A Multichannel Approach to Fingerprint Classification

Anil K. Jain, *Fellow*, *IEEE*, Salil Prabhakar, *Student Member*, *IEEE*, and Lin Hong

Abstract—Fingerprint classification provides an important indexing mechanism in a fingerprint database. An accurate and consistent classification can greatly reduce fingerprint matching time for a large database. We present a fingerprint classification algorithm which is able to achieve an accuracy better than previously reported in the literature. We classify fingerprints into five categories: whorl, right loop, left loop, arch, and tented arch. The algorithm uses a novel representation (FingerCode) and is based on a two-stage classifier to make a classification. It has been tested on 4,000 images in the NIST-4 database. For the five-class problem, a classification accuracy of 90 percent is achieved (with a 1.8 percent rejection during the feature extraction phase). For the four-class problem (arch and tented arch combined into one class), we are able to achieve a classification accuracy of 94.8 percent (with 1.8 percent rejection). By incorporating a reject option at the classifier, the classification accuracy can be increased to 96 percent for the five-class classification task, and to 97.8 percent for the four-class classification task after a total of 32.5 percent of the images are rejected.

Index Terms—Biometrics, fingerprint classification, Gabor filters, neural networks, FingerCode.

1 Introduction

Fingerprints are the ridge and furrow patterns on the tip of the finger [1] and are used for personal identification of people [2]. Fig. 1 shows some examples of fingerprints. Large volumes of fingerprints are collected and stored everyday in a wide range of applications, including forensics, access control, and driver license registration. An automatic recognition of people based on fingerprints requires that the input fingerprint be matched with a large number of fingerprints in a database (the FBI database contains more than 70 million fingerprints!). To reduce the search time and computational complexity, it is desirable to classify these fingerprints in an accurate and consistent manner such that the input fingerprint needs to be matched only with a subset of the fingerprints in the database. Fingerprint classification is a technique used to assign a fingerprint into one of the several prespecified types already established in the literature which can provide an indexing mechanism. Fingerprint classification can be viewed as a coarse level matching of the fingerprints. An input fingerprint is first matched at a coarse level to one of the prespecified types and then, at a finer level, it is compared to a subset of the database corresponding to that fingerprint type. In this study, we classify fingerprints into five classes, namely, *whorl* (*W*), *right loop* (*R*), *left loop* (*L*), *arch* (*A*), and *tented arch* (*T*) (Fig. 1).

There are two main types of features in a fingerprint:

1) global ridge and furrow structures which form special patterns in the central region of the fingerprint, and
2) local ridge and furrow minute details.

A fingerprint is classified based on only the first type of features and uniquely identified based on the second type

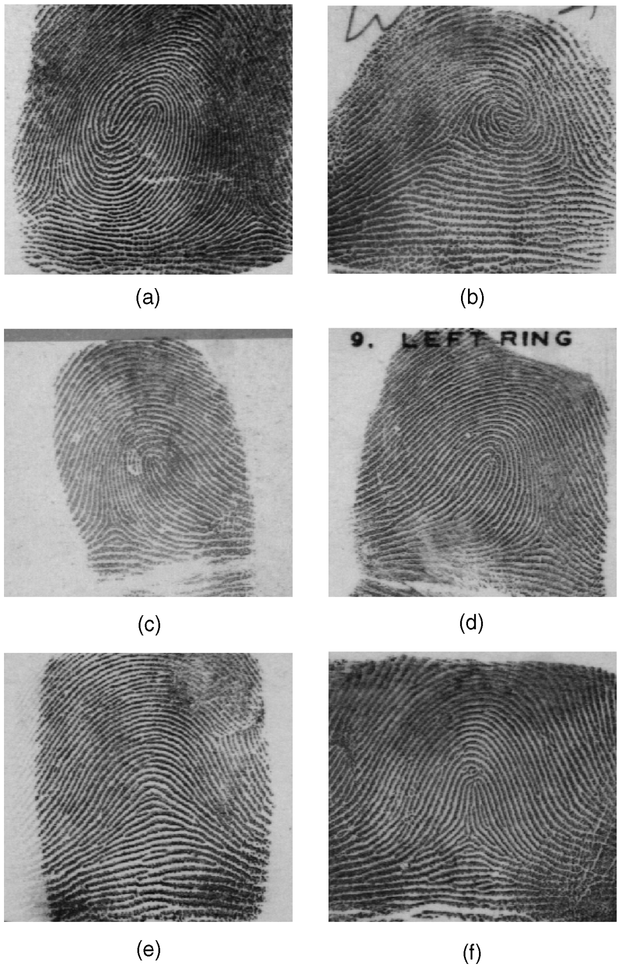


Fig. 1. Major fingerprint classes. Twin loop images are labeled as whorl in the NIST-4 database. (a) Twin loop (W), (b) Whorl (W), (c) Right loop (R), (d) Left loop (L), (e) Arch (A), (f) Tented arch (T).

• *A.K. Jain and S. Prabhakar are with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824. E-mail: {jain, prabhaka}@cse.msu.edu.*
• *L. Hong is with Visionics Corporation. E-mail: honglin@cse.msu.edu.*

Manuscript received 16 July 1998; revised 22 Dec. 1998. Recommended for acceptance by K. Bowyer.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 107855.

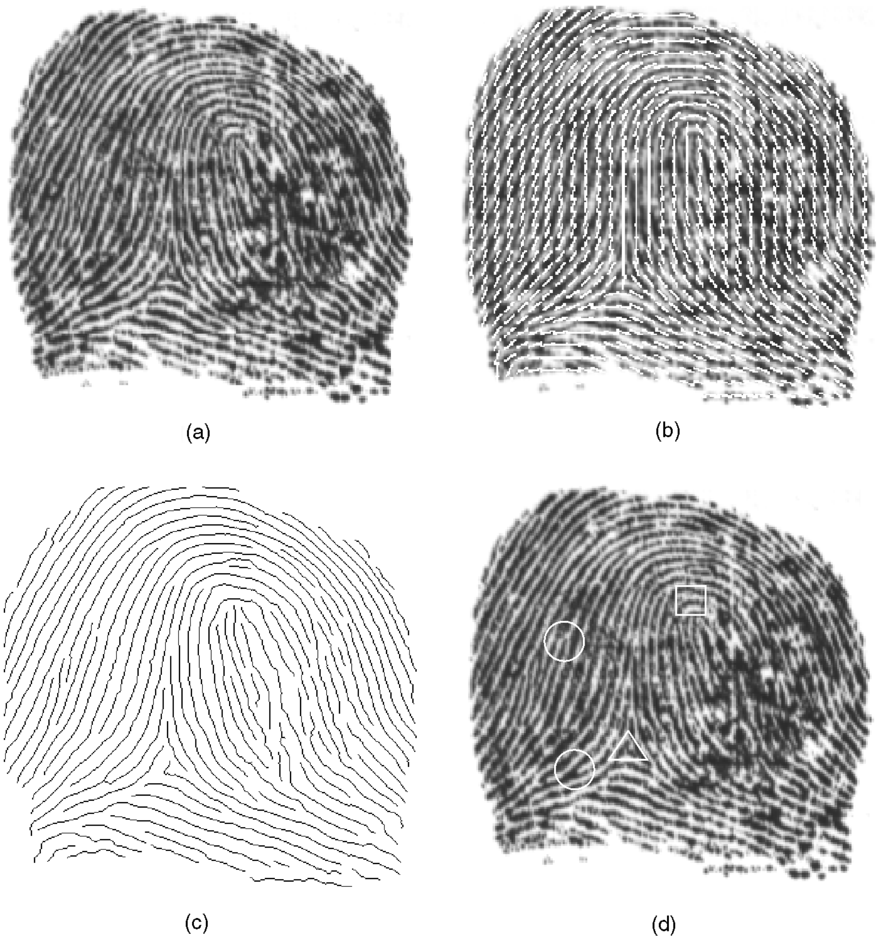


Fig. 2. Orientation field, thinned ridges, minutiae, and singular points. (a) Gray scale image. (b) Orientation field. (c) Thinned ridges. (d) Minutiae (○), Core (□), and Delta (△).

of features (one such feature is the ridge endings and bifurcations, also known as minutiae). See Fig. 2 for examples of ridges, minutiae, orientation field, and singular points in a fingerprint image.

