applied sciences

MDPI

Article

Finger-Vein Verification Based on LSTM Recurrent Neural Networks

Huafeng Qin [1,*] and Peng Wang [2]

[1] Chongqing Engineering Laboratory of Detection Control and Integrated System, School of Computer Science and Information Engineering, Chongqing Technology and Business University, Chongqing 400067, China
[2] National Research Base of Intelligent Manufacturing Service, Chongqing Technology and Business University, Chongqing 400067, China; 2017658007@email.ctbu.edu.cn
* Correspondence: qinhuafengfeng@163.com; Tel.: +86-138-8375-8680

Received: 27 February 2019; Accepted: 8 April 2019; Published: 24 April 2019

Abstract: Finger-vein biometrics has been extensively investigated for personal verification. A challenge is that the finger-vein acquisition is affected by many factors, which results in many ambiguous regions in the finger-vein image. Generally, the separability between vein and background is poor in such regions. Despite recent advances in finger-vein pattern segmentation, current solutions still lack the robustness to extract finger-vein features from raw images because they do not take into account the complex spatial dependencies of vein pattern. This paper proposes a deep learning model to extract vein features by combining the Convolutional Neural Networks (CNN) model and Long Short-Term Memory (LSTM) model. Firstly, we automatically assign the label based on a combination of known state of the art handcrafted finger-vein image segmentation techniques, and generate various sequences for each labeled pixel along different directions. Secondly, several Stacked Convolutional Neural Networks and Long Short-Term Memory (SCNN-LSTM) models are independently trained on the resulting sequences. The outputs of various SCNN-LSTMs form a complementary and over-complete representation and are conjointly put into Probabilistic Support Vector Machine (P-SVM) to predict the probability of each pixel of being foreground (i.e., vein pixel) given several sequences centered on it. Thirdly, we propose a supervised encoding scheme to extract the binary vein texture. A threshold is automatically computed by taking into account the maximal separation between the inter-class distance and the intra-class distance. In our approach, the CNN learns robust features for vein texture pattern representation and LSTM stores the complex spatial dependencies of vein patterns. So, the pixels in any region of a test image can then be classified effectively. In addition, the supervised information is employed to encode the vein patterns, so the resulting encoding images contain more discriminating features. The experimental results on one public finger-vein database show that the proposed approach significantly improves the finger-vein verification accuracy.

Keywords: biometrics; finger-vein verification; deep learning; convolutional neural network; representation learning

1. Introduction

With the wide application of internal and increasing risk of terrorist attacks, information security became a hot topic and received more and more attention. A key point is how to recognize a person to protect personal poverty and privacy. Biometrics as an authentication method of recognizing a person has been widely investigated in past years. Currently, various biometric characteristics such as fingerprints [1], palm-print [2], finger-vein [3,4], hand-vein [5], palm-vein [6], face [7], iris [8], voice [9], signature [10] have been employed for verification and can be broadly classified into two

categories. (1) Extrinsic characteristics (e.g., fingerprints, palm-print, face, voice, signature); (2) Intrinsic characteristics (e.g., finger-vein, palm-vein, hand-vein). The extrinsic characteristics are prone to be attacked because faked face and fingerprint can successfully cheat the verification system [11]. As the intrinsic characteristics such as finger-vein conceal the skin and not easily copied and forged, they show high security and privacy in practical application.

However, vein verification faces serious challenges. In practical applications, various factors such as environmental illumination [12–14], ambient temperature [3,14,15], light scattering [16,17], and user behavior [12,13] affect the finger-vein image quality. Generally, these factors are not controlled, so many capturing images not only contain vein patterns but also noise and irregular shadowing. Generally, the separability between the vein and non-vein patterns is poor in the regions associated with noise and irregular shadowing. Performing matching from such regions degrades the verification accuracy. To solve this problem, many segmentation-based methods are proposed to segment robust vein network for finger-vein recognition. Broadly, they can be categorized into two groups.

(1) Handcraft-based segmentation approaches. In this category, researchers employed the existing mathematical models to detect vein features based on attribute assumptions such as valleys and straight-lines. For example, they assume that the vein patterns can be approximated to line-like texture in a predefined neighborhood region and the descriptors such as Gabor filters are proposed to extract the vein pattern. The representative works include wide line detector (WLD) [13], Gabor filters [3,18–21], and matched filters [22]. Some researchers observe that the cross-sectional profile of a vein pattern shows the attribute of valley shape. Therefore, many models are built to detect the valley for vein pattern extraction. For instance, the curvature is sensitive to valley, so various approaches are proposed to enhance the vein patterns by computing mean curvature [14], difference curvature [23], and maxim curvature [15] of pixels in an image. In [24–27], the vein patterns are detected by computing the depth of the valley. In the region growth approach [27], both depth and symmetry of valley are combined to extract vein pattern. Recently, according to the anatomical knowledge, some characteristics of finger-vein structure, e.g., directionality, continuity, width variability, smoothness, and solidness are taken into account for finger-vein texture extraction in [28].

(2) Deep learning-based segmentation approaches. Unlike handcrafted approaches, the deep learning-based approaches are capable of extracting the vein patters from a raw image without the manual attribute distribution assumption and have shown promising performance in medical image segmentation such as neuronal membrane segmentation [29], prostate segmentation [30], retinal blood vessels [31], and brain image segmentation [32]. In work [33], the Convolutional Neural Network (CNN) model is firstly employed for finger-vein segmentation, and outperforms handcrafted feature-based approaches in terms of verification errors improvement. In their work, the pixels are automatically labeled and a patch-based dataset is built for CNN training. For testing, an image is split into various patches and each patch is put into the CNN to predict the probability of its center point being a vein pattern.

