Latent Fingerprint Recognition: Role of Texture Template

Kai Cao and Anil K. Jain
Department of Computer Science and Engineering
Michigan State University, East Lansing, Michigan 48824
Email: {kaicao,jain}@cse.msu.edu

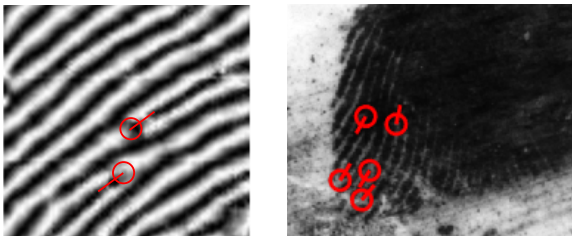
Abstract

We propose a texture template approach, consisting of a set of virtual minutiae, to improve latent fingerprint recognition accuracy. To compensate for the lack of a sufficient number of minutiae in poor quality latent prints, we generate a set of virtual minutiae. However, due to a large number of these regularly placed virtual minutiae, texture based template matching has a large computational requirement compared to matching true minutiae templates. To improve both the accuracy and efficiency of the texture template matching, we investigate: i) both original and enhanced fingerprint patches for training convolutional neural networks (ConvNets) to improve the distinctiveness of descriptors associated with each virtual minutiae, ii) smaller patches around virtual minutiae and a fast ConvNet architecture to speed up descriptor extraction, iii) reducing the descriptor length, iv) a modified hierarchical graph matching strategy to improve the matching speed, and v) extraction of multiple texture templates to boost the performance. Experiments on NIST SD27 latent database show that the above strategies can improve the matching speed from 11 ms (24 threads) per comparison (between a latent and a reference print) to only 7.7 ms (single thread) per comparison while improving the rank-1 accuracy by 8.9% against a gallery of 10K rolled prints.

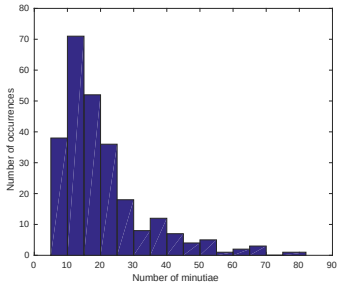
1. Introduction

Ever since minutiae were introduced for comparing fingerprints in 1888 by Sir Francis Galton [7], minutiae have been considered as the foundation for the science of fingerprint identification, which has expanded and transitioned to a wide variety of applications for person recognition over the past century [14]. In fact, the first Automated Fingerprint Identification System (AFIS) launched by FBI in early 1970s only stored an acquired fingerprint's type and minutiae instead of its digital image because of the compact and efficient representation offered by minutiae[1]. Following decades of research and development, advances in processors, memory, and sensor designs, fingerprint recognition systems have now been deployed in a broad set of applications such as border control, employment background checks, secure facility access and national identity programs [14]. For example, Aadhaar[2], has the world's largest biometric ID system with an enrollment database that already exceeds 1.2 billion tenprints (along with corresponding irises and face photos) of supposedly unique individuals. All of these systems are still primarily based on minutiae based fingerprint matching algorithms.

However, minutiae based approaches may not be effec-



(a) (b)



(c)

Figure 1. Fingerprint texture for matching. (a) Only two minutiae in a fingerprint image (96× 96 at 500 ppi) acquired by a capacitive sensor embedded in a smartphone (provided by Goodix), (b) five manually marked minutiae in a latent fingerprint image from NIST SD27, and (c) histogram of the number of minutiae in latent fingerprints in NIST SD27 database (consisting of 258 operational latents).

[1] https://www.fbi.gov/file-repository/about-us-cjis-fingerprints_biometrics-biometric-center-of-excellences-fingerprint-recognition.pdf/view

[2] https://uidai.gov.in/about-uidai/about-uidai.html

978-1-5386-7180-1/18/$31.00 ©2018 IEEE

tive in some cases, for example, in poor quality latent fingerprints and fingerprint images captured by small area sensors in mobile phones. Latent fingerprints (latents) are one of the most important and widely used sources of evidence in law enforcement and forensic agencies worldwide [8]. Due to the unintentional deposition of the print by a subject, latents are typically of poor quality in terms of ridge clarity, large background noise and small friction ridge area. Hence, the number of minutiae in a latent may be very small, e.g., $\leq$10. Fig. 1 (c) shows the distribution of the number of manually marked minutiae on NIST SD27 latent database and Fig. 1 (b) shows a latent image with 5 manually marked minutiae overlaid on the image. Given this small number of minutiae, minutiae alone do not have enough information for latents. Another example where minutiae based matching does not work is for matching fingerprint images captured by the capacitive sensors embedded in smartphones. It is estimated that 67% of the smartphones in the world will have an embedded fingerprint sensor[3] by 2018. The size of these embedded fingerprint sensors (only 88×88 pixels for TocuhID[4]) are much smaller than the standalone sensors. Hence, the number of minutiae in these images is not sufficient as shown in Fig. 1 (a). For these reasons, accurate non-minutiae based (also called *texture based*) fingerprint matching algorithms are necessary. To our knowledge, all major latent AFIS vendors use texture templates.

