Fingerprint segmentation based on an AdaBoost classifier

Article *in* Frontiers of Computer Science in China · June 2011

DOI: 10.1007/s11704-011-9134-x · Source: DBLP

Front. Comput. Sci. China
DOI 10.1007/s11704-011-9134-x

RESEARCH ARTICLE

Fingerprint segmentation based on an AdaBoost classifier

Eryun LIU[1], Heng ZHAO[1], Fangfei GUO[1], Jimin LIANG (✉)[1], Jie TIAN[1,2]

1 Life Sciences Research Center, School of Life Sciences and Technology, Xidian University, Xi'an 710071, China
2 Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

Abstract Fingerprint segmentation is one of the most important preprocessing steps in an automatic fingerprint identification system (AFIS). Accurate segmentation of a fingerprint will greatly reduce the computation time of the following processing steps, and the most importantly, exclude many spurious minutiae located at the boundary of foreground. In this paper, a new fingerprint segmentation algorithm is presented. First, two new features, block entropy and block gradient entropy, are proposed. Then, an AdaBoost classifier is designed to discriminate between foreground and background blocks based on these two features and five other commonly used features. The classification error rate (*Err*) and McNemar's test are used to evaluate the performance of our method. Experimental results on FVC2000, FVC2002 and FVC2004 show that our method outperforms other methods proposed in the literature both in accuracy and stability.

Keywords fingerprint segmentation, entropy, gradient entropy, AdaBoost classifier, McNemar's test

1 Introduction

Fingerprint recognition techniques are widely applied in personal identification systems because fingerprint images are unique to individuals, invariant to age, and convenient in practice. Fingerprint segmentation is one of the most important steps in an automatic fingerprint recognition system (AFIS). The foreground areas of the fingerprint are regions of interests with clear texture patterns in which ridges and valleys appear alternately, while the background areas are not regions of interests. The task of fingerprint segmentation is to exactly distinguish between the foreground from the background. An accurate segmentation algorithm is helpful in the following processing steps mainly in two aspects: 1) speeding up computation; and 2) reducing spurious minutiae close to the boundary of foreground.

With the development of hardware techniques, many types of fingerprint sensors have become available in the market. Unfortunately, the variety of hardware types demands higher robustness to noise and adaptability of the AFIS. Fig. 1 shows some sample images from the first international fingerprint verification competition (FVC2000, the sensors used in DB1, DB2, DB3, and DB4 are low-cost optical sensor, low-cost capacitive sensor, optical sensor, and synthetic generator, respectively) [1]. The four sample images are collected by different sensors (In Fig. 1, DB4 is collected by a synthetic fingerprint generator). It can be seen that fingerprints collected by different sensors display different background noise. The different noise produced by different kinds of sensor increases the complexity and difficulty of a fingerprint segmentation algorithm. To develop a fingerprint segmentation method for each new sensor is impractical and also unnecessary.

Fingerprint segmentation is a binary classification task, foreground and background, and usually consists of three steps: 1) foreground feature extraction; 2) classifier design; and 3) post processing. Many types of noise are produced during image acquisition, and are different for different kinds of sensors. The features extracted from

Received September 22, 2009; accepted July 11, 2010

E-mail: jimleung@mail.xidian.edu.cn

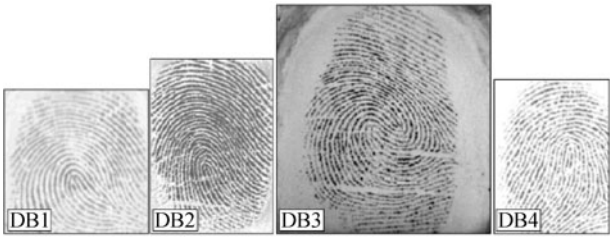

Fig. 1 Sample images of FVC2000 database

fingerprint images should have the ability to capture the intrinsic properties of the foreground, and a classifier decides which of the pixels of the fingerprint image belong to the foreground or background. Finally, a post processing step is needed to obtain a smooth contour.

Some fingerprint segmentation algorithms have been proposed previously. Hong et al. [2] proposed to use smoothed ridge frequency for segmentation; when too little information is used, this method is vulnerable to noise. Bazen and Gerez [3] proposed to extract three features: the coherence, mean, and variance of each pixel, and used a linear classifier to determine foreground or background classification. Their method was only tested on the FVC2000 database 1 and 2. Chen et al. [4] proposed to use a feature called block cluster degree (CluD) for fingerprint segmentation alongside a linear classifier. Their algorithm was tested on FVC2002 but only the classification error rate was used for performance evaluation. Zhan et al. [5] proposed a method using Monte Carlo Markov Chains for fingerprint segmentation. However, the existing segmentation algorithms lack adaptability to background noise. There are two main reasons: 1) the features used for segmentation are not representative or sufficient; 2) the classifiers used for feature fusion find it difficult to correctly classify pixels which are close to the region boundaries of background and foreground.

In this paper, a novel fingerprint segmentation algorithm is presented introducing two entropy based new features and an AdaBoost classifier. Moreover, McNemar's test is first applied for performance evaluation, together with the rate of classification error (*Err*). Experimental results on FVC2000, FVC2002, and FVC2004 show that our algorithm has lower *Err*, higher stability, and higher adaptability.

