Classification of Fingerprint Images

Lin Hong and Anil Jain
Department of Computer Science, Michigan State University, East Lansing, MI 48824
{honglin,jain}@cps.msu.edu

Abstract

Automatic fingerprint identification is one of the most important biometric technology. In order to efficiently match fingerprints in a large database, an indexing scheme is necessary. Fingerprint classification, which refers to assigning a fingerprint image into a number of pre-specified classes, provides a feasible indexing mechanism. In practice, however, large intraclass and small interclass variations in global pattern configuration and poor quality of fingerprint images make the classification problem very difficult. A fingerprint classification algorithm requires a robust feature extractor which should be able to reliablely extract salient features from input images. We present a fingerprint classification algorithm with an improved feature extraction algorithm and a novel classification scheme. This algorithm has been tested on the NIST-4 fingerprint database. For the 4,000 images in this database, error rates of 12.5% for the five-class problem and 7.7% for the four-class problem have been achieved. With a 20% reject rate (which eliminates most of the poor quality images in the database), the error of the four-class problem drops to 2.4%.

1. Introduction

Accurate *automatic* personal identification is critical in a wide range of application domains such as national ID card, electronic commerce, and automated banking [9]. *Biometrics*, which refers to automatic identification of a person based on her physiological or behavioral characteristics [9], is inherently more reliable and more capable in differentiating between an authorized person and a fraudulent imposter than traditional methods such as passwords and PIN numbers. *Automatic fingerprint identification* is one of the most reliable biometric technology. The high accuracy and reliability of fingerprint identification have long been established and justified [7]. However, automatic fingerprint identification is computationally demanding especially for a large database. Without an effective fingerprint indexing scheme, one needs to exhaustively match a query fingerprint against all the fingerprints in the database, which is not desirable in practice. *Fingerprint classification*, which classifies fingerprints into a number of pre-specified categories, provides an indexing scheme to facilitate efficient matching for large fingerprint databases; if two fingerprint images are the impressions of the same finger, then they must belong to the same category. Therefore, a query fingerprint needs to be compared only with the database fingerprints of the same category in the fingerprint matching process. Figure 1 shows a typical fingerprint image which needs to be classified into five major categories.

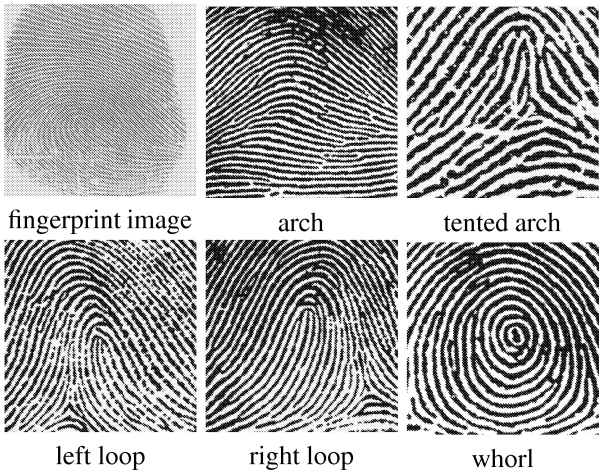


Figure 1. A fingerprint image and five major fingerprint classes.

The central problem in designing a fingerprint classification algorithm is to determine what features should be used and how categories are defined based on these features. There are mainly two types of features that are useful for fingerprint identification: (i) *local ridge and furrow details* (minute details) which have different characteristics for different fingerprints, and (ii) *global*