A few non-minutiae based fingerprint matching algorithms have been proposed in literature. FingerCode by Jain et al. [12] uses a bank of Gabor filters to capture both the local and global details in a fingerprint. However, FingerCode relies on a reference point and its accuracy is much lower than minutiae based approaches. Some approaches based on keypoints, e.g., SIFT [16, 20] and AKAZE [15], have been proposed to generate dense keypoints for fingerprint matching. However, these keypoints as well as their descriptors are not sufficient to distinguish fingerprints from different fingers.

Deep learning based approaches have also been proposed for fingerprint recognition. Zhang et al. [22] proposed a deep learning based feature, called deep dense multi-level feature, for partial high resolution fingerprint recognition which achieved promising performance on their own database. However, their approach could not handle fingerprint rotation. Cao and Jain [4] proposed a virtual minutiae based approach for latent fingerprint recognition, where the virtual minutiae locations are determined by a raster scan with a stride of 16 pixels; the associated descriptors are obtained by three convolutional neural networks. Experimental results on two latent databases, NIST SD27 and WVU, showed that the recognition performance of the virtual minutiae based "texture template" when fused with two different true minutiae templates boosted the rank-1 accuracy from 58.5% to 64.7% against a 100K reference print gallery for NIST SD27 [4]. However, the virtual minutiae feature extractor and matcher were shown to be quite slow.

The objective of this paper is to improve both the accuracy and efficiency of virtual minutiae (texture template) based latent matching. The main contributions of this paper are as follows:

1. Reduced the average recognition time between a latent texture template and a rolled texture template from 11 ms (24 threads) to 7.7 ms (single thread);
2. Improved the rank-1 identification rate of the texture templates in [4] by 8.9% (from 59.3% to 68.2%) for a 10K gallery;
3. Boosted the rank-1 identification rate in [4] by 2.7% through fusion of the proposed three texture templates with three templates in [4] (from 75.6% to 78.3%) for a 10K gallery. This means that out of the 258 latents in NIST SD27, improvements in the texture template push 7 additional latents to rank 1.

2. Proposed Approach

In this section, we describe the proposed texture-based latent matching approach, including feature extraction and matching. Fig. 2 shows a flowchart of the proposed approach.

2.1. Virtual Minutiae Extraction

For both latents and reference prints, the texture template is similar to that in [4] and consists of locations, orientations and descriptors of virtual minutiae. We first describe the virtual minutiae localization and then discuss the associated descriptors.

For reference fingerprints which are typically of high quality, the region of interest (ROI) is segmented by magnitude of the gradient and the orientation fields with a block size of 16×16 pixels as in [6]. The locations of virtual minutiae are sampled by raster scanning with a stride of s and their orientations are the same as the orientations of their nearest blocks in the orientation field. The virtual minutiae close to the mask border are ignored. Fig. 3 shows the virtual minutiae on two rolled prints.

For latents, the same manually marked ROIs and automatically extracted ridge flow with a block size of 16×16 pixels as in [3] are used for virtual minutiae extraction. Suppose that (x, y) are the $x-$ and $y-$coordinates of a sampling point and θ denotes the orientation of the block which is closest to (x, y) in the ridge flow. Two virtual minutiae, i.e., (x, y, θ) and $(x, y, \theta + \pi)$, are created to handle the ambiguity in ridge orientation. Fig. 4 show virtual minutiae on two enhanced latents from NIST SD27 with $s = 32$.

2.2. Descriptors for Virtual Minutiae

A minutia descriptor contains attributes of the minutia based on the image characteristics in its neighborhood.

[3] https://www.statista.com/statistics/522058/global-smartphone-fingerprint-penetration/

[4] https://assets.documentcloud.org/documents/1302613/ios-security-guide-sept-2014.pdf

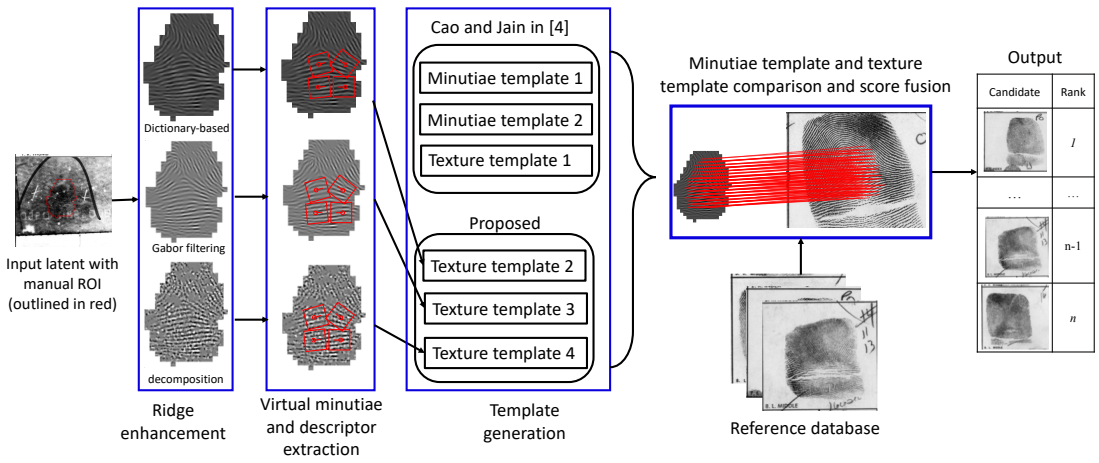
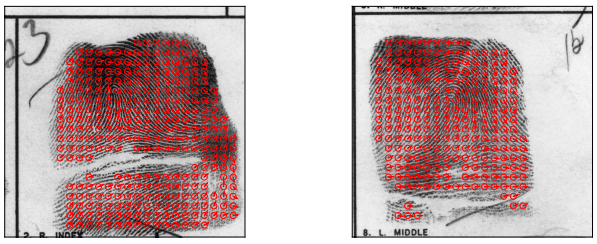
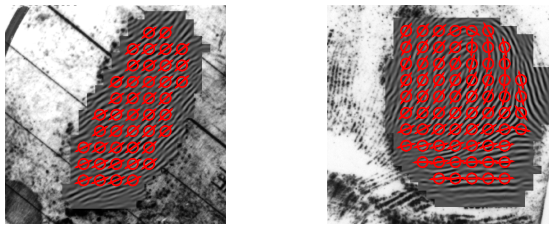


