Fingerprint Recognition Based on Spectral Minutiae Representation and Deep Learning

Shu Yu
Faculty of Electrical Engineering, Mathematics & Computer Science
University of Twente
Supervised by:
Prof.Dr.lr.R.N.J. Veldhuis(1^{st}), Chris Zeinstra, Yang Miao

***Abstract*—This paper proposes to apply spectral minutiae representation [1]–[7] and deep learning for fingerprint recognition. Fingerprint is one important biometric feature, and its recognition typically incorporates four steps: image acquisition, processing, feature extraction and comparison. The powerful functionality of deep learning in imaging processing makes it plausible to recognize the fingerprint patterns. Conventionally, deep learning has mainly been used to extract the minutiae or the feature vectors from raw fingerprint images. There has been no hybrid use of the two. In this paper, we propose to use the spectral minutiae representation and the convolutional neural network (CNN) in combine to advance direct matching of spectral minutiae representation in fingerprint recognition. In the proposed approach, a minutiae set is represented by a spectrum with fixed size, specifically, this spectral minutia representation converts a minutiae set into a 128×256 sized magnitude spectrum. This spectrum serves as the input to CNN, while the output of CNN is a 128-dimentional feature vector. The fingerprint recognition is then completed by feature vector comparison. In this paper, the CNN with 19 layers is used and the whole network is trained by triplet loss. This proposed approach makes the fingerprint recognition using CNN more efficient, as no complicated pre-processing is needed compared to process endowing raw image to CNN. The performance of the proposed approach is compared to direct matching of complex spectral minutiae representation [1].**

***Keywords*—fingerprint recognition, spectral minutiae representation, convolutional neural network**

I. Introduction

Biometric technology determines a person's identity by extracting and comparing human biological or behavioral characteristics. As the possibility of hacker intrusion increases with the development of IT technology [8], the interest in biometric technology for authentication has also greatly increased. In order to distinguish one person from the others, we need a convenient, safe and effective identification technology. Biometrics has the characteristics of portability, security, distinctiveness and stability, which gives it strong advantages compared with traditional identification technology such as password recognition. Biometric technology can be divided into behavioral features, which include signature and walking rhythm and physiological features which include fingerprint, iris and face recognition [4]. Physiological features have been widely used in forensics, transaction authentication and cellphone unlocking [9].

With the increasing use cases of fingerprint recognition, the algorithm of fingerprint recognition is also being improved gradually. Currently, minutiae-based fingerprint recognition algorithms [10], [11] are the most traditional. The starting points, end points, joint points and bifurcation points of ridges are called minutiae. In general, minutiae-based fingerprint recognition algorithm refers to the comparison of the position and the direction of the minutiae extracted from the fingerprint images. Good results have been obtained by deep-learning based minutiae extraction [12]. However, in different fingerprint images, different number of minutiae can be extracted which means the minutiae sets used for matching have different lengths. The number of extracted minutiae may even vary if different images of the same identity are used, as shown in figure 1. This problem leads to the hard computation to compare two unordered sets of different sizes. Spectral minutiae representation has been proposed to solve this problem. It uses a fixed-size magnitude spectrum to represent a minutiae set of a fingerprint image and compares magnitude spectra to achieve fingerprint recognition. Another approach to fingerprint recognition is to extract feature vectors from the original fingerprint images by using deep learning directly. These feature vectors are used to determine whether the two fingerprints are from the same identity. This idea was first proposed in [13], however, at that time, it was an early stage of deep learning development, only fully connected layers were used for image feature extraction. In 2019, Shervin et al. applied a convolutional neural network to fingerprint identification [14]. They preprocessed the original fingerprint images with traditional image processing methods such as desiccation and filtering, after which Resnet-50 is used to extract feature vectors and the vectors are compared.

In this paper, we study to what extent deep-learning based feature extraction can be applied successfully to spectral minutiae images for fingerprint comparison. Using NIST Biometric Image Software, we extract a minutiae set from every fingerprint image. The complex spectral minutiae representation (SMC) [1] is used to transform the position and direction of the minutiae into a fixed-size image by means of a Fourier transform. The fixed-size image is then used as the input of the convolutional neural network to extract fixed size feature vectors. Several similarity measures are used to compare the feature vectors and these measures are compared here.

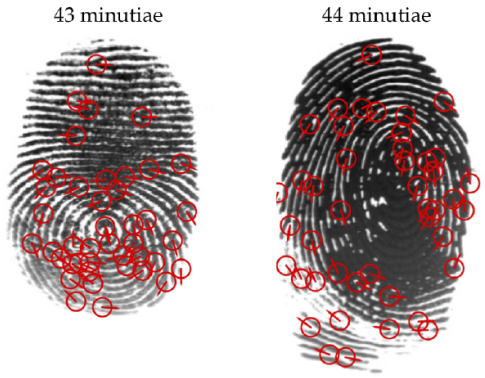


Fig. 1. Different images of the same identity have different numbers of minutiae

In this paper, Darknet-19 [15] is used as the convolutional neural network. It contains nineteen convolutional blocks and one fully connected layer. For each convolutional block, there is one convolutional layer, one activation layer and one batch normalization layer. Triplet loss [16] is used as the loss function to train all the parameters of the network.

The rest of the paper is organized as follows. Related work and background theory is shown in Sec.II . Sec. III describes the proposed method. Experiments are shown in Sec. IV. The last part is the conclusion.

II. Related Work and Background Theory

A. Spectral Minutiae Representation

Fingerprint recognition is usually achieved by minutiae comparison. Each minutia can be described by parameters (x, y, θ) [17], where (x, y) is the location of the minutia in Cartesian coordinate system. The orientation θ of a minutia can be incorporated by using the spatial derivative of m(x, y) in the direction of the minutia orientation. This is the reason that minutiae-based fingerprint recognition has some drawbacks. First of all, mostly, the numbers of minutiae extracted from fingerprint images are different, so it is difficult to obtain a fixed length feature. Secondly, the extracted minutiae are often with different orders, and the corresponding relationship of the minutiae is also unknown, therefore, comparing the minutiae is hard. In addition, the translation or rotation of a fingerprint image will also affect the comparison result. Spectral minutiae representation can be used to solve these problems.

Location-based spectral minutiae representation (SML) [2] only uses the location information of the minutiae. In the spatial domain, every minutia is represented by a Dirac pulse. Given Z minutiae, these minutiae are represented by $m_i(x,y) = \delta(x - x_i, y - y_i), i = 1...Z$, where (x_i, y_i) is the location of the i-th minutia in the fingerprint image. The Fourier transform of $m_i(x,y)$ is given by:

$$F\{m_i(x,y)\} = exp(-j(w_x x_i + w_y y_i)) \tag{1}$$

where ω_x and ω_y are the spatial frequencies in x and y direction, respectively. Therefore, the location-based spectral minutiae representation can be defined as:

$$M_L(w_x, w_y) = \sum_{i=1}^{Z} exp(-j(w_x x_i + w_y y_i)) \tag{2}$$

In order to reduce the sensitivity to small variations in minutiae locations in the spatial domain, a Gaussian low-pass filter is used to attenuate the higher frequencies. Then the magnitude of M is:

$$\left|M_L(w_x, w_y; \sigma_L^2)\right| = \left| exp\left(-\frac{w_x^2 + w_y^2}{2\sigma_L^{-2}}\right) \sum_{i=1}^{Z} exp(-j(w_x x_i + w_y y_i)) \right| \tag{3}$$

where σ is the standard deviation of the Gaussian filter. After these transforms, the magnitude of M has become translation invariant. Finally, the magnitude spectrum is re-mapped to a polar-logarithmic grid to make the rotation and scaling of the input become translations. The result of SML is shown in figure 2. And in figure 2 (b), the horizontal axis represents the rotation angle of the spectral magnitude; the vertical axis represents the frequency of the spectral magnitude.