The approaches described above achieve good performance on some finger-vein recognition tasks, but they suffer from the following problems. For example, existing handcrafted approaches segment vein pattern based on assumptions. However, these assumptions are not always effective to detect the finger-vein patterns because some vein pixels may be created by more complicated distributions than valleys or straight lines. Also, they explicitly extract some vein features by an image processing method, which might discard relevant information about the finger-vein pattern. In addition, they do not get any prior knowledge from the different images as they segment each image independently from the others. For the deep learning-based approach [33], these problems are alleviated to an extent because it directly uncovers hierarchical features from raw images to minimize its decision errors on vein patterns without the attribute distribution assumptions. Meanwhile, rich prior knowledge is harnessed by training it on a huge patch-based training data from different images. However, these approaches, including CNN in [33], segment each pixel independently based on a predefined neighborhood region (e.g., patch) instead of considering spatial dependencies among the closed pixels.

Factually, finger-vein vein patterns extend from finger root to fingertip, and show clear direction and good connectivity [34]. Therefore, there exists spatial dependencies among the closed vein pixels. So, the performances of these existing approaches are still limited for finger-vein texture pattern extraction.

Recurrent neural networks have shown powerful capacity for the representation of long-term dependency information and have been successfully applied to human activity [35,36], speech recognition [37], and handwriting recognition [38]. In recent years, LSTM networks [39] as the most successful extensions of recurrent neural networks have received more and more attention. The Long Short-Term Memory (LSTM) model adopts a gating mechanism controlling the contents of an internal memory cell so that it is capable of learning a better and more complex representation of long-term dependencies in the input sequential data. Consequently, LSTM networks work well for feature learning over time series data. Some researches employ it to learn the complex spatial dependencies for scene labeling and action recognition [40–42].

Inspired by this idea, in this paper we proposed a stacked Convolutional Neural Networks and Long Short-Term Memory (SCNN-LSTM) for finger-vein texture segmentation by combining the CNN model and LSTM model. Compared to existing segmentation-based methods, our approach not only predicts the probability of a pixel based only on its pixels and their correlations in a local region, but it does so by relying also on the spatial dependencies in its neighboring contexts, through a feature representation learned by LSTM from a large sequence training set. The main paper contributions are summarized as follows:

(1) We proposed a stacked Convolutional Neural Network and Long Short-Term Memory model to automatically learn features from raw pixels for finger-vein verification. First, the vein and background pixels are automatically labeled based on several baselines. For each labeled pixel, we generated four sequences along different directions. As a result, there are various sequence-based training sets, on which several SCNN-LSTMs are independently trained to form a complementary and an over-complete representation. Secondly, for a testing image, the probability of each sequence being to vein pattern is predicted and the scores from patch-based sequences are conjointly input to P-SVM to segment the vein patterns. As the CNN model has the capacity for representation of vein texture features in a local region (i.e., patch) and the LSTM model captures the spatial dependencies among the closed regions, the proposed SCNN-LSTM model is capable of predicting the probability of belonging to a vein pattern. The rigorous experimental results on a public finger-vein database imply that the proposed approach is able to extract vein pattern, which results in a significant improvement for finger-vein verification accuracy.

(2) This paper investigates a new approach to encode the finger-vein for verification. Generally, the existing finger-vein segmentation approaches encode an image to extract binary vein patterns based on one or more thresholds, which are not related to verification error reduction. Different from them, an effective supervised scheme is employed to automatically select the threshold for vein pattern encoding. We search for a robust threshold to encode image by maximizing the inter-class distance and minimizing intra-class distance, which is not based on human domain knowledge. So the proposed scheme directly targets biometrics verification performance instead of human perception. We analyze the experimental results and estimate the verification performance.

2. The Proposed Approach

To learn compositional representations of the texture feature and spatial dependencies information, a SCNN-LSTM model is proposed for finger-vein feature extraction. First, we employed seven baselines to label the pixel from a training set and validation set. Secondly, for each labeled pixel, different sequences are created along different orientations. Thirdly, each sequence is forwarded to SCNN-LSTM to predict its probability of belonging to a vein patten. As a result, there are several labeled scores for different orientations, which are taken out of the input of SVM to extract a vein feature. Applying the proposed SCNN-LSTM model to the whole image in this way, the vein images are enhanced.

To achieve verification, the resulting enhancement image is encoded by a supervised encoding scheme. The framework of the proposed approach is illustrated in Figure 1.

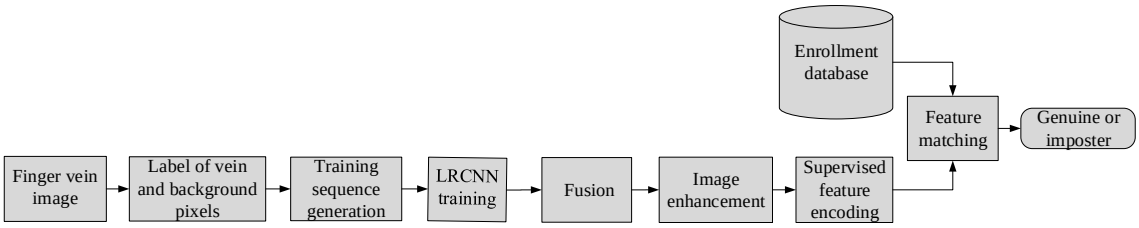


Figure 1. The framework of the proposed approach.

