FV-GAN: Finger Vein Representation Using Generative Adversarial Networks

Wenming Yang, Changqing Hui, Zhiquan Chen, Jing-Hao Xue, Qingmin Liao

***Abstract*—In finger vein verification, the most important and challenging part is to robustly extract finger vein patterns from low-contrast infrared finger images with limited *a priori* knowledge. Although recent convolutional neural network (CNN)-based methods for finger vein verification have shown powerful capacity for feature representation and promising perspective in this area, they still have two critical issues to address. First, these CNN-based methods unexceptionally utilize fully connected layers, which restricts the size of finger vein images to process and increases the processing time. Second, the capacity of CNN for feature representation generally suffers from the low quality of finger vein ground-truth pattern maps for training, particularly due to outliers and vessel breaks. To address these issues, in this paper we propose a novel approach termed FV-GAN to finger vein extraction and verification, based on generative adversarial network (GAN), as the first attempt in this area. Unlike the CNN-based methods, FV-GAN learns from the joint distribution of finger vein images and pattern maps rather than the direct mapping between them, with the aim at achieving stronger robustness against outliers and vessel breaks. Moreover, FV-GAN adopts fully convolutional networks as the basic architecture, and discards fully connected layers, which relaxes the constraint on the input image size and reduces the computational expenditure for feature extraction. Furthermore, we design an adversarial training strategy and propose a hybrid loss function for FV-GAN. The experimental results on two public databases show a significant improvement by FV-GAN in finger vein verification in terms of both verification accuracy and equal error rate.**

***Index Terms*—Finger vein verification, pattern extraction, convolutional neural network (CNN), generative adversarial network (GAN), cycle-consistent adversarial network (CycleGAN), deep convolutional generative adversarial network (DCGAN), U-Net.**

I. INTRODUCTION

WITH the growing demand for secured systems, biometric identification has drawn increasing attention and become one of the most critical and challenging tasks in information security. For biometric identification, a number of biometric modalities have been researched and developed. These modalities can be generally grouped into two categories: 1) extrinsic modalities, such as face [1], iris [2] and palmprint [3]; and 2) intrinsic modalities, such as finger vein [4] and palm-vein [5]. Extrinsic modalities-based verification systems are vulnerable to faked input, causing some concerns on privacy and security in practice. In contrast, intrinsic modalities are hidden under the skin and are difficult to be faked; in particular, vein verification can provide better security and privacy for users.

This work was partly supported by the National Natural Science Foundation of China (No.61471216 and No.61771276), the National Key Research and Development Program of China (No.2016YFB0101001 and 2017YFC0112500) and the Special Foundation for the Development of Strategic Emerging Industries of Shenzhen (No.JCYJ20170817161845824, No.JCYJ20170307153940960 and No.JCYJ20150831192224146). (Corresponding author: Wenming Yang)

W. Yang, C. Hui, Z. Chen and Q. Liao are with the Shenzhen Key Lab. of Info. Sci &Tech / Shenzhen Engineering Lab. of IS&DCP, Department of Electronic Engineering / Graduate School at Shenzhen, Tsinghua University, China. E-mail: yangelwm@163.com

J.-H. Xue is with the Department of Statistical Science, University College London, UK.

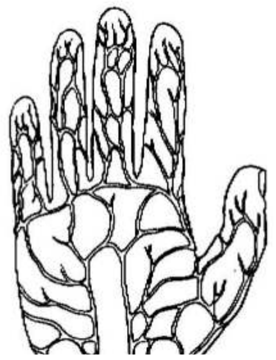


Fig. 1: Vein pattern in the palm and fingers [6].

A. Finger Vein Verification and Related Work

The veins and arteries are located under the skin. The finger vein pattern, extending from finger root to fingertip, shows a clear network and good connectivity (as shown in Fig.1). In general, the finger vein pattern is hardly observed in visible light because of the occlusion by fat and blood. Fortunately, since different components of skin have different responses to infrared light, the vein pattern can be observed under infrared light. In [7]–[10], infrared illumination with wavelength of 850 nm was employed to capture finger vein images.

Although finger vein verification has been investigated for many years, it is still a challenging task because the acquisition process is inherently affected by environmental illumination, ambient temperature and light scattering in imaging finger tissues and other factors. In general, noise and irregular shadow will ultimately compromise the performance of an authentication system. To alleviate the influence of these factors, a number of approaches have been proposed for extracting the vein pattern from finger vein images. They can be broadly classified into three categories:

1) Cross-sectional profile-based methods: In general, the cross-sectional profile is valley-shaped because the vein point has a lower gray value than its neighboring points on the profile. Some classical methods, including repeated line tracking (RLT) [4], region growth [11] and local maximum curvature point (LMC) [7], have been developed based on

this assumption. In repeated line tracking [4], the vein point is detected and tracked by the depth of the valley, and the vein pattern is then obtained by the binarization of the tracking times. Region growth [11] considers also the symmetry of the valley in vein point detection. Unlike these two approaches in which the gray value is used, LMC [7] calculates the local maximum curvature of the cross-sectional profile to extract the center line of the vein pattern; it is performed on four different directions, and the extracted center lines from different directions are fused by maximizing the curvature value on each pixel.

2) Neighborhood region-based methods: Wide line detector (WLD) [8], Gabor filters [9] and their improvements [12], [13] are the typical methods of this kind. These methods classify a point based on the information of its neighborhood region with a predefined radius. In WLD, the current pixel is labeled as a vein point if the number of its neighbors with higher gray values is larger than a predefined threshold. In [9], a set of self-similar Gabor filter templates is constructed and applied to a finger vein image for vein pattern extraction.
3) Whole image-based methods: Mean curvature [14] is a representative of the whole image-based methods. These methods view the finger vein image as a three-dimensional geometric shape and detects the valley-like vein pattern by the negative mean curvature.

These traditional methods extract finger vein patterns based on some attribute assumptions such as valleys and straight-lines. As a result, they suffer from the following problems:

A) Most methods perform poorly on low-quality images. Environmental illumination, variation of finger thickness and noise can lead to low image quality. Actually a valley shaped cross-sectional profile may not be detected in such images, in which case most methods cannot work well.
B) The vein pattern has been labeled by different thresholds, for example, tracking times, curvature value, and the number of special neighbors. However, it is difficult to determine these various thresholds. Take the threshold of tracking times in repeated line tracking as an example: In each tracking, the starting point is randomly selected from an image, which means that the points in the vein pattern and in the non-vein region may have similar probabilities of being selected. The tracking process may be ended by the point with a non-valley shaped cross-sectional profile, thereby increasing the tracking times. Consequently, a noise point in the non-vein area may have longer tracking times than a vein point, causing difficulty in threshold selection for binarization.
C) Some unique defects may occur in specific methods. For example, extracting vein patterns by the maximum curvature point brings gaps and burrs. The location of extracted vein line varies with the direction used in extraction, which also causes gaps and burrs in the fused vein lines.

To reduce such constrained attribute assumptions of the finger vein patterns and overcome the problems of these traditional methods, various approaches have been explored by researchers. In recent years, deep learning-based methods have been developed into finger vein verification due to their powerful capacity for feature representation. Radzi *et al.* [15] propose a 3-layers convolutional neural network (CNN) for classifying feature maps into 50 subjects and achieves 100% accuracy in finger vein biometric verification. Itqan *et al.* [16] develop a CNN-based finger vein verification system, which shows superior performance in finger vein datasets. Hyung *et al.* [17] present an effective finger vein verification system based on VGG-16, which achieves excellent results in realistic databases. Qin *et al.* [6] design a 5-layers CNN model for finger vein feature extraction and construct a fully convolutional network to recover pattern maps; their experimental results illustrate that the recovered pattern maps can be powerful in finger vein feature representation.

However, these CNN-based methods still have two critical issues. First, all of them utilize fully connected layers and learn the connection between a patch of finger vein image and the probability of its center to belong to a vein pattern, so there are some complex pre-processing and post-processing in their frameworks besides using CNN for vein feature extraction. It is difficult for these methods to be run in real time when extracting vein features. Second, the capacity of CNN for feature representation is restricted by the quality of the ground-truth pattern maps of finger vein images used in training. The wrong binary labels of foreground veins in pattern maps often occur due to outliers and vessel breaks, and cannot be recovered by these deep learning-based methods.

