Tạp chí Tin học và Điều khiển học, T. 15, S. 3 (1999), 24-30

NEURAL NETWORK AND FUZZY LOGIC: AN APPLICATION TO FINGERPRINT RECOGNITION

HOANG KIEM, LE HOAI BAC, LE HOANG THAI

Abstract. In this paper, we utilize a feedforward fuzzy neural network to solve the problem of fingerprint-images classification into four classes: Whorl, Left Loop, Right Loop and Arch.

At first, each fingerprint image is reduced noise, removed the background and then binarized.

Then, one of thinning processes is chosen to perform on the image.

Consequently, the 6x5 directional matrix which represents the global flow shapes of the fingerprint is established.

After being transformed into 6 × 5 directional matrices, the four matrices of the above classes will be utilized as training patterns for a ANN.

After learning from the four training patterns above, our trial FNN has been experimented through 53 fingerprint samples and results in the classification rate of 96.23%.

The obtained testing results demonstrate the system's power of classification.

1. INTRODUCTION

Fingerprints are considered as a reliable feature for human identifier. It can be widespreadly used in many applications such as: access control (securing area), credit card security and investigation of crimes, etc. Therefore, it is very necessary to produce an automatic fingerprint matching algorithm, and to design an efficient system based on that algorithm. Fingerprint classification takes an important role in fingerprint matching system because its target is to classify fingerprints into several different types, and consequently, shortens the executable time of matching algorithms.

Some classification algorithms used to be suggested such as the syntactic method of Fu [1], the classification method which is performed basing on features of singular points proposed by Kwagoe and Tojo [5], or the fingerprint classification method which is based on qualities and locations of the detected singular points by Karu and Jain [3].

In general, classification algorithms are divided into two main approaches:

- The first one is the syntactic method: in which, a context-free grammar is employed. However, it is a very complicated task to establish this grammar.
- The second one is based on feature points, it has a disadvantage that it is not an easy work to extract these feature points.

To overcome these weaknesses, we have proposed a new classification method: in which a feedforward neural network is employed, and fingerprints are classified into four classes: Whorl, Right Loop, Left Loop and Arch.

In our proposed method, each fingerprint is firstly performed under a thinning process. Then, a 6 × 5 directional matrix is established to represent the direction of finger ridges. After that, this matrix will be digitally force to numeric values (0, 1, 2 and 3).

Four standard fingerprint images represent the four classes above then being transformed into digitalized directional matrices. After that, those matrices will be utilized as training patterns for the FNN.

After learning from the four standard patterns above, the FNN will be able to classify fingerprints efficiently. Fingerprint images which need testifying will be transformed into corresponding digitalized directional matrices and will be classified by the FNN itself.

The four-layer feedforward FNN operates according to the below mechanism: when each input pattern is provided, at first the neural network fuzzifies this pattern and then calculates the similarity density between the input pattern and all learned patterns. Finally, the network will reach its conclusion by choosing the pattern which is most similar to the input pattern in density, and by performing a defuzzification process. After that the network sends it to an output. (Note: *This fuzzy neural network will still have the capacity of recognition even if it is applied in the area of language-understanding, where we only need to supply training patterns of those languages for the learning procedure*).

The next section presents in details about the FNN's structure and operation.

On Section 3, we discuss the proposed fingerprint classification method in details.

Section 4 illustrates our experiments, results and conclusions.

2. THE FOUR-LAYER FEEDFORWARD FUZZY NEURAL NETWORK [2]

2.1. Fuzzy neurons

The output of a typical nonfuzzy neuron can be expressed as follows:

$$y = f\left[\sum_{i=1}^{N} w_i x_i - T\right], \tag{1}$$

where

x_i (for $i = 1$ to N): N weighted inputs.

w_i ($i = 1$ to N): the corresponding weights.

Y: the output of the nonlinear activation function $f[\,]$.

T: the internal threshold of the neuron.

A. Definition of Fuzzy Neuron (FN)

A fuzzy neuron can be expressed through equations below:

$$z = h[w_1x_1, w_2x_2, \dots, w_Nx_N], \tag{2}$$

$$s = f[z - T], \tag{3}$$

$$y_j = g_j[s] \text{ for } j = 1 \text{ to } M. \tag{4}$$

Where

z is the net input of the fuzzy neuron,

$h[\,]$ is the aggregation function,

s is the state of the fuzzy neuron.

