$th real-valued chromosome, $B_k = [c_k \sigma_k \theta^i_k]^T$, where $c_k = [c^i_{1k}, \ldots, c^i_{nk}, \ldots, c^n_{1k}, \ldots, c^n_{nk}]$, $\sigma_k = [\sigma^i_{1k}, \ldots, \sigma^i_{mk}, \ldots, \sigma^n_{1k}, \ldots, \sigma^n_{mk}]$, and $\theta^i_k = [\theta^i_{1k}, \ldots, \theta^i_{Nk}]$. After applying the proposed GA approach as stated in Sections 2, the best chromosome (solution) corresponding to the smallest fitness value can be obtained.

4. Design examples

In order to evaluate how effectively the fuzzy identifier with our proposed approach, the illustrated
Examples as same as in [3] and [7] are used in this section. Moreover, the identified model f has the form of (8) with M=20 for all examples. Also four numerical design examples are presented to demonstrate the design results.

Example 1: The plant is governed by the difference equation

\[ y(k+1) = 0.3y(k) + 0.6y(k-1) + g[u(k)] \]  

(9)

where the unknown function g has the form

\[ g(u) = 0.6\sin(\pi u) + 0.3\sin(3\pi u) + 0.1\sin(5\pi u). \]

In order to identify the plant, the model is governed by the difference equation

\[ y(k+1) = 0.3y(k) + 0.6y(k-1) + f[u(k)]. \]  

(10)

After applying the GA approach with 5,000 generations with a random input u(k) for k=200 whose magnitude was uniformly distributed in the interval [-1, 1], the best chromosome corresponding to the smallest fitness value at each generation is presented in Fig. 1. When the best fitness value does not decrease any more with a long time, it means that the proposed approach finds the best chromosome (solution) of this problem. Therefore, the best solution is obtained in the 4,636th generation. For k=700 and the input u(k) = \sin(2\pi k/250), the outputs of the plant (solid line) and the identification model (dashed line) are shown in Fig. 2. The error between the plant and the identified model is presented in Fig. 3. Moreover, the same plant in [3] was identified using a neural network identifier which failed to follow the plant at k=500. The peak error of our method is 0.13 which is smaller than that of [7] whose peak error is 0.4. Obviously, the performance of our approach is much better. Furthermore, for another input

\[ u(k) = \begin{cases} \sin(2\pi k/250), & 1 \leq k \leq 250 \text{ and } 501 \leq k \leq 700 \\ 0.5\sin(2\pi k/250) + 0.5\sin(2\pi k/25), & 251 \leq k \leq 500 \end{cases}, \]

(11)

the outputs of plant and the identification model are shown in Fig. 4. It shows that the identified model approximates the plant very well under the same membership functions without training the model once again and error is shown in Fig. 5. Furthermore, the performance of our method compared with that of back-propagation fuzzy systems [7] and neural networks [3] is listed in Table I for reference. It is shown that the peak errors, number of rules, and number of generations (steps) of the proposed algorithm are better than that of back-propagation fuzzy systems [7] and neural networks [3]. Obviously, it presents that our approach is superior.

Example 2: The plant is described by the second-order difference equation

\[ y(k+1) = g[y(k), y(k-1)] + u(k) \]  

(12)

where

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

and \[ u(k) = 0.07 - 0.02\cos(2\pi k/25) \]. The identified model is described by the equation

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

(14)

After applying the same method as mentioned above, the results and error are presented in Figures 6 and 7, respectively. It shows that the trained model approximates the plant quite well. Moreover, compared results are also listed in Table I for reference.

Example 3: The plant is of the form

\[ y(k+1) = g[y(k), y(k-1), y(k-2), u(k), u(k-1)] \]  

(15)

where the unknown function g has the form

\[ g(x_1, x_2, x_3, x_4, x_5) = \frac{x_1x_2x_3(x_1-1)+x_4}{1+x_1^2+x_2^2} \]  

(16)

and

\[ u(k) = \begin{cases} \sin(2\pi k/250), & k \leq 500 \\ 0.8\sin(2\pi k/250) + 0.2\sin(2\pi k/25), & k > 500. \end{cases} \]  

(17)

The identification model is described by the equation

\[ y(k+1) = f[y(k), y(k-1), y(k-2), u(k), u(k-1)] \]  

(18)

With the proposed technique, Figures 8 and 9 present the results and the error, respectively. It shows that the performance of the identified plant is very well.