Several approaches have been developed for automatic fingerprint classification. These approaches can be broadly categorized into four main categories:

1) model-based,
2) structure-based,
3) frequency-based, and
4) syntactic.

The model-based fingerprint classification technique uses the locations of singular points (core and delta) to classify a fingerprint into the five above-mentioned classes [3], [4]. A model-based approach tries to capture the knowledge of a human expert by deriving rules for each category by hand-constructing the models and therefore, does not require training. Accuracies of 85 percent [3] and 87.5 percent [4] have been reported on the NIST-4 database [5] using these approaches. A structure-based approach uses the estimated orientation field in a fingerprint image to classify the fingerprint into one of the five classes. An accuracy of 90.2 percent with 10 percent rejection is reported on NIST-4 [6]. The neural network used in [6] was trained on images from 2,000 fingers (one image per finger) and then tested on an independent set of 2,000 images taken from the same fingers. The error reported is, thus, optimistically biased. A *later* version of this algorithm [7] was tested on the NIST-14 database, which is a *naturally* distributed database, resulting in a better performance. However, this performance improvement should be expected since the NIST-14 database contains only a small percentage of arch-type fingerprints, which pose the most difficulty for fingerprint classifiers and the neural network used in the algorithm implicitly takes advantage of this information. A similar structure-based approach, which uses hidden Markov models for classification [8], depends on a reliable estimation of ridge locations, which is difficult in noisy images. In another structure-based approach, B-spline curves are used to represent and classify fingerprints [9]. A syntactic approach uses a formal grammar to represent and classify fingerprints [10]. Frequency-based approaches use the frequency spectrum of the fingerprints for classification [11]. Hybrid approaches combine two or more approaches for classification [12], [13]. These approaches show some promise but have not been tested on large databases. For example, Chong et al. [9] report results on 89 fingerprints, Fitz and Green [11] on 40 fingerprints, and Kawagoe and Tojo [13] on 94 fingerprints.

Most of the category information about a fingerprint is contained in the central part of the fingerprint. The model-based techniques which use both the core and delta points

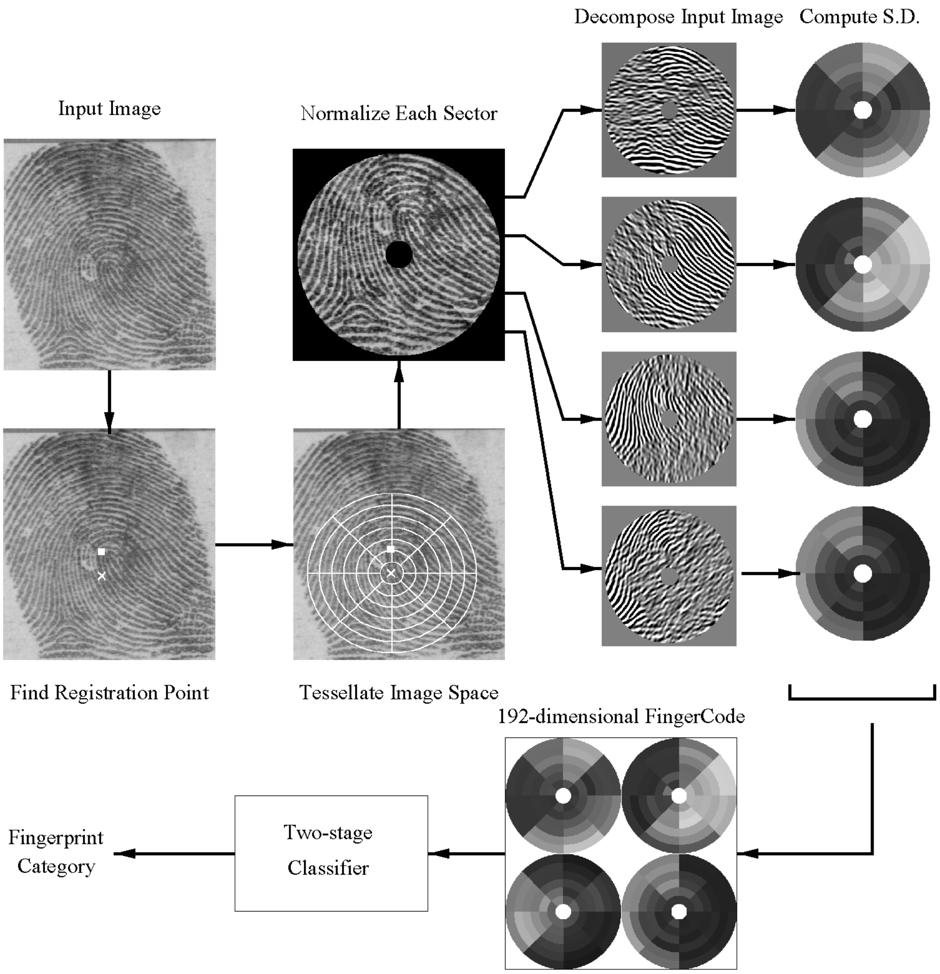


Fig. 3. Flow diagram of our fingerprint classification algorithm.

for classification require that these singular points be present in the image. The dab fingerprint images obtained by optical scanners do not always capture the entire fingerprint and often have the delta point(s) missing. The new solid-state fingerprint capture devices are small in size and so, capture only a part of the fingerprint (e.g., the FPS100 solid-state sensor from Veridicom is about the size of a postage stamp). Also, the core or delta point(s) are difficult to detect in noisy fingerprint images. There is, however, sufficient information available in the ridge pattern itself to classify a fingerprint. While the structure-based approach does not depend upon the core or delta points, it requires a reliable estimate of the orientation field which is again very difficult to obtain in low quality fingerprint images.

We propose a fingerprint classification algorithm (Fig. 3) based on a novel representation scheme which is directly derived from local ridge structures. The representation does not use the core, delta, and orientation field, explicitly. It is more capable of tolerating poor image quality, which is a major difficulty in fingerprint classification.

The main steps of our classification algorithm are as follows:

1) Locate a registration point in the input image and define a spatial tessellation of the region around the registration point (sectors).
2) Decompose the input image into a set of component images, each of which preserves certain ridge structures; compute the standard deviation of the component images in each sector to generate the feature vector (called FingerCode).
3) Feed the feature vector into a classifier; in our algorithm, a two-stage classifier is used.

This two-stage classifier uses a *K*-nearest neighbor classifier in its first stage and a set of neural network classifiers in its second stage to classify a feature vector into one of the five fingerprint classes.

In the following sections, we will present the details of our fingerprint classification algorithm. Section 2 presents our feature extraction scheme. In Section 3, we present our classification scheme. In Section 4, we present our experimental results on the NIST-4 database. The conclusions and future research directions are presented in Section 5.