Figure 2. Overview of the proposed latent fingerprint recognition algorithm.



(a) (b)

Figure 3. Virtual minutiae on two rolled prints with stride s=32.

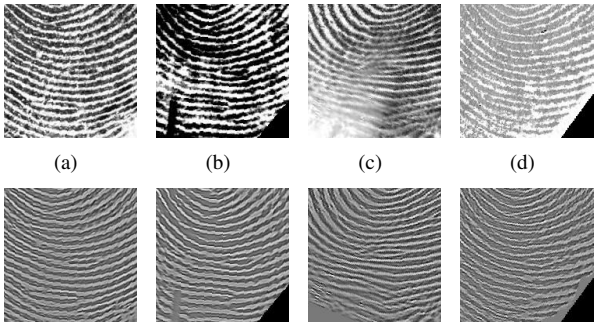


(a) (b)

Figure 4. Virtual minutiae on two enhanced latent fingerprints with stride s=32. Note that each circle represents two virtual minutiae with opposite orientations to handle the ambiguity in ridge orientation.

Salient descriptors are needed to establish minutiae correspondences and compute the similarity between a latent and reference prints. Instead of specifying the descriptor in an ad hoc manner, Cao and Jain [4] trained ConvNets to learn the descriptor from local fingerprint patches around a minutia and demonstrated its superior performance. In this paper, we improve both the distinctiveness of the descriptor and the efficiency of descriptor extraction.

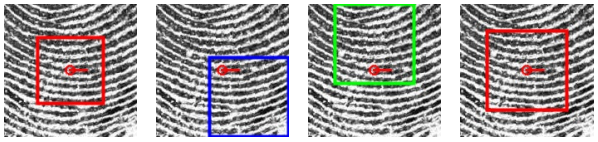
Cao and Jain [4] extracted approximately 800K 160 $\times$ 160 fingerprint patches from around 50K minutiae from images in a large longitudinal fingerprint database [21], to train the ConvNets. On average, there are around 16 fingerprint patches around each minutia for training. Cao and



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 5. Examples of training fingerprint patches.



(a) (b) (c) (d)

Figure 6. Four different patch types used in [4] for descriptor extraction. Patch types in (a)-(c) were determined to be the best combination in terms of identification accuracy. The patches in (b), (c) and (d) are of sizes 96 $\times$ 96 pixels while the patch in (a) is of size 80 $\times$ 80 pixels. In this paper, we used patch types in (b), (c) and (d) for descriptor extraction because all of them are of the same size and hence no resizing is needed.

Jain also reported that descriptors extracted from enhanced latent images give better performance. In order to augment the training dataset and improve the descriptor distinctiveness between enhanced latent and original rolled prints, we use patches from both the original and enhanced fingerprint patches for training ConvNets; this results in around 1.6 million fingerprint patches for training. Fig. 5 shows some example patches from the training dataset; Figs. 5 (a) to (d) are from the original image and (e) to (h) are from the corresponding enhanced images.

Location and size of the patches were also evaluated in

[4]. The three patch types shown in Figs. 6 (a)-(c) were determined to be the best patch types via forward sequential selection. The two patches in Figs. 6 (b) and (c) are of the same size (96×96 pixels) while the patch in Fig. 6 (a) is of size 80×80 pixels. All the patches had to be resized to 160×160 pixels in [4], by bilinear interpolation. In this work, to avoid resizing, we train a ConvNet on 96×96 images and use patch types in Figs. 6 (b)-(d) for descriptor extraction. Note that the patch in Fig. 6 (d) is minutiae centered whereas patches in Figs. 6 (b) and (c) are offset from the center minutiae.

Among the various ConvNet architectures [17], [18], [9], [11], MobileNet-v1 [11] uses depth-wise separable convolutions, resulting in a drastic reduction in model size and training/evaluation times while providing good recognition performance. The number of original model parameters to be trained in MobileNet-v1 (4.24M), is significantly smaller than the number of model parameters in Inception-v3 (23.2M) and VGG (138M), requiring significantly lower efforts in terms of regularization and data augmentation need, to prevent overfitting. For these reasons, we utilize the MobileNet-v1 architecture. In order to feed 96×96 fingerprint patches for training and to reduce the descriptor length, we make the following modifications to the MobileNet-v1 architecture: i) change the input image size to $96 \times 96 \times 1$, ii) remove the last two convolutional layers to accommodate the smaller input sizes, and iii) add a fully connected layer, also called the embedding layer, before the classifier layer to obtain a compact feature representation. The modified architecture is shown in Table 1.