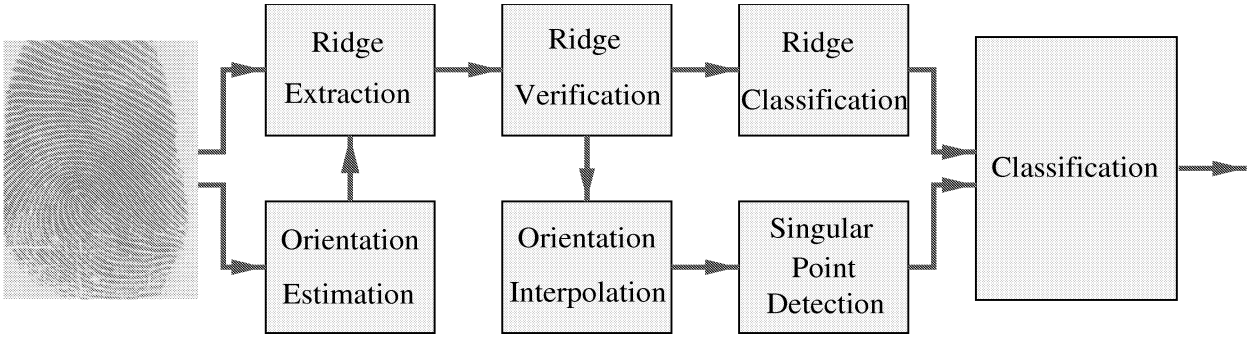


Figure 3. Stages in our fingerprint classification algorithm.

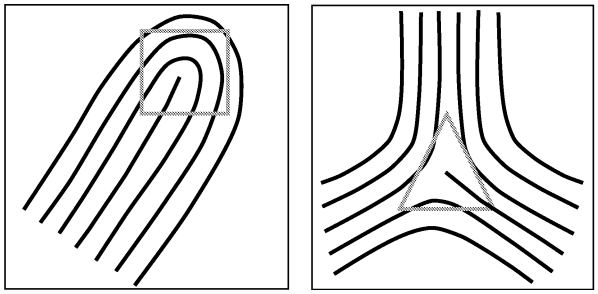


Figure 2. Singular points.

pattern configurations which form special patterns of ridges and furrows in the central region of the fingerprint. The first type of features carry the individuality information about the fingerprints and the second type of features carry information about the fingerprint class. Therefore, for fingerprint classification, the features derived from the global pattern configurations should be used. These features should be invariant to the translation and rotation of the input fingerprint images. Generally, global fingerprint features can be derived from the orientation field and global ridge shape. The orientation field of a fingerprint consists of the ridge orientation tendency in local neighborhoods and forms an abstraction of the local ridge structures. It has been shown that the orientation field is highly structured and can be roughly approximated by a core-delta model [8]. Therefore, *singular points* (see Figure 2) and their relationship can be used to drive fingerprint categories. On the other hand, global ridge shape also provides important clues about the global pattern configuration of a fingerprint image.

Previous approaches to fingerprint classification can be roughly divided into two categories: (i) statistical approach [2, 1] and (ii) structural approach [3, 5, 6]. A statistical approach classifies a fingerprint using feature vectors derived directly from the orientation field or the input images. A structural approach extracts and represents fingerprints using a number of salient fingerprint properties and their relationships. These algorithms perform reasonably well when the input fingerprint images are of good quality. When the quality of the input fingerprint images is poor, the performance of these algorithms degrades rapidly. The major reason for the brittleness of these algorithms is that they do not utilize robust features. We have designed a fingerprint classification algorithm based on the features mentioned above to classify a fingerprint into five categories *(arch, tented arch, left loop, right loop, and whorl)*. The main stages of the classification algorithm are depicted in Figure 3. Our algorithm spends a significant amount of effort to improve the quality of extracted orientation field and ridges, which results in a more robust feature extraction and classification.

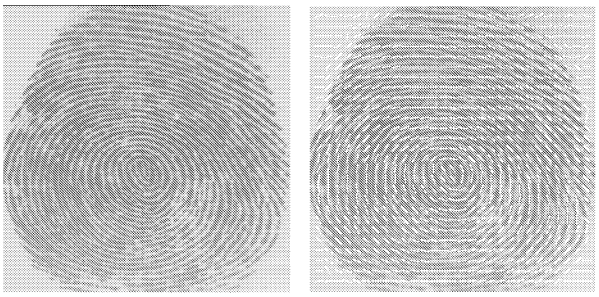
2. Feature Extraction

Our feature extraction algorithm extracts two types of features: (i) singular points, and (ii) fingerprint ridges. Following is a list of the definitions needed to understand these features.