2 Feature Extraction

The category of a fingerprint is determined by its global ridge and furrow structures. A valid feature set for fingerprint classification should be able to capture this global information effectively. We have developed a novel representation scheme (FingerCode) which is able to represent both the minute details and the global ridge and furrow structures of a fingerprint. For the purpose of classification, we use a low-level representation which is very effective in representing the global ridge and furrow structures and which is invariant to individual minute details.

The main steps of our feature extraction algorithm are as follows:

1) Find a registration point (center point) and define a spatial tessellation of the image space around the registration point (represented by a collection of sectors).
2) Decompose the input image into a set of component images, which preserve global ridge and furrow structures.
3) Compute the standard deviation of gray level values in each sector to form the feature vector or the FingerCode.

Let $I(x, y)$ denote the gray level at pixel (x, y) in an $M \times N$ fingerprint image and let (x_c, y_c) denote the center point. The spatial tessellation of the image space which consists of the region of interest is defined by a collection of sectors S_i, where the ith sector S_i is computed in terms of parameters (r, θ) as follows:

$$S_i = \{(x, y) \mid b(T_i + 1) \le r < b(T_i + 2), \quad \theta_i \le \theta < \theta_{i+1}, \\ 1 \le x \le N, \quad 1 \le y \le M\}, \tag{1}$$

where

$$T_i = i \ div \ k, \tag{2}$$

$$\theta_i = (i \ mod \ k)\left(\frac{2\pi}{k}\right), \tag{3}$$

$$r = \sqrt{(x - x_c)^2 + (y - y_c)^2}, \tag{4}$$

$$\theta = tan^{-1}\left(\frac{y - y_c}{x - x_c}\right), \tag{5}$$

b is the width of each band and k is the number of sectors considered in each band. We use six concentric bands around the center point. Each band is 20-pixels wide (b = 20), and segmented into eight sectors (k = 8) (Fig. 4). The innermost band is not used for feature extraction because the sectors in the region near the center contain very few pixels. Thus, a total of 8 × 6 = 48 sectors (S_0 through S_{47}) are defined.

2.1 Center Point Location

Any point that can be consistently detected in a fingerprint image can be used as a registration point (or center point because we prefer this point to be positioned at the center of the image). In a fingerprint image, the core point presents such a consistent point. Therefore, in our algorithm, we define core point as the center point (x_c, y_c). We used the core point detection algorithm described in [4] which is presented below.

1) Estimate the orientation field O using the least square orientation estimation algorithm [14]. Orientation field O is defined as an $N \times N$ image, where $O(i, j)$ represents the local ridge orientation at pixel (i, j). An image is divided into a set of $w \times w$ nonoverlapping windows and a single local orientation is defined for each window.
2) Smooth the orientation field in a local neighborhood. Let the smoothed orientation field be represented as O'.
3) Initialize $\mathcal{A}$, a label image used to indicate the core point.
4) For each pixel (i, j) in O', compute the *Poincaré index* and assign the corresponding pixels in $\mathcal{A}$ a value of one if the *Poincaré index* is (1/2). The *Poincaré index* at pixel (i, j) enclosed by a digital curve, which consists of a sequence of pixels that are on or within a distance of one pixel apart from the corresponding curve, is computed as follows:

$$Poincaré(i, j) = \frac{1}{2\pi}\sum_{k=0}^{N_\Psi - 1}\Delta(k), \tag{6}$$

$$\Delta(k) = \begin{cases} \delta(k), & \text{if } |\delta(k)| < \frac{\pi}{2} \\ \pi + \delta(k), & \text{if } \delta(k) \le -\frac{\pi}{2} \\ \pi - \delta(k), & otherwise, \end{cases} \tag{7}$$

$$\delta(k) = O'(\Psi_x(k'), \Psi_y(k')) - O'(\Psi_x(k), \Psi_y(k)), \tag{8}$$

$$k' = (k + 1) \ mod \ N_\Psi, \tag{9}$$

where $\Psi_x(\cdot)$ and $\Psi_y(\cdot)$ are the x and y coordinates of the closed digital curve with N_Ψ pixels.
5) Find the connected components in $\mathcal{A}$. If the area of a connected component is larger than seven, a core is detected at the centroid of the connected component. If the area of a connected component is larger than 20, two cores are detected at the centroid of the connected component.
6) If more than two cores are detected, go back to Step 2.
7) If two cores are detected, the center is assigned the coordinates of the core point with the lower y value

Fig. 4. Core (□), center (×), the region of interest and 48 sectors.

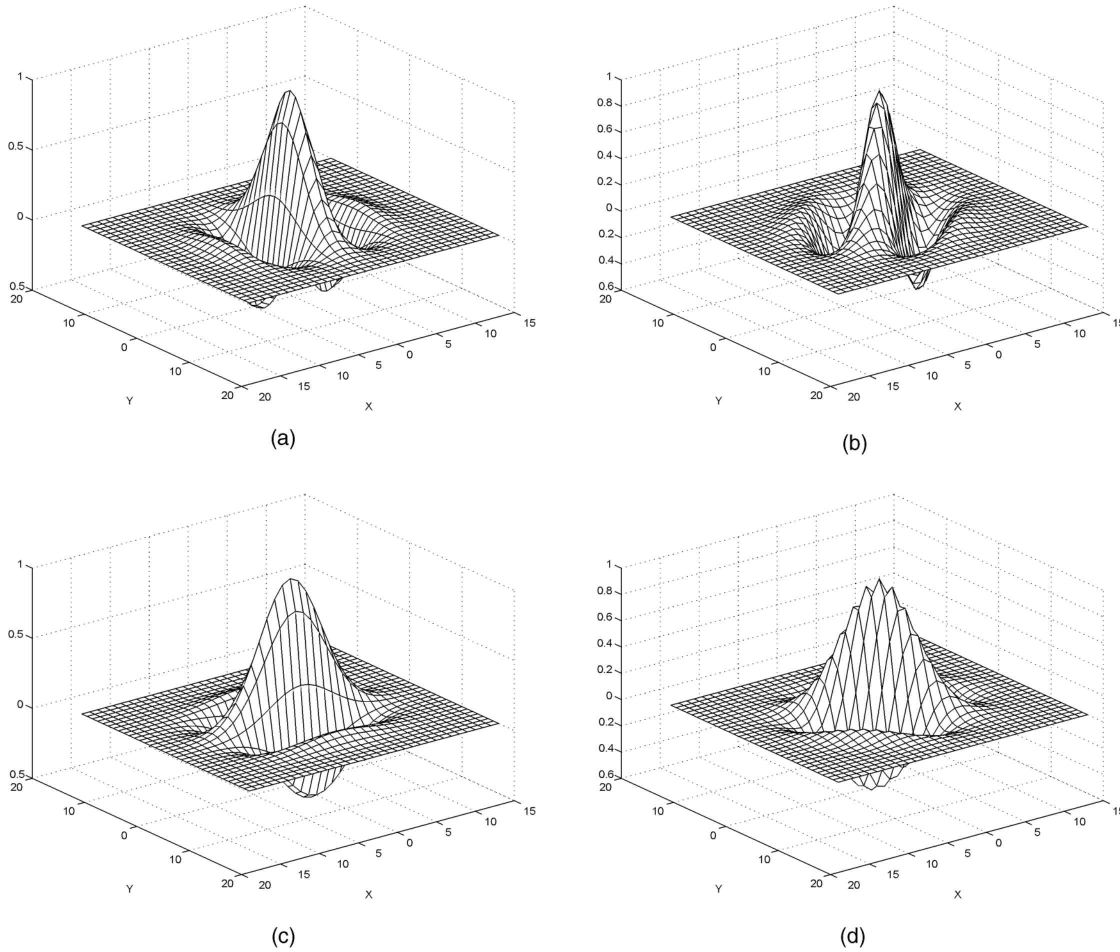


Fig. 5. Gabor filters (size = 33 × 33, $f = 0.1$, $\delta_x = 4.0$, $\delta_y = 4.0$). (a) 0° orientation. (b) 45° orientation. (c) 90° orientation. (d) 135° orientation.