Due to these issues, the performance of above-mentioned CNN-based approaches has been limited for finger vein pattern extraction, which is still the most crucial and difficult step in a system. Hence it would be desirable to design a new deep learning-based method of finger vein feature extraction, which should have no fully connected layers and can exhibit robustness against wrong training labels of foreground veins. Moreover, the fully convolutional network (FCN) [18] can process the whole image in one forward propagation without any pre-processing or post-processing, which has the potential to reduce computational time for vein pattern extraction, and thus may act as the alternative to the fully connected layers.

B. GANs and Our Contributions

Recently, generative adversarial networks have been successfully applied to computer vision, including image style transfer [19], object generation [20] and image resolution [21]. In spite of their strong capacity for data (or feature) generation, there has been no research yet to bring them into finger vein verification. To the best of our knowledge, this paper makes the first attempt to accommodate GANs on finger vein verification and address the wrong label problem by learning from the joint distribution of finger vein images and pattern maps.

Generative adversarial networks (GANs) were proposed by Goodfellow *et al.* [22] in 2014. GANs are designed for training generative parametric models, and have been shown to be able to produce high-quality images [22], [23]. They contain two networks, a generator G and a discriminator D, as illustrated in Fig.2, where G maps a random vector z from the latent space to the image space while D classifies an input image as

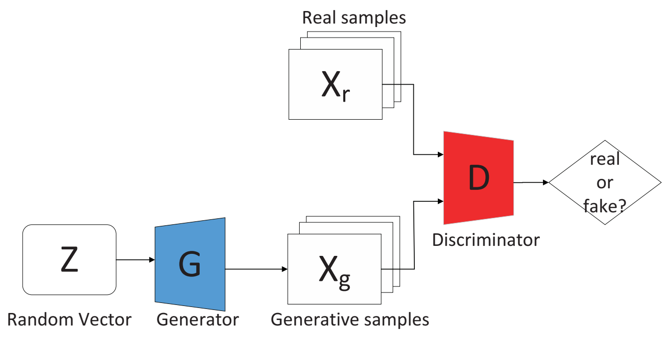


Fig. 2: Diagram of GANs.

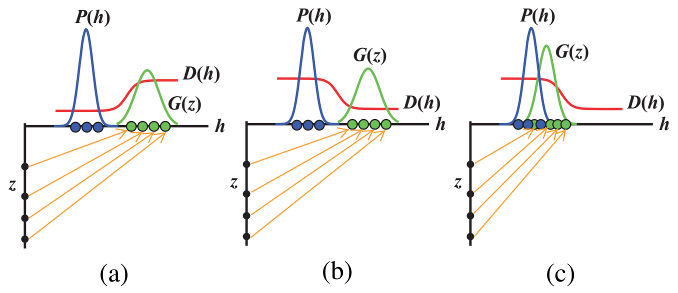


Fig. 3: GANs are trained by updating the discriminative distribution (D, red line) so that it discriminates between the samples from the real data distribution (blue line) p_z and those from the generated distribution (green line). The process from (a) to (c) shows the changes of these distributions with the training procedure going on.

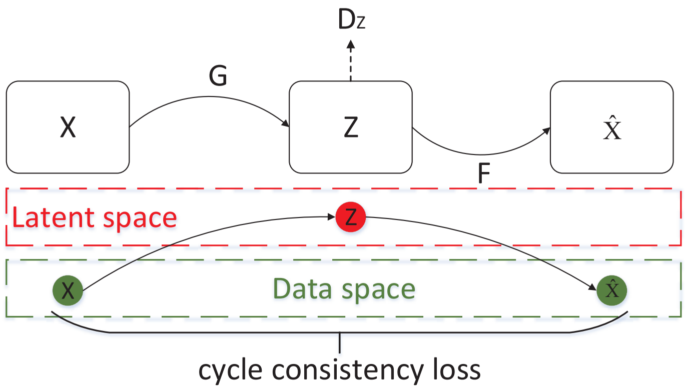


Fig. 4: A forward cycle of CycleGAN, which contains two generators: $G : X \to Z$ and $F : Z \to X$ and an adversarial discriminator D_z. The cycle consistency loss is forward as $X \to G(X) \to F(G(X)) \approx X$.

a real image or a fake image. The purpose of G is to generate realistic images, while the purpose of D is to discriminate between the image generated from G and the real image sampled from the data distribution P_{data}.

The G and D networks are trained by optimizing a loss function $V(G, D)$ as

$$\min_G \max_D V(G,D) = \mathbb{E}_{h\sim P_{data}(h)}[\log(D(h))] + \mathbb{E}_{z\sim P_Z(z)}[\log(1 - D(G(z)))], \tag{1}$$

where h is a sample from the P_{data} distribution, and z is a random vector in the latent space.

As shown in Fig.3, GANs search in the joint distribution of z and h for the best function to represent h using z. At the start of the training procedure shown in Fig.3(a), the generated distribution is different from the real data distribution, because the learned generator is not capable enough to represent h using z. After updating the discriminative distribution as in Fig.3(b), the space of the probable joint distribution is shrunk and an appropriately generated distribution is learned easily under the discriminative restriction as in Fig.3(c). Generally z represents *a priori* knowledge of the real data h and it is a Gaussian distribution if no *a priori* knowledge is known.

In finger vein verification, *a priori* knowledge can be obtained by labeling the finger vein features (as shown in Section II-B). Thus, z can be the labeled pattern in this paper: In our case, h represents a finger vein image from the image space, and z is a pattern map from the latent space. Furthermore, to make z learnable, we use z as the input of the discriminator, as illustrated in Fig.5. In the adversarial training between the generator and the discriminator, the best functions to represent h using z and represent z using h are both learned.

Although GANs are simple and effective generative methods, training GANs is difficult because of several reasons: firstly, it is difficult for GANs to converge and be trained stably; secondly, GANs often fall into 'mode collapse', in which the generator produces the same or similar images for different noise vectors z. Various extensions of GANs thus have been developed to improve the training stability.

Mirza *et al.* [24] extended GANs into a conditional model and proposed conditional generative adversarial networks (CGANs). In CGANs, the generator G and the discriminator D are conditioned on some extra information c, which provides additional controls on the generative results. Radford *et al.* [25] presented deep convolutional generative adversarial networks (DCGANs). They propose a number of architectural constrains of convolutional networks for both the generator and the discriminator. The architectural constraints include: 1) replacing all pooling layers with strided convolutions and fractional-strided convolutions; 2) using batchnorm layers; 3) removing fully connected hidden layers; 4) in the generator, using Tanh as the activation function and using the rectified linear units (ReLU) activation for other layers; and 5) in the discriminator, using the LeakyReLU activation in the discriminator for all layers. Experimental results show that DCGANs perform outstandingly in the training stability and are able to generate high-resolution images.

Besides exploring the network architecture and the training techniques of GANs, researchers also focus on developing appropriate loss functions of GAN. The log-likelihood loss function of original GAN leads to the vanishing gradient and the training instability. To overcome the vanishing gradient problem, Mao *et al.* [26] proposed least squares generative adversarial networks (LSGAN), which replace the log function by least squares function in the adversarial loss. Arjovsky *et al.* [27] proposed Wasserstein generative adversarial networks

(WGAN). They first theoretically show that the earth-mover distance produces better gradient behaviors in distributional learning compared to other distance metrics. Accordingly, they made several changes to regular GANs: 1) removing the sigmoid layer and adding weight clipping in the discriminator; and 2) replacing the log-likelihood function with the Wasserstein loss function in the adversarial loss. The Wasserstein loss function is shown as

$$WL(P_{data}, P_Z) = \max_{\|D\|_L \leq 1} \mathbb{E}_{h\sim P_{data}}[D(h)] - \mathbb{E}_{z\sim P_Z}[D(G(z))], \tag{2}$$

where $\|D\|_L \leq 1$ means that the discriminator represents the parameterized family of functions satisfying the 1-Lipschitz condition; h is a sample from the P_{data} distribution; and z is a random vector in the latent space.