Section 2 presents the features used in our method, including two new features. Section 3 introduces the AdaBoost classifier for fingerprint segmentation. Section 4 describes our post processing procedures. Section 5 shows our experimental results, and conclusions are made in Section 6.

2 Features extraction

Feature extraction is the first step for segmentation. Although many features for representing the foreground of a fingerprint image have been proposed, reliable feature extraction is still a critical aspect affecting the segmentation result.

Given a fingerprint image, it can be divided into $\omega \times \omega$ sized non-overlapping blocks (e.g. $\omega = 12$). We denote the $\omega \times \omega$ block centered at site s as b_s. Pixels within the same block are treated equally, and feature extraction procedure is performed block by block. A feature vector, $v_s = \{v_s^1, v_s^2, ..., v_s^d\}$, corresponding to block b_s is obtained in the feature extraction procedure, where d is the dimension of the feature vector. In this study, seven features are extracted, two of them are proposed initially.

For the ease of description, we suppose that the processed images in our algorithm are grayscale images with 256 gray levels. Naturally, this procedure can be extended to higher levels of gray scale fingerprint images.

2.1 Entropy based feature extraction

In this paper, two entropy based features, block entropy (local entropy), and block gradient entropy, are presented.

Entropy feature: For a given block, b_s, of size $\omega \times \omega$, its entropy is defined as

$$H_s = -\sum_{k=0}^{255} p_k \log p_k, \tag{1}$$

where p_k is defined as

$$p_k = \frac{n_k}{\omega \times \omega}, \tag{2}$$

and n_k is the number of all pixels whose gray value equals to k. So p_k indicates the probability that the gray level k appears in b_s.

The entropy is a measure of the average information of an image. Entropy has many applications in image coding [6], medical image processing [7], and image reconstruction [8].

It is reasonable that the background regions of a fingerprint have lower entropy, and the foreground regions with clear ridges and valleys have higher entropy. Fig. 2(a)

shows an original image and Fig. 2(b) shows the corresponding entropy image, where the light blocks indicate high entropy values.

Gradient-entropy feature: In the fingerprint image, the edges of ridges (or valleys) contain much more important information in the foreground regions. Given a fingerprint image, $I(x, y)$, its gradient image, $g(x, y)$, can be obtained by filtering using gradient operators

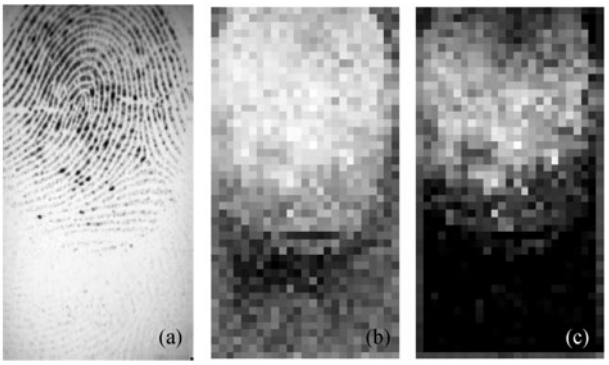

Fig. 2 Entropy and Gradient-entropy feature image of fingerprint. (a) Original image; (b) entropy image; (c) gradient-entropy image

$$g_x(x,y) = I(x,y) * h_x, \tag{3}$$

$$g_y(x,y) = I(x,y) * h_y, \tag{4}$$

$$g(x,y) = \sqrt{g_x^2 + g_y^2}, \tag{5}$$

where $*$ represents a convolution operation, and h_x, h_y are gradient operators (e.g. sobel operator) along x and y axes, respectively. By these operations, we can focus on the gradient image.

For a given block, b_s, of size $\omega \times \omega$ in the gradient image, $g(x, y)$, the block gradient entropy, H_s^g, is defined as

$$H_s^g = -\sum_{k=0}^{255} p_k^g \log p_k^g, \tag{6}$$

where p_k^g is defined as

$$p_k^g = \frac{n_k^g}{\omega \times \omega}, \tag{7}$$

and n_k^g is the number of pixels in the block, b_s, whose gray value is equal to k.

Figure 2 shows the gradient image of a fingerprint image and its gradient entropy. From Fig. 2(c) we can see that the background regions have a lower gradient value, and the entropy of gradient image of foreground regions is much higher than that of the background regions.

The block entropy value, H_s, and the entropy, H_s^g, of the gradient image of the fingerprint are used as the first two features, v_s^1 and v_s^2.

2.2 Further feature extraction

To achieve a more accurate segmentation result, two features are sometimes insufficient. So, five other features are used.

Coherence: Coherence is a measure of gradient consistency [3]. The block coherence is defined as

$$Coh = \frac{\sqrt{(g_{xx} + g_{yy})^2 + 4g_{xy}^2}}{g_{xx} + g_{yy}}, \tag{8}$$

where $g_{xx} = \sum_{s'} g_x^2$, $g_{yy} = \sum_{s'} g_y^2$, and $g_{xy} = \sum_{s'} g_x g_y$, g_x, g_y are the gradients value at site s' in a horizontal and vertical direction, respectively, and $s' \in b_s$.

The coherence feature Coh is assigned to v_s^3 as one component of the feature vector v_s.