(the upper core). If only one core is detected, the center is assigned the coordinates of the core point.

8) If no core point is detected, compute the covariance matrix of the vector field in a local neighborhood ($q \times q$) of each point in the orientation field. Define a feature image $\mathcal{F}$ with the largest eigenvalue of the covariance matrix for each element in the orientation image. A core is detected at the centroid of the largest connected component in the thresholded image of $\mathcal{F}$ and the center is assigned the coordinates of the core.

The center found above is shifted 40 pixels down for further processing based on the fact that most of the category information in a fingerprint lies in the lower part of the fingerprint. This value was empirically determined. See Figs. 4, 12, 13, 14, and 15 for results of our center location algorithm.

2.2 Decomposition

Fingerprint images present a strong orientation tendency and have a well-defined spatial frequency in each local neighborhood that does not contain singular point(s) (Fig. 6b). Gabor filters are band-pass filters which have both orientation-selective and frequency-selective properties and have optimal joint resolution in both spatial and frequency domains [18]. By applying properly tuned Gabor filters to a fingerprint image, the true ridge and furrow structures can be greatly accentuated. These accentuated ridges and furrow structures constitute an efficient representation of a fingerprint image.

An even symmetric Gabor filter has the following general form in the spatial domain:

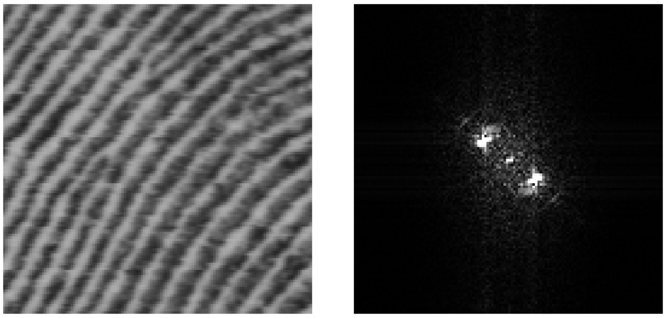
$$G(x, y; f, \theta) = exp\left\{\frac{-1}{2}\left[\frac{x'^2}{\delta_x^2} + \frac{y'^2}{\delta_y^2}\right]\right\}cos(2\pi f x'), \qquad (10)$$

$$x' = x sin\theta + y cos\theta, \qquad (11)$$

$$y' = x cos\theta - y sin\theta, \qquad (12)$$

where f is the frequency of the sinusoidal plane wave along the direction θ from the x-axis, and δ_x and δ_y specify the Gaussian envelope along x and y axes, respectively, which determine the bandwidth of the Gabor filter.

In our algorithm, the filter frequency f is set to the average ridge frequency ($1/K$), where K is the interridge distance. The average interridge distance is approximately 10 pixels in a 500 *dpi* fingerprint image. If f is too large, spurious ridges may be created in the filtered image, whereas if f is too small, nearby ridges may be merged into one. The bandwidth of the Gabor filters is determined by δ_x and δ_y. The selection of the values of δ_x and δ_y is based on the following trade-off. If they are too large, the filter is more robust to noise, but is more likely to smooth the image to the extent that the ridge and



(a) (b)

Fig. 6. Fingerprints have well defined local frequency and orientation. (a) Ridges in a local region. (b) Fourier spectrum.

furrow details in the fingerprint are lost. On the other hand, if they are too small, the filter is not effective in removing noise. In our algorithm, the values of δ_x and δ_y were empirically determined and both were set to 4.0.

A fingerprint image is decomposed into four component images corresponding to four different values of θ (0°, 45°, 90°, and 135°) with respect to the x-axis (Fig. 5). A fingerprint image is convolved with each of the four Gabor filters to produce the four component images. Convolution with a 0°-oriented filter accentuates ridges parallel to the x-axis, and it smoothes ridges which are not parallel to the x-axis. Filters tuned to other directions work in a similar way. According to our experimental results, the four component images capture most of the ridge directionality information present in a fingerprint image and thus form a valid representation. We illustrate this by reconstructing a fingerprint image by adding together all the four filtered images. The reconstructed image is similar to the original image but the ridges have been enhanced (Fig. 7).

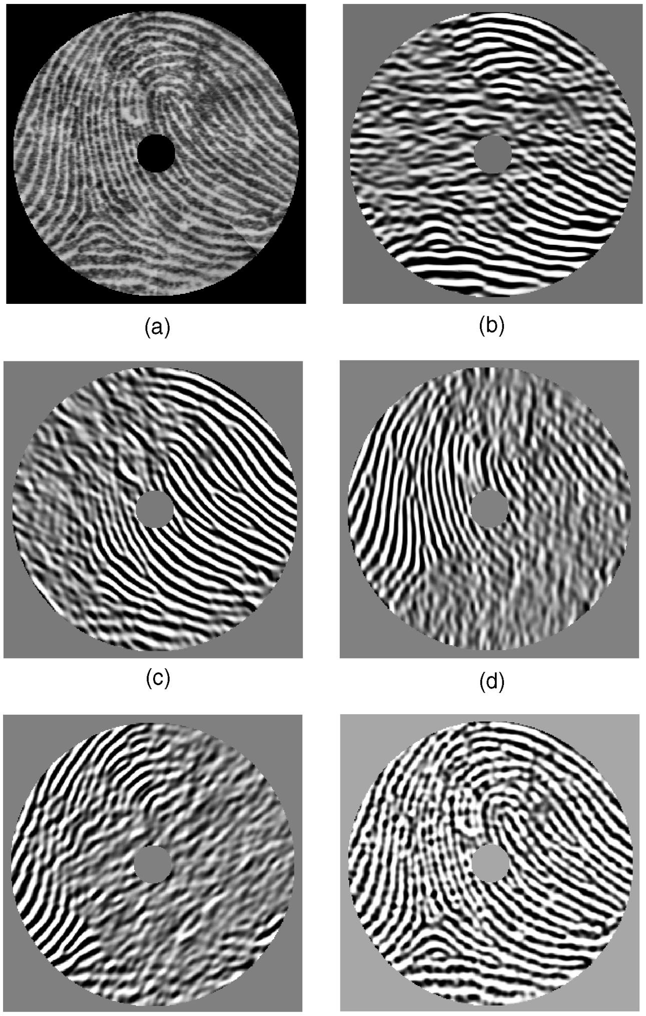
Before decomposing the fingerprint image, we normalize the region of interest in each sector separately to a constant mean and variance. Normalization is done to remove the effects of sensor noise and finger pressure differences. Let $I(x, y)$ denote the gray value at pixel (x, y), M_i and V_i, the estimated mean and variance of sector S_i, respectively, and $N_i(x, y)$, the normalized gray-level value at pixel (x, y). For all the pixels in sector S_i, the normalized image is defined as:

$$N_i(x,y) = \begin{cases} M_0 + \sqrt{\dfrac{(V_0)\times\left(I(x,y)-M_i\right)^2}{V_i}}, & \text{if } I(x,y) > M_i \\ M_0 - \sqrt{\dfrac{(V_0)\times\left(I(x,y)-M_i\right)^2}{V_i}}, & \text{otherwise,} \end{cases} \tag{13}$$

where M_0 and V_0 are the desired mean and variance values, respectively. Normalization is a pixel-wise operation which does not change the clarity of the ridge and furrow structures. If normalization is done on the entire image, then it cannot compensate for the intensity variations in the different parts of the finger due to finger pressure differences. Normalization of each sector separately alleviates this problem. For our experiments, we set both M_0 and V_0 to a value of 100. Normalized, filtered, and reconstructed images for the fingerprint shown in Fig. 4 are shown in Fig. 7.