$f[\,]$ is the activation function,

T is the activating threshold,

$\{g_j[\,],\ j = 1, 2, \dots, M\}$ are the M output functions which represent the membership functions of the input pattern $\{x_1, x_2, \dots, x_N\}$ in all the M fuzzy sets $\{g_j[\,] \in [0,1]\}$.

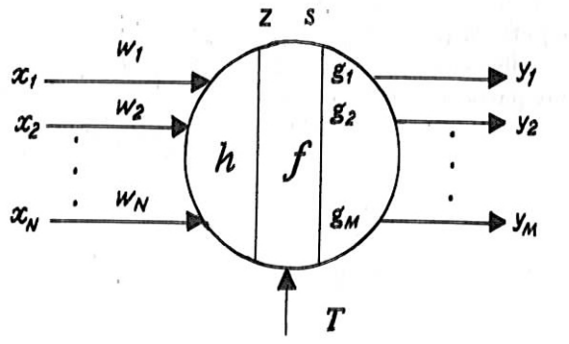


Figure 1. A fuzzy neuron

B. Input-FN

$$z = x \tag{5}$$

An Input-FN has only one input x.

C. Max-FN

$$z = \max_{i=1}^{N}(w_i x_i). \tag{6}$$

The aggregation function $h[]$ of a fuzzy neuron is maximum function.

D. Min-FN

$$z = \min_{i=1}^{N}(w_i x_i). \tag{7}$$

The aggregation function $h[\ ]$ of a fuzzy neuron is a maximum function.

E. Competitive-FN

$$y = g[s - T] = \begin{cases} 0 & \text{if } s < T \\ 1 & \text{if } s > T \end{cases} \tag{8}$$

$$T = t[c_1, c_2, \ldots, c_k], \tag{9}$$

where s the state of the fuzzy neuron, $t[\ ]$ is the threshold function, c_k (for $k = 1$ to K) are competitive variables of the fuzzy neuron.

2.2. Structure and mechanism of the fuzzy neural network

Operation mechanism: When each input pattern is provided, at first the neural network fuzzifies this pattern and then calculates the similarity density between the input pattern and all learned patterns. Next, the network will reach its conclusion by choosing the pattern which is most similar to the input pattern in density. Finally, the network performs a defuzzification process and sends the result to an output.

The network's structure:

The first layer (the input layer): accepts patterns into the network including INPUT-FN's.
Each INPUT-FN in this layer corresponds to one element of an input pattern. Assuming each input pattern has $N_1 \times N_2$ elements, then the first layer has $N_1 \times N_2$ INPUT-FN's.
The algorithm for the $(i, j)^{\text{th}}$ element in first layer is

$$s_{ij}{}^{[1]} = z_{ij}{}^{[1]} = x_{ij}, \text{ for } i = 1 \text{ to } N_1,\ j = 1 \text{ to } N_2 \tag{10}$$

$$y_{ij}{}^{[1]} = s_{ij}{}^{[1]}/P_{\nu\max}, \text{ for } i = 1 \text{ to } N_1,\ j = 1 \text{ to } N_2 \tag{11}$$

where $X_{i,j}^{\text{th}}$ element value of an input pattern ($x_{ij} \geq 0$), $P_{\nu\max}$ is the maximum element value among all input patterns.