Another inspiring progress on loss functions was achieved by CycleGAN [19], [28]. Compared with traditional CGAN-based methods, CycleGAN brings in a new cycle framework which couples up two GANs for adversarial training. Two generators G and F are designed for data translation: Generator G transforms images (from the data space) to feature maps (in the latent space), while Generator F transforms feature maps to images. Simultaneously two corresponding discriminators are designed for adversarial training. A complete CycleGAN contains two cycles: forward cycle and backward cycle. The backward cycle focuses on learning the mapping function from feature maps to images, and the forward cycle aims to learn the mapping function from images to feature maps as illustrated in Fig.4. Based on this attractive architecture, another powerful loss, namely cycle consistency loss, was proposed for reducing the space of mapping functions and accelerating the convergence of adversarial training. The cycle consistency loss is expressed as

$$\begin{aligned} V_{cyc}(G,F) =& \mathbb{E}_{x\sim P_{data}(x)}[\| F(G(x)) - x \|_1] + \\ & \mathbb{E}_{z\sim P_Z(z)}[\| G(F(z)) - z \|_1], \end{aligned} \tag{3}$$

where G and F are two generators; x is a sample from the data space and z is a sample from the latent space. Experimental results demonstrate that CycleGANs can generate images with comparable quality to well-designed DCGANs.

Inspired by DCGAN and CycleGAN, we propose in this paper a CycleGAN-based framework termed FV-GAN for finger vein verification. We design the network structure based on the architectural constraints of DCGANs and design the focal loss-based cycle consistency loss for our task. Main contributions of this paper can be summarized as follows:

1) This work makes the first attempt to accommodate GANs on the finger vein verification task. We propose a CycleGAN-based approach termed FV-GAN and design an appropriate loss function. In the FV-GAN framework (as shown in Fig.5), two generators (Image Generator and Pattern Generator) are designed as the major components for mapping between the finger-vein image space and the vein pattern space. The Image Generator is U-Net [29], which transforms vein patterns into vein images. Simultaneously an encoder-decoder network is utilized as the Pattern Generator, which maps vein images to vein patterns. A robust generator for pattern extraction can be achieved through adversarial training between the two generators and a discriminator in our framework.
2) The proposed FV-GAN adapts CycleGAN to finger vein extraction. Due to the lack of high-quality ground truth of foreground vein (i.e. the wrong label problem mentioned in Section I-A), CycleGAN cannot be utilized directly because accurate vein patterns are needed for the training of CycleGAN. Therefore, we develop an extra half cycle along with the forward cycle of CycleGAN in Fig.4, in which the generated fake vein images are input into the Pattern Generator again to obtain the second image of vein patterns, for the discriminator to distinguish the two vein pattern images, as illustrated in Fig.5.
3) The fully convolutional network (FCN) [18] is used as the basic architecture of the FV-GAN framework. Compared with other CNN-based methods such as DeepVein [17] and CNN+FCN [6] for vein pattern extraction, FV-GAN can predict binary vein patterns from vein images of any size in one forward propagation and transform a vein image into a pattern map directly without post-processing. The vein pattern extraction of our FV-GAN is time-saving, which enhances the practicality of finger vein verification.

The rest of the paper is organized as follows. Section II introduces the proposed FV-GAN framework. Section III shows the feature matching method for finger vein verification. The experimental results and analysis are given in Section IV. Finally, the paper is concluded in Section V.

II. Proposed FV-GAN framework

Traditional non-deep approaches extract vein patterns by assuming feature valleys and line segments. In this section, a new GAN called FV-GAN is presented to extract vein patterns without making any attribute distributional assumption.

A. The Proposed FV-GAN Framework

We present an adversarial training-based framework based on CycleGAN for finger vein verification (as shown in Fig.5). The training is implemented in the following steps. First, each pixel of a training image is initially labeled as either vein or background based on the combined results of several baseline methods. Then, for each patch on the labeled binary map, a corresponding patch in the raw ROI image is input into the Pattern Generator (an encoder-decoder network in the proposed FV-GAN) to extract patterns. Then, the extracted vein pattern is input into the Image Generator (a U-Net in the proposed FV-GAN) to generate a fake finger vein image. Then, we extract the vein pattern of the fake finger vein image through the Pattern Generator again. Finally, we compare the 'fake' vein pattern with the 'true' vein pattern in a Binary Discriminator to provide the loss. For testing, the Pattern Generator extracts finger vein patterns of a test image and outputs the probability of each pixel being a vein pattern. The resulting images of vein patterns and probabilities can be used for further vein verification.

The proposed FV-GAN is based on CycleGAN. Similar to CycleGAN, FV-GAN also contains two generators, namely Image Generator and Pattern Generator, which transform the

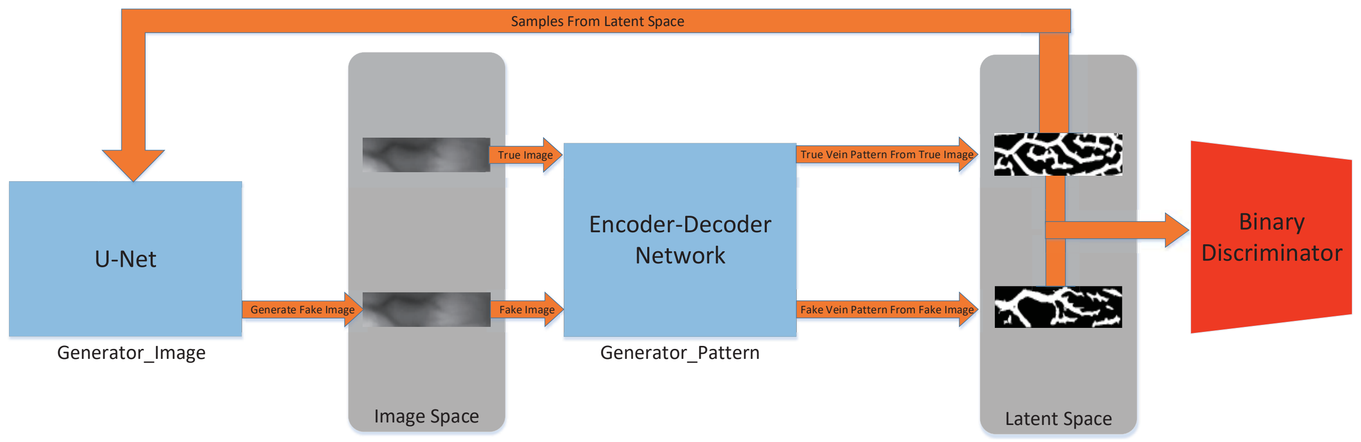


Fig. 5: Framework of FV-GAN. The discriminator contains a CNN for discrimination between true and false pattern maps. The Pattern Generator and the Image Generator transform data between the vein image space and the latent vein pattern space.

data between the raw vein image space and the latent vein pattern space respectively, and utilizes the cycle consistency loss for adversarial training. On the basis of CycleGAN, three adaptations have been made by FV-GAN for vein pattern extraction. Firstly, FV-GAN makes the first attempt to accommodate GANs on the finger vein verification task and creatively label foreground vein from obscure vein images to mitigate the problem of wrong training labels. The previous deep learning-based methods for vein pattern extraction are designed on the basis of common CNNs, such as AlexNet and VGG-16, and do not focus on the wrong label problem. Secondly, the architecture of FV-GAN is different from CycleGAN, in order to address the problem of wrong training labels. In the original CycleGAN, the discriminator D_Z in Fig.4 has to distinguish between the ground truth of vein pattern and the generated vein pattern. However, in finger vein verification, the captured vein images are usually seriously affected by noise, illumination and other environmental factors, and thus under current conditions, it is impossible for us to obtain the ground truth of foreground vein and is extremely difficult to label the foreground vein accurately. Thus, different from the original CycleGAN, the discriminator of FV-GAN is designed to distinguish between two generated vein patterns: One generated vein pattern is mapped from real vein images (corresponding to X in Fig.4) by the Pattern Generator (corresponding to G in Fig.4); and the other is mapped from fake vein images (corresponding to $\hat{X}$ in Fig.4) by the Pattern Generator again. In other words, FV-GAN not only contains the forward cycle of CycleGAN in Fig.4 but also adds another half cycle. In the proposed half cycle, generated $\hat{X}$ in Fig.4 is input into G again for obtaining the second generated vein pattern, to consider and mitigate the lack of accurate ground truth. Thirdly, instead of the $L1$ loss, the focal loss is utilized as the cycle consistency loss because the vein patterns in the latent space are binarized and the class imbalance exists in vein patterns. Based on CycleGAN and our adaptations designed for vein pattern extraction, the FV-GAN can achieve outstanding performance for vein verification.