2.3 Feature Vector

In each component image, a local neighborhood with ridges and furrows that are parallel to the corresponding filter direction exhibits a higher variation, whereas a local neighborhood with ridges and furrows that are not parallel to the corresponding filter tends to be diminished by the filter which results in a lower variation. The spatial distribution of the variations in local neighborhoods of the component images thus constitutes a characterization of the global ridge structures and is well captured by the standard deviation of grayscale values. In our algorithm, the standard deviation within the sectors defines the feature vector.



(a) (b) (c) (d) (e) (f)

Fig. 7. Normalized, filtered, and reconstructed fingerprint images. (a) Normalized image. (b) Component image 0°. (c) Component image 45°. (d) Component image 90°. (e) Component image 135°. (f) Reconstructed image.

Let $C_{i\theta}(x, y)$ be the component image corresponding to θ for sector S_i. For $\forall i$, $i = 0, 1, \ldots, 47$ and $\theta \in [0°, 45°, 90°, 135°]$, a feature is the standard deviation $F_{i\theta}$, defined as:

$$F_{i\theta} = \sqrt{\sum_{K_i}\left(C_{i\theta}(x,y) - M_{i\theta}\right)^2}, \tag{14}$$

where K_i is the number of pixels in S_i and $M_{i\theta}$ is the mean of the pixel values in $C_{i\theta}(x, y)$. The 192-dimensional feature vectors, also called FingerCodes (similar to the IrisCode introduced by Daugman [16]), for typical fingerprint images from different classes are shown as gray level images with four disks, each disk corresponding to one filtered image (Fig. 8). The gray level in each sector of a disk represents the feature value for that sector in the corresponding filtered image. One can see that visually this representation appears to discriminate the five fingerprint classes reasonably well.

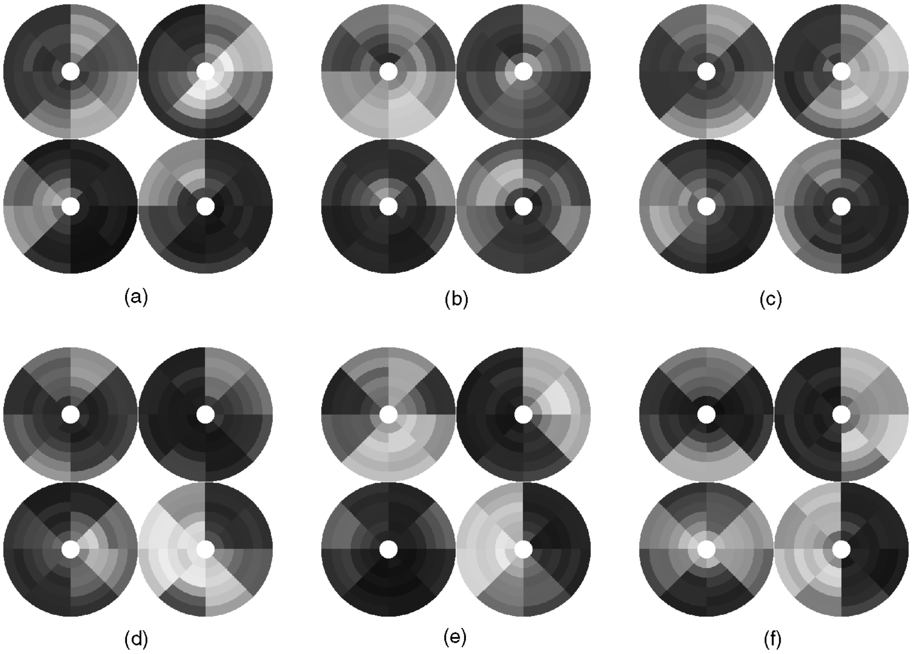


Fig. 8. Fingerprint representation using 192-dimensional feature vectors (In each representation, the top left disk represents the 0° component, the top right disk represents the 45° component, the bottom left disk represents the 90° component, and the bottom right disk represents the 135° component). The test pattern is a right loop. Each disk corresponds to one particular filter and there are 48 features (shown as gray values) in each disk ($8 \times 6 = 48$ sectors) for a total of 192 (48×4) features. (a) Test, (b) whorl, (c) right loop, (d) left loop, (e) arch, (f) tented arch.

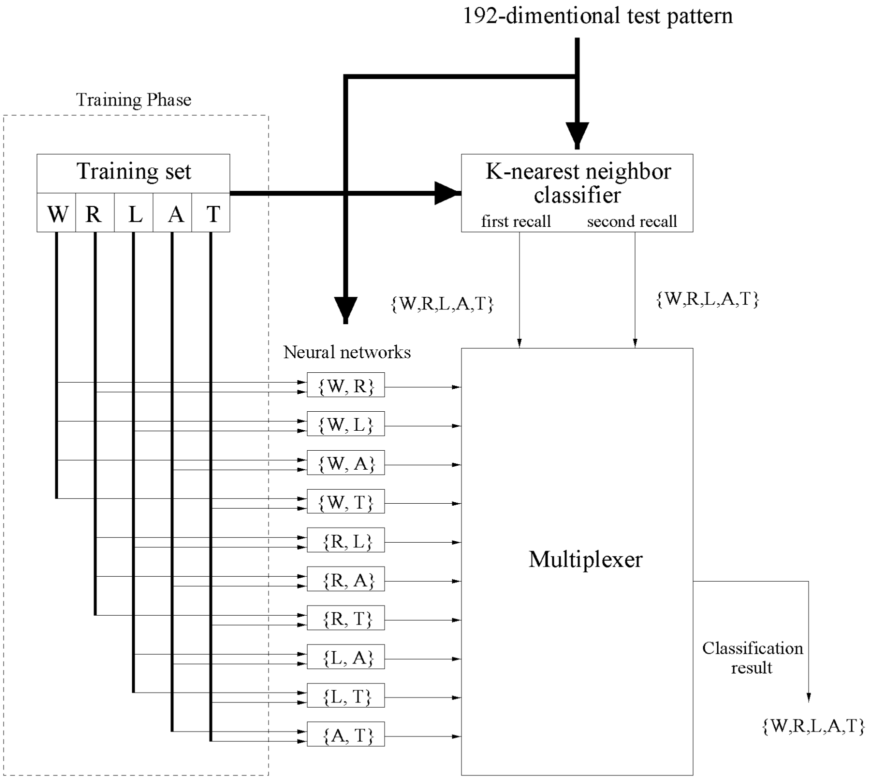


Fig. 9. Classification scheme using a combination of classifiers.

3 CLASSIFICATION

Automatic classification of fingerprints is a difficult problem because of the small *interclass* variability and large *intraclass* variability among the five classes under consideration.

In order to simplify the classification task, we decompose the five-class problem into a set of 10 two-class problems. Further, we use a two-stage classifier for fingerprint classification. In the first stage, we use a *K*-nearest neighbor classifier to find the two most probable classes for a given input pattern. The *K*-nearest neighbor decision rule first finds the *K* nearest neighbors of the test pattern in the feature space and, then, assigns the test pattern to the class which is most frequently represented among the *K* nearest neighbors. The top two categories can be retrieved from the *K*-NN classifier corresponding to the classes which have the highest and the second highest count among the *K* nearest neighbors, i.e., the first recall and the second recall. In the second stage of the classifier, a set of 10 (C_2^5) neural networks are trained to solve the 10 different two-class problems. The second stage uses the first and second recalls to select the specific neural network which has been trained to distinguish between the corresponding pair of classes and the input pattern is then sent to the selected neural network for further classification. This neural network makes the final decision between these two classes.

