A New Detection Method of Singular Points of Fingerprints Based on Neural Network

Liu Yong-xia
Electrical Engineering Department
University of Electronic Science and Technology of China
Chengdu, P.R.China
pencil1020@163.com

Qi Jin
Electrical Engineering Department
University of Electronic Science and Technology of China
Chengdu, P.R.China
qijin@ee.uestc.edu.cn

Xie Rui
Electrical Engineering Department
University of Electronic Science and Technology of China
Chengdu, P.R.China
xr89757@126.com

Abstract—Singular points detection, a crucial step for fingerprint identification system, is accurately robust, and reliable. In the processing of the fingerprint image matching and classification, many method use singular points to align two fingerprint images to surmount the problems about rotation and translation. The performances of the fingerprint recognition system rely on the effect of singular points detection. This paper introduces a new method through back-propagation neural network (BPNN) to detect singular points of fingerprint images. The algorithms that directly make use of the gray-scale of fingerprint images to detect singularities are different from the other methods, which are based on the neural network. The algorithm is tested accurately and reliably for a great many of fingerprint images in the FVC2002.

Keywords—singular points; BP neural network; detection; fingerprint image;

I. INTRODUCTION

In the biometric technology, fingerprint identification is the most important technology and has been used for a very long time, because of their uniqueness and immutability. Most fingerprint verification and identification are implemented in automatic fingerprint identification system using the ridge ending and bifurcation of fingerprint images. While the singularity is intrinsic global feature of fingerprints and is invariant to the rotation and transformation.

A fingerprint usually consists of 1-3 singularity. These singular points could be divided into two types: core and delta, also are considered the global feature of fingerprint image (See the Fig.1). The importance of singular points is mainly reflected in several aspects:

1) Position or rotation transformation of singular points is invariant. In the registration of fingerprint, the advantage can be taken as the landmark points.

2) The mount, type and relative position of singular points are limited, so they can be used to classify the fingerprint images into few categories.

Up to now, there have been great many algorithms to research singularity detection of fingerprint images. The reference [1] utilizes the Poincare index based method the used the orientation field on the curves. But the result exist many spurious detections, especially for some low quality fingerprint images. Jie Z. et. Propose a novel algorithm for singular points detection after an initial detection using the conventional Poincare Index method [2]. Nannan W. and Jie Z. introduced a model singular points detection algorithm to describe the orientation field around points and made use of Hough transform to compute the coefficients of the model [3]. Jun L. et. proposed to use interactive mechanism to detecting and validating a singular point [4]. Asker M. Bazen and Sabih H. Gerez shown a very accurate detection of singular points and obtain orientation of those points from
the directional field [5]. In this article, we use the fingerprint image gray values as input to the neural network value. Here, we have to determine in advance of each image block corresponding to the target, because the BP neural network, the input and output target values are determined. So we take the output value of network as singular point and non-singular point, or signed 1 and 0. The method is proved to be feasible and robust.

The paper from the following components: section 2 describes backpropagation neural network. Section 3 shows the method to detect singular points using the BP algorithm. Section 4 gives the experimental results by the BP neural network.

II. BACKPROPAGATION NEURAL NETWORK

BP neural network was created by generalizing the Widrow-Hoff learning rule to multiple-networks and nonlinear differentiable transfer functions. Input vectors and the corresponding vectors are used to train a network until it can approximate a function, associate input vectors with specific output vectors, or classify input vectors in an appropriate way as defined [6]. Networks with biases, a sigmoid layer, and a linear output layer are capable of approximating any function with a finite number of discontinuities. Standard Backpropagation is a gradient descent algorithm.

A. Backpropagation mode

BP neurons and other neurons are similar, except that the transfer function of BP neurons is nonlinear [7]. The model of neuron is shown in Fig.2.

![Figure 2. General model of neurons.](image)

In the BP network, there are no connections within a layer. The output of the hidden units is distributed over the next layer hidden layer, until the last layer of hidden units. See the Fig.3.