B. Labeling Finger Vein Pattern

In most image processing tasks [18], [25], images or image contents are often labeled manually in advance for training a deep neural network. This approach, however, is expensive and time-consuming. To alleviate this problem in our task, we apply an automatic labeling method. For an image M, we segment it into two groups, vein pattern or background, by three baseline algorithms, namely RLT [4], WLD [8] and LMC [7]. The image M is then labeled through merging three resultant patterns under a majority rule. Let M_i $(i = 1, 2, 3$ and $n = 3)$ be the vein pattern maps produced by the three methods. We assign a binary label $L(x, y)$ to pixel(x, y) as

$$L(x,y) = \begin{cases} 1 & \sum_{i=1}^{n} M_i(x,y) >= \frac{n}{2}, \\ 0 & \sum_{i=1}^{n} M_i(x,y) < \frac{n}{2}. \end{cases} \tag{4}$$

The labeled finger vein images are binary images, where the value of vein pixel is 1 and the value of background is 0. Then we divide the raw ROIs and the labeled binary maps into patches of size $N \times N$ and input these patches into FV-GAN for training. During testing only the Pattern Generator is utilized and the raw ROIs are directly input into the network without any pre-processing or post-processing. In the experiments, the size of patches is 32 $\times$ 32.

C. Image Generator Architecture

This Image Generator transforms pattern maps into finger vein images, which is an image-to-image problem. Many previous solutions to this problem used a U-Net, hence in our FV-GAN, we develop a U-Net to generate finger vein images from vein patterns.

The U-Net [29] in the Image Generator is shown in Fig.6. It consists of a contracting path (left side) and an expansive path (right side). The contracting path follows the typical architecture of a convolutional network. It consists of repeated applications of a 3×3 convolution (unpadded convolutions) with stride 2, followed by a rectified linear unit (ReLU). At each downsampling step we double the number of feature channels. Every step in the expansive path consists of an

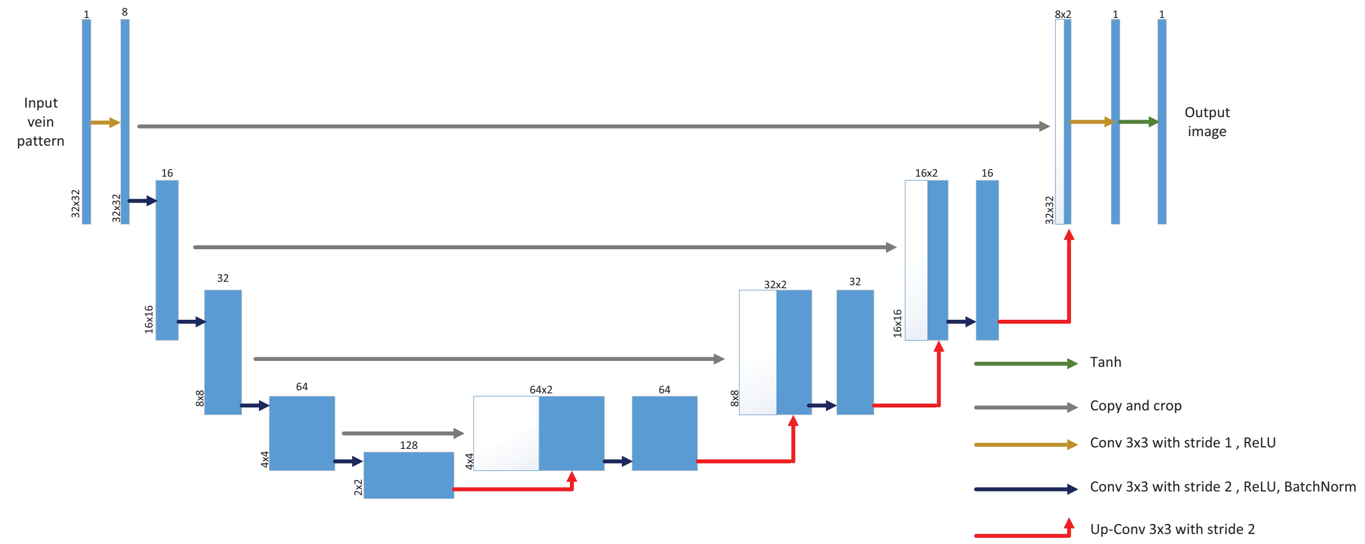


Fig. 6: A basic architecture of the Image Generator: U-Net. U-Net is an encoder-decoder with skip connections between mirrored layers in the encoder and decoder stacks.

TABLE I: Architecture of the Pattern Generator. Input size is 32×32.

Layers	Output Size	Type	Filter Size	Num	Stride
1	32×32	Conv	3	8	1
2	16×16	Conv	3	16	2
3	8×8	Conv	3	32	2
4	4×4	Conv	3	64	2
5	2×2	Conv	3	128	2
6	2×2	Conv	3	128	1
7	4×4	Up-Conv	3	64	2
8	8×8	Up-Conv	3	32	2
9	16×16	Up-Conv	3	16	2
10	32×32	Up-Conv	3	8	2
11	32×32	Conv*	3	2	1
-	32×32	Softmax	-	-	-

upsampling of the feature map followed by a 3×3 convolution (“up-convolution”) which halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, a 3×3 convolution followed by a ReLU and a batchnorm layer. In total the network has 13 convolutional layers.

D. Pattern Generator Architecture

The Pattern Generator is developed to extract vein patterns from raw finger-vein images. Compared with the Image Generator, this Pattern Generator obtains binary vein patterns with few details or textures and designed as an encoder-decoder network. Its network architecture is shown in Table I. It includes an encoder path (1-4 layers) and a decoder path (6-9 layers). The encoder path consists of repeated applications of a 3×3 convolution (unpadded convolutions) with stride 2, each followed by a leaky rectified linear unit (LeakyReLU) and a batchnorm layer. At each downsampling step we double the number of feature channels. Every step in the decoder path consists of an upsampling of the feature map followed by a 3×3 convolution (“up-convolution”) which halves the number of feature channels. In the final layer a Softmax classifier is used to generate the binary pattern maps. In total the network has 11 convolutional layers. In Table I, Conv is the combination of convolution, ReLU and BathcNorm; Up-Conv is the combination of up-convolution, ReLU and BathcNorm; and Conv* is the single convolution layer.

E. Binary Discriminator

For the Binary Discriminator, an 11×11 PatchGAN, which divides the whole image into 11×11 image patches and classify them as true or fake, is utilized. Such a patch-based discriminator network contains fewer parameters than a full connected network and can work on arbitrarily-sized images in a fully convolutional fashion.

F. Loss Function

We propose to use a hybrid loss function of the weighted sum of two terms for the two generators.

The first term in the loss function for the Pattern Generator is a focal loss [30]. Because the ground-truth pattern maps are binary images (as shown in (4)), the focal loss in our FV-GAN is two-class. Given a pattern maps $\hat{y}$ of size H×W×1 and a corresponding ground-truth pattern maps y of the same size, the two-class focal loss is defined as

$$FL(\hat{y}, y) = \frac{1}{H \cdot W} \cdot \sum_{i=1}^{H} \sum_{j=1}^{W} -0.25 \cdot [1 - p_t(i,j)]^2 \cdot \log(p_t(i,j)), \tag{5}$$

$$p_t(i,j) = \begin{cases} \hat{y}(i,j) & y(i,j) = 1, \\ 1 - \hat{y}(i,j) & y(i,j) = 0. \end{cases} \tag{6}$$

The two-class focal loss term encourages the GAN model to predict the right label for every image patch. Our tasks, extracting patterns of finger vein through segmenting foreground vein pixels from background non-vein pixels, can be considered as

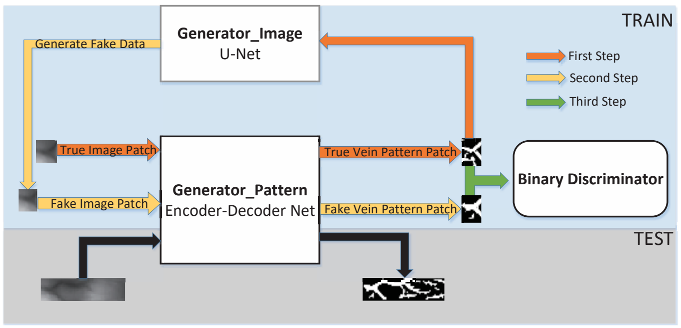


Fig. 7: Training and test processes of FV-GAN: The upper-panel sketch is for the training process; the lower-panel sketch is for the test process.

a two-class image segmentation problem. The cross-entropy loss is a standard loss function in state-of-the-art semantic segmentation models, e.g. [18], [31], [32]. But among the ground-truth labels, the foreground vein pixels are only about 10% percent of all pixels and serious class imbalance exists. To overcome this problem, a focal loss [30] is designed and utilized specifically for our tasks, as shown in (5).