4 EXPERIMENTAL RESULTS

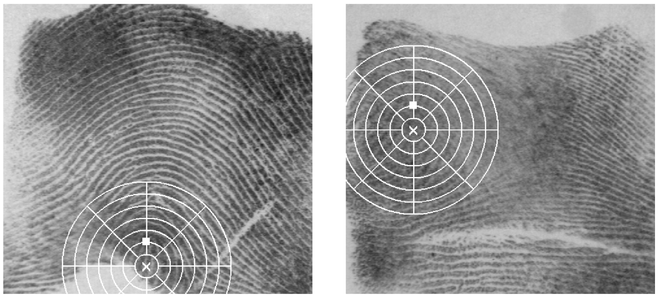
4.1 Dataset

We report the results of our fingerprint classification algorithm on the NIST-4 database for the five-class fingerprint classification problem. Since fingerprint classes *A* (*arch*) and *T* (*tented arch*) have a substantial overlap, it is very difficult to separate these two classes. Therefore, we also report our results for the four-class classification problem, where classes *A* and *T* have been merged into one class. By incorporating a rejection option, classification accuracy can be increased. We report the improvement in error rates at different rejection rates for both the five-class and the four-class classification problems.

The NIST-4 database consists of 4,000 fingerprint images (image size is 512 × 480) from 2,000 fingers. Each finger has two impressions (*f* and *s*). Each image is labeled with *one or more* of the five classes (*W*, *R*, *L*, *A*, and *T*). To simplify the training procedure, we make use of only the first label of a fingerprint to train our system. For testing, however, we make use of all the labels for a fingerprint and consider the output of our classifier to be correct if the output matches any one of the labels. This is in line with the common practice used by other researchers in comparing the classification results on the NIST-4 database. The images in the NIST-4 database are numbered *f*0001 through *f*2000 and *s*0001 through *s*2000. Each number represents a fingerprint from a different finger. We form our training set with the first 2,000 fingerprints from 1,000 fingers (*f*0001 to *f*1000 and *s*0001 to *s*1000) and the test set contains the remaining 2,000 fingerprints (*f*1001 to *f*2000 and *s*1001 to *s*2000). The natural proportion of fingerprints belonging to each class is 0.279, 0.317, 0.338, 0.037, and 0.029 for the classes W, R, L, A, and T, respectively [6]. Classification accuracies can be significantly increased by using datasets whose records follow the natural distribution of fingerprint classes because the more common types of fingerprints (*loop* and *whorl*) are easier to recognize. However, we do not use datasets with a natural class distribution. Twenty-eight fingerprints from the training set were rejected by our feature extraction algorithm because the center was detected at a corner of the image and, therefore, a valid tessellation could not be established for these images (Fig. 10). Thirty-five fingerprints were rejected from the test set for the same reason. So, our training set contains 1,972 fingerprint images. The 35 images rejected from the test set of 2,000 fingerprints amounts to a reject rate of 1.8 percent.

4.2 *K*-Nearest Neighbor Classifier

The *K*-nearest neighbor classifier results in an accuracy of 85.4 percent for the five-class classification task when 10 nearest neighbors ($K = 10$) are considered. Classification accuracy does not always increase with increasing *K*; there exists an optimal value of *K* for finite training sample size classification problems (Fig. 11) [19]. For the four-class classification task (where classes *A* and *T* were collapsed into one class), an accuracy of 91.5 percent is achieved. The confusion matrix for the *K*-nearest neighbor classification is shown in Table 1. The diagonal entries in this matrix show the number of test patterns from different classes which are correctly classified. Since a number of fingerprints in the



(a) (b)

Fig. 10. Example of images which were rejected because a valid tessellation could not be established.

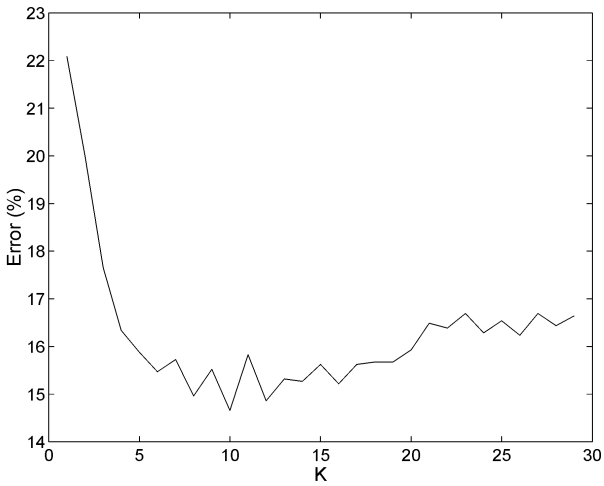


Fig. 11. *K* vs. percent error for the *K*-nearest neighbor classifier.

TABLE 1
CONFUSION MATRIX FOR THE K-NEAREST NEIGHBOR CLASSIFICATION; $K = 10$

True class	Assigned class				
	W	R	L	A	T
W	**320**	38	31	6	0
R	1	**368**	2	10	21
L	0	1	**359**	13	8
A	1	3	7	**422**	20
T	0	15	16	95	**208**

TABLE 2
CONFUSION MATRIX FOR THE NEURAL NETWORK CLASSIFICATION

True class	Assigned class				
	W	R	L	A	T
W	**352**	29	10	2	2
R	6	**374**	1	9	17
L	10	2	**353**	10	7
A	0	6	8	**384**	48
T	1	16	19	64	**235**

TABLE 3
CONFUSION MATRIX FOR THE TWO-STAGE CLASSIFICATION

True class	Assigned class				
	W	R	L	A	T
W	**366**	16	8	4	1
R	3	**372**	1	8	17
L	6	0	**364**	6	7
A	2	1	3	**405**	39
T	0	6	14	55	**261**

NIST-4 database are labeled as belonging to two different classes, row sums of the confusion matrices in Tables 1, 2, and 3 are not identical.

4.3 Neural Network Classifier

We trained a multilayer feed-forward neural network using a quick propagation training algorithm [22]. The neural network has one hidden layer with 20 neurons, 192 input neurons, and five output neurons corresponding to the five classes. We obtain an accuracy of 86.4 percent for the five-class classification task. For the four-class classification task, an accuracy of 92.1 percent is achieved. The confusion matrix for the neural network classification is shown in Table 2.

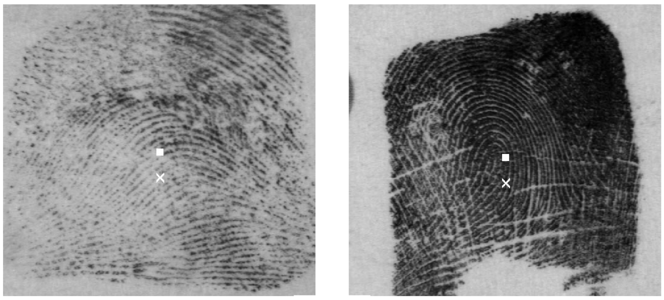
4.4 Two-Stage Classifier

The objective here is to perform a "simple" classification task using a *K*-NN classifier and then use a bank of two-class neural network classifiers to handle more subtle discriminations. The first stage uses the *K*-nearest neighbor ($K = 10$) classifier to yield the two most probable classes. We observed that 85.4 percent of the time, the class with the maximum frequency among the *K* nearest neighbors is the correct class and 12.6 percent of the time, the class with the second highest frequency is the correct class. In other words, the *K*-nearest neighbor classifier yields the top two classes with an accuracy of 98 percent. This result itself can be used to accurately classify fingerprints into two out of the five classes. Each fingerprint will have an entry in two of the five partitions of the database and the matching is required to be performed only in the corresponding two partitions of the database. The second stage uses 10 different neural networks for 10 different pairwise classifications. These neural networks have 192 input neurons, 20–40 hidden neurons in one hidden layer, and 2 output neurons. Each neural network is trained using the patterns from only the two corresponding classes in the training set. For example, the neural network which distinguishes between *R* and *W* is trained using only the patterns labeled *R* and *W* in the training set.