2.1. Definitions

An *orientation image*, $\mathcal{O}$, is defined as an $N \times N$ image, where $\mathcal{O}(i, j)$ represents the *local ridge orientation* at pixel (i, j). Local ridge orientation is usually specified for a region (block) rather than at every pixel; an image is divided into a set of $w \times w$ non-overlapping blocks and a single local ridge orientation is defined for each block. Note that in a fingerprint image, the ridges oriented at 0^o and the ridges oriented at 180^o in a local neighborhood can not be differentiated from each other.

A *ridge map*, $\mathcal{R}$, is an $N \times N$ binary image, where $\mathcal{R}(i, j) = 1$ indicates that pixel (i, j) is a ridge pixel and $\mathcal{R}(i, j) = 0$ indicates that pixel (i, j) is not a ridge pixel. A ridge in a ridge map is an 8-connected component. A



(a) (b)

Figure 4. Orientation image; (a) input image; (b) orientation image superimposed on the input image.

(a) (b)

Figure 5. Ridge map; (a) extracted ridges from the input image in Figure 4; (b) thinned ridge map.

thinned ridge has a width of 1 pixel and a *thinned* ridge map consists of thinned ridges.

A *Recurring ridge* is defined as a chain of pixels, $r_1, r_2, ..., r_n$, in a thinned ridge map, where r_1 is the first ridge pixel, r_n is the last ridge pixel, and each pair of consecutive pixels, (r_{i-1}, r_i) is eight connected, which cumulatively turns more than a certain degree when traveling from r_i to r_j, where $1 \leq i < j \leq n$.

A *singular point* is either a core point or a delta point which is characterized by its position and type (See Figure 2). A core is defined as a point in the orientation field where the orientation in a small local neighborhood around the point presents semi-circular tendency. A delta is defined as a point in the orientation field where a small local neighborhood around the point forms three sectors and the orientation in each sector presents hyperbolic tendency.

2.2. Local Orientation and Ridges

The orientation image represents an intrinsic property of a fingerprint image. We have developed a least mean square orientation estimation algorithm [4], which provides a fairly smooth orientation field estimate for a reasonable quality fingerprint image. Figure 4 shows an example of the orientation image estimated with our algorithm. After orientation image has been estimated from an input image, a ridge extraction algorithm [4] is applied to extract the ridge map and the thinned ridge map (Figure 5).

2.3. Ridge Verification

In an ideal fingerprint image, ridges and furrows are well-defined in each local neighborhood. The local ridge orientation can be reliably estimated from the sinusoidal-shaped plane waves of ridges and furrows.

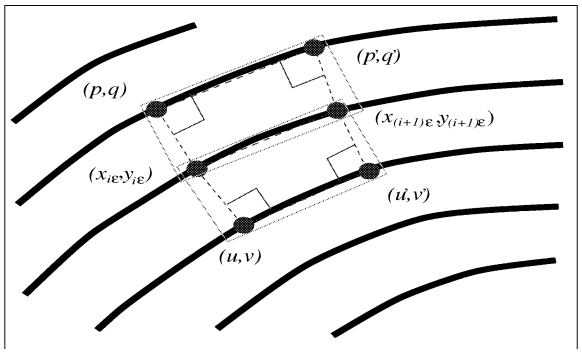


Figure 6. Ridge verification.

In practice, due to variations in impression conditions, ridge configuration, skin conditions, and characteristics of the acquisition devices, a significant percentage (10-15% based on our experience) of acquired fingerprint images is of poor quality. The ridge structures in poor-quality fingerprint images are not always well-defined, which may lead to: (i) incorrect local ridge orientation estimates and (ii) incorrect extracted ridges. It is very difficult to correctly classify a fingerprint based on the incorrect orientation field and incorrect ridge structures. Therefore, a noise removal algorithm should be applied to obtain more precise orientation estimates and ridges.