2.1. Label Vein Sequences

2.1.1. Label Vein Patterns

Similar to work [33], for each input finger-vein image, seven baselines, i.e., Repeated line tracking [24], Maximum Curvature points [15], Mean curvature [14], Different Curvature [43], Region growth [27], Wide line detector [13], and Gabor filters [3] are employed to segment vein pattern, resulting in seven binary images (as shown from Figure 2a–i). The values in each binary image (0 and 1 denote background and vein pixels, respectively) are treated as labels of corresponding pixels in the input image. We compute the average of seven binary images and obtain an average image F (Fin.3(i)). For a pixel (x, y), it is labeled as vein pattern if $F(x, y) = 1$ white region in Figure 2f), and it is labeled as vein for $F(x, y) = 0$ (black region in Figure 2j). We do not label the pixels in the remaining region (the color region in Figure 2j).

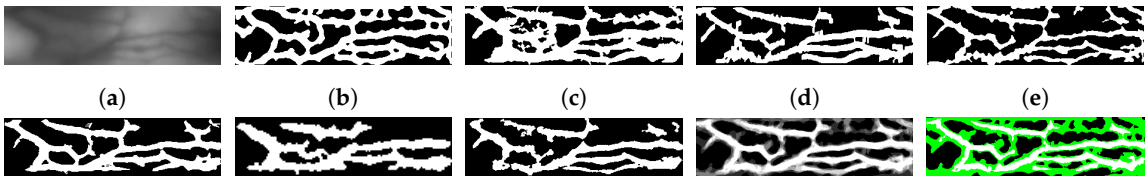


Figure 2. Segmented results. (**a**) Original finger-vein image; (**b**) Gabor filter; (**c**) Difference curvature; (**d**) Maximun curvature point; (**e**) Mean curvature; (**f**) Region growth; (**g**) Repeated line tracking; (**h**) Wide line detector; (**i**) Probability map; (**j**) Labeled pixels (wight region and black region denote vein and background, respectively).

2.1.2. Labeling Vein Sequences

In this section, the training sequences are produced based on labeled pixel (as shown in Figure 2j) for SCNN-LSTM training. Firstly, we select a labeled pixel as a current point c_0 and determine its $K-1$ adjacent pixels along a given orientation θ. This results in a set of K pixels $\{c_{-(K-1)/2,\theta}, ..., c_0, ...c_{(K-1)/2,\theta}\}$ for orientation θ, where $0 \leq \theta < \pi$ and K is the odd number to enforce symmetry. Then, we create K patches of $s \times s$ centered on c_0 and its $K-1$ adjacent pixels from image in training, and the resulting patches construct a sequence $S_\theta = \{P_{-(K-1)/2,\theta}, ..., P_0, ...P_{(K-1)/2,\theta}\}$. Similarly, the labels of K pixels create a labeled sequence $L_\theta = \{l_{-(K-1)/2,\theta}, ..., l_0, ...l_{(K-1)/2,\theta}\}$ for S_θ. Here, we quantize all the possible vein orientations θ into a set of C values by

$$\theta_i = \frac{i\pi}{C} \tag{1}$$

where $i = 1, 2, ..., C$ and C is heuristically set as 4. Namely, $\theta \in \{0°, 45°, 90°, 135°\}$, as shown in Figure 3a. Therefore, there are four sequences for each labeled pixel c_0. Figure 3 shows a sequence $S_{0°} = \{P_{-(K-1)/2,0°}, ..., P_0, ...P_{(K-1)/2,0°}\}$ of current pixel c_0 along the 0° orientation.

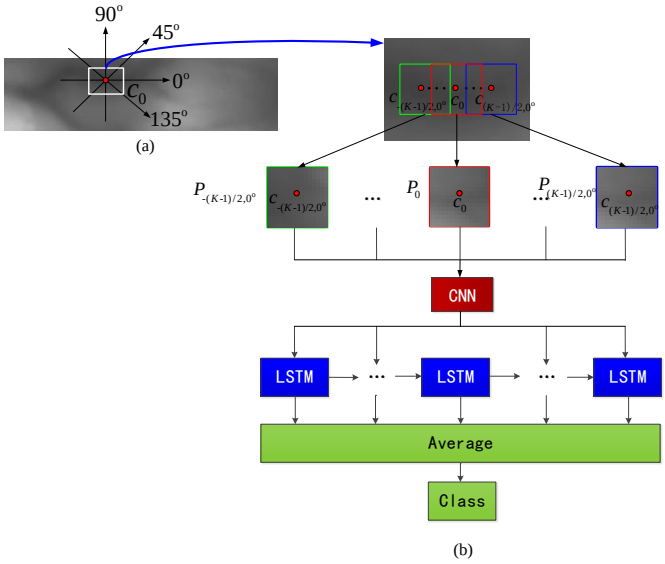


Figure 3. Illustration of the SCNN-LSTM model. (**a**) The four orientations for a pixel c_0; (**b**) SCNN-LSTM for prediction. A sequence sampled along 0° orientation is taken as an input of SCNN-LSTM to predict the probability of the centered point c_0 being to vein pattern. The LSTMs share same weights.