![Figure 3. Multilayer feed forward networks model.](image)

In the model of the Fig 3, the \( N_i \) inputs are fed into the first layer of \( N_{h1} \) hidden units. The activation of a hidden unit is a function \( f_j \) of the weighted inputs plus a bias, as the equation (1).

\[
y_j(t+1) = f_j(z_j(t)) = f_j(\sum w_{ji} x_i(t) + \theta_j(t))
\]

B. Backpropagation training process and algorithm

Typically, the input leading to an output is similar to the correct output for input vectors used in training that are similar to the new input being presented. The training functions are usually used to solve specific problems through training feed-forward neural networks. The process of training includes four parts, as the Fig.4.

![Figure 4. A general training process in the neural network.](image)

The implementation of Backpropagation learning rule updates the network weights and biases in the direction in which the performance function decreases rapidly following the negative of the gradient. The formula (2) is the iteration of algorithm.

\[
x_{i+1} = x_i - \alpha_i g_i
\]

Where, \( x_i \) is a vector of current weight and biases; \( g_k \), \( \alpha_k \) is the current gradient and learning rate respectively.

The central idea behind this solution is that the errors for the units of the hidden layer are determined by back-propagating the errors of the units of the output layer. For this reason, the method is often called the back propagation learning rule. When a learning pattern is clamped, the activation values are propagated to the output units, and the actual network output is compared with the desired output values, we usually end up with an error in each of the output units. Then we must bring to error to zero.

Let us suppose that the input nodes, hidden layer nodes and output nodes are \( x_i \), \( y_j \) and \( z_l \) in the three-layer backpropagation network, respectively. The weights of network are \( w_{ji} \) between input nodes and hidden layer nodes. Similarly, the weights of hidden-output nodes are \( v_{kl} \). When the expectation of output nodes is \( t_j \), the formulas of network model as shown below:

\[
y_j = f(\text{net}_j), \text{and } \text{net}_j = \sum w_{ji} x_i - \theta_j
\]

\[
z_l = f(\text{net}_l), \text{and } \text{net}_l = \sum v_{kl} y_j - \theta_l
\]

\[
E_n = \frac{1}{2} \sum (t_l - z_l)^2
\]

The equation (3) represents the output value of hidden layer, (4) express the value of output nodes in the last layer of network and (5) indicates the error of output nodes.

The BP algorithm is to amend vales of the network weights and thresholds along the direction with the fastest decline of performance function. Accordingly, we can exploit the derivative of error function on output and hidden
layer to amend weights. We can adopt similar method to carry out amendment on threshold value.

1) For output nodes derivative.

\[ \frac{\partial E}{\partial v_j} = \sum \frac{\partial E}{\partial z_j} \cdot \frac{\partial z_j}{\partial v_j} \]

With equations (3) (4) and (5), we can obtain

\[ \frac{\partial E}{\partial v_j} = -(t_j - z_j) \cdot f'(net_j) \]

Set the error of output nodes as \( \delta_j \)

\[ \delta_j = (t_j - z_j) \cdot f'(net_j) \]

Then,

\[ \frac{\partial E}{\partial v_j} = -\delta_j \]

2) For hidden-layer nodes derivative.

\[ \frac{\partial E}{\partial w_{ij}} = \sum \frac{\partial E}{\partial z_j} \cdot \frac{\partial z_j}{\partial v_j} \cdot \frac{\partial v_j}{\partial w_{ij}} \]

The same resolution as the condition (7),

\[ \frac{\partial E}{\partial w_{ij}} = -(t_j - z_j) \cdot f'(net_j) \cdot v_j \cdot f'(net_i) \cdot x_i \]

\[ = -\sum \delta_j v_j f'(net_i) \cdot x_i \]

Suppose the error of hidden layer nodes as \( \delta_i \), the expression (12) show that the errors of output nodes \( \delta_j \) become the error of hidden-layer nodes through back-propagation from the weights \( v_j \) to the hidden-layer nodes \( y_j \).