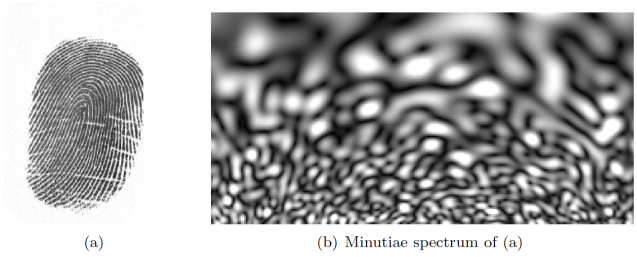


Fig. 2. Example of minutiae magnitude spectrum using SML. (a) is a given fingerprint image, (b)is the magnitude spectrum of (a) sampled on a polar-logarithm grid

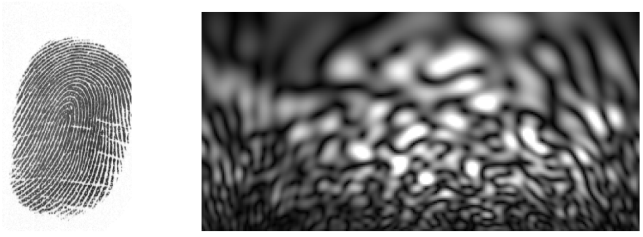
Orientation-based spectral minutiae representation (SMO) [3] not only uses the location information of the minutiae, but also the orientation information. Thus, to every minutia in a fingerprint, a function $m_i(x,y,\theta)$ is assigned being the derivative of $m_i(x,y)$ in the direction θ_i, such that

$$F\{m_i(x,y,\theta)\} = j(w_x cos\theta_i + w_y sin\theta_i) exp(-j(w_x x_i + w_y y_i)) \tag{4}$$

And then, as in SML, a Gaussian filter is used and the magnitude of the spectrum yields is kept

$$\left|M_O(w_x, w_y; \sigma_O^2)\right| = \left| exp\left(-\frac{w_x^2 + w_y^2}{2\sigma_O^{-2}}\right) \sum_{i=1}^{Z} j(w_x cos\theta_i + w_y sin\theta_i) \cdot exp(-j(w_x x_i + w_y y_i)) \right| \tag{5}$$

As with the SML algorithm, the magnitude spectrum is also re-mapped into the polar-logarithmic grid. And the result of SMO is shown in figure 3. And in figure 3 (b), the horizontal axis represents the rotation angle of the spectral magnitude; the vertical axis represents the frequency of the spectral magnitude.



(a) (b) Minutiae spectrum of (a)

Fig. 3. Example of minutiae magnitude spectrum using SMO. (a) is a given fingerprint image, (b)is the magnitude spectrum of (a) sampled on a polar-logarithm grid

SMO did not show better results than SML, even though SMO incorporates the orientation of minutiae [3]. That is because in SMO, the orientation of the minutiae is incorporated as a derivative of the delta function, which amplifies the minutiae noise in the high frequency part of the SMO. Therefore, a Gaussian kernel with higher σ is needed for SMO to attenuate the noise in higher frequencies. However, the high frequency part also contains discriminative information, especially when the minutiae are of good quality. Another spectral minutiae representation called complex spectral minutiae representation (SMC) [1] compensates for these limitations, and it shows better performance than SML and SMO. Therefore, SMC is used for spectral minutiae representation in this paper, and it is described in more detail in Sec.III.

B. Neural Networks

Convolution is an important component of neural networks. The name of convolutional neural network comes from its convolutional operation. The main purpose of convolution is to extract features from the input images. Convolution can learn image features from a small piece of input data, and preserve the spatial relationship between pixels. The convolution used to process images in a convolutional neural network is usually a two dimensional convolutional computation. For a two dimensional convolutional computation, the convolutional kernel can only slide on the direction of x and y, but not on the depth(channel). Convolutional kernel defines a certain pattern, and the convolutional (correlation) operation is to calculate the similarity between each position of the image and the pattern. The more similar the current position is to the convolutional kernel, the stronger the response would be. Therefore, the feature map which is the result of the convolutional operation can represent the output of the feature extraction. The value of the response is the sum of the product of the kernel weights and its corresponding pixel value on the input image, which can be represented by (6)

$$conv_{x,y} = \sum_{i}^{p \times q} w_i v_i \tag{6}$$

where x, y are the coordinates of the input image, $p \times q$ is the size of the kernel, v_i is the value of pixel and w_i is the weight of the kernel. The progress can be shown in figure 4.

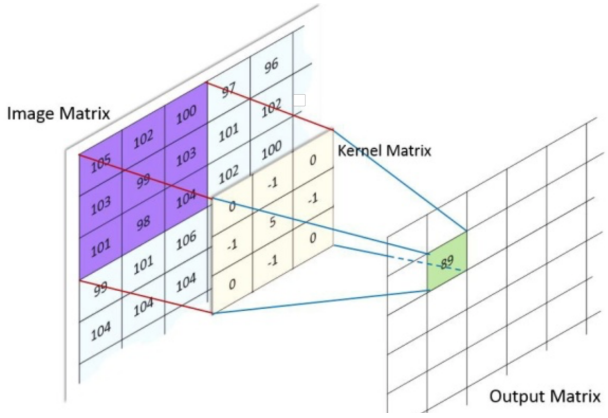


Fig. 4. The process of convolutional response calculation

Besides, network structure is an important factor in the effect of feature extraction. A multi-layer perceptron flattens all the pixels in an image into a long vector for feature extraction. However, this method has some limitations. Each entry of the vector is multiplied by a parameter, which results in too many parameters. The network trains slowly and is susceptible to overfitting. Convolutional neural networks [21] try to solve this problem. It shares weights resulting in an major parameter space reduction. Different structures of the convolutional layers will have different effects. Several classical convolutional neural networks are introduced as follows.

Lenet-5 [22] is a simple and basic network structure. It contains three convolutional layers having sigmoid activation functions to extract spatial features, and contains two pooling layers to down sampling. Finally, it contains two-layer fully connected structure is used to classify the features. The size of the output is in relation to the number of classes.

Deep-learning based image recognition was first proposed in Alexnet [23] in 2012. It contains eight layers for feature extraction. The first five layers are convolutional layers, and the first two layers and the fifth layer are followed by a pooling layer. The last three layers are fully connected layers. In order to facilitate the calculation, the model is divided into two blocks at the first two layers, and then the feature maps are concatenated to extract the overall features at the third convolutional layer. In addition, the network uses ReLU as the activation function to improve the network computing speed, and dropout is used to reduce the possibility of over fitting.

The VGG network [24] is an enhanced version of Alexnet. It emphasizes the depth of convolutional neural network in network design. The depth of the network is increased to 19 layers. The VGG network demonstrates that a good result can

be achieved by very small convolutional kernels if the whole structure of the network is deep enough. It also demonstrates that an effective way to improve network performance is to deepen the network structure. Even though this method increases the complexity of the network and, hence, the computational load, it enables the network to solve more complex problems. According to this paper, two convolutional layers with a kernel size of 3×3 have the same receptive field (the area mapped on an original image by a point on the feature map) of one convolutional layer with a kernel size of 5×5; three convolutional layers with a kernel size of 3×3 have the same receptive field of one convolutional layer with a kernel size of 7×7. In order to increase the depth of the network, in VGG network, two convolutional layers with a kernel size of 3×3 are used to replace the one 5×5-kernel size convolutional layer, and three convolutional layers with a kernel size of 3×3 are used to replace one convolutional layer with a kernel size of 7×7. This makes the number of parameters smaller, saves computing resources and decreases the possibility of overfitting.

The Siamese network [25] is a network structure used to measure the similarity of two inputs. It maps two inputs onto two vectors($G_w(x)$) by two networks that share weights, and uses the distance (L1 norm) between the two vectors to represent the difference between the two inputs. The structure is shown in the figure 5.

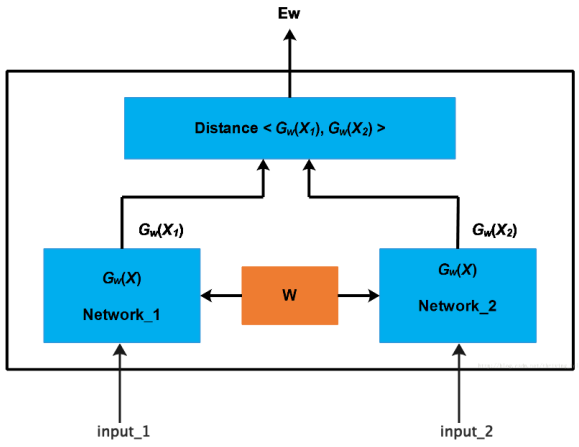


Fig. 5. The structure of Siamese Network

C. Loss Function

For image classification and recognition, in addition to improving the network structure, choosing an appropriate loss function will also bring great benefits to the result. As for the influence of the loss on the network, the most intuitive way is to update the parameters of the network by calculating the loss and then apply propagation. Different loss functions can make the model focus on learning different characteristics of data, and better extract this "unique" feature. Therefore, loss plays a guiding role in the process of network optimization.

A contrastive loss function [25] has been proposed for Siamese network. This loss function can effectively deal with the relationship between paired data in Siamese network. The expression of the contrastive loss is as follows:

$$L(W,(Y,X_1,X_2)) = \frac{1}{2N}\sum_{n=1}^{N} YD_w^2 + (1-Y)max(m-D_w,0)^2 \tag{7}$$

$$D_w(X_1,X_2) = \|X_1 - X_2\|_2 = (\sum_{i=1}^{P}(X_1^i - X_2^i)^2)^{\frac{1}{2}} \tag{8}$$

D_w is the euclidean distance (L2 norm) between two input vectors X_1 and X_2. Y is the label of whether the two samples match, $Y = 1$ means matching while $Y = 0$ means mismatching. N is the number of samples. m is a distance threshold set for the unmatched samples. If the distance of the unmatched samples is large enough, the loss is 0.