For each of the three patch types shown in Figs. 6 (b)-(d), 1.6 million fingerprint patches are used to train a MobileNet. Given a fingerprint patch around a virtual minutiae, the output (l-dimensional feature vector) of the last fully connected layer is considered as the virtual minutiae descriptor. In the experiments, three value of l, namely $l = 32, 64$, and 128, are investigated. The concatenation of the three outputs for the same virtual minutiae is regarded as the descriptor with length $l_d = 3 \times l$. The set of virtual minutiae and the associated descriptors define a texture template.

2.3. Texture Template Matching

The algorithm for comparing two texture templates, one from latent and the other from a reference print, as proposed in [4] can be summarized as: i) compute pair-wise similarities between the latent and reference print virtual minutiae descriptors using cosine similarity; normalize the similarity matrix, ii) select the top N ($N = 200$) virtual minutiae correspondences based on the normalized similarity matrix, iii) remove false minutiae correspondences using second-order graph matching, iv) further remove false minutiae correspondences using third-order graph matching, and (v) finally compute the overall similarity between the two texture templates based on final minutiae correspondences. Although the second-order graph matching can remove most false correspondences, it is still time-consuming as it involves $N(N-1)/2$ computations. Furthermore, since the locations of virtual minutiae do not have large variations due to the raster scan, the larger complexity third order graph matching does not help too much in virtual minutiae correspondences.

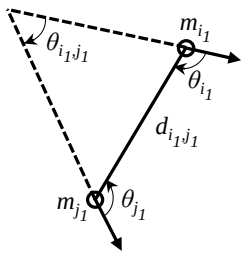
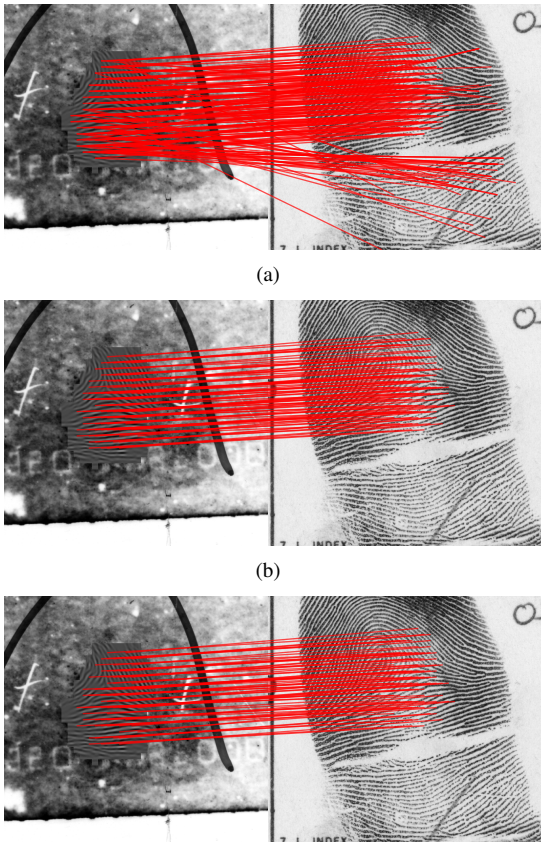


Figure 7. Illustration of second-order graph matching [4], where m_{i_1} and m_{j_1} are the two virtual minutiae, d_{i_1,j_1} is the Euclidean distance between m_{i_1} and m_{j_1}, and θ_{i_1,j_1}, θ_{i_1} and θ_{j_1} are the three angles formed by two virtual minutiae orientations and the line segment connecting m_{i_1} and m_{j_1}.

Table 1. MobileNet Architecture, where l is the length of feature vector output by the modified MobileNet and c is the number of classes used for training.

Type / Stride	Filter Shape	Input Size
Conv / s2	3×3×1×32	96×96×1
Conv dw / s1	3×3×32 dw	48×48×32
Conv / s1	1×1×32×64	48×48×32
Conv dw / s2	3×3×64 dw	48×48×64
Conv / s1	1×1×64×128	24×24×64
Conv dw / s1	3×3×128 dw	56×24×128
Conv / s1	1×1×128×128	24×24×128
Conv dw / s2	3×3×128 dw	24×24×128
Conv / s1	1×1×128×256	12×12×128
Conv dw / s1	3×3×256 dw	12×12×256
Conv / s1	1×1×256×256	12×12×256
Conv dw / s2	3×3×256 dw	12×12×256
Conv / s1	1×1×256×512	6×6×256
5×Conv dw / s1	3×3×512 dw	6×6×512
Conv / s1	1×1×512×512	6×6×512
Conv dw / s2	3×3×512 dw	6×6×512
Avg Pool / s1	Pool 6×6	6×6×512
FC / s1	512×l	1×1 × 512
FC / s1	$l \times c$	1×1 × l
Softmax / s1	Classifier	1 × 1 ×c

As shown in Fig. 7, the computation of the compatibility between two virtual minutiae pairs in second-order graph matching involves one Euclidean distance computation and three angular distance computations. In our preliminary experiments, we found that the Euclidean distance alone is good enough to remove most false correspondences



(a) (b) (c)

Figure 8. Illustration of virtual minutiae correspondences by the proposed graph matching strategy. Figures in (a), (b) and (c) show the top $N = 200$ minutiae correspondences, 88 virtual minutiae correspondences by using the modified second order graph matching and 73 virtual minutiae correspondences by applying the second order graph matching used in [4] to the 88 minutiae correspondences in (b).

as shown in Fig. 8. The approach we propose here is to use a modified second-order graph matching to remove most false virtual minutiae correspondences and use the second-order graph matching in [4] to get the final virtual minutiae correspondences.