This two-stage classifier yields an accuracy of 90 percent for the five-class classification task and an accuracy of 94.8 percent is achieved for the four-class classification task. The confusion matrix for the two-stage classification is shown in Table 3. Although our classifier is robust to noise and is able to correctly classify most of the poor quality fingerprints in the NIST-4 database (Fig. 12), it fails on some very bad quality fingerprint images where no ridge information is present in the central part of the fingerprint (Fig. 13). In poor quality fingerprints, it is very difficult to detect the center point correctly (Fig. 10b). Our classifier also fails to correctly classify *twin loop* images which are labeled as *whorl* in the NIST-4 database. For these images, our center location algorithm picks up the upper core and on considering that as the center, the image looks like a loop in the region of interest which leads to a misclassification of *W* as *L* or *R*. See Fig. 14 for these misclassifications. About 3 percent of the errors result from *loop-arch* misclassification because of the subtle difference between *loop* and *arch* types (see Fig. 15a). The *A*-*T* misclassification accounts for about 5 percent of the errors. An example of this type of confusion is shown in Fig. 15b.

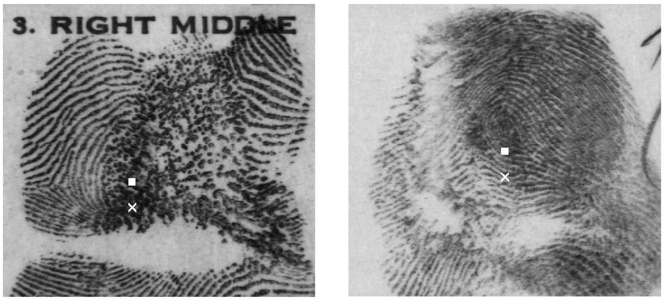
4.5 Reject Option

Classification accuracies can be further increased by incorporating a rejection option. We use the (K, K')-nearest neighbor classifier [23] for rejection and the proposed



(a) (b)

Fig. 12. Poor quality images which were correctly classified. (a) Arch, (b) left loop.



(a) (b)

Fig. 13. Poor quality images which were misclassified as arch. (a) Whorl, (b) right loop.

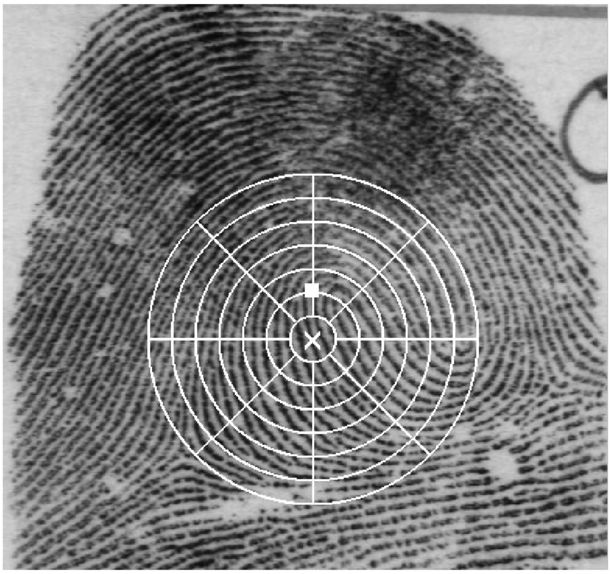
two-stage classifier for classification. If the number of training samples from the majority class among the K nearest neighbors of a test pattern is less than K' ($K' < K$), we reject the test pattern and do not attempt to classify it. Most of the rejected images using this scheme are of poor quality (Figs. 16a and 16b). Other rejected images are those images which "appear" to belong to different classes. For example, for the fingerprint image shown in Fig. 16c, three of its nearest neighbors belong to class R, three to class A, and four to class T. By rejecting 19.5 percent of the images for the five-class problem, the classification accuracy can be increased to 93.5 percent and for the four-class classification problem, the accuracy can be increased to 96.6 percent (Table 4).

5 Conclusions

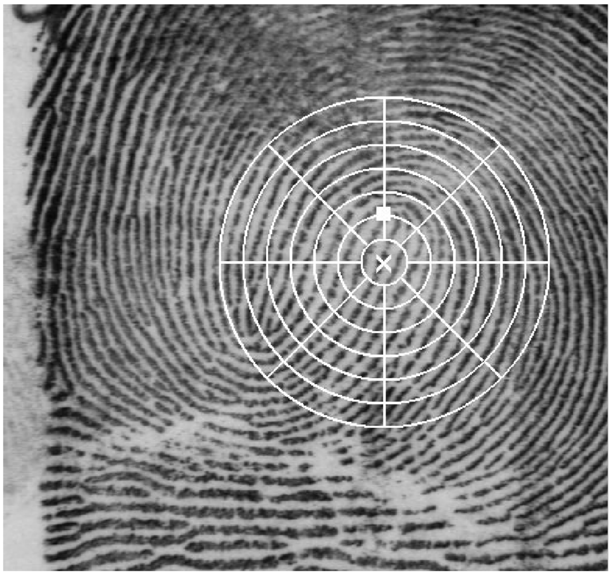
We have developed a novel multichannel filter-based classification algorithm which gives better accuracy than previously reported in the literature on the NIST-4 database. Our feature vector, called FingerCode, is more representative of

TABLE 4
Error-Reject Trade-Off

Classifier	(10,0)-NN	(10,5)-NN	(10,6)-NN	(10,7)-NN
Rejection rate	1.8%	8.5%	19.5%	32.5%
5-class error	10%	8.8%	6.5%	4%
4-class error	5.2%	4.5%	3.4%	2.2%

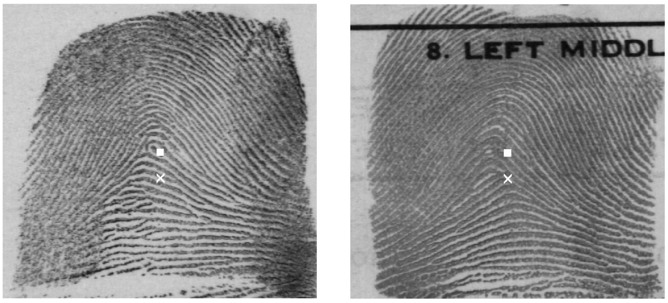


(a)



(b)

Fig. 14. Misclassification of whorl (twin loop) as (a) right loop, (b) left loop.



(a) (b)

Fig. 15. Misclassifications; (a) a right loop misclassified as an arch; (b) an arch misclassified as a tented arch.