The first term in the loss function for the Image Generator is the Huber loss. Given a true finger vein image x of size H×W×1 and a corresponding fake finger vein image $\hat{x}$ of the same size generated by the Image Generator, the Huber loss is defined as

$$HL(\hat{x}, x) = \frac{1}{H \cdot W} \cdot \sum_{i=1}^{H} \sum_{j=1}^{W} \delta(i,j), \tag{7}$$

$$\delta(i,j) = \begin{cases} 0.5 \cdot [x(i,j) - \hat{x}(i,j)]^2 & |(x(i,j) - \hat{x}(i,j))| \leq 1, \\ |(x(i,j) - \hat{x}(i,j))| - 0.5 & \text{otherwise.} \end{cases} \tag{8}$$

The two generators are tasked not only to fool discriminators but also to be near the ground-truth output in a distance metric. The Huber loss can accelerate the convergence of CNN and encourage less blurring.

The second loss term is based on the adversarial training. Our Binary Discriminator aims at enlarging the distance between true and false pattern maps, so we calculate the distance between the pattern maps generated from the fake finger vein images and the inversion of ground-truth label maps as the adversarial loss. The inversion of ground-truth label maps means swapping the finger vein and background in the ground-truth label maps. This means that the Binary Discriminator is trained to enlarge the distance between fake finger vein images and true finger vein images.

Mathematically, given a data set of N training images x_n and N ground-truth pattern maps y_n, we use $f(x_n)$ to denote the extracted probability map of patterns from image x_n that the Pattern Generator produces, and we use $g(y_n)$ to denote the generated finger vein image from pattern maps y_n that the Image Generator produces. Then the loss of Pattern Generator is defined as

Algorithm 1 FV-GAN training procedure.

```
Require: Finger vein image patch set $U$; vein pattern patch set $V$; generator parameters $\theta_{ig}$ and $\theta_{pg}$; batch size $m$; number of critic iterations $n$
 1: initialize $\theta_{ig}$, $\theta_{pg}$ with the Kaiming initialization method with the negative slope of the rectifier 0 [33]
 2: repeat
 3:     for $t = 1, \ldots, n$ do
 4:         sample images $\{u^{(k)}\}_{k=1}^m \sim U$ $\{v^{(k)}\}_{k=1}^m \sim V$
 5:         input $\{u^{(k)}\}_{k=1}^m$ into the Pattern Generator and obtain vein patterns $\{v_T^{(k)}\}_{k=1}^m$
 6:         input $\{v_T^{(k)}\}_{k=1}^m$ into the Image Generator and obtain fake finger vein images $\{u_F^{(k)}\}_{k=1}^m$
 7:         input $\{u_F^{(k)}\}_{k=1}^m$ into the Pattern Generator and obtain vein patterns $\{v_F^{(k)}\}_{k=1}^m$
 8:         minimize $\frac{1}{m}\sum_{k=1}^m L_{pg}(v_F^{(k)}, v_T^{(k)}, v^{(k)})$ and update $\theta_{pg}$. $L_{pg}$ is shown in (9)
 9:     end for
10:     sample image $\{u^{(k)}\}_{k=1}^m \sim U$ $\{v^{(k)}\}_{k=1}^m \sim V$
11:     input $\{u^{(k)}\}_{k=1}^m$ into the Pattern Generator and obtain vein patterns $\{v_T^{(k)}\}_{k=1}^m$
12:     input $\{v_T^{(k)}\}_{k=1}^m$ into the Image Generator and obtain fake finger vein images $\{u_F^{(k)}\}_{k=1}^m$
13:     input $\{u_F^{(k)}\}_{k=1}^m$ into the Pattern Generator and obtain vein patterns $\{v_F^{(k)}\}_{k=1}^m$
14:     minimize $\frac{1}{m}\sum_{k=1}^m L_{ig}(u^{(k)}, u_F^{(k)}, v_F^{(k)}, v^{(k)})$ and update $\theta_{ig}$. $L_{ig}$ is shown in (10)
15: until convergence
```

$$L_{pg}(\theta_{pg}) = \sum_{n=1}^{N} FL(f(x_n), y_n) + \lambda \cdot FL(f(g(f(x_n))), \overline{y}_n), \tag{9}$$

and the loss of Image Generator is defined as

$$L_{ig}(\theta_{ig}) = \sum_{n=1}^{N} HL(g(f(x_n)), x_n) + \lambda \cdot FL(f(g(f(x_n))), y_n), \tag{10}$$

where θ_{ig}, θ_{pg} denote the parameters of Image Generator and Pattern Generator, respectively; $FL(\cdot, y)$ denotes the focal loss for prediction y; $HL(g(f(x_n)), x_n)$ denotes the Huber loss for prediction x_n; $\overline{y}_n$ is the inversion of ground-truth label maps; and λ is a hyper-parameter to balance two terms of loss function (λ is set to 0.03 in our experiments). We minimize the loss function with respect to parameters θ_{ig} and θ_{pg}.

G. Parameter Update

In this work, we use stochastic gradient descent to train our network with a batch size of 32. The update rule for weight w_k in the k-th iteration is

$$w_{k+1} = \nabla_{k+1} + w_k \tag{11}$$

$$\nabla_{k+1} = 0.9 \cdot \nabla_k - 0.004 \cdot \eta \cdot w_k - \eta \cdot \frac{\partial L}{\partial w_k} \tag{12}$$

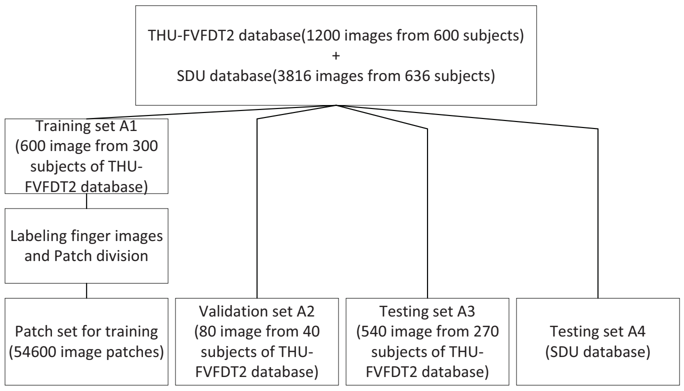

THU-FVFDT2 database(1200 images from 600 subjects)
+
SDU database(3816 images from 636 subjects)
Training set A1 (600 image from 300 subjects of THU-FVFDT2 database)
Labeling finger images and Patch division
Patch set for training (54600 image patches)
Validation set A2 (80 image from 40 subjects of THU-FVFDT2 database)
Testing set A3 (540 image from 270 subjects of THU-FVFDT2 database)
Testing set A4 (SDU database)

Fig. 8: Data partitioning and training set construction on the THU-FVFDT2 database and the SDU database

where ∇_k is the momentum variable, η is the learning rate, and $\frac{\partial L}{\partial w_k}$ is the derivative of the objective. The constants 0.9 and 0.004 are the momentum and weight decays which can reduce the training errors [13]. The weights in each layer are initialized using the Kaiming initialization method with the negative slope of the rectifier 0 [33]. We employ a learning rate of 0.001 which is decreased by 10 percent every 6 epochs.

H. Training and Testing of FV-GAN

For the FV-GAN training, the input data consists of patches of size $N \times N$. When the input size changes, the width and height of map in each layer will change accordingly. The output dimension is decided according to the number of classes to predict. The adversarial training step of FV-GAN in Fig.5 is illustrated in Fig.7 and described in Algorithm 1.

