Pattern recognition is an important application of both neural network techniques and fuzzy sets theory. Various crisp neural networks such as the Hopfield network [74], the Hamming net [74], the Carpenter/Grossberg ART [18], Martin-Pittman BP-based neural net [75] and the Fukushima Neocognitron [29,30] have been used in pattern recognition. On the other hand, fuzzy sets theory is able to deal with uncertainty and fuzziness in pattern recognition, therefore fuzzy logic is a useful tool for simulating the human brain's perception and decision. In recent years, fuzzy neural networks have been used in pattern recognition. Kosko proposed a Fuzzy Associate Memory (FAM) using fuzzy matrices to represent fuzzy mappings [60]. Kwan and Cai designed a fuzzy neural network to recognize the 26 English letters and the 10 Arabic numericals [65]. However, the Kwan-Cai Fuzzy Neural Network (FNN) has two disadvantages which are (1) the important parameter $\alpha$ is defined subjectively and (2) the learning algorithm may greatly increase the number of fuzzy neurons if many training patterns are not similar. To solve these problems, we have developed a genetic fuzzy neural network which is capable of using genetic algorithms to optimize the parameter $\alpha$ and reducing the number of fuzzy neurons. The simulations have indicated that the genetic fuzzy neural network can recognize various distorted patterns with high recognition rates.

10.1 Structure of a Genetic Fuzzy Neural Network

A Genetic Fuzzy Neural Network (GFNN) is a genetic-algorithms-based Kwan-Cai FNN, and consists of 4 layers (see Figure 10.1). Each pattern has $N_1 \times N_2$
Layer 1  Layer 2  Layer 3  Layer 4

Figure 10.1. A Genetic Fuzzy Neural Network.

pixels and there are $K$ training patterns. For clarity, the functions of fuzzy neurons in different layers are described layer by layer as follows:

Layer 1: Input Layer

$N_1 \times N_2$ input neurons in the first layer are oval nodes with simple normalization functions defined by

$$O_{ij}^{[1]} = \frac{x_{ij}^k}{\max_{i=1}^{N_1} (\max_{j=1}^{N_2} (\max_{k=1}^{K} (x_{ij}^k)))},$$

(10.1)

where $x_{ij}^k$ is the $(i,j)$th pixel value of the $k$th pattern for $i = 1, 2, ..., N_1$, $j = 1, 2, ..., N_2$ and $k = 1, 2, ..., K$.

Layer 2: Genetic Fuzzification Layer

$N_1 \times N_2$ Genetic fuzzification neurons in the second layer are square nodes with either the triangular fuzzification functions used by Kwan and Cai [65] or the new Gaussian fuzzification functions.

The triangular fuzzification functions are given by:

$$O_{pqm}^{[2]} = \begin{cases} 1 - \frac{2|\tau_{pqm} - f_{pq}|}{\alpha} & \text{for } O_{pqm}^{[2]} \geq 0 \\ 0 & \text{for otherwise,} \end{cases}$$

(10.2)
the Gaussian fuzzification functions are given by:

\[ O_{pqm}^{[2]} = e^{-\frac{\gamma_{pqm} - f_{pq}}{\alpha}} \]

where

\[ f_{pq} = \max_{i=1}^{N_1}(\max_{j=1}^{N_2}(O_{ij}^{[1]} e^{-\beta^2[(p-i)^2+(q-j)^2]})) \]

where \( \gamma_{pqm} \) and \( \beta \) are parameters to adjust centers and widths of fuzzification functions \( f_{pq} \), respectively, and \( \alpha \) is a parameter to change the shapes of functions \( O_{pqm}^{[2]} \) for \( p = 1, 2, \ldots, N_1, q = 1, 2, \ldots, N_2 \) and \( m = 1, 2, \ldots, M \).

Genetic algorithms are used to optimize \( \alpha \) for \( K \) input patterns in order to reduce the number of neurons in the layers 2, 3 and 4.

**Layer 3: Fuzzy Clustering Layer**

\( M \) fuzzy neurons in the 3rd layer are trapezoidal nodes with a min function defined by

\[ O_{m}^{[3]} = \min_{p=1}^{N_1}(\min_{q=1}^{N_2}(O_{pqm}^{[2]})) \]

where \( m = 1, 2, \ldots, M \). \( M \) will be determined by the learning algorithm.