An overview of the modified virtual minutiae matching algorithm is enumerated in **Algorithm 1**. The details of the modified second-order graph matching are as following. Suppose $\{i = (i_1, i_2)\}_{i=1}^{N}$ is the set of N selected minutiae correspondences between a latent L and a rolled print R, where i_1 and i_2 denote the i^{th} correspondence between the i_1^{th} and i_2^{th} virtual minutiae in the latent F_l and the rolled print F_r, respectively. Given two minutiae correspondences (i_1, i_2) and (j_1, j_2), $H_{i,j}^2$ ($H^2 \in R^{N \times N}$) measures the compatibility between (i_1, j_1) from the latent and (i_2, j_2) from the rolled prints:

$$H_{i,j}^2 = Z(D_{i,j}, \mu, \tau, t), \tag{1}$$

where $D_{i,j} = |d_{i_1,j_1} - d_{i_2,j_2}|$ and Z is a truncated sigmoid function:

$$Z(v, \mu, \tau, t) = \begin{cases} \frac{1}{1+e^{-\tau(v-\mu)}}, & \text{if } v \leq t, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Here μ, τ and t are the parameters of function Z.

The goal of graph matching is to find an N-dimensional correspondence vector Y, where the i^{th} element (Y_i) indicates whether i_1 is assigned to i_2 ($Y_i = 1$) or not ($Y_i = 0$). This can be represented in terms of maximizing the following objective function:

$$S_2(Y) = \sum_{i,j} H_{i,j}^2 Y_i Y_j. \tag{3}$$

A strategy of power iteration followed by discretization used in [4] is used to remove false minutiae correspondences.

Suppose $\{i = (i_1, i_2)\}_{i=1}^{n}$ represent the final n matched minutiae correspondences between F_l and F_r. The similarity S between F_l and F_r is defined as:

$$S = \sum_{i=1}^{n} DesSim(i_1, i_2), \tag{4}$$

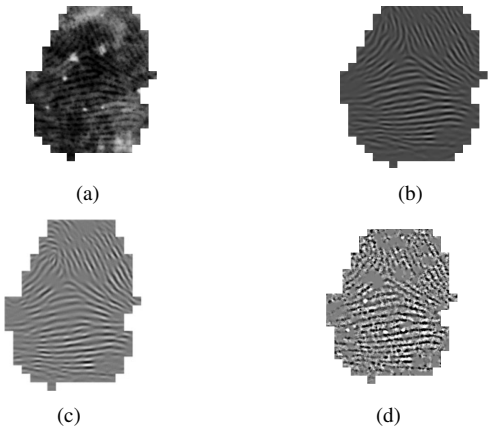
where $DesSim(i_1, i_2)$ is the descriptor similarity between i_1^{th} virtual minutiae in latent template F_l and i_2^{th} virtual minutiae in reference template F_r.

Algorithm 1 Modified virtual minutiae matching algorithm

1: **Input:** Latent template F_l and reference template F_r
2: **Output:** Similarity between F_l and F_r
3: Compute descriptor similarity matrix
4: Normalize similarity matrix
5: Select the top N minutiae correspondences based on the normalized similarity matrix
6: Construct H^2 based on these N minutiae correspondences
7: Remove false correspondences using modified second-order graph matching
8: Further remove false correspondences using original second-order graph matching
9: Compute similarity between F_l and F_r

2.4. Constructing texture templates

The locations and orientations of virtual minutiae are determined by the ROI and ridge flow, while the descriptors depend on the images input to the trained ConvNets. Three different processed latent images for each latent are investigated for texture template construction. Two different enhancement algorithms were proposed in [4] and the result-



(a) (b) (c) (d)

Figure 9. Illustration of processing strategies applied to the input latent shown in (a) for virtual minutiae descriptor extraction. (b) and (c) are the two enhanced latent fingerprint images used for minutiae template 1 and 2 extraction in [4], and (d) is the texture component after decomposition [5].

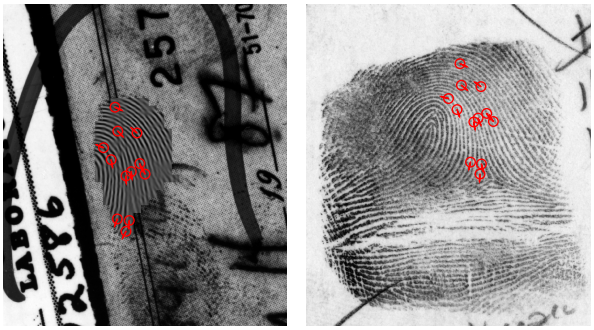
ing two minutiae sets were extracted, one per enhanced images. Both of the enhanced images for virtual minutiae descriptor extraction are also investigated here. However, the fingerprint enhancement performance critically depends on the estimates of ridge flow and ridge spacing. If ridge flow or ridge spacing are not estimated correctly, spurious ridge structures are created in the enhanced images. In addition to the two enhanced images, we also consider the texture image obtained by image decomposition [5], which essentially removes large scale background noise and enhances ridge contrast. Figs. 9 (b)-(d) illustrate the three processed images for the input latent image in Fig. 9 (a). For each latent, three different texture templates, T_{e_1}, T_{e_2} and T_t, can then be extracted. Given a latent to reference print pair, three texture template similarities are computed which are then fused to improve the overall latent recognition performance.