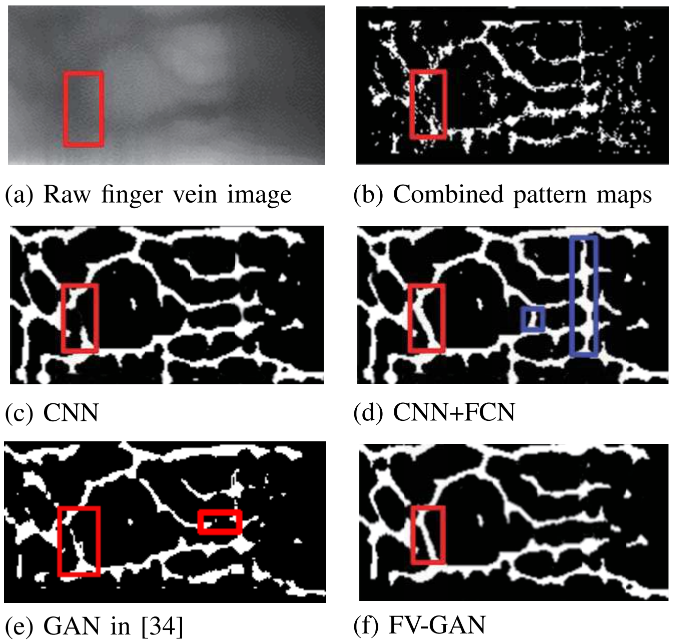
When testing, only the Pattern Generator is preserved and utilized. Given a training or test image, the CNN computes the probability of any pixel in the image being a vein pattern within one forward propagation, and labels it with the winning class with a probability threshold of 0.5. Then the vein pattern network is extracted and stored in a binary image. The binary images are subsequently used for feature matching and finger vein verification.

III. Feature Matching

After extracting the region of interest (ROI), we apply preprocessing to normalize translation and rotation variations. As some variations remain there due to inaccurate localization and normalization, a matching method is employed to compute the non-overlapping region between two images with possible spatial shifts.

Let R and T denote registered (reference template) and test binarized feature maps of size $m \times n$, respectively. The template $\bar{R}$ is an expanded image of R obtained by extending its width and height to $2w+m$ and $2h+n$; $\bar{R}$ is expressed as

$$\bar{R}(x,y) = \begin{cases} R(x-w, y-h) & 1 \leq x-w \leq m,\ 1 \leq y-h \leq n, \\ -1 & otherwise. \end{cases} \tag{13}$$



(a) Raw finger vein image (b) Combined pattern maps (c) CNN (d) CNN+FCN (e) GAN in [34] (f) FV-GAN

Fig. 9: Visual comparison between different feature maps. Recovered wrong labels are shown in red box. The blue box presents the over-recovering by FCN [6]

The matching score between R and T is computed as

$$\phi(R,T) = \underset{0\leq i\leq 2w, 0\leq j\leq 2h}{\arg\max} \frac{\sum_{x=1}^{m}\sum_{y=1}^{n} \odot(\bar{R}(x+i,y+j), T(x,y))}{m \times n}, \tag{14}$$

where

$$\odot(x,y) = \begin{cases} 0 & x \neq y, \\ 1 & x = y. \end{cases} \tag{15}$$

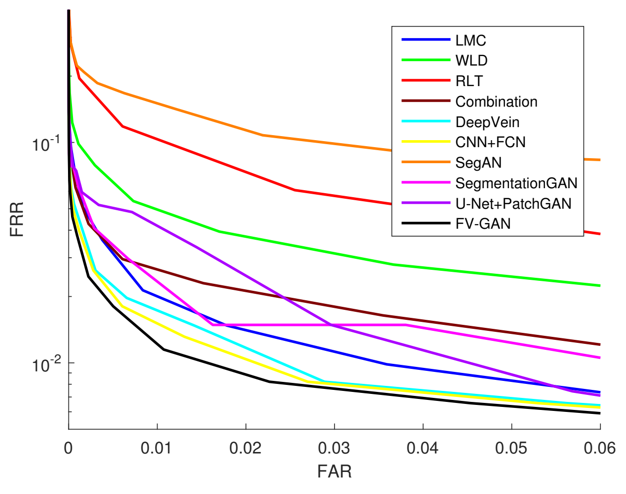
The score $\phi(R,T)$ basically computes the amount of overlap between R and T excluding the pixels located in the expanded region. The parameters w and h are employed to control the translation distance over horizontal and vertical directions and are heuristically set to 15 and 10, respectively. The template size $m \times n$ is 200×100 in our experiments.

IV. Experiments and Results

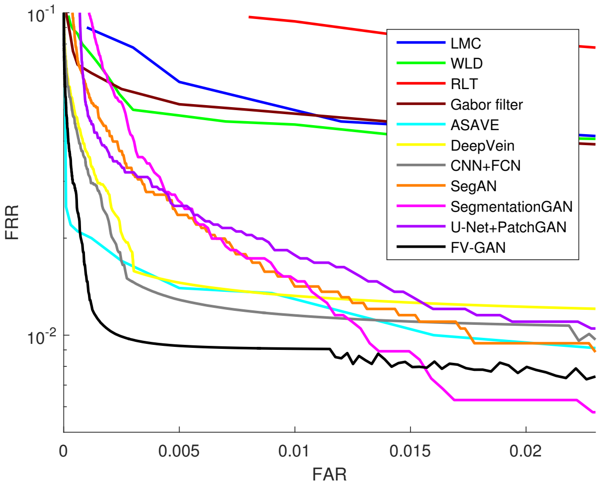
In order to evaluate the verification performance of the proposed framework in terms of effectiveness and robustness, we carry out experiments on two real databases.

A. Databases

1) The Tsinghua University Finger Vein and Finger Dorsal Texture Database 2 [35] (THU-FVFDT2 for short) consists of 610 images acquired from 610 subjects by using an open and contactless imaging device. The majority of the subjects are students and staff volunteers from Graduate School at Shenzhen, Tsinghua University. Each subject provided index finger from left hand. The captured finger images are collected in two sessions, separated by more



(a)



(b)

Fig. 10: (a) The ROC curves of LMC, RLT, WLD, combination scheme (results from LMC, RLT, WLD based on Eq.4), DeepVein, CNN+FCN, three CGAN-based methods and FV-GAN on dataset A3. (b) The ROC curves of LMC, RLT, WLD, Gabor filter, ASAVE, DeepVein, CNN+FCN, three CGAN-based methods and FV-GAN on dataset A4 (the SDU database).

than one week. In each session, each finger provides 1 image and thus there are a total of 1220 images from two sessions. Original images have the spatial resolution of 720×576. In [35], a preprocessed dataset with ROI images of size 200×100 is provided for finger vein verification.

2) The ShanDong University finger vein database [36] (SDU database for short) were captured from 106 volunteers by using a closed device designed by Joint Lab for Intelligent Computing and Intelligent Systems of Wuhan University. Each subject provided index, middle and ring fingers from two hands, resulting in a total of 636 finger classes. The captured finger images are collected in one session. In the session, each finger provided 6 images and thus 3816 images are obtained. Original images have the spatial resolution of 320×240. Since no preprocessed dataset is provided for finger vein verification in their work, we preprocessed the dataset in our experiment and obtained ROI images with size of 200×100.

B. *Experimental Setting*

To test the proposed approach, we split the two public databases into three sub-datasets for training, validation and testing. Fig.8 shows the data partition and the training set construction schemes. For THU-FVFDT2, there are 610 subjects associated with 610 fingers to provide the images in two sessions. For SDU, there are 636 subjects associated with 3816 fingers to provide the images in one session. We divide these two datasets into four subsets: a training dataset with 600 (300 fingers$\times$2 images in THU-FVFDT2) images, a validation dataset with 80 (40 fingers$\times$2 images in THU-FVFDT2) images, the first test dataset with 540 (270 fingers$\times$ 2 images in THU-FVFDT2) images and the second test dataset with 3816 (636 fingers$\times$ 6 images in SDU) images. The four datasets are denoted as datasets A1, A2, A3 and A4, respectively. The GAN is trained as described in subsection II-H. The validation set A2 is used to confirm the generalization of GAN, and datasets A3 and A4 are employed for verification.

In order to evaluate the performance of the proposed FV-GAN for finger vein verification, we conduct extensive experiments with several compared methods. Some classical approaches such as RLT [4], LMC [7], WLD [8] and Gabor filter method [37] are carried out on two public datasets. Besides, the state-of-the-art conventional method, namely ASAVE [38], is also conducted in our experiments. Based on the labeling scheme in (4), we extract the finger vein patterns and obtain the binarized map $L(x, y)$ computed by combining the segmentation outputs of the three baseline methods (RLT, LMC and WLD). To simplify the description, we denote it as the combination scheme. In addition to these classical methods, two CNN-based methods (DeepVein [17] and CNN+FCN [6]) and three CGAN-based methods (SegmentationGAN [39],U-Net+PatchGAN [34] and SegAN [40]) are also conducted. All deep learning-based methods are set as described in their works and trained by using dataset A1. We compare all these finger vein extraction approaches and the combination scheme with the proposed FV-GAN. Their corresponding performance is recorded in the following experiments. All the methods are implemented with Matlab2015b on a PC with E5-2687W CPU 3.30 GHz, GTX 1080 GPU and 64.00 G memory.