2.2. Stacked Convolutional Neural Networks and Long Short-Term Memory

The proposed stacked Convolutional Neural Networks and Long Short-Term Memory (SCNN-LSTM) consist of a CNN model and LSTM model (as shown in Figure 3) and are trained to learn the joint texture and spatial dependency representations for finger-vein texture segmentation. Our SCNN-LSTM takes a sequence associated with K patches as its input. In SCNN-LSTM, a deep CNN model is built by removing the output layer of an existing CNN model [33] for the vein texture representation. Then we take any patch as an input of the CNN model and it outputs a fixed-length vector representation which is further forwarded to a recurrent sequence learning module (LSTM) to learn the compositional representations in space, as shown Figure 3b.

Figure 4 shows the architecture of the proposed SCNN-LSTM. As shown in Figure 4, our approach consists of a CNN model and LSTM model. This CNN model (as shown in the red box in Figure 4) consists of three convolutional layers and one fully connected layer. There are 24 kernels of 5×5 in the first convolutional layer, 48 kernels of 5×5 in the second convolutional layer, and 100 kernels in the fully connected layer. The LSTM model (the blue region in Figure 4) includes 128 kernels. For SCNN-LSTM training, its input is a sequence of 7 patches with size of 11×11. Each patch in the sequence is forwarded to CNN model to obtain a 100 dimensional vector. As a result, there are 7 vectors for an input sequence with length of 7. The resulting vectors are taken as an input of LSTM model to obtain a 100 dimensional representative vector. Finally, the output of LSTM model is put into the last layer for classification. The output of last layer is a 2 dimensional vector because there are two classes (vein and background) for vein segmentation. When the input size changes, the width and height of the map in each convolutional layer changes accordingly. Along the forward direction, a patch-based sequence is represented effectively.

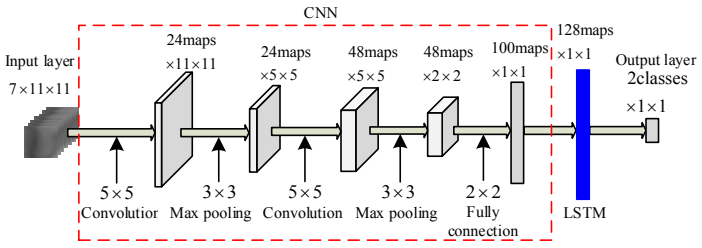


Figure 4. Architecture of SCNN-LSTM.

2.2.1. CNN Module

As the existing CNN model with three layers described in [33] has achieved promising performance for vein feature segmentation, we create a CNN module for feature representation of vein or background patch by removing the output layer of CNN in work [33]. During the training stage, our CNN is initialized using weights of an existing CNN [33]. Our CNN model consists of one input layer, three convolutional layers, two max-pooling layers, and one full-connection layer, respectively. The number of kernels in the three layers are 24, 48, and 100 respectively, and the sizes of kernels in both convolutional layers are 5. Each layer is detailed as follows.

Convolutional layer: The concept of Rectified Linear Units (y = max(0, x)) is used to active the hidden neurons.

Pooling alyer: The max-pooling is employed to extract location information by ensuring robustness to translation.

$$R_{i,j}^{k} = \max_{0 \le m,n < s} (r_{i.s+m,j.s+n}^{k}) \tag{2}$$

where r_k denotes as k-th output map obtained by the k-th filter; The value $R_{i,j}^{k}$ pools over non-overlapping $r \times r$ local regions in I_k to extract the compact feature.

Dropout: The drop-out technique [44] is applied in three fully connected layers. The overfitting can be greatly prevented by randomly omitting half of the hidden units.

2.2.2. LSTM Module

The LSTM module is a subnet of our SCNN-LSTM which allows to easily memorize the context information for long periods of time in sequence data. In general, LSTM is proposed to model the temporal dependencies. In images, this temporal dependency learning is converted to the spatial domain [41]. Therefore, we employ a LSTM unit as described in [39] to model spatial dependencies by mapping the deep feature sequences produced from CNN to hidden states. To predict a distribution over spatial step, the softmax is employed in output layer. Finally, we average the outputs of the LSTM network's softmax layer to compute the predicted distribution, as shown in Figure 3b. Given inputs x_t, h_{t-1}, and c_{t-1}, the LSTM updates at the position t are

$$i_t = \sigma(W_{xi}x_t + W_{hi}x_{t-1} + b_i) \tag{3}$$
$$f_t = \sigma(W_{xf}x_t + W_{hf}x_{t-1} + b_f) \tag{4}$$
$$o_t = \sigma(W_{xo}x_t + W_{ho}x_{t-1} + b_o) \tag{5}$$
$$g_t = tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{6}$$
$$c_t = f_t * c_{t-1it} * g_t \tag{7}$$
$$h_t = o_t * tanh(c_t) \tag{8}$$

where σ and $tanh$ are logistic sigmoid (sigm) and hyperbolic tangent ($tanh$), which are defined as

$$\sigma(x) = (1 + e^{-x})^{-1}, \tag{9}$$

$$tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{10}$$

and $*$ is the element-wise product. In addition, h_t, i_t, f_t, o_t, g_t, and c_t denote hidden unit, input gate, forget gate, output gate, input modulation gate, and memory cell, respectively, at the position t.

Output layer: The outputs from the last hidden layer are normalized with the softmax function:

$$y_m = \frac{\exp(z_m)}{\sum_{n=1}^{N} \exp(z_n)} \tag{11}$$

where z_n is a linear combination of outputs in LSTM hidden states.