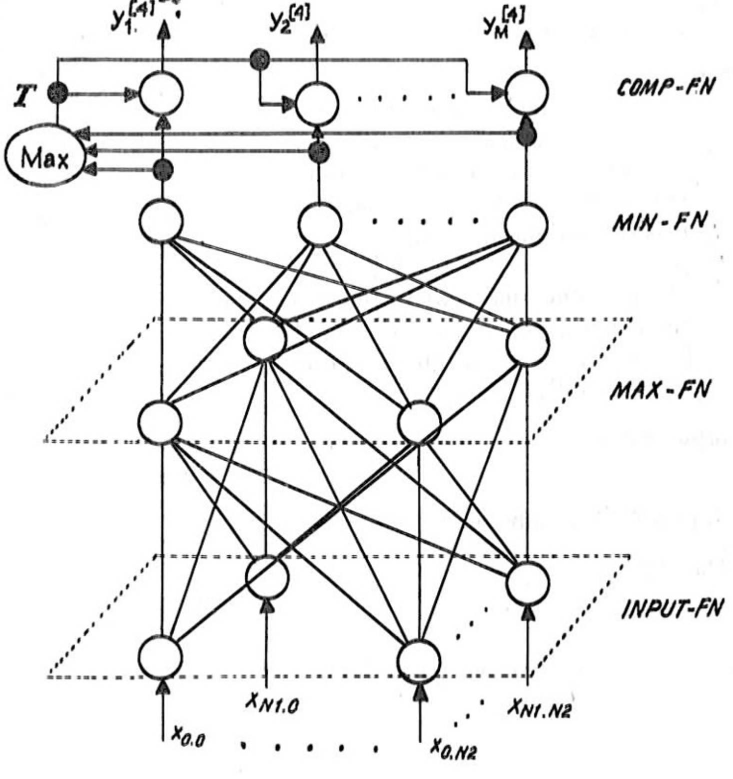


Figure 2. Four-layer feedforward fuzzy neural network

The second layer: is to fuzzify input patterns through a weight function $w[m,n]$ and consisted of $N_1 \times N_2$ MAX-FN's. The algorithm of the $(i,j)^{th}$ MAX-FN in this layer as follows:

- The state of the $(p,q)^{th}$ MAX-FN in this layer is:

$$s_{pq}{}^{[2]} = \max_{i=1}^{N_1} \left(\max_{j=1}^{N_2} \left(w[p-i, q-j] y_{ij}{}^{[1]} \right) \right) \quad \text{for } p = 1 \text{ to } N_1,\ q = 1 \text{ to } N_2, \tag{12}$$

where $w[p-i, q-j]$ is the weight connecting to the $(i,j)^{\text{th}}$ INPUT-FN in the first layer to the $(p,q)^{\text{th}}$ MAX-FN in the second layer which is specified by:

$$w[m,n] = \exp(-\beta^2(m^2+n^2)) \quad \text{for } m = -(N_1-1) \text{ to } (N_1-1),\ n = -(N_2-1) \text{ to } (N_2-1), \tag{13}$$

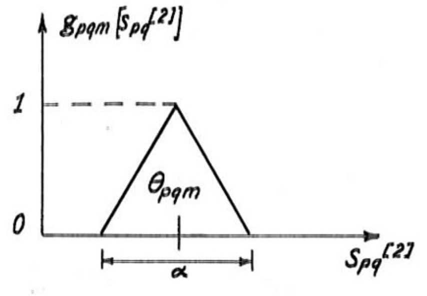
$w[m,n]$ can also be called the fuzzyfication function.
β can be called the fuzzification coefficient.

$$y_{pqm}{}^{[2]} = g_{pqm}\left[s_{pq}{}^{[2]}\right] = \begin{cases} 1 - 2|s_{pq}{}^{[2]} - \theta_{pqm}|/\alpha & \text{if } \alpha/2 \geq |s_{pq}{}^{[2]} - \theta_{pqm}| \geq 0 \\ 0 & \text{if in any other case} \end{cases} \tag{14}$$

for $\alpha \geq 0$, $p = 1$ to N_1, $q = 1$ to N_2, $m = 1$ to M.

- The outputs of the $(p,q)^{\text{th}}$ MAX-FN in this layer are:

Each MAX-FN has M outputs which is equal to the number of FN's in the third layer. $y_{pqm}{}^{[2]}$ is the m^{th} output of the $(p,q)^{\text{th}}$ MAX-FN whose values belonging into $[0, 1]$, they can be connected to the m^{th} MIN-AN in the third layer.



θ_{pqm} is the central point of the vase of function $g_{pgm}[S_{pq}^{[2]}]$,
α is the corresponding coefficient which can be selected depending on each set patterns, (α and θ_{pqm} are determined by the learning algorithm).

Figure 3. The output function of a MAX-FN in the second layer

The third layer (represents learned patterns): Including M MIN-FN's, each MIN-FN in the third layer represents one learned pattern.

The output of the m^{th} MIN-FN in the third layer is:

$$y_m{}^{[3]} = s_m{}^{[3]} = \min_{p=1}^{N_1} \left(\min_{q=1}^{N_2} \left(y_{pqm}{}^{[2]} \right) \right) \quad \text{for } m = 1 \text{ to } M. \tag{15}$$

The fourth layer (the output layer): We use COMP-FN's in this layer, one for each learned pattern (in M learned patterns), in order to supply nonfuzzy outputs.