Mean and Variance: The mean and variance of block b_s is defined as

$$Mean = \frac{1}{\omega \times \omega} \sum_{s' \in b_s} I_{s'}, \tag{9}$$

$$Var = \sum_{s' \in b_s} (I_{s'} - Mean)^2. \tag{10}$$

Mean and variance values are assigned to v_s^4 and v_s^5, respectively, as two components of the feature vector, v_s.

Gabor features: Gabor features were proposed in [9] as measures of fingerprint quality. The Gabor features of b_s are obtained as follows: h_θ is a Gabor filter of size $\omega \times \omega$ with angle θ, and deviations δ_x, δ_y along the x and y axis, respectively.

$$g_\theta = \sum_{s' \in b_s} b_s \cdot h_\theta, \tag{11}$$

where $\theta \in \left\{0, \frac{\pi}{m}, \frac{2\pi}{m}, \ldots, \frac{(m-1)\pi}{m}\right\}$, and m is a predetermined scalar. The mean and standard deviation value of g_θ are obtained by

$$M_g = \frac{1}{m} \sum_{\theta} g_\theta, \tag{12}$$

$$D_g = \frac{1}{m-1}\sum_{\theta}(g_\theta - M_g)^2. \quad (13)$$

For more details we refer the reader to [9]. M_g and D_g are assigned to v_s^6 and v_s^7 of v_s, respectively.

A single feature is usually not enough to segment a fingerprint image accurately, because one feature only represents one characteristic aspect of a fingerprint image. The above described 7 features can be divided into two categories: gray intensity-based features and texture based features. Block entropy (H_s), mean (*Mean*), and variance (*Var*) belong to the first category and the remaining four features are texture-based. They are not redundant, but instead, complementary to each other. A successful fingerprint segmentation algorithm should consider both the textual properties and gray intensity properties. We have experimented on all these features; it is difficult to obtain satisfying segmentation results with too few features. A fusion of multiple features into fingerprint segmentation is a potential solution to improving the accuracy of segmentation. In the following sections, an AdaBoost classifier based segmentation method is adopted to fuse all these features.

3 Classification by AdaBoost classifier

In this paper, we adopt the supervised classifier, AdaBoost, to segment the fingerprint image. The AdaBoost method, which was introduced in 1995 by Freund and Schapire [10], is used to determine each of the block; $+1$ or -1, respectively represents foreground and background. AdaBoost is an integrated method for pattern classification which allows designers to continue adding weak learners until some acceptable low training error has been achieved. In AdaBoost, each training pattern receives a weight that determines its probability of being selected for a training set as an individual component classifier. If a training pattern is accurately classified, then its chance of being used again in a subsequent component classifier is reduced; conversely, if a training pattern is not accurately classified, then its chance of being used again is raised. In this way, AdaBoost focuses on difficult patterns [11]. For more details, please refer to [12].

The training procedures of the AdaBoost algorithm are given as follows:

1) Initialization: $S = \{(v_1, c_1), (v_2, c_2),..., (v_n, c_n)\}$, $\omega_1(i) = \frac{1}{n}$, $i = 1, 2,..., n$, $k_{\max}$, $k = 0$;

2) Do $k = k + 1$;

3) Train weak learner C_k using a resampled set of S according to $\omega_k(i)$;

4) Apply C_k on S to get training error e_k, and computing $\alpha_k = \frac{1}{2}\ln\frac{1-e_k}{e_k}$;

5) Update weight vector

$$\omega_{k+1}(i) = \frac{\omega_k(i)}{Z_k} \times \begin{cases} e^{-\alpha_k}, \text{ if } h_k(v_i) = c_i, \\ e^{\alpha_k}, \text{ if } h_k(v_i) \neq c_i, \end{cases} \quad (14)$$

where Z_k is a normalizing constant computed to ensure that $\omega_k(i)$ represents a true distribution, and $h_k(v_i)$ is the category label($+1$ or -1) given to pattern v_i by component classifier C_k;

6) If $k = k_{\max}$, goto next step, otherwise goto step 2;

7) Combine all weak component classifiers

$$H(v_i) = \mathrm{Sgn}\left(\sum_{k=1}^{k_{\max}} \alpha_k h_k(v_i)\right), \quad (15)$$

where Sgn(α) is a sign function: when $\alpha > 0$, Sgn(α) = 1; otherwise 0.

4 Post processing

Because the segmentation procedures are performed block-wise, the boundaries of foreground regions have a block effect, and some small clusters may be produced. To get a smooth contour and more compact clusters, some post processing steps are necessary.

In this paper, the roughly segmented binary image, which is the output of AdaBoost classifier, is first processed by a morphology operation: an open operation is used to remove the small clusters that are incorrectly assigned to the foreground, and a close operation is used to remove the small clusters that are incorrectly assigned to the background. Then the binary images are processed by contour filtering in a complex Fourier transform domain [13] as follows. The foreground boundary is represented as a complex point sequence $q = \{(x_1 + i\cdot y_1), (x_2 + i\cdot y_2),..., (x_l + i\cdot y_l)\}$, where $\{(x_k, y_k) | k = 1,2,...,l\}$, are the coordinates of the boundary point and $i = \sqrt{-1}$. Taking a complex Fourier transform on the sequence q. The highest five coefficients are kept and the others are set to zero, then inverse Fourier transform is taken.

After post processing, a smooth contour can be obtained.

Fig. 3 shows two examples of the post processing results on FVC2000 database 3 and FVC2004 database 2.