Softmax loss (9) is a popular loss function for classification problems.

$$L_s = -\frac{1}{m}\sum_{i=1}^{m} log\left(\frac{e^{x_{y_i}}}{\sum_{j=1}^{n} e^{x_j}}\right) \tag{9}$$

where x_j represents the output of a fully connected layer, the proportion of x_{y_i} must be increased, so that the trained parameters can make more samples fall into the decision boundary of its class. However, softmax loss function mainly considers whether the samples can be classified correctly, and lacks the constraints of the distance of intra class and inter classes.

Centerloss is used to reduce the distance of intra classes [26]. Centerloss randomly initializes a center point for each class, and calculates the distance between each sample and its corresponding center point, and then the parameters are updated gradually through back-propagation to decrease the distance.

$$L = L_s + L_c = -\frac{1}{m}\sum_{i=1}^{m}\log\left(\frac{e^{W_{y_i}^T x_i + b_i}}{\sum_{j=1}^{n} e^{W_j^T x_i + b_j}}\right) + \frac{\lambda}{2}\sum_{i=1}^{m}\|x_i - c_{y_i}\|^2 \tag{10}$$

Centerloss (10) adds penalty to the intra class samples with large distance, which makes the intra class distance compact during training.

The network trained by centerloss can make the intra class distance more compact, but it cannot increase inter class distance. In Arcface [27], a new loss function is proposed, which can not only reduce the intra class distance, but also increase the inter class distance, which further improves the classification effect. The loss of Arcface is based on the softmax loss. It can be found that $W_{y_i}^T x_i = \left\|W_{y_i}^T\right\| \|x_i\| \cos\theta$. If the bias b is ignored, the result depends on the angle θ.

In order to increase the distance of different classes, an extra angle t is added.

$$L_{Arcface} = -\frac{1}{m}\sum_{i=1}^{m}\log\left(\frac{e^{s(\cos(\theta_{y_i}+t))}}{e^{s(\cos(\theta_{y_i}+t))}+\sum_{j=1,j\neq y_i}^{n}e^{s\cdot\cos\theta_j}}\right) \quad (11)$$

The extra angle t can be considered as a margin between different classes. The Angle interval between the different classes is at least equal to this extra angle, which makes the intra class distance more compact and increases the inter class distance.

D. Classifier

Classifiers calculate the similarity score and compare it with a threshold to classify the fingerprint. Many classifiers are based on vectors comparing. The difference between two vectors is compared. The greater the difference, the smaller the similarity.

Euclidean distance is a commonly used distance definition, which describes a kind of distance between two vectors in m-dimensional space, or the length of the vector (i.e. the distance from the vector to the origin). Given two vectors $\mathbf{x}$, $\mathbf{y}$, the distance between them is given by:

$$d(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=0}^{m}(\mathbf{x}_i-\mathbf{y}_i)^2}, \quad (12)$$

with m the dimension of the vectors. When Euclidean distance is used to express similarity, (13) can be used, the smaller the distance, the greater the similarity.

$$sim(\mathbf{x},\mathbf{y}) = \frac{1}{1+d(\mathbf{x},\mathbf{y})} \quad (13)$$

Pearson correlation coefficient uses linear correlation between vectors to express similarity. It can be defined as:

$$s(\mathbf{x},\mathbf{y}) = \frac{n\sum\mathbf{xy}-\sum\mathbf{x}\sum\mathbf{y}}{\sqrt{n\sum\mathbf{x}^2-(\sum\mathbf{x})^2}\sqrt{n\sum\mathbf{y}^2-(\sum\mathbf{y})^2}} \quad (14)$$

The closer the score to 1, indicating that the higher the similarity between the two vectors.

Log likelihood ratio can also be used to measure the similarity of two vectors [28]. Given two feature vectors $\mathbf{x}$ and $\mathbf{y}$, two hypotheses can be formulated. H_s: the two vectors are from the same identity, H_d: the two vectors are from different identities. Given these two hypotheses, the likelihood ratio is defined as:

$$l(\mathbf{x},\mathbf{y}) = \frac{P(\binom{\mathbf{x}}{\mathbf{y}} \mid H_s)}{P(\binom{\mathbf{x}}{\mathbf{y}} \mid H_d)} \quad (15)$$

During feature reduction based on principal component analysis and linear discriminant analysis, whitening transforms(W_H, W_L) is applied to the vectors $\mathbf{x}$ and $\mathbf{y}$, and then u, d, v are obtained by a singular value decomposition. D is the number of singular value v_i on the diagonal of the diagonal matrix d. Δ_{DIF} and Δ_{SUM} can be defined as (16) and (17).

$$\Delta_{\mathrm{DIF},i} = \frac{v_i}{v_i-1}, i = 1, ..., D \quad (16)$$

$$\Delta_{\mathrm{SUM},i} = \frac{v_i}{v_i+1}, i = 1, ..., D \quad (17)$$

Under the assumption that the feature have Gaussion distributions, after the feature reduction and ignoring some constants, the similarity score can be denoted as:

$$s(\mathbf{x}_c,\mathbf{y}_c) = -\sum_{i=1}^{D}\frac{v_i}{1-v_i}(\mathbf{x}_{c,i}-\mathbf{y}_{c,i})^2 + \sum_{i=1}^{D}\frac{v_i}{1+v_i}(\mathbf{x}_{c,i}+\mathbf{y}_{c,i})^2 \quad (18)$$

The block diagram of the similarity score according to (18) is shown in figure 6.

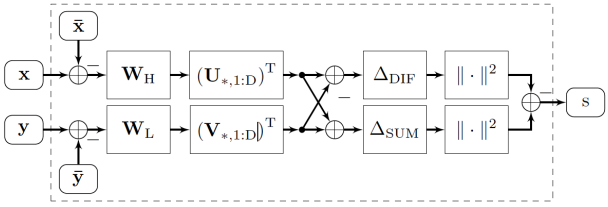


Fig. 6. Block diagram of calculating the similarity [28]

Finally, the complete expression for the log-likelihood ratio, including all the ignored constants is

$$\log(l(\mathbf{x}_c,\mathbf{y}_c)) = -\frac{1}{2}\sum_{i=1}^{D}log(1-v_i^2)+\frac{1}{4}s(\mathbf{x}_c,\mathbf{y}_c) \quad (19)$$

Direct matching [1] is a way to get the similarity score of SML, SMO and SMC. Suppose $R(m,n)$ and $T(m,n)$ are the minutiae spectra of the reference fingerprint and the test fingerprint on the polar-logarithmic(or polar-linear) grid, respectively. Both $R(m,n)$ and $T(m,n)$ are normalized to have zero mean and unit energy. The two-dimensional correlation coefficient (14) between these two spectra can be used as the similarity score.

In practice, rotation, translation and scaling may exist in the original fingerprint images which is caused by the sensor used to acquire the fingerprints. Since the magnitude spectrum is translation invariant, some methods should be used to compensate for the rotation and the scaling. As a magnitude spectrum is mapped onto a polar-logarithmic (or polar-linear) grid, angle is represented by the horizontal axis while scaling is represented by the vertical axis. The scaling becomes the shift in the vertical direction, and the rotation becomes the circular shift in the horizontal direction. Keep the magnitude spectrum of reference image $R(m,n)$ unchanged, and denote

$T(m-i, n-j)$ as a shifted version of $T(m, n)$, with a shift of i in the vertical direction and a circular shift j in the horizontal direction. Then, the correlation coefficient between R and T is defined as:

$$C^{(R,T)}(i,j) = \frac{1}{MN} \sum_{m,n} R(m,n)\, T(m-i, n-j) \quad (20)$$

Most of time, the scaling of the original fingerprint can be compensated for on the level of the minutiae sets [31]. Then, i in (20) can be set to zero. We chose to test rotations from -15 units to +15 units in steps of 3 units, which corresponds to a range from $-10°$ to $+10°$ in steps of $2°$. The maximum score (21) of these correlation coefficient between R and T is used as the similarity score of the reference fingerprint and the test fingerprint.

$$S^{R,T} = max_j \left\{ C^{(R,T)}(0,j) \right\} \quad (21)$$

with $j = 3k$ for $k = -5...5$.

III. The Proposed Approach

A. Problem Statement

In the field of fingerprint recognition, many algorithms based on minutiae comparison [4]–[6] have achieved good performance, and currently the algorithms based on deep learning attract more and more attention. This paper mainly wants to investigate whether the combination of these two algorithms can improve the performance of fingerprint recognition. In this paper, we use complex spectral minutiae representation (SMC) [1] to express the information of the minutiae, and the magnitude spectrum is considered as the input of deep learning for feature extraction. The similarity score of feature vectors extracted by the convolutional neural network is used as the decision criterion of fingerprint recognition. The specific framework is shown in figure 7.

As can be seen from the figure, the method proposed in this paper can be divided into four parts. The first part is minutiae extraction of fingerprint images. In this paper, Mindtct minutiae detector in the open source NIST biometric image software is used to automatically locate and record the minutiae in a fingerprint image. The coordinates and orientations of the minutiae are collected. In the second part, the complex spectral minutiae representation [1] is used to process the extracted coordinates and orientations of minutiae, and then the magnitude spectrum is obtained. The third part is feature vector extraction based on deep learning. The spectrum obtained from the second part is used as the input of the network structure. After the calculation of the deep convolutional neural network, the spectrum is transformed into a 128-dimensional feature vector. The fourth part is the comparison of feature vectors. The classifier is used to compare the two vectors, and then the fingerprint is identified by scoring.