If an input pattern is most similar to the m^{th} learned pattern, then the output of the m^{th} COMP-FN in the fourth layer is 1 while other outputs are 0.

The number of COMP-FN's in the output layer is equal to M. The algorithm of the m^{th} COMP-FN in the fourth layer is:

$$s_m{}^{[4]} = z_m{}^{[4]} = y_m{}^{[3]} \quad \text{for } m = 1 \text{ to } M, \tag{16}$$

$$y_m^{[4]} = g\left[s_m^{[4]} - T\right] = \begin{cases} 0 & \text{if } s_m^{[4]} < T \\ 1 & \text{if } s_m^{[4]} = T \end{cases} \quad \text{for } m = 1 \text{ to } M, \tag{17}$$

$$T = \max_{m=1}^{M} \left(y_m^{[3]}\right) \quad \text{for } m = 1 \text{ to } M, \tag{18}$$

where T is the activation threshold of all the COMP-FN's in the fourth layer.

2.3. Self-organizing learning algorithm of the FNN

Necessary parameters: The parameters of the output functions of the MAX-FN's in the second layer, α and θ_{pqm} (for each set of p, q and m); the parameter of the fuzzification function, β; and the number of FN's in each of the third and the fourth layer (M). T_f is the fault tolerance of the fuzzy neural network ($0 \leq T_f \leq 1$) and K is the total number of training patterns (for $k = 1$ to K). The steps of the learning algorithm are:

Step 1. Create $N_1 \times N_2$ MIN-FN's in the first layer and $N_1 \times N_2$ MAX-FN's in the second layer. Choose an appropriate value for α ($\alpha > 0$) and an appropriate value for β.

Step 2. Set $M = 0$ and $k = 1$.

Step 3. Set $M = M + 1$. Create the M^{th} MIN-FN in the third layer and the M^{th} COMP-FN in the fourth layer. Set:

$$\theta_{pqM} = s_{pqM}^{[2]} = \max_{i=1}^{N_1} \left(\max_{j=1}^{N_2} (w[p-i, q-j] x_{ijk}) \right) \quad \text{for } p = 1 \text{ to } N_1,\ q = 1 \text{ to } N_2, \tag{19}$$

where θ_{pqM} is the central point of the M^{th} output function of the $(p,q)^{th}$ MAX-FN in the second layer. $X_k = \{x_{ijk}\}$ is the k^{th} training pattern.

Step 4. Set $k = k + 1$. If $k > K$, then the learning procedure is finished. In any other case, input the k^{th} training pattern to the network and compute the output of the current fuzzy neural network (with M fuzzy neuron in the third and fourth layer). Set

$$\delta = 1 - \max_{j=1}^{M} \left(y_{ik}^{[3]}\right), \tag{20}$$

where $y_{jk}^{[3]}$ is the output of the j^{th} MIN-FN in the third layer for the k^{th} training pattern X_k. If $\delta \leq T_f$ go to Step 4. If $\delta > T_f$, go to Step 3.

3. THE PROPOSED FINGERPRINT CLASSIFICATION METHOD

At first, a fingerprint image is binarized using a bilevel-thresholding process.
After that, the image is executed under the thinning process of Baruch [6].

3.1. Directional matrix creation [4]

After being thinned, the image will be divided into 6×5 subareas. The main direction of ridges on each subarea is extracted, then stored in a 6×5 matrix so called a directional matrix. All directions of ridges are divided into four directions and are represented by symbols "\", "–", "/", "|". Each directional matrix describes the global flow shape of the original fingerprint.

Algorithm for extracting a directional matrix:

Step 1. In the thinned image, try to find two vertical boundary lines on the right and the left of the fingerprint.

Step 2. Insert 5 vertical lines in equal horizontal spacing between the two boundary lines.

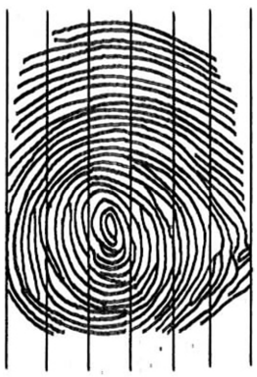
Step 3. In each vertical line, we navigate all the ridge points. For each ridge point, calculate its direction, then assign one of the four directions mentioned above to it.