C. *Visual Assessment*

Here we visually assess the extracted patterns by various methods, to get more insights into the proposed method.

Fig.9 illustrates the finger vein extraction results of CNN+FCN, combination scheme and the proposed FV-GAN. From the obtained results, we can see that, compared with the conventional approaches, the deep learning-based methods effectively suppress the noise and extract more smooth and

TABLE II: EER comparison between various approaches on the SDU database

Method	LMC [7]	RLT [4]	WLD [8]	Gabor [37]	ASAVE [38]	DeepVein [17]	CNN+FCN [6]	SegmentationGAN [39]	SegAN [40]	U-Net+PatchGAN [34]	FV-GAN
EER(%)	3.65	5.85	3.76	3.34	1.39	1.37	1.16	1.53	1.48	1.10	**0.94**

TABLE III: Verification performance of various approaches on the dataset A3 (THU-FVFDT2)

Method	Rank-One verification rate(%)	EER(%)
LMC [7]	94.44	1.74
RLT [4]	92.60	4.52
WLD [8]	94.81	3.28
Combination	95.56	2.20
DeepVein [17]	96.67	1.44
CNN+FCN [6]	97.04	1.37
SegmentationGAN [39]	97.41	1.34
SegAN [40]	97.04	1.42
U-Net+PatchGAN [34]	97.78	1.26
FV-GAN	**98.52**	**1.12**

TABLE IV: Inference time of various deep learning-based approaches with image size 200×100

Method	DeepVein [17]	CNN+FCN [6]	FV-GAN
Time(sec)	0.43	6.4	**0.09**

continuous vein features from raw finger vein images. As shown in Fig.9(a), there is much noise in the raw finger vein image, which leads to low-quality data. Some vein patterns are corrupted and marked by the red rectangle. Surprisingly, there are remarkable differences between vein networks of the same finger extracted by the combination scheme, CNN+FCN and FV-GAN. Compared with the combination scheme, FV-GAN and CNN+FCN extract more precisely the vein pattern and recover the vessel breaks in spite of various noise. Although CNN+FCN succeeds in recovering the vessel breaks, extra errors in blue box of Fig.9(d) are introduced into the vein patterns by fully convolutional networks. This may be explained by the fact that the FV-GAN learns the distribution of latent vein pattern space rather than the pixel-to-pixel mapping, that is learned by fully convolutional networks, between finger vein images and vein patterns. Compared with CGAN-based methods, the proposed CycleGAN-based FV-GAN accelerates the convergence of training procedure and achieves better performance on break recovering. The experimental results in Fig.9 imply that the proposed FV-GAN is able to robustly extract vein patterns from raw finger vein images.

D. Verification Results with Adversarial Training-based Feature Extraction

Now we present quantitative evaluation of the verification performance in terms of effectiveness and robustness of various algorithms on the finger vein image datasets. In the test dataset A3, there are 540 images associated with 270 fingers which were captured in two sessions. We select 1 finger vein image acquired during the first imaging session as training data while the corresponding 1 image acquired during the second session are employed as test data to assess the verification performance. In the test dataset A4, 3816 images from 636 fingers are employed for verification. We select the first 3 finger vein images of each finger for training while the last 3 images of the same finger are employed for testing to assess the verification performance. The genuine scores are computed by matching images from the same finger while the impostor scores are obtained by matching images from different fingers. This results in 270 (270×1) genuine scores and 72630 ($270\times269\times1$) impostor scores from dataset A3, and 1908 (636×3) genuine scores and 1211580 ($636\times635\times3$) impostor matching scores from dataset A4. The experimental results from various approaches are summarized in Table II and Table III. The receiver operating characteristics (ROC) curves for the compared methods are illustrated in Fig.10.

We observe from Fig.10, Table II and Table III that the proposed FV-GAN achieves on both datasets the best performance among all the approaches considered in this work, including the combination scheme of the three baseline approaches, DeepVein [17], CNN+FCN [6] and three CGAN-based methods. For instance, the proposed FV-GAN reduces by more than 12% the EER obtained by the best one (U-Net+PatchGAN [34]) among existing approaches on dataset A3. Such a good performance may be explained as follows: 1) The vein image is labeled based on several baseline methods, it can guide the learning approach to extract robust features for vein pattern representation. 2) The information between different images cannot be inferred by conventional methods because they segment each image independently from the others. By contrast, the deep learning-based approaches (DeepVein [17], CNN+FCN [6] and the proposed FV-GAN) obtain rich prior knowledge by training the deep convolutional network on a big patch set from different images. Therefore, deep learning-based methods are capable of predicting better the probability of a pixel being a vein pattern based on the rich prior knowledge. 3) CNN is designed to extracted high-level feature by back propagation. CNN takes as input the raw images and iteratively uncovers hierarchical features to mini-

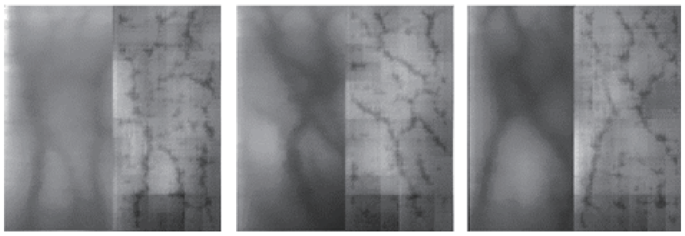


Fig. 11: Results of the generator. On each panel, the left-hand side is the original finger vein image, and the right-hand side is the output of the generator.

mize its errors on vein patterns. CNN avoids the need of first explicitly extracting some image processing-based features that might discard valuable and relevant information about the latent vein pattern space. 4) GANs learn from the joint distribution of the image space and the latent space (vein pattern space). The labeled vein patterns in the training dataset A1 can restrict the property of learned latent space. In the training dataset A1, only a small proportion of the whole database are abnormal data with wrong labels. The abnormal data thus has little effect on the chosen latent space and a proper distribution can be learned by FV-GAN. 5) Compared with CGAN-based methods (SegmentationGAN [39],U-Net+PatchGAN [34] and SegAN [40]), the proposed CycleGAN-based FV-GAN brings the cycle consistency loss into and restricts the distance of instance pair in both the latent space and the image space. These restrictions can reduce the space of possible mapping functions and accelerate the convergence of training procedure, which pushes FV-GAN to outperform CGAN-based methods on the vein pattern extraction task.

As shown in Table IV, we compare the proposed FV-GAN with two CNN-based methods in terms of inference time. The inference time is obtained by averaging the running time on the SDU database. Experiment results reveal that FV-GAN reduces by more than 98% the inference time of CNN+FCN [6]. Such a good performance may be explained as follows: 1) Compared with our proposed FV-GAN based on fully convolutional networks, the CNN in [6] contains two fully connected layers, which induce massive calculation. Accordingly, the inference time is also increased. 2) CNN+FCN in [6] predict the segment result of a single pixel in one forward pass. Therefore, the network needs to be passed forward for 20000 (200×100) times when inputing a 200×100 image. In contrast, our method can predict the vein patterns of an image with arbitrary size in one forward pass, which reduces the inference time substantially.

The baseline approaches that are based on detecting valleys have been shown to offer promising performance on their own finger vein databases, but they do not generalize and perform well on the THU-FVFDT2 and SDU databases that are more realistic. Such a poor performance may be attributed to their assumption that the profile of vein is valley-shaped, while noise and variation of infrared illumination can also create valleys in non-vein regions and erase valleys in vein regions, as shown in Fig.9(a). When matching in such regions, wrong vein patterns may generate additional verification errors. Similarly to the valley detector, the line feature detectors make the assumption that the vein pattern can be treated as line. Due to this assumption, the line feature detectors are also sensitive to noise and illumination, especially in these two realistic databases. The line feature-based approaches [4], [8] obtain the highest EER on dataset A3.