(a) (b) (c)

Fig. 16. Examples of images rejected by (K, K')-NN.

the fingerprint class information and is robust to noise which is reflected in the classification accuracy. We have tested our algorithm on the NIST-4 database and a very good performance has been achieved (90 percent for the five-class classification problem and 94.8 percent for the four-class classification problem with 1.8 percent rejection during the feature extraction phase). However, this algorithm suffers from the requirement that the region of interest be correctly located, requiring the accurate detection of center point in the fingerprint image. By improving the accuracy of registration point location, a better performance can be easily expected. Our system takes about 10 seconds on a Sun Ultra-1 machine to classify one fingerprint, which is another aspect of the algorithm that needs to be improved. Since image decomposition (filtering) steps account for 90 percent of the total compute time, special purpose hardware for convolution can significantly decrease the overall time for classification.

Acknowledgments

We would like to thank Dr. Ruud Bolle and Dr. Sharath Pankanti of the IBM T.J. Watson Research Center and Scott Connell of the PRIP lab, Michigan State University, for their many useful suggestions.

References

[1] *Advances in Fingerprint Technology*, H.C. Lee and R.E. Gaensslen, eds. New York: Elsevier, 1991.

[2] A.K. Jain, L. Hong, S. Pankanti, and R. Bolle, "An Identity Authentication System Using Fingerprints," *Proc. IEEE*, vol. 85, no. 9, pp. 1,365-1,388, 1997.

[3] K. Karu and A.K. Jain, "Fingerprint Classification," *Pattern Recognition*, vol. 29, no. 3, pp. 389-404, 1996.

[4] L. Hong and A.K. Jain, "Classification of Fingerprint Images," Technical Report MSUCPS:TR98-18, Michigan State Univ., June 1998.

[5] C.I. Watson and C.L. Wilson, "NIST Special Database 4, Fingerprint Database," Nat'l Inst. of Standards and Technology, Mar. 1992.

[6] C.L. Wilson, G.T. Candela, and C.I. Watson, "Neural Network Fingerprint Classification," *J. Artificial Neural Networks*, vol. 1, no. 2, pp. 203-228, 1993.

[7] G.T. Candela, P.J. Grother, C.I. Watson, R.A. Wilkinson, and C.L. Wilson, "PCASYS—A Pattern-Level Classification Automation System for Fingerprints," Technical Report NISTIR 5647, Apr. 1995.

[8] A. Senior, "A Hidden Markov Model Fingerprint Classifier," *Proc. 31st Asilomar Conf. Signals, Systems and Computers*, pp. 306-310, 1997.

[9] M.M.S. Chong, T.H. Ngee, L. Jun, and R.K.L. Gay, "Geometric Framework for Fingerprint Classification," *Pattern Recognition*, vol. 30, no. 9, pp. 1,475-1,488, 1997.

[10] C.V.K. Rao and K. Black, "Type Classification of Fingerprints: A Syntactic Approach," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 2, no. 3, pp. 223-231, 1980.

[11] A.P. Fitz and R.J. Green, "Fingerprint Classification Using Hexagonal Fast Fourier Transform," *Pattern Recognition*, vol. 29, no. 10, pp. 1,587-1,597, 1996.

[12] B.G. Sherlock and D.M. Monro, "A Model for Interpreting Fingerprint Topology," *Pattern Recognition*, vol. 26, no. 7, pp. 1,047-1,055, 1993.

[13] M. Kawagoe and A. Tojo, "Fingerprint Pattern Classification," *Pattern Recognition*, vol. 17, no. 3, pp. 295-303, 1984.

[14] A. Jain, L. Hong, and R. Bolle, "On-Line Fingerprint Verification," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 302-314, Apr. 1997.

[15] J.G. Daugman, "Two-Dimensional Spectral Analysis of Cortical Receptive Field Profiles," *Vision Research*, vol. 20, pp. 847-856, 1980.

[16] J.G. Daugman, "High Confidence Recognition of Persons by a Test of Statistical Independence," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 11, pp. 1,148-1,161, Nov. 1993.

[17] A.K. Jain and F. Farrokhnia, "Unsupervised Texture Segmentation Using Gabor Filters," *Pattern Recognition*, vol. 24, no. 12, pp. 1,167-1,186, 1991.

[18] J.G. Daugman, "Uncertainty Relation for Resolution in Space, Spatial Frequency, and Orientation Optimized by Two-Dimensional Visual Cortical Filters," *J. Optical Soc. Amer. A*, vol. 2, pp. 1,160-1,169, 1985.

[19] A.K. Jain and B. Chandrasekaran, "Dimensionality and Sample Size Considerations in Pattern Recognition Practice," *Handbook of Statistics*, vol. 2, P.R. Krishnaiah and L.N. Kanal, eds., pp. 835-855. North-Holland, 1982,

[20] N. Ratha, S. Chen, and A.K. Jain, "Adaptive Flow Orientation-Based Feature Extraction in Fingerprint Images," *Pattern Recognition*, vol. 28, no. 11, pp. 1,657-1,672, 1995.

[21] N. Ratha, K. Karu, S. Chen, and A.K. Jain, "A Real-Time Matching System for Large Fingerprint Databases," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 799-813, Aug. 1996.

[22] S.E. Fahlman, "Faster-Learning Variations on Back-Propagation: An Empirical Study," *Proc. 1988 Connectionist Models Summer School*, 1988.

[23] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.

[24] J.L. Wayman, "Technical Testing and Evaluation of Biometric Identification Devices," *Biometrics: Personal Identification in Networked Society*, A.K. Jain, R. Bolle, and S. Pankanti, eds. Kluwer, 1998.

[25] K. Woods, W.P. Kegelmeyer, and K.W. Bowyer, "Combination of Multiple Classifiers Using Local Accuracy Estimates," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 4, pp. 405-410, Apr. 1997.

Anil K. Jain is a University Distinguished Professor and Chair of the Department of Computer Science and Engineering at Michigan State University. His research interests include statistical pattern recognition, Markov random fields, texture analysis, neural networks, document image analysis, fingerprint matching, and 3D object recognition. He received the best paper awards in 1987 and 1991 and certificates for outstanding contributions in 1976, 1979, 1992, and 1997 from the Pattern Recognition Society. He also received *the 1996 IEEE Transactions on Neural Networks* Outstanding Paper Award. He was the editor-in-chief of the *IEEE Transactions on Pattern Analysis and Machine Intelligence* (1990-1994). He is the co-author of *Algorithms for Clustering Data* (Prentice Hall, 1988), edited the book *Real-Time Object Measurement and Classification* (Springer-Verlag, 1988), and co-edited the books, *Analysis and Interpretation of Range Images* (Springer-Verlag, 1989), *Markov Random Fields* (Academic Press, 1992), *Artificial Neural Networks and Pattern Recognition* (Elsevier, 1993), *3D Object Recognition* (Elsevier, 1993), and *BIOMETRICS: Personal Identification in Networked Society* (Kluwer, 1998). He is a fellow of the IEEE and the IAPR. He received a Fulbright research award in 1998.

Salil Prabhakar received his BTech degree in computer science and engineering from the Institute of Technology, Banaras Hindu University, Varanasi, India, in 1996. During 1996-1997, he worked with Tata Information Systems Ltd. (now IBM Global Services India Pvt. Ltd.), Bangalore, India, as a software engineer. In the fall of 1997, he joined the Department of Computer Science and Engineering, Michigan State University, where he is currently a doctoral degree candidate. His research interests include pattern recognition, image processing, computer vision, biometrics, network security, computer graphics, and multimedia systems.

Lin Hong received the BS and MS degrees in computer science from Sichuan University, China, in 1987 and 1990, respectively, and the PhD degree in computer science from Michigan State University in 1998. He is currently with Visionics Corporation. His current research interests include multimedia, biometrics, data mining, pattern recognition, image processing, and computer vision application.