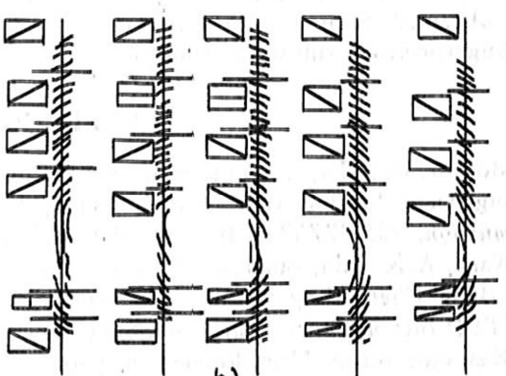
Step 4. Divide ridge point on each line into 5 parts, each part has the same number of ridge points, then divide the fifth part into two subparts.

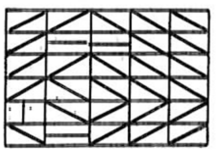
In each subpart, calculate the number of each appearing direction and choose the direction with maximum number.

After the major direction of each part is extracted, a 6×5 matrix is then established to store these major directions.

The above 6×5 directional matrix is digitalized into a corresponding 6×5 matrix with numeric elements (0 corresponds to "\", 1 corresponds to "/", 2 corresponds to "–", 3 corresponds to "|").


a)


b)


c)

$$\begin{array}{ccccc} 1 & 1 & 0 & 0 & 0 \\ 1 & 2 & 2 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 3 & 0 & 1 & 1 & 1 \\ 0 & 2 & 1 & 1 & 1 \end{array}$$

d) The matrix after digitalized

Figure 4. Illustration for step of the algorithm (WHORL fingerprint only)

3.2. Applying a FNN

Four standard image samples represent the four mentioned classes: Whorl, Left Loop, Right Loop, and Arch. After that they are transformed into digitalized directional matrices, those matrices will then be employed as first training patterns for a FNN.

With this set of patterns, perform the learning algorithm of the FNN (given $T_f = 0.3$, $\alpha = 2.0$, $\beta = 0.3$), we construct the corresponding desired set of patterns (those ones are four 6×5 status matrices with real-value elements (namely, fuzzy matrices of input images)).

After the FNN has learned from the above four patterns, the system makes experiments of classification ability through many different fingerprints. These patterns will be transformed into digitalized 6×5 directional matrices (using the algorithm for extracting directional matrices), then those matrices will continue to be classified by the FNN.

4. EXPERIMENTS AND RESULTS

The above system has been implemented in C Language on Pentium II-PC 266 Mz. Our experiments are performed with 53 fingerprint patterns (Given parameters of the FNN: $T_f = 0.3$, $\alpha = 2.0$, $\beta = 0.3$).

Experimental results as follows: classification rate is 96,23%, regardless of the executable time. Hence, the classification rate is high so that the system is considered entirely applicable.

5. CONCLUSIONS

In summary, in this paper, we have proposed a new fingerprint classification approach based on a combination between the approach of extracting features with the power of fuzzy neural networks. We have entirely employed the learning and understanding capacity of FNNs.

The obtained results can demonstrate that it is a reasonable and high available work to apply FNNs in fingerprint classification problems.

REFERENCES

[1] B. Moayer, K. S. Fu, A syntactic approach to fingerprint, *Pattern Recognition* **6** (1974) 1-23.

[2] Hoang Kiem, Le Hoai Bac, Le Hoang Thai, *A fuzzy neural network for Vietnamese character recognition*, VJFUZZY'98, Ha Long Bay, Vietnam, 1998.

[3] K. Karu, A. K. Jain, Fingerprint classification, *Pattern Recognition* **29** (3) (1996) 389-404.

[4] Ling-Hwei Chen, Ming-Yo-Wu, A new approach for fingerprint classification, *Proceedings of the IASTED International Conference*, Singapore, 1997, p. 211-214.

[5] M. Kawagoe and A. Tojo, Fingerprint pattern classification, *Pattern Recognition* **17** (3) (1984) 295-303.

[6] O. Baruch, Line thinning by line following, *Pattern Recognition Lett* **8** (1989) 217-276.

Received February 10, 1999
Revised August 2, 1999

Natural Science University, National University Ho Chi Minh City.