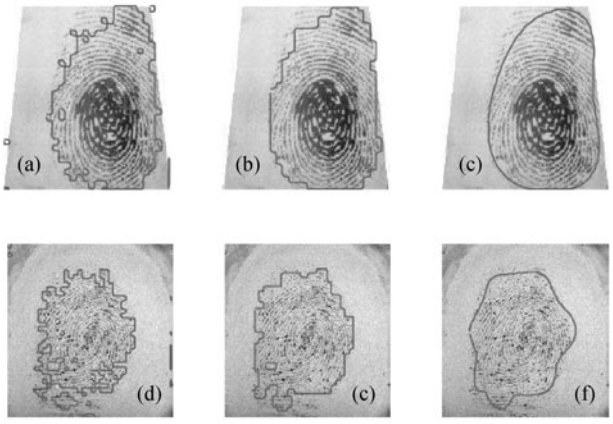

Fig. 3 Two examples of post processing. (a), (d) before post processing; (b), (e) the results after morphology operation; (c), (f) the results of contour smoothing

5 Experimental results

To evaluate the performance of our proposed algorithm, two performance measurements are used in our experiments. The first is *Err* [3], and the second is McNemar's test [14].

5.1 FVC databases

Our algorithm is tested on all databases of FVC2000 [1], FVC2002 [15] and FVC2004 [16]. The sensor type and resolution parameters of each database are shown in Table 1. The sensor types cover optical sensors, capacitive sensors, thermal sweeping sensors, and synthetic generators. For each database, we chose 20 images for training and testing, which include samples of good, normal, and bad quality. Five different training-testing ratios, {10:10, 8:12, 6:14, 4:16, 2:18}, of the 20 sample images are used to investigate the stability of the proposed segmentation method on different databases. For each training-testing ratio, the selection process of the training to testing samples is repeated 10 times to yield 10 different train-testing partitions. The proposed method is compared with the methods proposed in [3] and [4] on FVC databases. These sample images are divided into non-overlapping blocks of size 12×12. Each block's category is determined by manual inspection. Because the image size for different databases is labeled differently, the numbers of blocks used for training and testing are different.

Table 1 Parameters of the FVC databases [1,15,16]

		Sensor type	Resolution
FVC2000	DB1	Low-cost optical sensor	500 dpi
	DB2	Low-cost capacitive sensor	500 dpi
	DB3	Optical sensor	500 dpi
	DB4	Synthetic generator	≈500 dpi
FVC2002	DB1	Optical sensor	500 dpi
	DB2	Optical sensor	569 dpi
	DB3	Capacitive sensor	500 dpi
	DB4	SFinGe v2.51	≈500 dpi
FVC2004	DB1	Optical sensor	500 dpi
	DB2	Optical sensor	500 dpi
	DB3	Thermal sweeping sensor	512 dpi
	DB4	SFinGe v3.0	≈500 dpi

5.2 Comparison of fusion on decision level and feature level

There are two strategies for performing feature fusion: 1) feature level; 2) decision level. In our experiments, we investigate the feature fusion on both levels. Feature level fusion is performed by an AdaBoost classifier. And the decision level fusion is performed as follows: first, the segmentation procedure is performed on each of the extracted seven features. Then a majority of the features determine a block to a background (or foreground) region, the block is marked as a background (or foreground) region. Fig. 4 shows the segmentation results based on single feature, decision level fusion, and AdaBoost. We reach two conclusions from Fig. 4: 1) single feature based segmentation is not sufficient to obtain satisfying segmentation results; 2) feature level fusion is superior to decision level fusion.

5.3 Error rate classification

Each test, that runs the program on a large set of input data whose correct outputs are known, can field four possible results. In this paper, the correct outputs of the input data are marked manually. They are presented as *true positive*, *true negative*, *false positive* ($e1$), *false negative* ($e2$). So the total error rate of classification, represented as *Err*, includes both *false positive* ($e1$) and *false negative* ($e2$). *Err* is defined as

$$Err = \frac{Num_{e1} + Num_{e2}}{Num_{all}}, \tag{16}$$

where Num_{e1} indicates the total number of background blocks which are misclassified as foreground blocks,

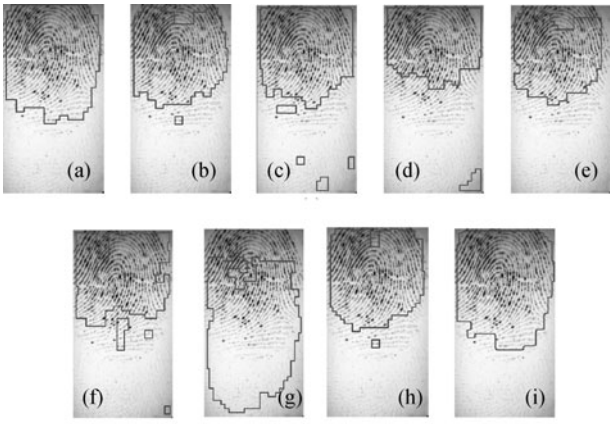

Fig. 4 Segmentation results by (a) entropy feature; (b) gradient-entropy feature; (c) coherence feature; (d) mean feature; (e) variance feature; (f) Gabor mean feature; (g) Gabor standard deviation feature; (h) majority voting; (i) AdaBoost classifier

Num_{e2} indicates the total number of foreground blocks which are misclassified as background blocks, Num_{all} indicates the total numbers of both foreground and background blocks.