2.3. Multi-SCNN-LSTM Feature Representation

For a pixel with a label $l \in \{0,1\}$ from a given finger-vein image F, we produce a sequence $S_{\theta^*} \in S^{s \times s \times K}$ along an orientation θ^* and label it as $L_{\theta^*} \in \Re^{K \times 1}$ using the scheme described in Section 2.1, where 0 and 1 denote respectively background and vein. The training set used for vein segmentation is represented as $\{(S_{\theta^*,1}, L_{\theta^*,1})\}, \{(S_{\theta^*,2}, L_{\theta^*,2})\}, \ldots, \{(S_{\theta^*,N}, L_{\theta^*,N})\}$, where N is the number of sequences from finger-vein images in the training database. As we quantize all the possible vein orientations into four orientations, we in this way obtain 4 training datasets. Let $\{(S_{\theta_i,1}, L_{\theta_i,1})\}, \{(S_{\theta_i,2}, L_{\theta_i,2})\}, \ldots, \{(S_{\theta_i,N}, L_{\theta_i,N})\}$ be the i-th training dataset $(i = 1, 2, \ldots, 4)$. A different SCNN-LSTM for each dataset is then trained independently, and each SCNN-LSTM produces a score from a particular sequence. We combine the outputs of the 4 SCNN-LSTMs to generate a 4-dimensional vector $v = [v_1, v_2, v_3, v_4]$, which is taken as an input of P-SVM to predict the probability of the pixel (Figure 5).

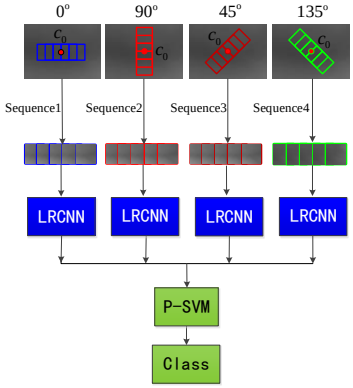


Figure 5. The framework of the Muilti-SCNN-LSTM. The prediction scores of a pixel are computed from four sequences along four orientations $(0°, 45°, 90°, 135°)$ and combined to generate a complementary score vector, taken as input of P-SVM to jointly predict the probability of centered pixel c_0 being to vein pattern.

2.4. Generating Score

A SVM model is employed to compute the probability of a pixel belonging to vein pattern based on its predicted distribution along four orientations. In this work, we employ the P-SVM model [45], which requires a set of vectors for training, to combine all features from all orientations (shows in Figure 5). Let v be a vector extracted from four sequences of a pixel with a label $l \in \{0,1\}$. The P-SVM is trained to provide a probabilistic value p (0 to 1: from background to vein)

$$p(q = 1|\varepsilon(v)) = \frac{1}{1 + \exp\left(w \cdot \varepsilon(v) + \gamma\right)} \tag{12}$$

where $\varepsilon(v)$ is the output of a general two-class SVM [46] with v as the input feature vector, and w and γ as fitting parameters trained by P-SVM. After training, we are able to compute the probability of any pixel based on its feature vector v and Equation (12).

2.5. Supervised Feature Encoding

In this section, we propose a scheme to obtain the threshold for vein feature encoding. After applying SCNN-LSTM for all pixels, an enhancing vein image is obtained and then we encode it for matching. In existing works [3,13–15,27,33,43], the vein patterns are encoded by one or more thresholds. For example, the probability of 0.5 is employed to obtain vein patterns in [33]. In [3], the vein image is enhanced by Gabor and then subject to binarzition using threshold of 0. In the classic repeated line tracking approach [24], two global thresholds (i.e., 85 and 175) are used to divide a image into three regions for matching. Some curvature-based approaches [14,15,23] enhanced vein patterns by computing the curvature of all pixels and an empirical threshold is employed to encode resulting enhancement image. For the finger-vein verification, the primary target of feature encoding is to improve performance, mainly verification error rates. However, the approaches determine the threshold based on human perception instead of minimizing the verification error, so the resulting binary code (vein texture features) may not be robust for finger-vein verification. To overcome this problem, in this section, a supervised scheme is proposed to encode vein pattern. Our approach decides the threshold by maximizing the distance between intra-class score set and inter-class score set computed from a training set, such that the resulting threshold is directly related to verification performance. The robust thresholds T are computed as follows.

Assume that there are N classes in the training set and each class provides M samples. Using the proposed SCNN-LSTM model (Figure 5), all finger-vein images are enhanced and we denote the mth enhancement image in the nth class as $x_{m,n}$, where $m = 1, 2, ..., M$ and $n = 1, 2, ...N$. We aim to find a function to map and quantize each enhancement image into a binary image $b_{m,n} \in \{0,1\}^{I \times J}$ which encodes a more discriminative information for verification error minimization. In our work, the binary code (vein texture pattern) $b_{m,n}$ of $x_{m,n}$ is computed by

$$b_{m,n} = 0.5 \times (sgn(x_{m,n} - T) + 1) \tag{13}$$

where $sgn(z)$ is equal to -1 if $z \leq 0$ and 1 otherwise and $T \in [0 \quad 1]$ is a parameter which is determined as follows.