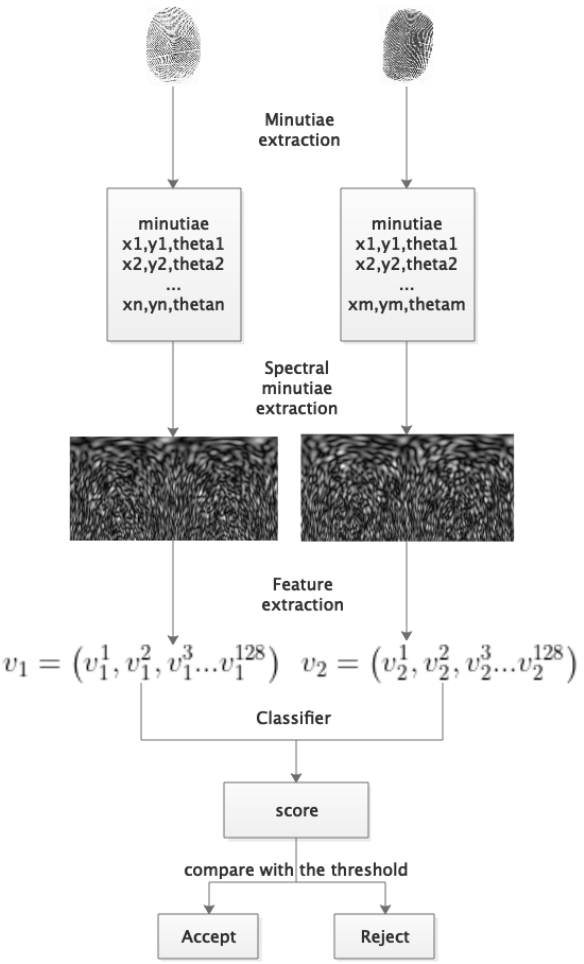


Fig. 7. The framework

B. Complex Spectral Minutiae Representation

The spectral minutiae representation is based on the shift, scale and rotation properties of the two-dimensional continuous Fourier transform. Given an input signal $f(\vec{x}), \vec{x} = (x,y)^T$(superscript T denotes the transpose of a vector), its continuous Fourier transform is

$$F\{f(\vec{x})\} = F(\vec{w}) = \int_{\vec{x} \epsilon R} f(\vec{x})\, exp(-j\vec{w}^T\vec{x}) d\vec{x} \quad (22)$$

with $\vec{w} = (w_x, w_y)^T$. The Fourier transform of a translated $f(\vec{x})$ is

$$F\{f(\vec{x} - \vec{x_0})\} = exp(-j\vec{w}^T\vec{x_0}) F(\vec{w}) \quad (23)$$

in which $\vec{x_0} = (x_0, y_0)^T$ the translation vector. The Fourier transform of an isotropically scaled $f(\vec{x})$ is

$$F\{f(a\vec{x})\} = a^{-2} F(a^{-1}\vec{w}) \quad (24)$$

with $a\,(a > 0)$ the isotropic scaling factor. The Fourier transform of a rotated $f(\vec{x})$ is

$$F\left\{f\left(\Phi\vec{x}\right)\right\} = F\left(\Phi\vec{w}\right) \tag{25}$$

with

$$\Phi = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \tag{26}$$

Here Φ is the (orthonormal) rotation matrix, and ϕ is the (anticlockwise) rotation angle of $f\left(\vec{x}\right)$.

It can be seen from (23) that if only the magnitude of the Fourier spectrum is retained, the representation of the input signal is translation invariant. Furthermore, from (24) and (25), it follows that scaling and rotation of the input signal results in a scaled and rotated Fourier spectrum. However, in most fingerprint databases, there is no scaling difference between the fingerprints, or the scaling can be compensated for on the level of the minutiae sets [31], and some additional operations are performed to reduce the effect of the rotation. Besides, SMC is sampled in a polar-linear grid. Compared with sampling in a polar-logarithmic grid, it provides more information in the higher frequency part.

There are three steps to convert the minutiae of a fingerprint into complex spectral minutiae representation. First of all, for each minutia, it is considered as a Dirac pulse, $m_i(x,y) = \delta(x-x_i, y-y_i), i = 1...Z$, where (x_i, y_i) is the location of the i-th minutia in a fingerprint image. A complex amplitude $e^{j\theta_i}$ represents the minutia orientation. Then the Fourier transform of $m_i(x,y)$ is given by:

$$F\left\{m_i(x,y)\right\} = exp(-j(w_x x_i + w_y y_i) + j\theta_i) \tag{27}$$

Secondly, the complex spectral minutiae representation is:

$$M_C(w_x, w_y) = \sum_{i=1}^{Z} exp(-j(w_x x_i + w_y y_i) + j\theta_i) \tag{28}$$

Finally, the continuous spectra SMC needs to be sampled on a polar-linear grid. A polar mapping transforms rotation to circular shift in the horizontal direction. In the radial direction λ, we use M = 128 samples between $\lambda_1 = 0.05$ and $\lambda_h = 0.58$. In the angular direction β, we use N = 256 samples uniformly distributed between β = 0 and $\beta = 2\pi$.

Examples of the minutiae spectra achieved with SMC are shown in figure 8. For each spectrum, the horizontal axis represents the angle of the spectral magnitude (from 0 to 2π); the vertical axis represents the frequency of the spectral magnitude (the frequency increases from top to bottom).

C. Network

The deep convolutional neural network takes the magnitude spectra as the input to extract the features from the spectra and then get feature vectors. The network structure used in this paper is darknet-19 [15], which is composed of 19 convolutional blocks and one fully connected layer. For the dataset used in this paper, the 19-layer convolutional structure is appropriate. Because the training dataset is relatively small, the 19-layer convolutional structure is neither too complex, resulting in a quick overfitting, nor too simple, making it difficult for the network to extract the features from the magnitude spectra.

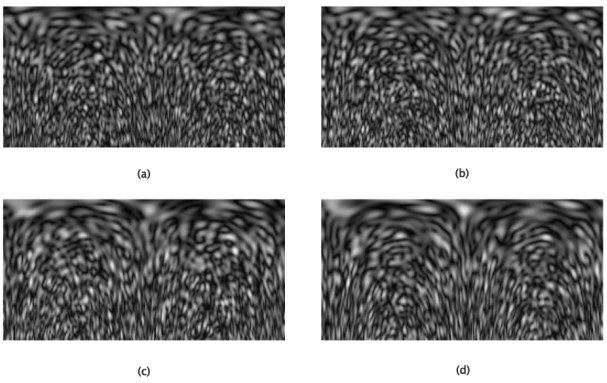


Fig. 8. The spectra of SMC. (a) and (b) are the SMC spectra from different fingerprint images of the same identity; (c) and (d) are the SMC spectra from different fingerprint images of the same identity.

As mentioned in Sec.II the essence of convolution is linear computation. In order to increase the nonlinear ability of feature extraction and increase the diversity of expression of the features, a layer of activation function is often added behind the convolutional computation. The activation function used in this paper is the Parametric Rectified Linear Unit (PReLU) function [30]. PReLU has made some improvements to Rectified Linear Unit (ReLU) [29]. In ReLU, when the input is less than zero, the output values are all set to be zero, which will cause some data missing. PReLU adds a variable slope at the negative half axis to ensure the integrity of the data. The functions of ReLU and PReLU are shown in figure 9.

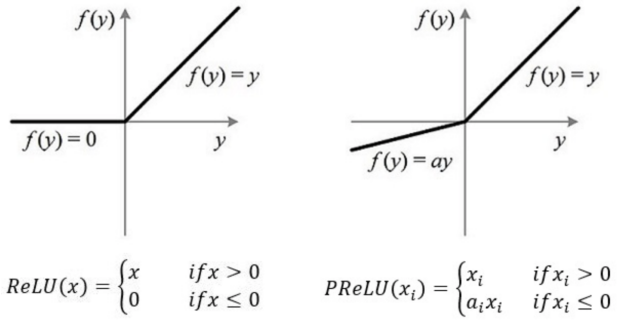


Fig. 9. The contrast of ReLU and PReLU

where a is the varialbe slope, which will be updated during the training process. After the covolutional operation, the values of the data will be changed, and the distribution of the outputs are different with the inputs, so the training convergence is slow. Generally, the values would be very large or very small, which results in the convergence of deep neural network slower and slower, and batch normalization is used to force the distribution of input values of any layer back to

the standard normal distribution with mean value of zero and variance of one. In other words, by using batch normalization, the increasingly biased distribution is forced to be pulled back to the standard distribution. The training convergence speed will be faster, and the problem of gradient vanishing or explosion can be solved at the same time.

A convolutional block (figure 10) is composed of one convolutional layer, one activation operation and one batch normalization operation. The convolutional block is taken as the basic unit of the network structure.