We have developed a ridge verification algorithm which receives as input a thinned ridge map and outputs a refined thinned ridge map, a refined orientation field, and a quality index which indicates the goodness of the input ridge map. Let $\mathcal{R}$, $\mathcal{O}'$, and $\mathcal{R}'$ be the input ridge map, the interpolated orientation field, and the verified ridge map, respectively. The major steps in our ridge verification algorithm are as follows:

1. *Initialize $\mathcal{O}'$, $\mathcal{R}'$, and $\mathcal{A}$ which is a map used to indicate the genuine regions.*
2. *Delete all ridge pixels in $\mathcal{R}$ which have more than*

two 8-connected neighboring pixels to ensure that each ridge is a single 8-connected chain.

3. *Trace and label all the ridges in $\mathcal{R}$. For each traced ridge, $r = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, do the following: (i) Smooth r; (ii) Let $(x_{i\epsilon}, y_{i\epsilon})$ and $(x_{(i+1)\epsilon}, y_{(i+1)\epsilon})$ denote the starting point and ending point of a segment in r, where ϵ is the length of the segment and $i = 0, \epsilon, 2\epsilon, ..., \lfloor \frac{n-\epsilon}{\epsilon} \rfloor \epsilon$. Find the 4 nearest neighboring ridge points, (u, v), (p, q), (u', v'), and (p', q') (Figure 6). $(x_{i\epsilon}, y_{i\epsilon})$, $(x_{(i+1)\epsilon}, y_{(i+1)\epsilon})$, (p, q), and (u, v) form a quadrilateral at one side of the segment. $(x_{i\epsilon}, y_{i\epsilon})$, $(x_{(i+1)\epsilon}, y_{(i+1)\epsilon})$, (p', q'), and (u', v') form a quadrilateral at the other side of the segment; (iii) For each quadrilateral, find the minimum rectangle that contains the quadrilateral. Compute the ratio, η, between the area of the quadrilateral and the area of the minimum rectangle. If η is larger than a threshold ($\eta_0 = 0.75$), then label all the pixels inside the quadrilateral as foreground pixels. Otherwise, label them as background pixels.*

4. *Remove all the foreground connected components whose area is less than a threshold ($\omega_0 = 15$) and fill all the background connected components in $\mathcal{A}$ whose area is less than a threshold ($\tau_0 = 15$).*

5. *Compute local orientation at all the pixels in $\mathcal{O}'$, where the corresponding pixels in $\mathcal{A}$ are foreground pixels, as the orientation of the nearest ridge segment. Interpolate the local orientation at all pixels in $\mathcal{O}'$, where the corresponding pixels in $\mathcal{A}$ are background pixels.*

6. *Return the percentage of the area of the foreground regions in $\mathcal{A}$ with respect to total area of $\mathcal{A}$ as the quality index.*

An example of ridge verification is depicted in Figure 7, which demonstrates that a better orientation field can be obtained by using our ridge verification algorithm.

2.4. Singular Point Detection

In an orientation field, the *Poincare index* of a core-shaped singular point has a value of $(1/2)$ and the Poincare index of a delta-shaped singular point has a value of $(-1/2)$ [8]. Let $\Psi_x(\cdot)$ and $\Psi_y(\cdot)$ represent the x and y coordinates of a closed digital curve with N_Ψ pixels. The Poincare index at pixel (i, j) which is enclosed by the digital curve can be computed as follows:

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

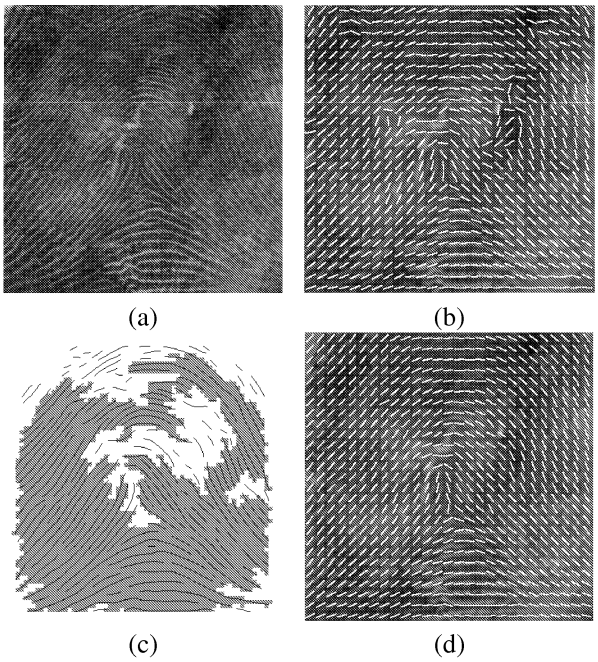


Figure 7. Ridge verification; (a) input image; (b) orientation field; (c) verified ridge map, where the verified ridges are marked with gray shade; (d) interpolated orientation field.

where

$$\Delta(k) = \begin{cases} \delta(k), & \text{if } |\delta(k)| < \pi/2, \\ \pi + \delta(k), & \text{if } \delta(k) \leq -\pi/2, \\ \pi - \delta(k), & otherwise, \end{cases}$$
$$\delta(k) = \mathcal{O}'(\Psi_x(i'), \Psi_y(i')) - \mathcal{O}'(\Psi_x(i), \Psi_y(i)),$$
$$i' = (i+1) mod N_\Psi.$$

The size of the closed digital curve is crucial for the performance of a singular point detection algorithm using the Poincare index. If it is too small, then a small perturbation of orientations may result in spurious singular points being detected. On the other hand, if it is too large, then a true pair of core and delta which are close to one another may be ignored because the Poincare index of a digital curve that includes an equal number of cores and deltas is 0. We have developed a singular point detection algorithm which uses a closed square curve with a length of 25 pixels. We have empirically determined that a curve of 25 pixels is a good trade-off between detections and misses of singular points. Let $\mathcal{O}'$ be the interpolated orientation field. The main steps in our singular point detection algorithm are as follows:

1. *Initialize $\mathcal{A}$, which is a label image used to indicate the singular points.*

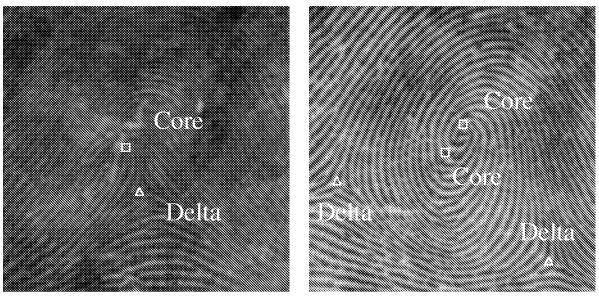


Figure 8. Singular point detection.

2. *For each pixel (i, j) in $\mathcal{O}'$, compute the Poincare index and assign the corresponding pixel in $\mathcal{A}$ a value 1 if the Poincare index is (1/2) and a value 2 if the Poincare index is (-1/2).*

3. *Find each connected component in $\mathcal{A}$ with pixel values 1. If the area of the connected component is larger than 7, a core is detected at the centroid of the connected component. If the area of the connected component is larger than 20, then two cores are detected at the centroid of the connected component.*

4. *Find each connected component in $\mathcal{A}$ with pixel values 2. If the area of the connected component is larger than 7, a delta is detected at the centroid of the connected component.*

5. *If more than two cores or more than two deltas are detected, smooth the orientation field $\mathcal{O}'$ and go back to step 1.*

Although the heuristic that at most two cores and two deltas exist in a fingerprint is not always true, it is rarely observed that a fingerprint has more than two cores and two deltas. Results of applying our singular point detection algorithm on two fingerprint images are shown Figure 8.