In this paper, the classification error rate, *Err*, is the shown the average and standard deviation of 10 experimental results for each training-testing ratio, on each database, so that we can investigate the stability of the proposed method. The *Err* of the proposed method is compared with methods proposed in [3] and [4] on FVC databases. The Classification results are plotted in Figs. 5–7, which show both average *Err* and its standard deviation. The length of vertical solid lines on the *Err* curves denote the values of standard deviation of *Err* at the corresponding training-testing ratio points (we conduct 10 experiments at each training-testing ratio point).

Results on FVC2000: From Fig. 5(a), we see that our algorithm has lower *Err* on DB1 and all three algorithms have high stability which can be explained by the high image quality of DB1. In Fig. 5(b) we see for DB2, when the training-testing ratio is higher, our algorithm performs better than the others, when training-testing ratio is below 4:16, our algorithm has a higher *Err*. In Fig. 5(c) for DB3, our algorithm's performance curve intersects the Chen curve but is superior to the Bazen method. In Fig. 5(d) on DB4, our algorithm performs better with a lower training-

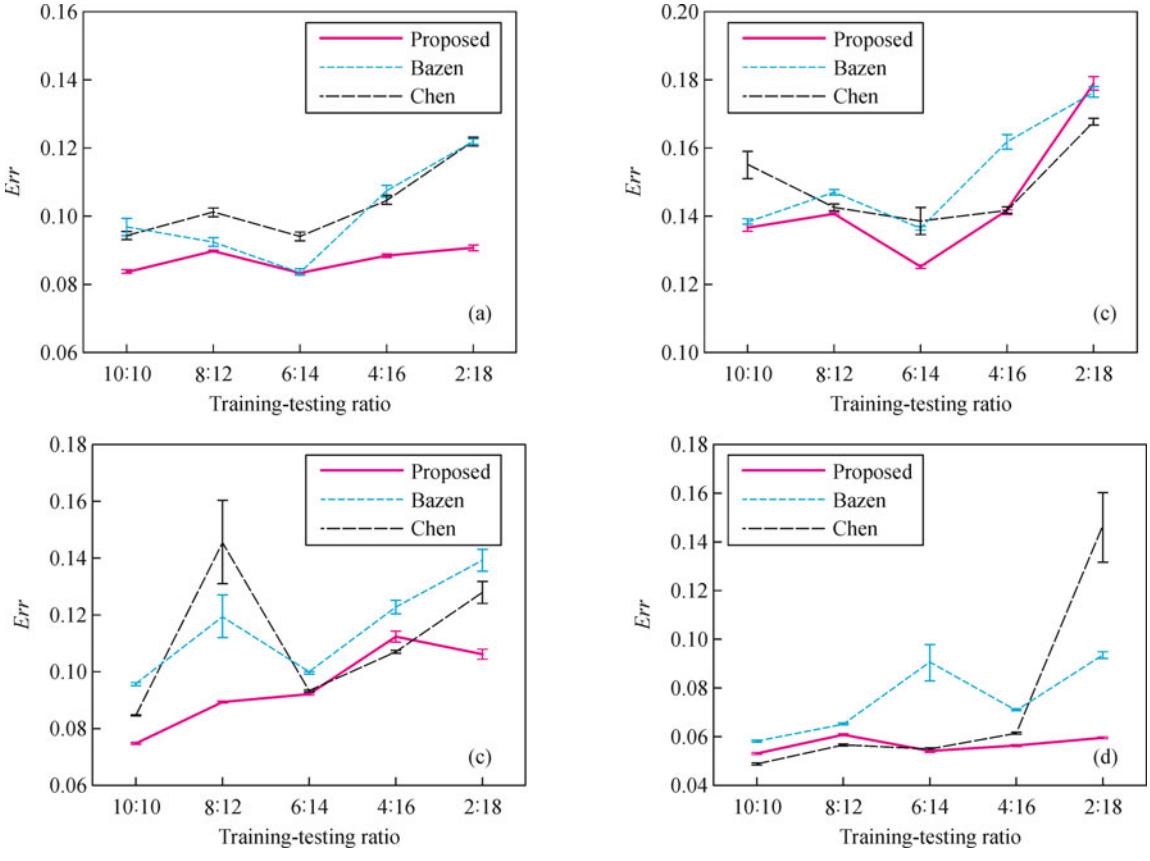

Fig. 5 *Err*(s) of FVC2000 for (a) DB1; (b) DB2; (c) DB3; (d) DB4

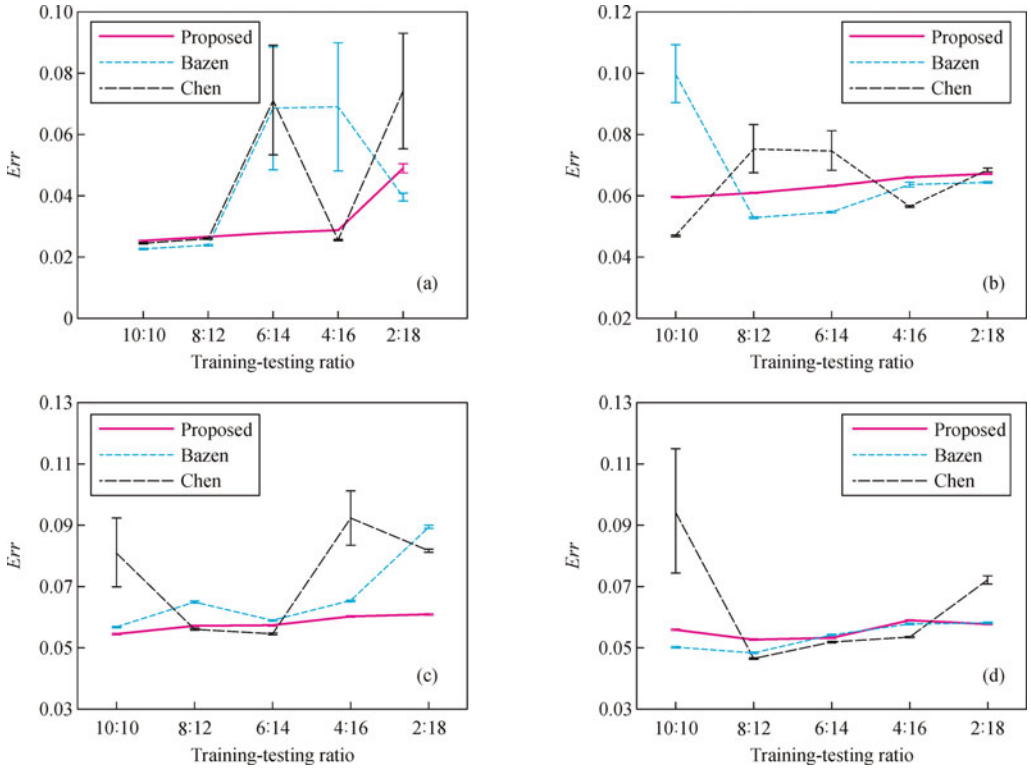

Fig. 6 *Err*(s) of FVC2002 for (a) DB1; (b) DB2; (c) DB3; (d) DB4

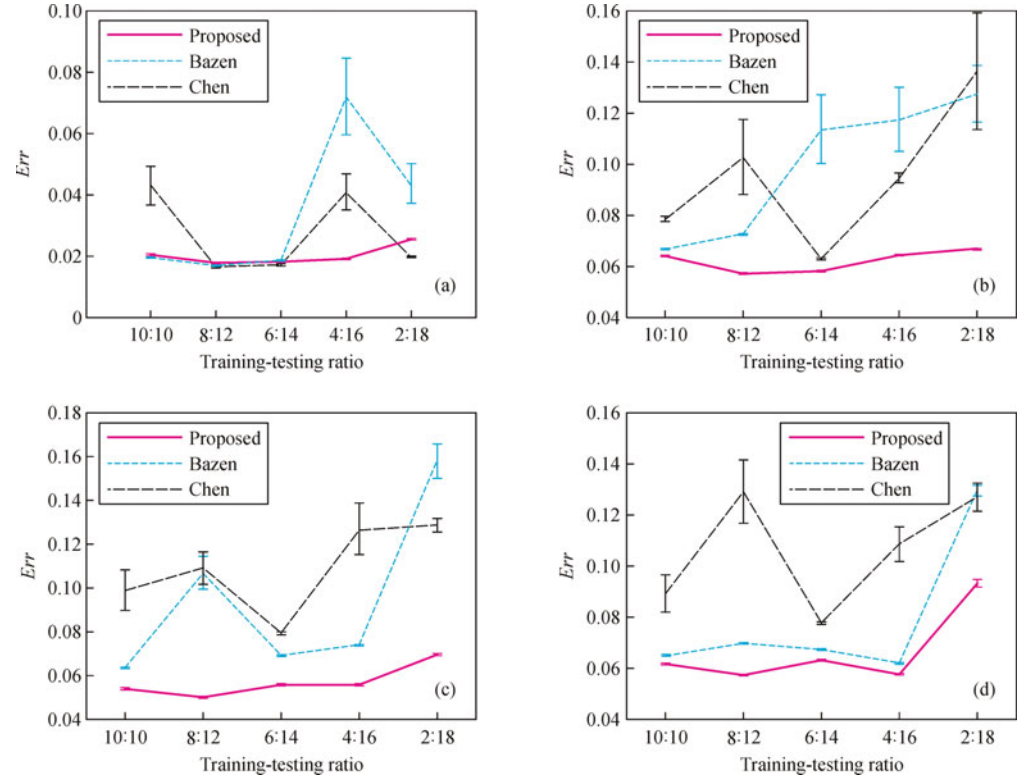

Fig. 7 *Err*(s) of FVC2004 for (a) DB1; (b) DB2; (c) DB3; (d) DB4

testing ratio than the others, but a higher *Err* than the Chen method when the training-testing ratio greater than 6:14.

Results on FVC2002: From Fig. 6(a) on DB1, we see that our algorithm has a higher stability than both the Chen and Bazen methods when changing the training-testing ratio. In Fig. 6(b) on DB2 we see that the Bazen method performs better than our algorithm, while the performance curve of the Chen method intersects with our algorithm. The stability of our algorithm is better than both Bazen's and Chen's method. In Fig. 6(c) on DB3, we can see that our algorithm outperforms the Bazen method, and that the Chen method has very low stability. Fig. 6(d) on DB4, we see both Bazen and Chen methods outperform our proposed algorithm. However, our algorithm maintains a higher stability.