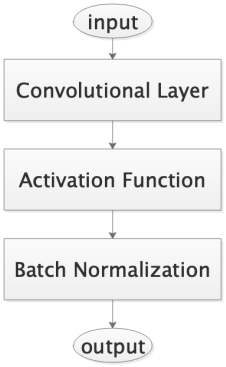


Fig. 10. The convolutional block

In addition to the convolutional blocks, there are some max pooling layers in the network, which can reduce the size of the feature maps, improve the computing speed, and improve the robustness of the extracted features. In this paper, a 2×2 max pooling kernel is used, that is, only the maximum value of 4 pixels is kept. The process can be shown in figure 11.

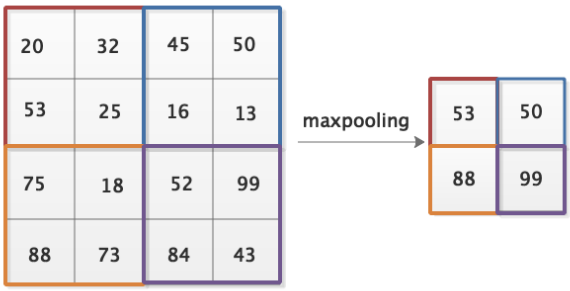


Fig. 11. The process of maxpooling

After all the convolutional blocks, there is a fully connected layer to convert the (channel, height, width) format into a 128 dimensional vector. The overall structure of the network can be shown in figure 12 and figure 13.

D. Loss Function

Triplet loss was first proposed in Facenet [16]. In this paper, triplet loss is used to train the embedding of the magnitude spectra. In a well-trained embedding space, similar images have similar results, therefore, the embedding space can be used to estimate whether the magnitude spectra belong to the same identity. In order to get a standard embedding space, the main goal of triplet loss is to: a). make the distance of the samples with the same identity as close as possible in the embedding space, and b). make the distance of the samples with different identities as far as possible in the embedding space. In order to achieve b), an extra margin is needed. So, the basic idea of triplet loss is to make the sum of the margin

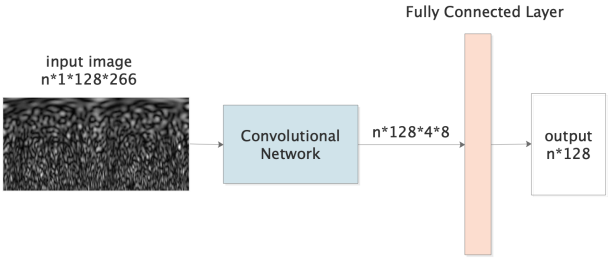


Fig. 12. The overall structure of the network

Convolutional Netwok

Type	Filters	Size/Stride	Output(channel*h*w)
Convolutional	32	3*3	32*128*256
Maxpooling		2*2/2	32*64*128
Convolutional	64	3*3	64*64*128
Maxpooling		2*2/2	64*32*64
Convolutional	128	3*3	128*32*64
Convolutional	64	1*1	64*32*64
Convolutional	128	3*3	128*32*64
Maxpooling		2*2/2	128*16*32
Convolutional	256	3*3	256*16*32
Convolutional	128	1*1	128*16*32
Convolutional	256	3*3	256*16*32
Maxpooling		2*2/2	256*8*16
Convolutional	512	3*3	512*8*16
Convolutional	256	1*1	256*8*16
Convolutional	512	3*3	512*8*16
Convolutional	256	1*1	256*8*16
Convolutional	512	3*3	512*8*16
Maxpooling		2*2/2	512*4*8
Convolutional	1024	3*3	1024*4*8
Convolutional	512	1*1	512*4*8
Convolutional	128	3*3	128*4*8
Convolutional	64	1*1	64*4*8
Convolutional	128	3*3	128*4*8

Fully Connected Layer

Input_features	Output_features
4096	128

Fig. 13. The structure of convolutional network and fully connected layer

and the distance between the samples of the same identity less than the distance between the samples of different identities.

A triple (A, P, N) consists of a randomly selected feature vector A (Anchor), another feature vector P (Positive) sharing the same identity as A, and a feature vector N (Negative) having an identity unequal to A and P. The distance of each two vectors can be define as:

$$d(a,b) = \left\| f(x_i^a) - f(x_i^b) \right\|_2^2 \tag{29}$$

The triplet loss can be defined as:

$$L = max(d(A,P) - d(A,N) + margin, 0) \tag{30}$$

In the training process, the loss should be minimized, and after optimizing, the distance between the anchor A and the positive P is close to zero and the distance between the anchor A and the negative N is close to margin. The process is shown in figure 14.

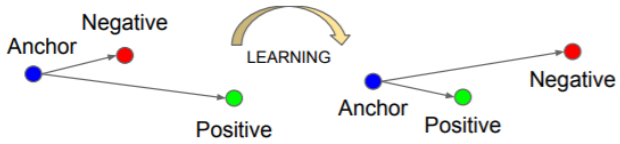


Fig. 14. The training process of triplet loss [16]

The triples can be divided into the following three categories:

Easy triplet : for some (A, P, N), the sum of the margin and the distance between Anchor and Positive always less than the distance of the Anchor and the Negative. These triples do not need to be used during training.

Semi-hard triplet: the distance between the Anchor and the Positive is less than the distance between the Anchor and the Negative, and at the same time, the distance between the Anchor and the Negative is less than the sum of the margin and the distance of Anchor and Positive. It means sometimes the fingerprints can be classified, but the distance between the feature vectors with the same identity is not small enough or the distance between the feature vectors with different identities is not big enough .

Hard triplet : the distance of the Anchor and the Negative is less than the distance of the Anchor and the Positive. For these triples, they should be trained more.

In order to create a more discriminative embedding space, hard triplets and some semi-hard triplets are used during training stage. There are two methods to train the network. One is called online training and the other one is called offline training.

Offline training : Consider every feature vector in the dataset as an Anchor and find all easy triplets, semi-hard triplets and hard triplets. Only hard triplets are used for training. As the parameters of the network are updated over time, the feature vectors of the fingerprints will change, and the distances between the different vectors will also change. Therefore, all triplets need to be reclassified, and hard triplets need to be updated after a few epochs of training. In order to be able to train continuously and get the appropriate parameters, the other training method: online training is used in this paper.

Online training is based on batch training. In the process of training, consider each sample of the feature vectors in a batch as an anchor, and then find its hardest positive (the positive sample with the furthest distance from the anchor) and hardest negative(the negative sample with the nearest distance from the anchor) to form a triplet, and use all the triplets to calculate the loss.

The distance between the anchor and its hardest positive and the distance between the anchor and its hardest negative can be found according to the matrix of the batch. These distances can be calculated as follows: Given a batch of feature vectors with batch size n, the batch can be expressed as,

$$\begin{bmatrix} \vec{v_1} \\ \vec{v_2} \\ \dots \\ \vec{v_n} \end{bmatrix} \tag{31}$$

the square of the vector is taken and expand it into an $n \times n$ matrix,

$$\begin{bmatrix} |\vec{v_1}|^2 & |\vec{v_1}|^2 & \dots & |\vec{v_1}|^2 \\ |\vec{v_2}|^2 & |\vec{v_2}|^2 & \dots & |\vec{v_2}|^2 \\ \dots & \dots & \dots & \dots \\ |\vec{v_n}|^2 & |\vec{v_n}|^2 & \dots & |\vec{v_n}|^2 \end{bmatrix} \tag{32}$$

,

then get the sum of the matrix and its transpose matrix, and subtract two times of the dot product of the vectors(31) and the vectors' transpose, then the matrix is,

$$\begin{bmatrix} |\vec{v_1}|^2 - 2v_1 \cdot v_1^T + \left|\vec{v_1}^T\right|^2 & \dots & |\vec{v_1}|^2 - 2v_1 \cdot v_n^T + \left|\vec{v_n}^T\right|^2 \\ |\vec{v_2}|^2 - 2v_2 \cdot v_1^T + \left|\vec{v_1}^T\right|^2 & \dots & |\vec{v_2}|^2 - 2v_2 \cdot v_n^T + \left|\vec{v_n}^T\right|^2 \\ \dots & \dots & \dots \\ |\vec{v_n}|^2 - 2v_n \cdot v_1^T + \left|\vec{v_1}^T\right|^2 & \dots & |\vec{v_n}|^2 - 2v_n \cdot v_n^T + \left|\vec{v_n}^T\right|^2 \end{bmatrix} \tag{33}$$

,

it can be found that each entry of the matrix is the square of the distance of two vectors. Take the square root of each entry, and the distance can be obtained(34).

$$\begin{bmatrix} \sqrt{|\vec{v_1} - \vec{v_1}|^2} & \sqrt{|\vec{v_1} - \vec{v_2}|^2} & \dots & \sqrt{|\vec{v_1} - \vec{v_n}|^2} \\ \sqrt{|\vec{v_2} - \vec{v_1}|^2} & \sqrt{|\vec{v_2} - \vec{v_2}|^2} & \dots & \sqrt{|\vec{v_2} - \vec{v_n}|^2} \\ \dots & \dots & \dots & \dots \\ \sqrt{|\vec{v_n} - \vec{v_1}|^2} & \sqrt{|\vec{v_n} - \vec{v_2}|^2} & \dots & \sqrt{|\vec{v_n} - \vec{v_n}|^2} \end{bmatrix} \tag{34}$$

For each row in this matrix(34), find all positive samples and all negative samples of the corresponding vector, and then find the hardest positive from the positive samples and find the hardest negative from the negative samples. Therefore, the triplet becomes (Anchor, Hardest Positive, Hardest Negative).