3. Experimental Results

The proposed texture template matching algorithm is evaluated on NIST SD27 latent database, which consists of 88 good quality, 85 bad quality and 85 ugly quality latent fingerprint images. For latent search experiments, 10,000 reference prints[5], including the 258 mates of NIST SD27 and others from NIST SD14, are used as the gallery.

3.1. Descriptor length evaluation

The performance of the new virtual minutiae descriptors with different feature lengths, namely, l_d = 96 ($l = 32$), 192 ($l = 64$) and 384 ($l = 128$), are evaluated by verification performance based on manually marked minutiae correspondences on NIST SD27[6] [1]. A total of 5,460 minutiae correspondences between the latent images and the mated rolled fingerprint images were provided with the NIST SD27 database. The average numbers of manually marked minutiae correspondences on 88 good quality, 85 bad quality and 85 ugly quality latent images are 31, 18 and 14, respectively. For a fair comparison with the descriptors used in [4], the descriptors for the latents are extracted on the same enhanced images as in [4] while the descriptors of the mated rolled prints are extracted on the original rolled prints. Fig. 10 shows the manually marked minutiae on an enhanced latent image and its mated rolled prints. The genuine scores are computed using the similarities of descriptors from the manually marked minutiae correspondences while the impostor scores are computed using the similarities of descriptors from the different minutiae. Thus, a total of 5,460 genuine scores and around 30 million impostor scores ($5,460 \times 5,460$) are computed. Fig. 11 compares the Receiver Operating Characteristic (ROC) curves of different descriptor lengths as well as the descriptors from [4] which used a descriptor length of 384. Note that the ROC curves of descriptors with feature lengths 384 and 192 are very close to each other, slightly better than descriptors with feature length 96 and significantly better than the descriptors in [4]. This can be explained by the use of both enhanced and original fingerprint patches along with a more appropriate ConvNet architecture for training.



(a) (b)

Figure 10. Manually marked minutiae correspondences are used to evaluate the proposed virtual minutiae descriptors. (a) Manually marked latent minutiae shown on the enhanced latent image and (b) manually marked rolled minutiae shown on mated rolled print image.

3.2. Texture template search

In this section, we evaluate the search performance of the proposed three texture templates, i.e., T_{e_1}, T_{e_2} and T_t extracted in section 2.4, as well as their fusion. The following seven scenarios are considered:

1. T_{e_1}: Texture template with enhanced image 1 (Fig. 9 (b)) for descriptor extraction;

[5]Results for the larger gallery of 100K reference prints are not yet available at the time of submitting this paper.

[6]NIST SD27 dataset is no longer available for download from the NIST site.

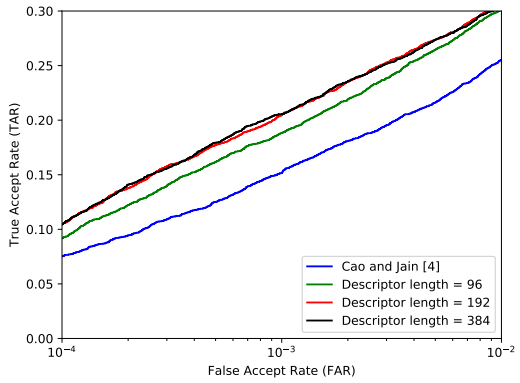


Figure 11. Receiver Operating Characteristic (ROC) curves under different descriptor lengths.

Table 2. Latent search accuracies under different scenarios for NIST SD27. Reference database size is 10K.

Input templates	descriptor length	rank-1 (%)	rank-5 (%)	rank-10 (%)
Cao&Jain [4]	384	59.30	70.16	73.26
T_{e_1}	192	68.22	73.64	74.81
T_{e_2}	192	66.67	72.48	74.42
T_t	192	60.47	67.83	70.93
T_{e_1}+T_{e_2}	192	**70.93**	74.81	77.91
T_{e_1}+T_t	192	**70.93**	76.36	79.07
T_{e_2}+T_t	192	67.05	75.19	77.13
T_{e_1}+T_{e_2}+T_t	192	70.16	**76.74**	**81.40**
T_{e_1}	384	69.38	75.58	77.13
T_{e_2}	384	66.28	72.48	73.64
T_t	384	58.91	66.28	69.77
T_{e_1}+T_{e_2}	384	70.16	75.58	78.29
T_{e_1}+T_t	384	69.38	76.74	77.91
T_{e_2}+T_t	384	67.83	74.03	75.97
T_{e_1}+T_{e_2}+T_t	384	**70.93**	75.97	78.68

2. T_{e_2}: Texture template with enhanced image 2 (Fig. 9 (c)) for descriptor extraction;
3. T_t: Texture template with texture image (Fig. 9 (d)) for descriptor extraction;
4. T_{e_1} + T_{e_2}: Score level fusion of T_{e_1} and T_{e_2};
5. T_{e_1} + T_t: Score level fusion of T_{e_1} and T_t;
6. T_{e_2} + T_t: Score level fusion of T_{e_2} and T_t;
7. T_{e_1} + T_{e_2} + T_t: Score level fusion of T_{e_1}, T_{e_2} and T_t.