Surprisingly, combining vein features from the three baselines does not show higher performance than each of them. For example, Local Maximum Curvature [7] achieves lower EER compared with the combination scheme. This may be explained by the fact that some approaches such as Wide Line Detector [8] is sensitive to the noise and variation of infrared illumination. The vein patterns extracted by these approaches are not accurate enough to provide useful information for the combination scheme, leading to poor EER of combined vein patterns. Furthermore, these conventional approaches may generate conflicting information due to their segmentation errors in feature extraction. Thus, deep learning-based methods using these combined vein patterns cannot achieve excellent performance, such as DeepVein [17] and CNN+FCN [6]. In fact, wrong labels in the ground-truth data for deep learning-based methods are unavoidable even if the ground-truth data is labeled manually, and sometimes it is difficult to distinguish between correct labels and wrong labels in some tasks. Therefore, the adversarial training based on GAN is necessary for deep learning-based methods to correct wrong labels and improve verification performance automatically.

V. Conclusion and Future Work

In this paper, we proposed a CycleGAN-based pattern extraction model called FV-GAN for finger vein verification. The FV-GAN framework was developed to predict the probability of pixels being veins or background by learning a deep pattern representation, and extract the vein patterns from finger vein images. Experimental results show that FV-GAN can robustly extract vein patterns and significantly improve the verification performance in terms of verification accuracy and EER, which verifies the effectiveness and efficiency of adversarial training.

In the future we plan to investigate the following interesting aspects to further improve the performance of finger vein verification. First, a larger database will be collected. Second, in the experiments, we found that training GAN is difficult and unstable even with the state-of-the-art techniques of WGAN [27], CycleGAN [19] and DCGAN [25]. Hence other deep techniques will be investigated to make the convergence of GAN training easier and more stable.

Another interesting aspect is with the generated finger vein images from the generator. We compared them with the raw finger vein images, as illustrated in Fig.11. We can see that the generated finger vein images contains less noise and outliers. The generator is able to generate enhanced finger vein image and finger vein skeleton from finger vein patterns. This generative process is attractive because it can be applied to solving many other problems in finger vein verification. For instance, the generator can be developed to enlarge the finger vein database. Vein patterns of raw data are extracted by conventional or deep learning-based methods. Then, we input vein patterns into the generator with random noise. The generated results will be different from each other since the noise changes but they share the same finger vein skeletons with raw images of vein patterns. Hence expansion of the raw database can be achieved by this approach. These attractive outputs clearly merit further investigation.

Acknowledgment

We are grateful to the Associate Editor and three anonymous reviewers for their constructive comments which largely improve our manuscript, in particular about the methodology presentation and empirical comparison.

References

[1] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR'91., IEEE Computer Society Conference on*. IEEE, 1991, pp. 586–591.

[2] J. Daugman, "How iris recognition works," in *The essential guide to image processing*. Elsevier, 2009, pp. 715–739.

[3] D. Zhang, W.-K. Kong, J. You, and M. Wong, "Online palmprint identification," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 25, no. 9, pp. 1041–1050, 2003.

[4] N. Miura, A. Nagasaka, and T. Miyatake, "Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification," *Machine Vision and Applications*, vol. 15, no. 4, pp. 194–203, 2004.

[5] Y. Zhou and A. Kumar, "Human identification using palm-vein images," *IEEE transactions on information forensics and security*, vol. 6, no. 4, pp. 1259–1274, 2011.

[6] H. Qin and M. A. El-Yacoubi, "Deep representation-based feature extraction and recovering for finger-vein verification," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 8, pp. 1816–1829, 2017.

[7] N. Miura, A. Nagasaka, and T. Miyatake, "Extraction of finger-vein patterns using maximum curvature points in image profiles," *IEICE TRANSACTIONS on Information and Systems*, vol. 90, no. 8, pp. 1185–1194, 2007.

[8] B. Huang, Y. Dai, R. Li, D. Tang, and W. Li, "Finger-vein authentication based on wide line detector and pattern normalization," in *Pattern Recognition (ICPR), 2010 20th International Conference on*. IEEE, 2010, pp. 1269–1272.

[9] J. Yang, J. Yang, and Y. Shi, "Finger-vein segmentation based on multi-channel even-symmetric Gabor filters," in *Intelligent Computing and Intelligent Systems, 2009. ICIS 2009. IEEE International Conference on*, vol. 4. IEEE, 2009, pp. 500–503.

[10] C. Liu and Y.-H. Kim, "An efficient finger-vein extraction algorithm based on random forest regression with efficient local binary patterns," in *Image Processing (ICIP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 3141–3145.

[11] H. Qin, L. Qin, and C. Yu, "Region growth-based feature extraction method for finger-vein recognition," *Optical Engineering*, vol. 50, no. 5, p. 057208, 2011.

[12] J. Yang and Y. Shi, "Finger–vein ROI localization and vein ridge enhancement," *Pattern Recognition Letters*, vol. 33, no. 12, pp. 1569–1579, 2012.

[13] W. Yang, X. Huang, F. Zhou, and Q. Liao, "Comparative competitive coding for personal identification by using finger vein and finger dorsal texture fusion," *Information sciences*, vol. 268, pp. 20–32, 2014.

[14] W. Song, T. Kim, H. C. Kim, J. H. Choi, H.-J. Kong, and S.-R. Lee, "A finger-vein verification system using mean curvature," *Pattern Recognition Letters*, vol. 32, no. 11, pp. 1541–1547, 2011.

[15] S. A. Radzi, M. Khalil-Hani, and R. Bakhteri, "Finger-vein biometric identification using convolutional neural network," *Turkish Journal of Electrical Engineering & Computer Sciences*, vol. 24, no. 3, pp. 1863–1878, 2016.

[16] K. Itqan, A. Syafeeza, F. Gong, N. Mustafa, Y. Wong, and M. Ibrahim, "User identification system based on finger-vein patterns using convolutional neural network," *ARPN Journal of Engineering and Applied Sciences*, vol. 11, no. 5, pp. 3316–3319, 2016.

[17] H. G. Hong, M. B. Lee, and K. R. Park, "Convolutional neural network-based finger-vein recognition using NIR image sensors," *Sensors*, vol. 17, no. 6, p. 1297, 2017.

[18] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[19] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *arXiv preprint arXiv:1703.10593*, 2017.

[20] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum, "Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling," in *Advances in Neural Information Processing Systems*, 2016, pp. 82–90.

[21] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, and Z. Wang, "Photo-realistic single image super-resolution using a generative adversarial network," *arXiv preprint*, 2017.

[22] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[23] E. L. Denton, S. Chintala, and R. Fergus, "Deep generative image models using a Laplacian pyramid of adversarial networks," in *Advances in neural information processing systems*, 2015, pp. 1486–1494.

[24] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[25] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[26] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2017, pp. 2813–2821.

[27] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," *arXiv preprint arXiv:1701.07875*, 2017.

[28] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," *arXiv preprint arXiv:1703.05192*, 2017.

[29] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[30] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *arXiv preprint arXiv:1708.02002*, 2017.

[31] G. Lin, C. Shen, A. van den Hengel, and I. Reid, "Efficient piecewise training of deep structured models for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3194–3203.

[32] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1520–1528.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[34] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint*, 2017.

[35] W. Yang, X. Yu, and Q. Liao, "Personal authentication using finger vein pattern and finger-dorsa texture fusion," in *Proceedings of the 17th ACM international conference on Multimedia*. ACM, 2009, pp. 905–908.

[36] Y. Yin, L. Liu, and X. Sun, "SDUMLA-HMT: a multimodal biometric database," in *Chinese Conference on Biometric Recognition*. Springer, 2011, pp. 260–268.

[37] A. Kumar and Y. Zhou, "Human identification using finger images," *IEEE Transactions on image processing*, vol. 21, no. 4, pp. 2228–2244, 2012.

[38] L. Yang, G. Yang, Y. Yin, and X. Xi, "Finger Vein Recognition with Anatomy Structure Analysis," *IEEE Transactions on Circuits & Systems for Video Technology*, vol. PP, no. 99, pp. 1–1, 2017.

[39] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, "Semantic segmentation using adversarial networks," *arXiv preprint arXiv:1611.08408*, 2016.

[40] Y. Xue, T. Xu, H. Zhang, L. R. Long, and X. Huang, "SegAn: Adversarial network with multi-scale l1 loss for medical image segmentation," *Neuroinformatics*, pp. 1–10, 2018.