Example 4: In this example, the fuzzy system with GA approach is introduced how to identify a multi-input and multi-output (MIMO) system described by the following equation

\[ \begin{bmatrix} y_1(k+1) \\ y_2(k+1) \end{bmatrix} = \begin{bmatrix} \frac{y_1(k)}{1+y_2^2(k)} \\ \frac{y_1(k)y_2(k)}{1+y_2^2(k)} \end{bmatrix} + \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix}. \]  

(19)

There are two fuzzy systems, \( f_1 \) and \( f_2 \), in the identification model described by the equation

\[ \begin{bmatrix} y_1(k+1) \\ y_2(k+1) \end{bmatrix} = \begin{bmatrix} f_1(y_1(k), y_2(k)) \\ f_2(y_1(k), y_2(k)) \end{bmatrix} + \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix}. \]  

(20)

After applying the proposed approach, the responses of the plant, \( y_1(k) \) and \( y_2(k) \), and the identified model, \( \hat{y}_1(k) \) and \( \hat{y}_2(k) \), are presented in Fig. 5. The error between \( y_1(k) \) and \( \hat{y}_1(k) \) is presented in Fig. 11. Similarly, for \( y_2(k) \) and \( \hat{y}_2(k) \), the responses and error are shown in Fig. 12 and Fig. 13, respectively.

We have summary the performance of these four examples in Table I and also indicate in the table the back-propagation fuzzy systems [7] and the neural networks [3] peak errors, number of rules, and number of generations (steps). It shows that the performance of our approach is much better.
5. Conclusions

In this paper, the fuzzy system can be represented as a linear combination of fuzzy basis functions. In order to ensure a high performance, an appropriate fuzzy system is automatically selected by GA approach. It also prevents the time-consuming and trial-and-error procedure. This method is not only simplicity but also easy implementation with satisfactory performance. The illustrated examples are given to demonstrate the design flexibility and usefulness by incorporating the proposed approach.

6. References


Table I Fuzzy identifier with GA approach design examples

<table>
<thead>
<tr>
<th>Peak error</th>
<th>Example 1</th>
<th>Example 2</th>
<th>Example 3</th>
<th>Example 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>GA approach</td>
<td>0.13</td>
<td>0.11</td>
<td>0.11</td>
<td>0.14</td>
</tr>
<tr>
<td>[7]</td>
<td>0.4</td>
<td>1</td>
<td>0.2</td>
<td>0.6</td>
</tr>
<tr>
<td>[3]</td>
<td>3</td>
<td>1.5</td>
<td>0.3</td>
<td>0.9</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Num. of rules</th>
<th>GA approach</th>
<th>Example 1</th>
<th>Example 2</th>
<th>Example 3</th>
<th>Example 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>[7]</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>[3]</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td>40</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Num. of gen.</th>
<th>GA approach</th>
<th>Example 1</th>
<th>Example 2</th>
<th>Example 3</th>
<th>Example 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>[7]</td>
<td>5,000</td>
<td>5,000</td>
<td>5,000</td>
<td>5,000</td>
<td></td>
</tr>
<tr>
<td>[3]</td>
<td>100,000</td>
<td>100,000</td>
<td>100,000</td>
<td>100,000</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 1 Fitness
Fig. 2 Outputs of the plant (solid line) and the identified model (dashed line)

Fig. 3 Error

Fig. 4 Outputs of the plant (solid line) and the identified model (dashed line)

Fig. 5 Error

Fig. 6 Outputs of the plant (solid line) and the identified model (dashed line)

Fig. 7 Error
Fig. 8 Outputs of the plant (solid line) and the identified model (dashed line)

Fig. 9 Error

Fig. 10 Outputs of the plant ($y_1(k)$) and the identified model ($\hat{y}_1(k)$)

Fig. 11 Error

Fig. 12 Outputs of the plant ($y_2(k)$) and the identified model ($\hat{y}_2(k)$)

Fig. 13 Error