\[ \delta_i = f'(net_i) \sum \delta_j v_j \]

Then,

\[ \frac{\partial E}{\partial w_{ij}} = -\delta_i x_i \]

3) For threshold of output nodes derivative.

\[ \frac{\partial E}{\partial \theta_j} = \sum \frac{\partial E}{\partial z_j} \cdot \frac{\partial z_j}{\partial \theta_j} \]

By equation (4) can get:

\[ \frac{\partial E}{\partial \theta_j} = (t_j - z_j) \cdot f'(net_j) \]

4) For threshold of hidden-layer nodes derivative.

\[ \frac{\partial E}{\partial \theta_i} = \sum \frac{\partial E}{\partial z_i} \cdot \frac{\partial z_i}{\partial \theta_i} \]

By equations (3) and (4) can be gained:

\[ \frac{\partial E}{\partial \theta_i} = -(t_j - z_j) \cdot f'(net_j) \cdot v_j \cdot f'(net_i) = \delta_i \]

5) For transfer function derivative.

\[ f(x) = \frac{1}{1 + e^{-x}} \]

Then,

\[ f'(x) = f(x)(1 - f(x)) \Rightarrow f'(net_j) = f'(net_j)(1 - f(net_j)) \]

Output nodes:

\[ z_j = f(net_j) \Rightarrow z'_j = f'(net_j)z_j(1 - z_j) \]

Hidden-layer nodes:

\[ y'_j = f'(net_j) \Rightarrow y'_j = y_j(1 - y_j) \]

As the weight amendment is proportional to the gradient descent along with the error function.

\[ v_j(k+1) = v_j(k) + \Delta v_j, \text{ and } \Delta v_j = -\eta \frac{\partial E}{\partial v_j} \]

\[ w_{ij}(k+1) = w_{ij}(k) + \Delta w_{ij}, \text{ and } \Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} \]

Lastly, there are two methods to solve function of gradient: incremental mode and batch mode. Here, we will select the second mode to deal with gradient function.

III. SINGULAR DETECTION USING THE BPNN

A. The training samples of fingerprint image

In order to train the BP neural network, we extract about 6000 image-blocks (see samples in the Fig.5) from db3_A in the FVC2002 as the train template.

![Image](https://example.com/image1)

(b)

Figure 5. The samples of network. (a) are non-singular points samples, their target values are set as 0; (b) are the singular points samples, their target values are 1.

B. The flow-chart of singular detection

The following flow-chart is the process of detection (see the Fig.6). The size of training’s image blocks is \( w \times w \). Similarly, we will segment detection image into the same size of training blocks.

![Image](https://example.com/image2)

Figure 6. Singular points detection flow-chart in the BP neural network.

C. The training samples of fingerprint image

The gray-scale of fingerprint images blocks are inputs to the neural network. The output value of neural network is set as 1 and 0. We adopt three layers neural network to train, because it has been suggested that three layers network achieve in approximating any finitely and discontinuity functions to arbitrary, provided the activation function of the hidden-layer units are non-linear function. Another critical issue is how to set the number of the hidden-layer, because of this is direct affect to final result. So we use the theoretical results tried many times by the predecessors, such as the formula (19).

\[ h = \sqrt{(n + m) + \beta} \]
Where, n is number of the input units, m is number of the output units and $\beta$ is a constant between 1 and 10.

The hidden and output layer transfer functions are non-linear function. Account for the speed of neural network training and target range, we take tansig function as hidden layer transfer function, and logsig as output layer transfer function.

We make use of variable learning rate methods to set neural network. Variable learning rate, in other words, it will change along the process of training. The learning rate is related to each the weight value changes on each epoch. The larger learning rate leads to the larger weight value and the quicker learning speed. However, learning rate can not be infinite, because it may also affect whether the network is to reach a stable solution.