For each one of the above seven scenarios, all three descriptor lengths, i.e., l_d = 96, 192 and 384, are considered. The latent search accuracies of different scenarios at three different ranks for NIST SD27 against 10K reference fingerprints are shown in Table 2. In addition, the performance of texture template used in [4] is also included for a comparison (row 1 of Table 2). Note that the enhanced images used for T_{e_1} extraction are the same as those in [4]. A descriptor length 96 performs much lower than the other two lengths (192 and 384) so its performance is not reported in Table 2. The findings from Table 2 can be summarized as follows: i) among the three texture templates, T_{e_1} performs the best for both descriptor lengths; ii) the performance of descriptor length of 192 is sufficiently close to that of descriptor length 384 in different scenarios; iii) rank-1 accuracy of T_{e_1} with descriptor length 192 is 8.98% higher than that in [4]; and iv) fusion of any two out of the three proposed texture templates is higher than that of any single template; the fusion of all three templates boosts the performance slightly at rank-1 (for l_d=192) and rank-5 (for l_d=384). Fusion of all three proposed templates with $l_d = 192$ is preferred because of its smaller feature length and higher accuracy at rank-10. As examiners typically evaluate top 10 candidate matches to identify the source of a latent [2], this will ensure that the true mate is frequently available in the candidate list.

3.3. Fusion with minutiae templates

In order to determine if our new texture templates can boost the performance over the results in [4] we fuse the proposed three texture templates with the three templates used in [4]. Fig. 12 compares the Cumulative Match Characteristic (CMC) curves of the fusion scheme on all 258 latents in NIST SD27 as well as subsets of latents of three different quality levels (good, bad and ugly). Plots in Fig. 12 show the proposed three texture templates when fused with the three templates in [4] can boost the overall performance by 2.7% at rank-1 (from 75.6% to 78.3%). In particular, the fusion of six templates (three proposed + three from [4]) improves the rank-1 accuracy by 4.7% on the subset of ugly latents, some of the most challenging latents with an average of only 5 minutiae per latent. Fig. 13 shows two ugly latents whose true mates were not retrieved at rank-1 by the method in [4], but are now correctly retrieved at rank-1 with the introduction of three new texture templates.

3.4. Computation time

The texture template matching algorithm was implemented in tensorflow and python and executed on a desktop with i7-6700K CPU@4.00GHz, GTX 1080 Ti (GPU), 32 GB RAM and Linux operating system. The average computation time for comparing a latent texture template to a rolled texture template is 7.7ms (single thread) compared to 11.0 ms (24 threads) in [4]. The average times for extracting one proposed latent texture template and one proposed rolled texture template are 0.7s and 1.5s (GPU), respectively; when fused with the three templates in [4] are 1.2s and 2.2s (24 threads).

4. Summary and future work

Texture template are critical in improving the search accuracy of latent fingerprints, especially for latents with

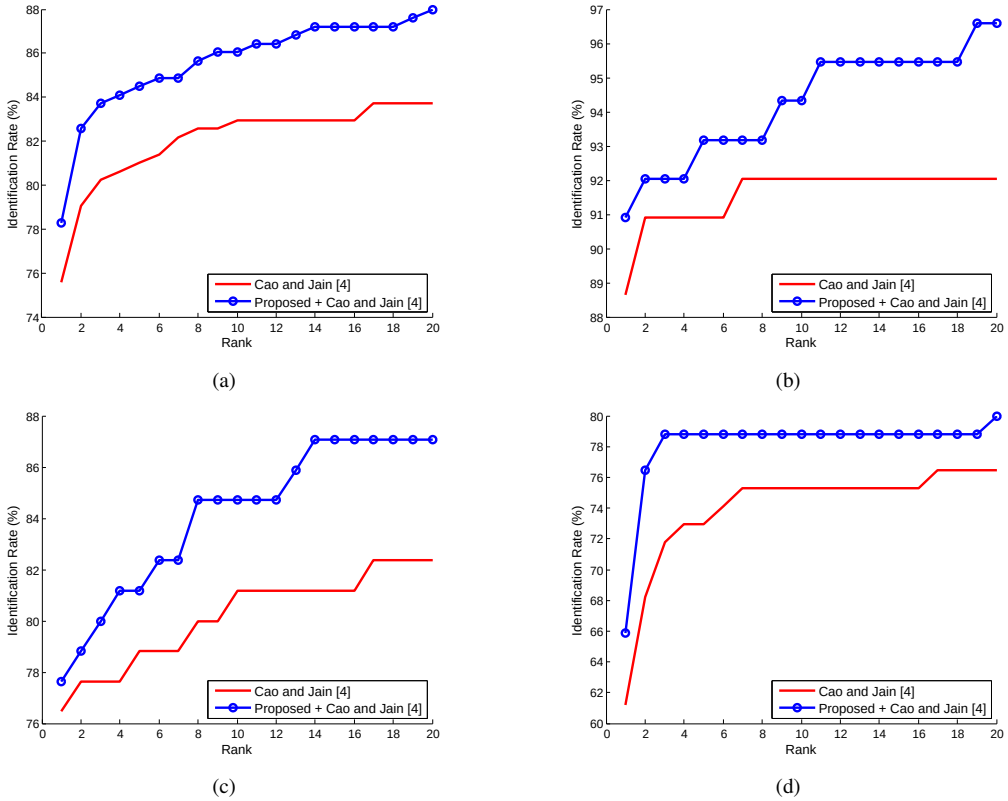
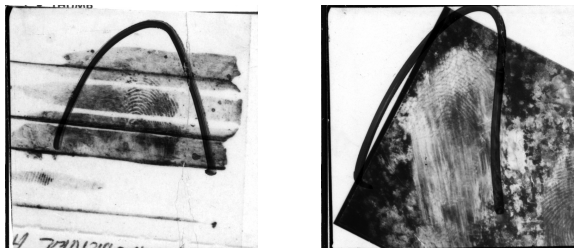