Based on the Equation (13), all training samples are mapped into Hamming space, so a Hamming distance in [47] is employed to match two images for verification. We match the binary codes from same class to generate intra-class scores while the inter-class scores are produced by matching the binary codes from different class. So there are $a_1 = N \times C_2^M$ genuine matching scores $\Omega_1 = \{d_1(T), d_2(T), ..., d_{a_1}(T)\}$ and $a_2 = N \times (N-1) \times M \times M/2$ impostor scores $\Omega_2 = \{d'_1(T), d'_2(T), ..., d'_{a_2}(T)\}$. To make $b_{m,n}$ discriminative, we enforce an important criterion to encode the enhancement images that the resulting binary codes should maximize the distance between two sets Ω_1 and Ω_2. Therefore, we formulate the following optimization objective function:

$$\max_T J(T) = \frac{|u_1(T) - u_2(T)|}{D_1(T) + D_2(T)} \tag{14}$$

where $|\cdot|$ represents the absolute value. $u_1(T)$ and $u_2(T)$ are the means of the scores in Ω_1 and Ω_2, and $D_1(T)$ and $D_2(T)$ are the variances of the scores in the sets Ω_1 and Ω_2

To facilitate to search the threshold T, all enhancement images are converted to gray-scale images with integer values between 0 and 255. The parameter T is assigned from 0 to 255 to transform the enhancement image into a binary code map according to Equation (13). So, 256 different values $J(T)$ $(T = 1, 2, ..., 256)$ are computed using Equation (14). The parameter T_*, which can maximize Equation (14), are selected to encode the vein pattern. The binary code of $x_{m,n}$ is computed by

$$b_{m,n} = 0.5 \times (sgn(x_{m,n} - T_*/255) + 1) \tag{15}$$

2.6. Feature Matching

After all training images are mapped into Hamming space, the Hamming distance is employed to match two images. In general, the capturing images are subject to translation and rotation normalization, but there are still some variations due to inaccurate localization and normalization. However, Hamming distance is not robust enough to reduce these variations. So, an enhanced Hamming distance is employed to compute the non-overlapping region between two images with possible spatial shifts for finger-vein matching. Assuming Q and B are enrolment and test binarized feature codes with size of $I \times J$, respectively (as shown in Figure 6), the height and width of Q are extended to $2E + I$ and $2H + J$, and then its expanded image $\bar{Q}$ is obtained and expressed as:

$$\bar{Q}(i,j) = \begin{cases} Q(i-E, j-H) & \text{if} \quad 1+E \le i \le I+E, \\ & \qquad 1+H \le j \le J+H \\ -1 & \text{otherwise} \end{cases} \tag{16}$$

Figure 6b illustrates the extended image $\bar{Q}$ of a template Q and the extend region with values of -1 is marked in color. The matching distance between Q and B is obtained by

$$d(Q,B) = \min_{0 \le e \le 2E, 0 \le h \le 2H} \frac{Hamdistance(\bar{Q}^{e,h}, B) - \Phi(\bar{Q}^{e,h})}{Hamdistance(\bar{Q}^{e,h}, U)} \tag{17}$$

In Equation (17), $\Phi(V)$ is the amount of -1 values in matrix V. U is a matrix with size of $I \times J$ and the values of its elements are -1. *Hamdistance* represents the hamming distance between two encoding images, i.e., summation of the number of positions that are different. $\bar{Q}^{(e,h)}$ is a matrix (the red rectangle box in Figure 6b) when the translation distances are e and h over horizontal and vertical directions. $d(Q,B)$ basically computes the minimal amount of non-overlap between Q and B at different spatial shifts, excluding the pixels located in the expanded region (e.g., the green region in Figure 6b). The parameters E and H control the translation distance in horizontal and vertical directions and are heuristically set to 20 and 60.

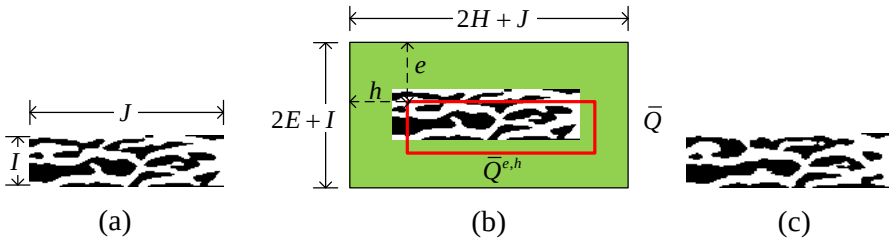


Figure 6. Matching sample. (**a**) A finger-vein template; (**b**) The extended image from (**a**); (**c**) A testing image. The values in green region are -1 in (**b**). The red rectangle box translates in the extended images from top left corner to lower right corner, and $\bar{Q}^{e,h}$ is a map in the red rectangle box when the translation distances are e and h over horizontal and vertical directions.

3. Experiments and Results

To estimate the performance of our approach, we compare various approaches with respect to verification performance improvement. In our experiments, we repeat the experimental results of classic approaches, i.e., Repeated line tracking [24], Maximum Curvature points [15], and recent approaches, i.e., Mean curvature [14], Different Curvature [43], Region growth [27], Wide line detector [13], and Gabor filter [3] for comparison. Also, we show the performance of the deep-based segmentation approach [33] to estimate the verification performance of our approach. In addition, based on the supervised encoding scheme in Equation (15), we can extract the finger-vein patterns from the probability map which is

computed by the proposed SCNN-LSTM approach. To simplify the description, we denote them as the SCNN-LSTM + Supervised encoding. To test our encoding approach, we also encode the resulting probability map using a probability threshold of 0.5. This scheme is presented as SCNN-LSTM + Unsupervised encoding. The corresponding performance is shown in the following experiments. We compare all finger-vein extraction approaches mentioned above with the proposed one to get more insights into the problem of finger-vein verification. All experiments are carried out on one public database, namely the PolyU [3] finger-vein database, which is described below.