2.5. Recurring Ridges

The global shape of ridges determines the global configuration of fingerprints. Ridges in fingerprints are highly structured. Generally, in the upper region (which can be roughly defined as the region above the highest core points in loops, tented arches, and whorls and the region above the most curved ridges in arches) of a fingerprint, ridges are a family of uni-modal smooth curve segments. In the bottom region, ridges form a family of relatively flat curves. In the middle region, depending on the fingerprint class, ridges may be of the following types: uni-modal curve segment, recurring segment, circular segments, multi-recurring segments, spiral segments, etc. The presence of a particular type of ridges defines the class of a fingerprint. If the ridge type can be accurately determined, then the fingerprint can be correctly classified.

We classify ridges into three categories: (i) *non-recurring ridge*, (ii) *type-1 recurring ridge*, and (iii) *type-2 recurring ridge*. Let $r = \{r_1, r_2, ..., r_n\}$ be a ridge of length n, where r_1 is the first ridge pixel, r_n is the last ridge pixel, and each pair of consecutive pixels is eight connected. Then $r' = \{r_1, r_{2\epsilon}, ..., r_{m\epsilon}\}$, where $m = \lfloor \frac{n-\epsilon}{\epsilon} \rfloor$, is obtained by sampling r at intervals of length ϵ. Define the cumulative orientation of r as:

$$
\begin{aligned}
AO(r) &= |\frac{1}{2\pi} \sum_{k=2}^{m-1} \varpi(k)| \\
\varpi(k) &= \begin{cases} \rho(k), & \text{if } |\rho(k)| < \pi, \\ 2\pi + \rho(k), & \text{if } \rho(k) \leq -\pi, \\ 2\pi - \rho(k), & otherwise, \end{cases} \\
\rho(k) &= \vartheta(k) - \vartheta(k-1),
\end{aligned}
$$

where $\vartheta(k)$ represents the angle from $r'(k)$ to $r'(k+1)$. Define any sequence of ridge pixels in r, $\{r_i, r_2, ..., r_j\}$, where $1 \leq i < j \leq n$, a sub-ridge of r. A non-recurring ridge, r, is a ridge such that the cumulative orientation of any sub-ridge of r is less than a threshold, $T_{non} = 150^o$. A type-1 recurring ridge, r, is a ridge such that the cumulative orientation of any sub-ridge of r is between the two thresholds, T_{non} and $T_{rec} = 270^o$. A type-2 recurring ridge, r, is a ridge such that the cumulative orientation of any sub-ridge of r is larger than a threshold, T_{rec} or a ridge such that there exist multiple disjoint sub-ridges of r, which are type-1 recurring ridges. Obviously, uni-modal ridge segments and flat ridge segments are non-recurring ridges. Circular ridge segments, multi-recurring ridge segments and spiral ridge segments are type-2 recurring ridges.

It is very difficult to correctly extract all the true ridges from an input fingerprint image, especially when the quality of the input fingerprint image is poor. It is essential that a ridge classification algorithm be able to handle the following undesirable situations: (i) spurious ridges, (ii) broken ridges, and (iii) missing ridges. Ridge verification (see Figure 7) can be used to remove all the spurious ridges from a ridge map. Broken ridges can be connected based on the information present near the end of broken ridges. However, it is very difficult to recover missing ridges. This needs both high-level structural analysis and local structural analysis of the ridge pattern, which is very difficult to formulate and implement. We have developed a ridge classification algorithm which traces each ridge in the verified ridge map and classifies each ridge into one of the three categories

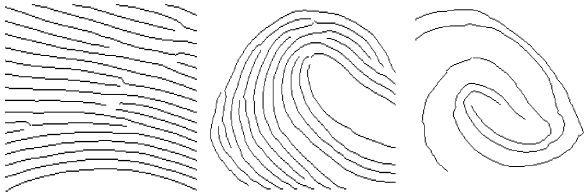


Figure 9. Ridge classification; ridges classified as (a) non-recurring, (b) type-1 recurring, and (c) type-2 recurring.

mentioned above (Figure 9).