Besides, the distance can be selected from the matrix (34) directly. Using (30) for a single triplet, we define the batch loss is

$$L = \sum_{i=0}^{n} max\left(d\left(Anchor_i, HardestPostive_i\right) - d\left(Anchor_i, HardestNegative\right) + margin, 0\right) \tag{35}$$

E. Classifier

Cosine similarity, also known as cosine distance, is a measure of the difference between two individuals by using the cosine value of the angle between two vectors in vector space. Cosine similarity is suitable for the similarity calculation of high dimensional vectors.

Given two vectors, (36) can be used to define the cosine similarity (37). The value of the cosine similarity is between -1 and 1. -1 is the most dissimilar case ("π" radians), 1 is the most similar case ($\vec{a} = \lambda\vec{b}$, with $\lambda > 0$).

$$\vec{a} \cdot \vec{b} = \|a\| \, \|b\| \cos\theta \tag{36}$$

$$\begin{aligned} similarity = \cos\theta &= \frac{a \cdot b}{\|a\| \, \|b\|} \\ &= \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} (a_i)^2} \sqrt{\sum_{i=1}^{n} (b_i)^2}} \end{aligned} \tag{37}$$

In our experiment, the cosine similarity of two fingerprint images represented by two feature vectors are calculated, and then the value of cosine similarity is used to achieve fingerprint recognition.

If the cosine similarity of two feature vectors is greater than or equal to a certain threshold, it can be considered that the fingerprint images represented by the two feature vectors belong to the same identity, if the cosine similarity of the feature vectors is less than the threshold value, the fingerprint images represented by the two feature vectors belong to different identities.

IV. Experiments

In this section, we will introduce the dataset used in the experiments, the process of four experiments and the results of these experiments.

A. Dataset

MCYT fingerprint database [19] is proposed and implemented by the Biometric Research Laboratory - ATVS of the Universidad Politecnica de Madrid. Two different devices: Digital Persona UareU (256x400 8-bit 500 dpi) and PRECISE BIOMETRICS SC-100 (300x300 8-bit 500 dpi) were used to collect fingerprints. 434 subjects from four different schools participated in the fingerprint collection. For each device and each subject, ten fingers were used. Consider each finger as an identity, for each identity, twelve samples were collected. All fingerprint images are grayscale images, and the fingerprints can be displayed in figure 15 and 16.

Fig. 15. The fingerprint images collected by Digital Persona UareU

Fig. 16. The fingerprint images collected by Precise Biometrics SC-100

The DB2 of FVC2002 [20] and FVC200 are also used in the experiment. These fingerprints are collected by the optical sensor "U.are.U 4000" by Digital Persona. For both FVC2002 and FVC200, each dataset contains 100 identities and for each identity, there are eight samples.

According to the above introduction to the two databases, there are about 104160 images in MCYT database and there are about 1600 images in the two FVC databases. In order to ensure the balance between the two databases, we take all FVC fingerprint images and a part of MCYT fingerprint images as our dataset. It can be found that in figure 15 and figure 16, thumb images always have different shape with others, so we drop all the thumb images in our dataset. The dataset is divided into training dataset, validation dataset and test dataset. In the training set, there are 3360 images from MCYT and FVC. 2040 images are from MCYT database (170 identities, and 12 samples for each identity), while 1320 are from FVC databases (165 identities, and 8 samples for each identity). In the validation dataset, there are 200 images. 120 images are from MCYT database (10 identities, and 12 samples for each identity), while 80 images are from FVC databases (10 identities, and 8 samples for each identity). In the testing dataset, there are 500 images. 300 images are from MCYT database (25 identities, and 12 samples for each identity), while 200 images are from FVC databases (25 identities, and 8 samples for each identity). In order to compensate for the

bias caused by the rotation of the fingerprints in the process of SMC, some rotated fingerprints are added into the training set. Two samples for each identity in the training dataset are selected and then rotated with a small and randomly-selected angle (from $-15°$ to $15°$) to obtain two new images. Therefore, there are 4030 images in total for training.

B. Experiment Setting

We use direct matching of SMC as our baseline experiment to compare with the methods we proposed in this paper.

1) Baseline experiment: For the direct matching of SMC, there is no training process, so only testing set is used. Extract the minutiae sets of these fingerprint images, and convert these minutiae sets into magnitude spectra. For each two magnitude spectra, the similarity score is calculated according to (20) and (21). Set a threshold, if the similarity score is greater than the threshold, the two fingerprints belong to the same identity. If the similarity score is less than the threshold, the two fingerprints belong to different identities.

The methods we proposed in this paper consists of four parts: (1) minutiae extraction; (2) complex spectral minutiae representation; (3) feature extraction by using deep learning; (4) feature vector comparison by using classifier.

In the process of training, there are only the first three steps and no process of comparison. In the first step, 3360 fingerprint images can be considered as the input of minutiae extraction, then the output of this part are 3360 minutiae sets, in each minutiae set, there is the information of the minutiae of the fingerprint, which contains the coordinates and the orientations. Before the step of complex spectral minutiae representation, two samples of each identity are randomly selected, and then the minutiae sets of these two samples are rotated with a very small angle. The rotated minutiae sets are also added to the dataset for training. Therefore, there are 4030 minutiae sets in total as the input of complex spectral minutiae representation. The rotation is calculated as follows (38).

$$\begin{cases} x = x'cos(\theta) \\ y = y'cos(\theta) \\ \theta = \theta' + \alpha \\ -\frac{\pi}{12} < \alpha < \frac{\pi}{12} \end{cases} \tag{38}$$

In the second step, each minutiae set is converted into a fixed-size magnitude spectrum by complex spectral minutiae representation. For validation dataset, we use the raw fingerprint images. Extract the minutiae sets and convert them into magnitude spectra. We train the network by using training dataset and observe the loss curve (figure 17) of the training dataset and the validation dataset. It can be found that the loss curve of the validation dataset reached its lowest point at around 30th epoch. Therefore, the network parameters of 30th epoch training were selected as the final parameters in our experiments.

When testing, consider the two samples for recognition as reference fingerprint and test fingerprint. The reference fingerprint is used as a baseline, and the test fingerprint is used to compare with the reference fingerprint. During the testing process, we conducted three experiments to compare the results of these three experiments to see if they could improve the result of baseline experiment.

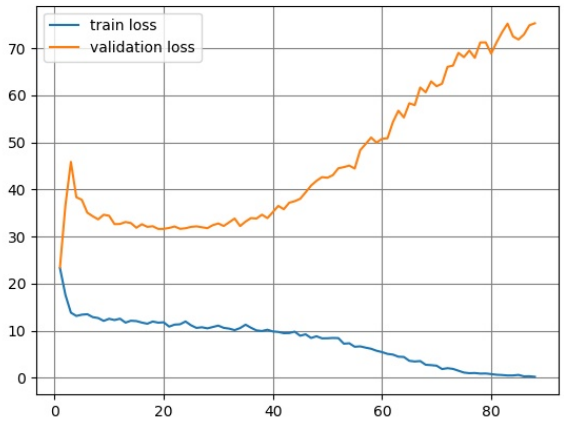


Fig. 17. The loss graph of the train set and the validation set after 88 epochs training

2) Experiment I: The testing process is as follows: first of all, extract the minutiae of the reference fingerprint and the test fingerprint. Secondly, the minutiae sets of the two fingerprints will be represented by the spectra. And thirdly, the convolutional neural network is used to extract the feature vectors of the reference magnitude spectrum and the test magnitude spectrum. Finally, these two vectors are compared by using cosine similarity and then a score is obtained. This score is used to compare with the threshold, and it can be determined that whether the reference fingerprint and the test fingerprint belong to the same identity.

However, in the process of fingerprints collection, the rotation of the fingerprints still exist (for example, depending on the sensor that is used to acquire an image). Additional measures aiming to compensate for the rotation can be taken to check whether it is useful of improving the performance of the fingerprint recognition system. Then we have Experiment II.