Results on FVC2004: In Fig. 7(a) on DB1, we see that the stability of both the Bazen and Chen methods decrease when fewer training samples are used whereas our algorithm remains stable under different training-testing ratios. In Fig. 7(b) on DB2, our algorithm outperforms both the Bazen and Chen method with respect to *Err* curves and standard deviations. The same conclusions can be drawn for DB3 and DB4 from Figs. 7(c) and 7(d) respectively.

Based on the above analysis, we conclude that our algorithm is more stable under a variety of training-testing ratios and sensors. On some databases (such as FVC2000: DB2, DB4 and FVC2002: DB2, DB4), the *Err* curves intersect, so that we cannot judge which method is better on those particular databases. In this way, we use the McNemar's test to comprehensively evaluate the algorithmic performances and determine out whether our proposed method is better than others.

5.4 McNemar's test

McNemar's test is a first-order check on the statistical significance of an observed difference in recognition performance. It is a paired success/failure trial using the binomial. Considering Table 2 as results for two algorithms, the McNemar's test is

$$\chi^2 = \frac{(|N_{sf} - N_{fs}| - 1)^2}{(N_{sf} + N_{fs})}. \tag{17}$$

If the number of tests is greater than 30 or so, then the central limit theorem applies. In such a case, the Z score (standard score) is obtained from Eq. (17) as

$$Z = \frac{|N_{sf} - N_{fs}| - 1}{\sqrt{N_{sf} + N_{fs}}}. \tag{18}$$

Table 2 Test results for two algorithms (A and B)

	A succeeded	A failed
B succeeded	N_{ss}	N_{sf}
B failed	N_{fs}	N_{ff}

If algorithm A and algorithm B give very similar results, then $Z \to 0$. With the difference increasing, Z will increase. Confidence limits can be obtained from the Z-value by looking up on standard tables. More information can be obtained by observation of N_{sf} and N_{fs}. If $N_{sf} > N_{fs}$, then we can say that algorithm A outperforms algorithm B.

In this paper, the proposed method (represented as algorithm A) is compared with that of [3] and [4] (represented as algorithm B) by means of McNemar's test on FVC databases. For each database, the test results of the 10 different partitions with training-testing ratios from 10:10 to 2:18 are summed to obtain the statistical evaluation for each database. The performance comparison results are shown in Tables 3–5 in which the meanings of N_{ss}, N_{sf}, N_{fs}, and N_{ff} can be found in Table 2. The Z-values which are marked with asterisks (*) indicate $N_{sf} > N_{fs}$.

In Tables 3, 4 and 5, the first number in columns 3, 4, 5 and 6 corresponds to row N_{ss}. N_{sf} denotes the number of blocks that both methods (A and B) succeed in classifying (N_{ss}). The second number in the column denotes the number of blocks that A method successfully classifies but method B fails to classify (N_{sf}), where A is the proposed method and B is the compared method.

We make the following observations on our results:

1) Comparison with the method from [3]: From Tables 3–5 we can see that on the test databases, our algorithm is superior to [3] with higher confidence in all bar FVC2002 DB4. On FVC2002 DB4, the algorithm in [3] is superior to our algorithm with a 100% confidence limit[1)]. These results are consistent with *Errs*.

2) Comparison with the method from [4]: Our algorithm is absolutely superior to [4] with a confidence of 100%.

From the above comparisons, we can state that, our algorithm performs well through most of the tested databases by means of classification error rate and

1) The main reason that we get such a high confidence limit is because we have a large number of test samples. Each block is a test sample and each test image can be partitioned into a lot of such blocks.

Table 3 Comparison of proposed algorithm with [3] and [4] on FVC2000 databases

		DB1	DB2	DB3	DB4
proposed vs [3]	N_{ss}, N_{sf}	379584, 13493	384279, 19479	906367, 32412	344643, 4179
	N_{fs}, N_{ff}	19904, 24978	23085, 51047	54118, 71506	12147, 17541
	Z-value	35.0755	17.4737	73.7864	62.3527
	confidence	100%	100%	100%	100%
proposed vs [4]	N_{ss}, N_{sf}	377891, 14020	381987, 24094	905953, 37859	343360, 5018
	N_{fs}, N_{ff}	21590, 24458	25382, 46434	54563, 66069	13426, 16723
	Z-value	40.1100	5.7860	54.9423	61.9033
	confidence	100%	100%	100%	100%

Table 4 Comparison of proposed algorithm with [3] and [4] on FVC2002 databases

		DB1	DB2	DB3	DB4
proposed vs [3]	N_{ss}, N_{sf}	693619, 11229	753707, 15478	400977, 6473	501607, 6964
	$N_{fs}N_{ff}$	21762, 13094	16573, 37360	11064, 19189	6144, 23372
	Z-value	57.9847	6.1108	34.6605	7.1535*
	confidence	100%	100%	100%	100%
proposed vs [4]	N_{ss}, N_{sf}	693162, 11618	751748, 17600	397466, 7650	496397, 7802
	N_{fs}, N_{ff}	22208, 12724	18532, 35232	14591, 18002	11344, 22565
	Z-value	57.5744	4.8978	46.5353	25.5610
	confidence	100%	100%	100%	100%