**Layer 4: Output Layer**

\( M \) output neurons in the 4th layer are triangular nodes with the functions defined by

\[ O_{m}^{[4]} = \begin{cases} 0 & \text{for } O_{pqm}^{[3]} < \max_{m=1}^{M}(O_{pqm}^{[3]}) \\ 1 & \text{for } O_{pqm}^{[3]} = \max_{m=1}^{M}(O_{pqm}^{[3]}) \end{cases} \]

where \( m = 1, 2, \ldots, M \).

### 10.2 Genetic-Algorithms-Based Self-Organizing Learning Algorithm

By adding genetic algorithms into the Kwan-Cai learning algorithm [65], we propose a genetic-algorithm-based self-organizing learning algorithm which is able to reduce \( M \) by adjusting \( \alpha \). The total number of fuzzy neurons in a GFNN is \((1 + M)N_1N_2 + 2M\). \( N_1 \) and \( N_2 \) are fixed, therefore we can only try to reduce \( M \) to reduce the complexity of the GFNN. The genetic-algorithm-based self-organizing learning algorithm is given below,

**Step 1**: Choose values of \( \alpha (\alpha \geq 0) \) and \( \beta \) for the second layer of the GFNN.
Sec. 10.3. Simulations

We use 10 kinds of the distorted 26 English letters and 10 Arabic numerals to verify the new GFNN. For example, 10 kinds of distorted patterns for “H” and “4” are shown in Figures 10.2, 10.3, 10.4, 10.5 and 10.6. In the simulations, $N_1 = N_2 = 16$, $M_{\text{max}} = 36$, $K = 72$, $\beta = 0.3$, the initial $\alpha = 2.0$. We used 36 LA & 36 SM, 36 TT & 36 SQ, and 36 SH & 36 DC distorted patterns to train a GFNN, respectively, then used 360 distorted patterns to check recognition rates. In order to compare the GFNN with the Kwan-Cai FNN, we adopted the triangular fuzzification function Eq. (10.2) in the simulations. By using genetic algorithms, we finally get $\alpha = 1.684$ for the trained GFNN.
Figure 10.4. SH and DC distorted patterns (SH: shaking, DC: disconnected).

Figure 10.5. HL and HS distorted patterns (HL: half-part larger, HS: half-part shifted).

Figure 10.6. AP and MP distorted patterns (AP: added small parts, MP: missed small parts).
The simulation results are given in Tables 10.1, 10.2 and 10.3. As a result, the new GFNN is better than the Kwan-Cai FNN because the GFNN can optimize $\alpha$ by using genetic algorithms.

Table 10.1: *Comparison between the Kwan-Cai FNN’s Recognition Rate and our GFNN’s Recognition Rate for $T_f = 0.52$.*

<table>
<thead>
<tr>
<th>Training Patterns</th>
<th>36 LAs and 36 SMs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kwan-Cai FNN</td>
<td>87.78%</td>
</tr>
<tr>
<td>Our GFNN</td>
<td>94.44%</td>
</tr>
</tbody>
</table>

Table 10.2: *Comparison between the Kwan-Cai FNN’s Recognition Rate and our GFNN’s Recognition Rate for $T_f = 0.48$.*

<table>
<thead>
<tr>
<th>Training Patterns</th>
<th>36 TTs and 36 SQs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kwan-Cai FNN</td>
<td>87.78%</td>
</tr>
<tr>
<td>Our GFNN</td>
<td>93.05%</td>
</tr>
</tbody>
</table>

Table 10.3: *Comparison between the Kwan-Cai FNN’s Recognition Rate and our GFNN’s Recognition Rate for $T_f = 0.515$.*

<table>
<thead>
<tr>
<th>Training Patterns</th>
<th>36 SHs and 36 DCs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kwan-Cai FNN</td>
<td>87.78%</td>
</tr>
<tr>
<td>Our GFNN</td>
<td>89.17%</td>
</tr>
</tbody>
</table>

### 10.4 Conclusions

By applying genetic algorithms to the Kwan-Cai fuzzy neural network, we have designed a more adaptive genetic fuzzy neural network for pattern recognition. A genetic-algorithms-based self-organizing learning algorithm is capable of reducing the total number of fuzzy neurons and increasing recognition rates for a fixed number of output neurons. The simulations have indicated that the genetic fuzzy neural network is effective for recognizing various distorted patterns with good recognition rates.
Bibliography