Figure 12. Cumulative Match Characteristic (CMC) curves of the three texture templates proposed here and their fusion with three templates used in [4] on (a) all 258 latents in NIST SD27, (b) subset of 88 good latents, (c) subset of 85 bad latents and (d) subset of 85 ugly latents. Note that the scales of the y-axis in these four plots are different to show the differences between the two curves.



(a) (b)

Figure 13. Two ugly latents whose true mates were not retrieved at rank 1 by the algorithm in [4]. The use of proposed texture templates and their fusion with templates in [4] correctly retrieves their true mates at rank 1.

small friction ridge area and large background noise. We have proposed a set of three texture templates, defined as a set of virtual minutiae along with their descriptors. Different virtual minutiae descriptors lead to different texture templates. The contributions of this paper are as follows. i) Use patches from original fingerprints and enhanced fingerprints to improve the distinctiveness of virtual minutiae descriptors, ii) three different texture templates, and iii) a modified second-order graph matching. Identification results on NIST SD27 latent database demonstrate that the proposed texture templates when used alone can improve the rank-1 accuracy by 8.9% (from 59.3% in [4] to 68.2%). Our ongoing research includes i) improving ridge flow estimation, ii) using latent-rolled pairs for learning minutiae descriptors and similarities, and iii) using multicore processors to improve the search speed.

Acknowledgement

This research is based upon work supported in part by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via IARPA R&D Contract No. 2018-18012900001. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

[1] NIST Special Database 27. `http://www.nist.gov/srd/nistsd27.cfm`.

[2] T. A. Busey and F. J. Parada. The nature of expertise in fingerprint examiners. *Psychonomic Bulletin & Review*, 17(2):155–160, Apr 2010.

[3] K. Cao and A. K. Jain. Latent orientation field estimation via convolutional neural network. In *International Conference on Biometrics*, pages 349–356, 2015.

[4] K. Cao and A. K. Jain. Automated latent fingerprint recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, DOI 10.1109/TPAMI.2018.2818162*, 2018.

[5] K. Cao, E. Liu, and A. Jain. Segmentation and enhancement of latent fingerprints: A coarse to fine ridgestructure dictionary. *IEEE TPAMI*, 36(9):1847–1859, Sept 2014.

[6] S. Chikkerur, A. N. Cartwright, and V. Govindaraju. Fingerprint enhancement using STFT analysis. *Pattern Recognition*, 40(1):198–211, 2007.

[7] F. Galton. *Finger Prints*. Macmillan, 2009.

[8] M. Hawthorne. *Fingerprints: Analysis and Understanding*. CRC Press, 2008.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[10] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. R. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath, and B. Kingsbury. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012.

[11] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *CoRR*, abs/1704.04861, 2017.

[12] A. Jain, S. Prabhakar, L. Hong, and S. Pankanti. Filterbank-based fingerprint matching. *IEEE Transactions on Image Processing*, 9(5):846–859, 2000.

[13] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105. 2012.

[14] D. Maltoni, D. Maio, A. Jain, and S. Prabhakar. *Handbook of Fingerprint Recognition (Second Edition)*. Springer, 2009.

[15] S. Mathur, A. Vjay, J. Shah, S. Das, and A. Malla. Methodology for partial fingerprint enrollment and authentication on mobile devices. In *2016 International Conference on Biometrics (ICB)*, pages 1–8, June 2016.

[16] U. Park, S. Pankanti, and A. K. Jain. Fingerprint verification using sift features. In *Defense and Security Symposium, Biometric Technologies for Human Identification, BTHI, Proc. SPIE*, 2008.

[17] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[18] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, 2014.

[19] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701–1708, 2014.

[20] M. Yamazaki, D. Li, T. Isshiki, and H. Kunieda. Sift-based algorithm for fingerprint authentication on smartphone. In *2015 6th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES)*, pages 1–5, March 2015.

[21] S. Yoon and A. K. Jain. Longitudinal study of fingerprint recognition. *Proceedings of the National Academy of Sciences*, 112(28):8555–8560, 2015.

[22] F. Zhang, S. Xin, and J. Feng. Deep dense multi-level feature for partial high-resolution fingerprint matching. In *2017 IEEE International Joint Conference on Biometrics (IJCB)*, pages 397–405, Oct 2017.