3. Classification

The fingerprint classification algorithm classifies input fingerprints into *five* categories according to the number of singular points detected, their relative positions and presence of type-1 and type-2 recurring ridges. A prototype of each class is shown in Figure 10. Let $\mathcal{O}'$ be the interpolated orientation field; N_c and N_d be the number of cores and deltas detected from $\mathcal{O}'$, respectively; N_1 and N_2 be the number of type-1 recurring ridges and type-2 recurring ridges in $\mathcal{R}'$. The classification criteria used in our algorithm is as follows:

1. *If* $(N_2 > 0)$ *and* $(N_c = 2)$ *and* $(N_d = 2)$*, then a whorl is identified.*
2. *If* $(N_1 = 0)$ *and* $(N_2 = 0)$ *and* $(N_c = 0)$ *and* $(N_d = 0)$*, then an arch is identified.*
3. *If* $(N_1 > 0)$ *and* $(N_2 = 0)$ *and* $(N_c = 1)$ *and* $(N_d = 1)$*, then classify the input using the core and delta assessment algorithm given below.*
4. *If* $(N_2 > T_2)$ *and* $(N_c > 0)$*, then a whorl is identified.*
5. *If* $(N_1 > T_1)$ *and* $(N_2 = 0)$ *and* $(N_c = 1)$ *then classify the input using the core and delta assessment algorithm.*
6. *If* $(N_c = 2)$*, then a whorl is identified.*
7. *If* $(N_c = 1)$ *and* $(N_d = 1)$*, then classify the input using the core and delta assessment algorithm.*
8. *If* $(N_1 > 0)$ *and* $(N_c = 1)$*, then classify the input using the core and delta assessment algorithm.*
9. *If* $(N_c = 0)$ *and* $(N_d = 0)$*, then an arch is identified.*
10. *If none of the above conditions is satisfied, then reject the fingerprint.*

The core and delta assessment algorithm is used to classify a one-core and one-delta fingerprint into one of the following categories: (i) left loop, (ii) right loop, and (iii) tented arch. The steps of this algorithm are:

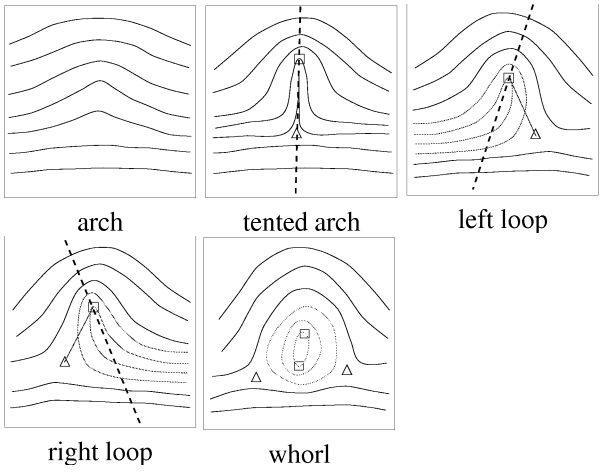


Figure 10. Fingerprint class prototypes; the dashed lines in (b), (c), and (d) are the symmetric axis.

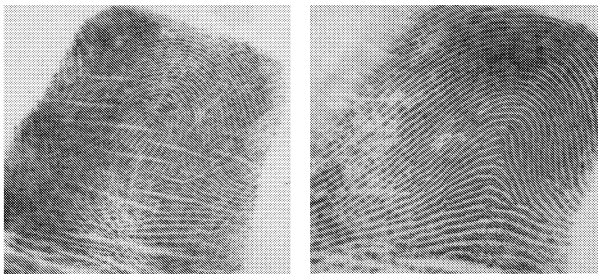
1. *Estimate the symmetric axis which crosses the core in its local neighborhood.*
2. *Compute the angle, α, between the line segment from the core to the delta and the symmetric axis.*
3. *Compute the average angle difference, β, between the local ridge orientation on the line segment from the core to the delta and the orientation of the line segment.*
4. *Count the number of ridges, γ, that cross the line segment from the core to the delta.*
5. *If* $(\alpha < 10^o)$ *or* $(\beta < 15^o)$ *and* $(\gamma = 0)$*, then classify the input as a tented arch.*
6. *If the delta is on the right side of the axis, then classify the input as a left loop.*
7. *If the delta is on the left side of the axis, then classify the input as a right loop.*