D. Simulate of the neural network

In the time of detection singular points, we extract the image blocks as the input of network from the fingerprint images. Each pixels of the fingerprint image must be input neural network for testing. The specific method is to get one-pixel point and take an area as $w \times w$ image block with this pixel as the center in its surrounding area, into the trained neural network to be simulated.

IV. EXPERIMENT

We use FVC2002 public db3_A to test our approach. The db3 is generated through precise biometrics 100 SC, and the size of each fingerprint image is $300 \times 300$, 500dpi [8]. The singular points of these fingerprints are manually labeled by experts as the ground truth beforehand.

The size of the image blocks must be determined. Because the detected fingerprint images mainly are in the db3, so we can set the parameters according to the sum of ridge and valley. From the reference [9], the value of W is about 10 pixels. The area of singular point includes three ridges or three valleys at least, so we take sample size as 35. The number of output layer is only one unit. Through multiple tests and experience value, we use 40 layers of hidden layer neurons to experiment. The goal, leaning rate and epochs of network training are set as $1 \times 10^{-3}$, 0.02 and 1500.

From the Tab1 we can get that the target value of singular points and non-singular points samples are more than 0.5 and less than 0.5, respectively. The Tab2 display the tested sample blocks results. As the result of this situation, fingerprint images singular points can be detected. The fingerprint imaged detection results through BP neural network are showed in the Fig.7.

| TABLE I. SOME TRAINING SAMPLES ( BLOCK SIZE:35 × 35) |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| Sm 0            | Res (10^-2)     | Sm 1            | Res (10^-2)     |
| ![Sample Image](image1) | 6.43            | ![Sample Image](image2) | 64.07          |
| ![Sample Image](image3) | 2.88            | ![Sample Image](image4) | 88.24          |
| ![Sample Image](image5) | 11.81           | ![Sample Image](image6) | 76.88          |
| ![Sample Image](image7) | 5.34            | ![Sample Image](image8) | 78.12          |
| ![Sample Image](image9) | 7.48            | ![Sample Image](image10) | 70.05          |
| ![Sample Image](image11) | 3.19            | ![Sample Image](image12) | 70.32          |
| ![Sample Image](image13) | 1.41            | ![Sample Image](image14) | 82.59          |

| TABLE II. SOME TEST SAMPLES ( BLOCK SIZE:35 × 35) |
|-----------------|-----------------|-----------------|-----------------|
| Sm 0            | Res (10^-2)     | Sm 1            | Res (10^-2)     |
| ![Sample Image](image15) | 9.27            | ![Sample Image](image16) | 68.35          |
| ![Sample Image](image17) | 4.23            | ![Sample Image](image18) | 75.42          |
| ![Sample Image](image19) | 4.81            | ![Sample Image](image20) | 55.45          |
| ![Sample Image](image21) | 16.57           | ![Sample Image](image22) | 59.85          |
| ![Sample Image](image23) | 6.35            | ![Sample Image](image24) | 66.65          |
| ![Sample Image](image25) | 10.32           | ![Sample Image](image26) | 63.35          |
| ![Sample Image](image27) | 4.60            | ![Sample Image](image28) | 70.48          |

Figure 7. Fingerprint image samples singular detection results. (a)–(d) and (i)–(l) present the fingerprint images whose size are 25% of the original images; (e)–(h), (m)–(p) show detected fingerprint singular points images. Here, the brighter spots represent detected singular points and the black region are others region of fingerprint without singular points.

V. CONCLUSION

This paper proposes a new algorithm to detect singular points approach for fingerprint images using the gray value in the backpropagation neural network. We know that neural network is often used in the pattern recognition. Therefore, we used this pattern recognition method to distinguish singular points and non-singular points to realize the fingerprint singular points detection. Although singular detection is quick, sample blocks collection is a very time-consuming job. Moreover, the collected samples are impossible to include all conditions of fingerprint. In addition, we can extract the fingerprint image’s features (such as orientation, gradient and other information) as the inputs of neural network to detect singularity to improve the process.

REFERENCES