Table 5 Comparison of proposed algorithm with [3] and [4] on FVC2004 databases

		DB1	DB2	DB3	DB4
proposed vs [3]	N_{ss}, N_{sf}	1446598, 9815	534474, 10088	619718, 11585	488083, 5688
	N_{fs}, N_{ff}	34674, 21220	35531, 28013	39887, 29197	13120, 31234
	Z-value	117.8528	119.1183	124.7430	54.1846
	confidence	100%	100%	100%	100%
proposed vs [4]	N_{ss}, N_{sf}	1460410, 11390	537737, 10666	609469, 13425	469098, 10840
	N_{fs}, N_{ff}	20889, 19674	32296, 27457	50118, 27353	32081, 26063
	Z-value	52.8655	104.3504	145.5585	102.5226
	confidence	100%	100%	100%	100%

McNemar's test. While the algorithm proposed in [3] performs well just on FVC2002 DB4 which is a synthetic fingerprint database. So, for statistical evaluation, the proposed method is superior to [3] and [4] on real fingerprint databases.

6 Conclusions

In this paper, a systematic fingerprint segmentation algorithm is presented. Firstly, two new entropy based fingerprint foreground feature representations are extracted. These features distinguish the foreground and background blocks by measuring their average information. Secondly, an AdaBoost classifier is designed to decide the type of each block. Our algorithm is tested on the FVC2000, FVC2002, and FVC2004 databases, and compared with the methods proposed in [3] and [4]. The comparison results based on classification error rate and McNemar's test show that our proposed algorithm has the best performance.

Though our algorithm performs well on the FVC databases, some problems still exist. The definition of background and foreground regions is ambiguous and does not have a standard criterion. In our opinion, blocks that can be repaired by the enhancement procedures should be treated as foreground blocks, and those that cannot, as background blocks. In our method, the criteria of classification are only based on visual inspection,

which is not objective in all situations. Some blocks treated as background by our method may still possibly be repaired by a subsequent enhancement algorithm. In our future works, we will research how to create more objective criteria to guide segmentation algorithms.

Acknowledgements This paper was supported by the National High Technology Research and Development Program of China (2008AA01Z411), the National Natural Science Foundation of China (Grant Nos. 60902083, 60803151, and 60875018), the Beijing Natural Science Fund (4091004), and the Fundamental Research Funds for the Central Universities (K50510100003).

References

1. Maio D, Maltoni D, Cappelli R, Wayman J L, Jain A K. FVC2000: fingerprint verification competition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2002, 24(3): 402–412
2. Hong L, Wan Y F, Jain A K. Fingerprint image enhancement: algorithm and performance evaluation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998, 20(8): 777–789
3. Bazen A M, Gerez S H. Segmentation of fingerprint images. In: Proceedings of ProRISC 2001 Workshop on Circuits, Systems and Signal Processing. 2001, 276–280
4. Chen X J, Tian J, Cheng J G, Yang X. Segmentation of fingerprint images using linear classifier. EURASIP Journal on Applied Signal Processing, 2004, 2004(4): 480–494
5. Zhan X S, Sun Z C, Yin Y L, Chen Y. A method based on the Markov chain Monte Carlo for fingerprint image segmentation. In: Proceedings of 2nd International Conference on Fuzzy Systems and Knowledge Discovery. 2005, 240–248
6. Houlding D, Vaisey J. Low entropy image pyramids for efficient lossless coding. IEEE Transactions on Image Processing, 1995, 4 (8): 1150–1153
7. Wallace K D, Marsh J N, Baldwin S L, Connolly A M, Keeling R, Lanza G M, Wickline S A, Hughes M S. Sensitive ultrasonic delineation of steroid treatment in living dystrophic mice with energy-based and entropy-based radio frequency signal processing. IEEE Transactions on Ultrasonics, Ferroelectrics and Frequency Control, 2007, 54(11): 2291–2299
8. Battle D J, Harrison R P, Hedley M. Maximum entropy image reconstruction from sparsely sampled coherent field data. IEEE Transactions on Image Processing, 1997, 6(8): 1139–1147
9. Shen L L, Kot A C, Koo W M. Quality measures of fingerprint images. In: Proceedings of 3rd International Conference on Audio- and Video-Based Biometric Person. 2001, 266–271
10. Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 1997, 55(1): 119–139
11. Duda R O, Hart P E, Stork D G. Pattern Classification. 2nd ed. New York: John Wily & Sons, Inc, 2000
12. Freund Y, Schapire R E. A short introduction to boosting. Journal of Japanese Society for Artificial Intelligence, 1999, 14(5): 771–780
13. Zheng X L, Wang Y S, Zhao X Y. Fingerprint image segmentation using active contour model. In: Proceedings of 4th International Conference on Image and Graphics. 2007, 437–441
14. Clark A F, Clark C. Performance characterization in computer vision: a tutorial. http://citeseerx.ist.psu.edu/viewdoc/download?
15. Maio D, Maltoni D, Cappelli R, Wayman J L, Jain A K. FVC2002: second fingerprint verification competition. In: Proceedings of 16th International Conference on Pattern Recognition. 2002, 811–814
16. Maio D, Maltoni D, Cappelli R, Wayman J L, Jain A K. FVC2004: Third fingerprint verification competition. In: Proceedings of 1st International Conference on Biometric Authentication. 2004, 1–7