3.1. HKPU Database

The Hong Kong Polytechnic University (HKPU) finger-vein image database [3] includes 3132 images with a resolution of 513 × 256 pixels. All images are collected from 156 subjects using an open and contactless imaging device. The first 105 subjects provided 2520 finger images (105 subjects × 2 fingers × 6 images × 2 sessions) in two separate sessions with a minimum interval of one month and a maximum of over six months, with an average of 66.8 days. In each session, each subject provided 2 fingers (index finger and middle finger) and each finger provided 6 image samples. The remaining 51 subjects only provided image data in one session. To verify our approach, the 2520 finger images captured in two sessions are employed in our experiment because it is closer to a practical captured environment. A pre-processing method [3] is employed to extract the region of interest (ROI) image and carry out translation and orientation alignment. In addition, the image background is cropped because it contributes matching errors and computation cost. As a result, all images are normalized to 39 × 146.

3.2. Experimental Setting

To test our approach, we split the database into three data sets: training set associated with 660 (55 fingers × 12) images, validation associated with 600 (50 fingers × 12) images, and testing set associated with 1260 (105 fingers × 12) images. Based on the label scheme described in Section 2.1, we label vein and background pixels from the training set and validation set. To train our model, we select the sequences centered on vein pixel as positive samples and sequences centered on background pixels as negative ones. For each image in training set, we only employ about 80 positive sequences and negative sequences, respectively. As the length of sequences is fixed to 11 using next experiments in Section 3.3, there are about 1760 (80 sequences × 11 (length of sequences) × 2 (positive sequences and negative sequences)) patches for an image. This results in a total of 100,000 training sequences (50,000 positive sequences and 50,000 negative sequences) from 660 images. In the testing phase, we generate a patch for each pixel in a test image. So, for an image with size of 39 × 146, there are 5694 (39 × 146) patches, based on which a sequence is created for each pixel along a given orientation. In our work, the length of the sequence is 11. Therefore, for a pixel, the patches centered on its 11 adjacent pixels form a sequence along a given orientation (shown in Figure 3), which results 5694 sequences for a test image with size of 39 × 146. Then, the sequence of each pixel is put into our model, the output of which is taken as the probability of this pixel to belong to vein pattern.

3.3. Parameter Estimation

As described in Section 2.1, each sequence from images in training set consists of K patches with size of $s \times s$. The CNN module in our SCNN-LSTM is trained by fine-tuning the CNN with an input of 11 × 11 patch in [33]. Such a size has also shown good performance in work [33], so the patch size s for SCNN-LSTM is fixed to 11. The length of the sequence is important to achieve high verification accuracy. If K is too small, more detailed vein patterns are extracted but including more noise. Matching pixels in noisy regions can create errors which result in verification accuracy reduction. On the contrary, sequences with large K will suppress vein feature details, leading to smooth vein features, which also degrades the verification accuracy. Therefore, we determine the appropriate size of sequence for SCNN-LSTM experimentally. Firstly, we train the proposed SCNN-LSTM model to extract the vein feature of the finger-vein images in the training and validation at different lengths of

sequence. To reduce the redundant information, we obtain patches with sampling intervals of one pixel to create training sequences. Secondly, the first 6 images acquired at the first session are employed as registration templates and the remaining as testing images. Therefore, there are 300 (50×6) genuine scores and 14,700 ($50 \times 49 \times 6$) impostor scores. The False Rejection Rate (FRR) is computed by the genuine scores and the False Acceptance Rate is computed by impostor scores. The Equal Error Rate (EER) is the error rate when FAR is equal to FRR. Figure 7 illustrates the relationships between length of sequence and EER, and the results are obtained by using only the validation data. From Figure 7, we can see that a smaller equal error rate is achieved at a sequence of length 11 and 13. With increasing the length K, the computation time will be increased. Therefore, we fix the length of a sequence to 11 in our experiments.

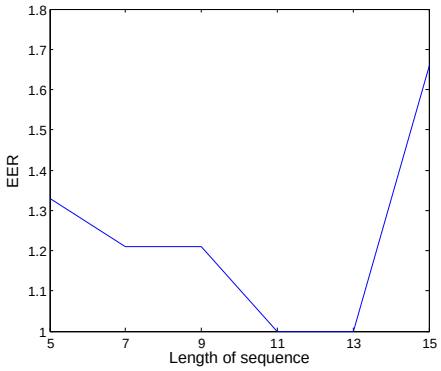
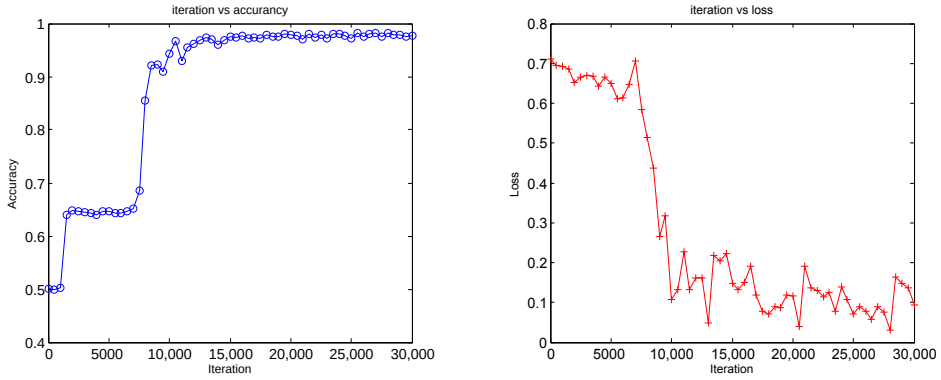


Figure 7. Relationship between the length of sequence and EER.