3) Experiment II: When this rotation is applied to the testing process, it can be described as follows: Step 1: extract the minutiae sets from the original fingerprint images of reference fingerprint and test fingerprint; Step 2: these two minutiae sets are represented by two magnitude spectra by using complex spectral minutiae representation. Keep the reference spectrum unchanged, rotate the test spectrum for several times. As the minutiae set is remapped to the polar-linear grid after Fourier transform, the rotation of the fingerprint image becomes circular shift in the horizontal direction on the magnitude spectrum. If $T(m,n)$ is used to represent the magnitude spectrum, $T(m,n-j)$ can be used to represent the the circular shift of the spectrum, where m and n are the coordinates of the spectral representation and j $(-15 < j < 15, step = 3)$ is the shift. This circular shift is the same as the rotation of the

original fingerprint image with the angle from -10^o to $+10^o$ in steps of 2^o. After the circular shift, we have 11 test spectra to compare with the reference spectrum; Step 3: the reference spectrum and the 11 test spectra are forward to the network as inputs, then the feature vector of the reference spectrum and the feature vectors of the test spectra can be obtained; Step 4: the feature vector of the reference spectrum is compared with the feature vectors of the test spectra, and 11 scores are got by using cosine similarity. The highest score will be selected as the final score of this testing process. This score will also be used to compare with the threshold we set.

4) Experiment III: Since the log likelihood ratio has achieved good performance in many fields such as face recognition, we also apply the log likelihood ratio as the classifier to recognize the two vectors which are obtained from the convolutional neural network. According to the log likelihood ratio, the classifier needs to be trained first to get some parameters(described in Sec.2), then it can be used for feature vector comparison.

There are still four steps for fingerprint recognition. Firstly, extract the minutiae sets of the reference fingerprint and the test fingerprint; Secondly, the minutiae sets are converted to two spectra by using SMC; Thirdly, two feature vectors are got from the convolutional neural network; Finally, in order to use the log likelihood ratio as classifier, these two vectors should be concatenated. For example, two vectors of size 128×1 are concatenated to a vector of size 256×1. Then the similarity score can be calculated by LLR. A threshold is also used to compare with the similarity score.

C. Result

The performance of a fingerprint recognition system can be evaluated by means of several measures. The receiver operating characteristics (ROC) curve and the equal error rate (EER) are used in this paper. Some concepts about these measures are described as below.

During the testing process, four situations will occur when achieve fingerprint recognition, and it can be shown in the confusion table (tableI).

TABLE I
CONFUSION TABLE

Actual	Prediction		
	Positive	Negative	Total
Positive	True Positive	False Negative	Actual Positive
Negative	False Positive	True Negative	Actual Negative
Total	Predicted Positive	Predicted Negative	

Compare the reference fingerprint and test fingerprint, four kinds of results will occur.

• The reference and the test are with the same identity. After using the fingerprint recognition system, it is predicted that these two fingerprints are from the same identity. It means that the system predicts correctly. This case can be represented by True Positive in table I.

• The reference and the test are with the same identity. After using the fingerprint recognition system, it is predicted that these two fingerprints are from different identities. It means that the system makes a wrong prediction. This case can be represented by False Negative in table I.

• The reference and the test are with different identities. After using the fingerprint recognition system, it is predicted that these two fingerprints are from the same identity. It means that the system makes a wrong prediction. This case can be represented by False Positive in table I.

• The reference and the test are with different identities. After using the fingerprint recognition system, it is predicted that these two fingerprints are from different identities. It means that the prediction of the system is correct. This case can be represented by True Negative in table I.

When the reference and the test are from the same identity, this kind of pairs can be represented by Actual Positive in table I, and they are also called genuine pairs. When the reference and the test are from different identities, this kind of pairs can be represented by Actual Negative in table I, and they are also called impostor pairs.

True match rate (TMR) [18] is the ratio of the number of True Positive cases and the number of Actual Positive cases. It can be represented by (39).

$$\mathrm{TMR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = \frac{\mathrm{Genuine}(\mathrm{score} \geq \mathrm{threshold})}{\mathrm{Genuine\ Pairs}} \tag{39}$$

False match rate (FMR) [18] is the ratio of the number of False Positive cases and the number of Actual Negative cases. It can be represented by (40).

$$\mathrm{FMR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} = \frac{\mathrm{Impostor}(\mathrm{score} \geq \mathrm{threshold})}{\mathrm{Impostor\ Pairs}} \tag{40}$$

False non-match rate (FNMR) is the ratio of the number of false negative cases and the number of actual positive cases. It can be represented by (41), which is equal to $1 - TPR$.

$$\mathrm{FNMR} = \frac{\mathrm{FN}}{\mathrm{TP} + \mathrm{FN}} = \frac{\mathrm{Genuine}(\mathrm{score} < \mathrm{threshold})}{\mathrm{Genuine\ Pairs}} \tag{41}$$

The ROC curve is drawn with FMR as the horizontal axis and TMR as the vertical axis. The ROC curve is threshold independent and it presents the performance of the fingerprint recognition system under different threshold settings. In the process of fingerprint recognition to determine whether two fingerprints belong to the same identity, different thresholds are set. Each threshold corresponds to a TMR and a FMR, that is, a corresponding point on the ROC curve. The greater the TMR and the smaller the FMR, the better the performance of the recognition system. Therefore, the closer the ROC curve is to the point (0,1) in the figure, the better the performance of the fingerprint recognition system will be.

Equal Error Rate is the common value of FMR and FNMR when the FMR and FNMR are equal. The smaller the EER, the better the performance of fingerprint recognition system.

In order to know whether adding deep learning to the complex spectral minutiae representation can improve the performance of fingerprint recognition system, we will compare the ROC curve and the EER of the three experiments mentioned above and Baseline Experiment.

In Experiment I, cosine similarity is used as the classifier, and no rotation process is used during testing. The ROC curve of Baseline Experiment and Experiment I is shown in figure 18.

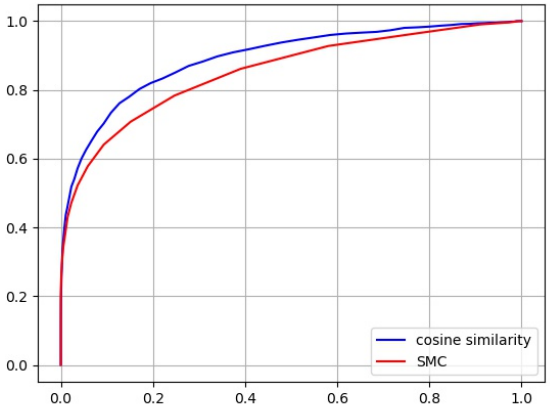


Fig. 18. The red line is the ROC curve of Baseline Experiment; the blue line is the ROC curve of experiment I

In figure 18, it can be found that compared with Baseline Experiment, Experiment I has some improvement.

In Experiment II, circular shift is applied to the complex spectral minutiae representations before the feature extraction of convolutional neural network. Then the ROC curve of Baseline Experiment, Experiment I and II is shown in figure 19.

In figure 19, it can be found that the performance of Experiment II is still better than the performance of Baseline Experiment, but worse than Experiment I. According to the ROC curve of Experiment I and Experiment II, we can derive that when the similarity scores of the reference fingerprint and the test fingerprints with circular shift on are calculated, and the maximum score is kept, the influence on the impostor pairs is greater than the influence on the genuine pairs. The increase of the number of the False Positive cases predicted by the fingerprint recognition system is more than the increase of the number of the True positive cases. In our case, circular shift on the complex spectral minutiae representations does not help to improve the performance.

In Experiment III, log likelihood ratio is used as the classifier during the testing process. The comparison of the ROC curves of the four experiments is shown in figure 20.

It can be found in figure 20, Experiment I still has the

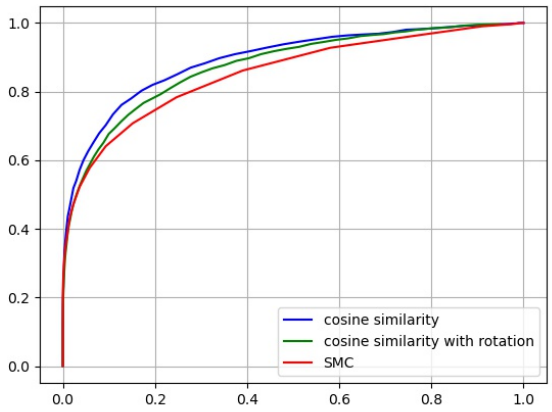


Fig. 19. The red line is the ROC curve of Baseline Experiment; the blue line is the ROC curve of Experiment I; the green line is the ROC curve of Experiment II

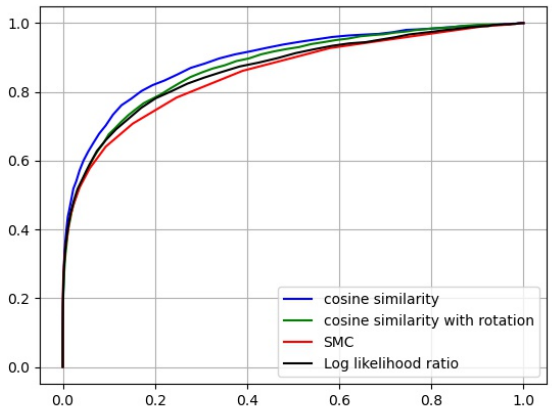


Fig. 20. The red line is the ROC curve of Baseline Experiment; the blue line is the ROC curve of Experiment I; the green line is the ROC curve of Experiment II; the black line is the ROC curve of Experiment III

best performance among these four experiments. When FMR is smaller than 0.2, the performance of Experiment II and Experiment III are almost the same, but when FMR is greater than 0.2, Experiment II performs better than Experiment III.