4. Experimental Results

We have tested our fingerprint classification algorithms on a number of databases. Here, we present the performance of our classification algorithm on the NIST-4 database which contains 4,000 images (image size is 512×480) taken from 2,000 different fingers, 2 images per finger. Five fingerprint classes are defined: (i) Arch, (ii) Tented arch, (iii) Left Loop, (iv) Right Loop, and (v) Whorl. Fingerprints in this database are uniformly distributed among these five classes. The five-class error rate in classifying these 4,000 fingerprints is 12.5%. The confusion matrix is given in Table 1; numbers shown in bold font are correct classifications. Since a number of

True Class	Assigned Class				
	A	T	L	R	W
A	**885**	13	10	11	0
T	179	**384**	54	14	5
L	31	27	**755**	3	20
R	30	47	3	**717**	16
W	6	1	15	15	**759**

Table 1. Five-class classification results on NIST-4 database; A-Arch, T-Tented Arch, L-Left Loop, R-Right Loop, W-Whorl.



(a) (b)

Figure 11. Misclassifications: (a) a left loop as an arch and (b) a tented arch an arch.

fingerprints in NIST-4 database are labeled as belonging to two different classes, each row of the confusion matrix in Table 1 does not sum up to 800. For the five-class problem, most of the classification errors are due to misclassifying a tented arch as an arch. By combining these two arch categories into a single class, the error rate drops to 7.7%. Besides the tented arch-arch errors, the other errors mainly come from misclassifications between arch/tented arch and loops and due to poor image quality. Two examples of misclassified fingerprints are shown in Figure 11. A lower error rate can be achieved by adding the reject option, which is based on the quality index of the input image. The error rates corresponding to different reject rates are listed in Table 2.

Reject rate	0%	5%	10%	20%
5-class Error	12.5%	11.6%	10.1%	7.5%
4-class Error	7.7%	6.6%	5.1%	2.4%

Table 2. Error-reject tradeoff.

5. Summary and Conclusions

Fingerprint classification provides an important indexing mechanism for automatic fingerprint identification. At a first glance, the fingerprint classification problem appears to be rather simple. But because of large intraclass and small interclass variations in global pattern configuration and poor quality of input images, the desired accuracy of 1% error rate at 20% reject rate is very difficult to achieve. We have designed a fingerprint classification algorithm which classifies input fingerprints into five categories according to the number of singular points detected, their relative position, and presence of type-1 and type-2 recurring ridges. Our algorithm invests a significant amount of effort in feature extraction to make robust to interclass variations as well as poor quality of input images. Experiment results demonstrate that our algorithm has better classification performance than previously reported in the literature on the same database. Currently, we are studying an alternative classification scheme which classifies fingerprints according to the pattern similarity. We are also investigating an image enhancement algorithm based on a bank of Gabor filters.

Acknowledgments

This research was supported by an IBM university partnership program.

References

[1] P. Baldi and Y. Chauvin. Neural networks for fingerprint recognition. *Neural Computation*, 5(3):402–418, 1993.

[2] 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*. NIST Tech. Report NISTIR 5647, August 1995.

[3] M. Chong, T. Ngee, L. Jun, and R. Gay. Geometric framework for fingerprint image classification. *Pattern Recognition*, 30(9):1475–1488, 1997.

[4] A. Jain, L. Hong, and R. Bolle. On-line fingerprint verification. *IEEE Trans. Pattern Anal. and Machine Intell.*, 19(4):302–314, 1997.

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

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

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

[8] D. Monro and B. Sherlock. A model for interpreting fingerprint topology. *Pattern Recognition*, 26(7):1047–1055, 1993.

[9] E. Newham. *The Biometric Report*. SJB Services, New York, 1995.