Equal error rates of the four experiments are also compared, the values can be shown in table II.

TABLE II
THE COMPARISON OF EQUAL ERROR RATE

Experiment	Equal Error Rate
Baseline Experiment	0.22999626
Experiment I	0.18640349
Experiment II	0.20953675
Experiment III	0.21226133

The comparison result of EER is almost the same as the comparison result of ROC curve. Experiment I has the

best result. The performance of Experiment II is better than the performance of Experiment III. All the results of the experiments we proposed in this paper is better than the result of Baseline experiment.

V. Conclusion

Spectral minutiae representation solves some problems of minutiae-based fingerprint recognition such as the difficulty of comparing different numbers and the dislocation of the minutiae. The process of fingerprint recognition using spectral minutiae representation is to calculate the correlation coefficient between the spectra of the reference fingerprint and the test fingerprint as the similarity score. In this paper, the spectral minutiae representation is combined with deep learning, the convolutional neural network is used to extract feature vectors from the spectral minutiae representations, and then the feature vectors can be used to represent the original fingerprints. Therefore, the pointwise comparison of the spectra is converted to the similarity calculation of two vectors. In this paper, three experiments are implemented to observe whether the combination of the spectral minutiae representation and deep learning can improve the performance from the direct matching of spectral minutiae representation. In Experiment I, cosine similarity is used as the classifier to calculate the similarity score of two vectors which represent the reference fingerprint and the test fingerprint. In Experiment II, circular shift is used on the test spectrum to compensate for the rotation of the fingerprint image. Then cosine similarity is still used to calculate the similarity scores, and the maximum score is selected as the final score of the two fingerprints. In Experiment III, log likelihood ratio is used as classifier to compare two vectors of the reference fingerprint and the test fingerprint, and then similarity score is obtained. The ROC curve and EER are used to evaluate the three experiments and the direct matching of SMC. Experiment I shows the best performance. Experiment II performs better than Experiment III. All these experiments perform better then direct matching of SMC.

Even though the convolutional neural network transforms a magnitude spectrum with size 128×256 into a feature vector with size 1×128, like feature reduction, it extracts the global features of the spectrum and performs better than the pointwise comparison. Further research can be carried out in the direction of deep learning and log likelihood ratio classifier. On the one hand, more complex network structure such as Resnet [32] can be tried; more loss functions and optimizers can be tried to train the network; the dimension of the feature vectors can be increased, on the other hand, more data can be used to train the LLR classifier and get the more appropriate parameters for testing. These methods can be used to find whether the performance will be improved.

References

[1] H. Xu and R. N. J. Veldhuis, *Complex spectral minutiae representation for fingerprint recognition*, In Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on, pp. 1–8, 13-18 2010.

[2] H. Xu and R. N. J. Veldhuis, *Spectral minutiae representations for fingerprint recognition*, 2010.

[3] H. Xu, R. Veldhuis, A. Bazen, T. Kevenaar, T. Akkermans, and B. Gokberk, *Fingerprint verification using spectral minutiae representations*, Information Forensics and Security, IEEE Transactions on, vol. 4, pp. 397–409, Sept. 2009.

[4] H. Xu, A. M. Bazen, R. N. J. Veldhuis, T. A. M. Kevenaar, and A. H. M. Akkermans, *Spectral representation of fingerprints.*, In Proceedings of the 28th Symposium on Information Theory in the Benelux, Enschede, The Netherlands, pages 313–319, Eindhoven, June 2007. Werkgemeenschap voor Informatie- en Communicatietechniek.

[5] H. Xu, R. N. J. Veldhuis, T. A. M. Kevenaar, A. H. M. Akkermans, and A. M. Bazen, *Spectral Minutiae: A Fixed-length Representation of a Minutiae Set*, In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshop on Biometrics, Anchorage, USA, 2008.

[6] H. Xu, R. N. J. Veldhuis, A. M. Bazen, T. A. M. Kevenaar, A. H. M. Akkermans, and B. Gokberk, *Fingerprint verification using spectral minutiae representations*, Information Forensics and Security, IEEE Transactions on, 4(3):397–409, Sept.2009.

[7] H. Xu and R. N. J. Veldhuis, *Spectral representations of fingerprint minutiae subsets*, In Image and Signal Processing, 2009. CISP '09. 2nd International Congress on, pages 1–5, Oct. 2009.

[8] "Hacker steals fingerprint from photo, suggests politicians wear gloves in public." *https://www.rt.com/news/218587-hacker-fingerprint-minister-photo/*

[9] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar, *Handbook of Fingerprint Recognition*, Springer, 2nd ed., 2009. 1

[10] A. Jain, L. Hong, and R. Bolle, *On-line fingerprint verification*, IEEE Trans. PAMI, vol. 19, pp. 302–314, Apr. 1997.

[11] A. Bazen and S. Gerez, *Fingerprint matching by thin-plate spline modelling of elastic deformations*, Pattern Recognition, vol. 36, pp. 1859–1867, Aug. 2003.

[12] Tang Y , Gao F , Feng J , et al, *FingerNet: An Unified Deep Network for Fingerprint Minutiae Extraction[J]*, 2017.

[13] W. F. Leung, S. H. Leung, W. H. Lau and A. Luk, *Fingerprint recognition using neural network*, Neural Networks for Signal Processing Proceedings of the 1991 IEEE Workshop, Princeton, NJ, USA, 1991, pp. 226-235, doi: 10.1109/NNSP.1991.239519.

[14] Shervin Minaee and Elham Azimi and Amirali Abdolrashidi, *FingerNet: Pushing The Limits of Fingerprint Recognition Using Convolutional Neural Network*,2019.

[15] Redmon, Joseph, et al, *You only look once: Unified, real-time object detection*, Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[16] F. Schroff, D. Kalenichenko, and J. Philbin, *FaceNet: A Unified Embedding for Face Recognition and Clustering*, In CVPR, 2015.

[17] A. Bazen, R. Veldhuis, and S. Gerez, *Hybrid fingerprint matching using minutiae and shape*, in Computer Aided Intelligent Recognition Techniques and Applications, John Wiley, 2004.

[18] R. Bolle, J. H. Connell, S. Pankanti, N. K. Ratha, and A. W. Senior, *Guide to Biometrics. Springer Verlag*, 2003.

[19] Ortega-Garca, J., et al., *MCYT baseline corpus: a bimodal biometric database*, in IEE Proc. Vision, Image and Signal Processing 150(6), pp. 395–401, 2003.

[20] D. Maio, D. Maltoni, R. Cappelli, J. Wayman, and A. Jain, *FVC2002: Second fingerprint verification competition*, vol. 3, pp. 811–814, Aug. 2002.

[21] Y. LeCun , B. Boser , J. S. Denker , D. Henderson , R. E. Howard , W. Hubbard and L. D. Jackel, *Backpropagation applied to handwritten zip code recognition*, Neural Computation, vol. 1, no. 4, pp. 541-551, 1989.

[22] Lecun Y , Bottou L , Bengio Y , et al. *Gradient-based learning applied to document recognition[J]*, Proceedings of the IEEE, 1998, 86(11):2278-2324.

[23] Alex Krizhevsky,Ilya Sutskever,Geoffrey E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks*, 2012.

[24] K. Simonyan and A. Zisserman, *Very deep convolutional networks for large-scale image recognition*, in International Conference on Learning Representations, May 2015.

[25] S. Chopra, R. Hadsell, and Y. LeCun, *Learning a similarity metric discriminatively, with application to face verification*, In Computer

Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, volume 1, pages 539–546. IEEE, 2005.
[26] Wen Y, Zhang K, Li Z, et al.*A discriminative feature learning approach for deep face recognition[C]*, European Conference on Computer Vision. Springer, Cham, 2016: 499-515.MLA.
[27] Deng, J., Guo, J., Zafeiriou, S, *Arcface: Additive angular margin loss for deep face recognition*, arXiv preprint arXiv:1801.07698 (2018)
[28] Peng, Y., Spreeuwers, L., Veldhuis, R, *Lowresolution face alignment and recognition using mixed-resolution classifiers*, IET Biometrics 6(6) (2017) 418–428.
[29] Xavier Glorot, Antoine Bordes, Yoshua Bengio, *Deep Sparse Rectifier Neural Networks*
[30] Maas, A.L., Hannun, A.Y. and Ng, A.Y., *Rectifier Nonlinearities Improve Neural Network Acoustic Models*, Proceedings of the 30th International Conference on Machine Learning, Vol. 28, 3.
[31] *ISO/IEC 19794-2, Information Technology - Biometric Data Interchange Format - Part 2: Finger Minutiae Data*, 2005.
[32] K.He, X.Zhang, S.Ren, and J.Sun, *Deep residual learning for image recognition*, In CVPR 2016.