To verify over-fitting of our model, we shows learning curves in Figure 8. Figure 8a,b show the accuracy on the validation dataset and loss on the training dataset. From Figure 8, we can observe that the accuracy of validation dataset increases to about 65% and the loss decreases slowly after 2000 backpropagations. When the number of iteration steps is between 5000 and 10,000, the accuracy increases to more than 90% and the loss dramatically reduces. After 10,000 iterative steps, the loss fluctuates but it still decreases slowly. Therefore, our SCNN-LSTM model has good convergence for finger-vein segmentation.



(**a**) (**b**)

Figure 8. Training curves (**a**) Accuracy vs. iteration and (**b**) Loss vs. iteration.

3.4. Visual Assessment

In this experiment, we visually analyze the extracted finger-vein patterns from various approaches to get more insights into the proposed approach. The seven baselines and a state of the art [33] are employed to segment the vein texture, respectively. Also, the vein patterns encoded by a threshold of 0.5 and supervised threshold are reported in our experiment. Figure 9 shows the extracted results of various approaches. We can see from Figure 9 that the deep learning-based approaches suppresses

the noise, and extract more connective and smoothness vein texture compared to the seven baselines. Observed the experiments in Figure 9i,j,f, it sees that the SCNN-LSTM-based approaches outperform the CNN in terms of extracting the connective vein patterns.

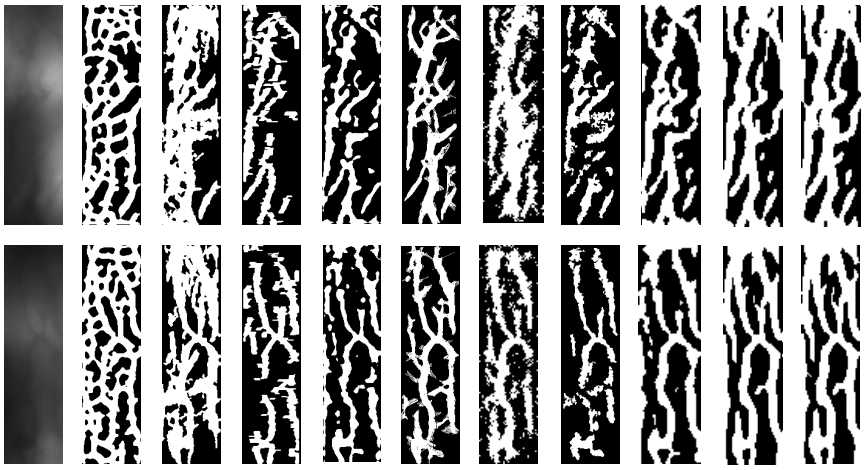


Figure 9. Experimental results from various approaches. (**a**) Original image; (**b**) Gabor filters; (**c**) Difference curvature; (**d**) Maximun curvature point; (**e**) Mean curvature; (**f**) Region growth; (**g**) Repeated line tracking; (**h**) Wide line detector; (**i**) CNN; (**j**) SCNN-LSTM + Unsupervised encoding; (**k**) SCNN-LSTM + Supervised encoding.

3.5. Verification Results Based on Image Dataset from One Session

In this section, we evaluate the performance of various approaches on the HKPU finger-vein dataset by considering vein images collected in each of the two sessions. First, the performance is evaluated in each session, individually. In one session, there are 630 images from 105 fingers. Therefore, the total number of genuine scores and impostor scores is 1575 ($105 \times C_2^6$) and 196,560 ($105 \times 104 \times 36/2$). To compute the impostor score, the symmetric matches are not executed. Second, the performance of combining scores from two sessions is reported. So, there are 3150 (1575×2 sessions) genuine scores and 393,120 (196,560 $\times 2$ sessions) impostor scores. Table 1 lists the verification error of various approaches for each session taken separately, and then for the two sessions, mixed. The receiver operating characteristics (ROC) curve for the corresponding performances is illustrated in Figure 10. The experimental results from Table 1 imply that the proposed SCNN-LSTM approach outperforms existing approaches including CNN [33] and achieves low errors, e.g., 1.12%, 0.62%, and 1.01% for data in the first session, second session, and two mixed sessions, respectively. The ERRs are further reduced to 1.08%, 0.58%, and 0.95% using the proposed encoding approach. We also observe from Figure 10 that the SCNN-LSTM-based approaches significantly improve FRR when the FAR is lower than 0.01%, which implies that our system achieve lower verification error than the methods considered in our work at high security level system.

Table 1. EER of various approaches on image dataset from one session.

Methods	First Session	Second Session	Two Sessions
Repeated line tracking [24]	4.76	5.67	5.21
Maximum curvature point [15]	3.91	3.27	3.59
Region growth [27]	2.32	1.24	1.75
Wide line detector [13]	3.68	3.11	3.39
Gabor filters [3]	2.10	1.84	1.95
Mean curvature [14]	2.06	1.50	1.73
Difference curvature [43]	3.61	3.64	3.64
CNN [33]	1.21	0.86	1.12
SCNN-LSTM + Unsupervised encoding	1.12	0.62	1.01
SCNN-LSTM + Supervised encoding	1.01	0.58	0.95

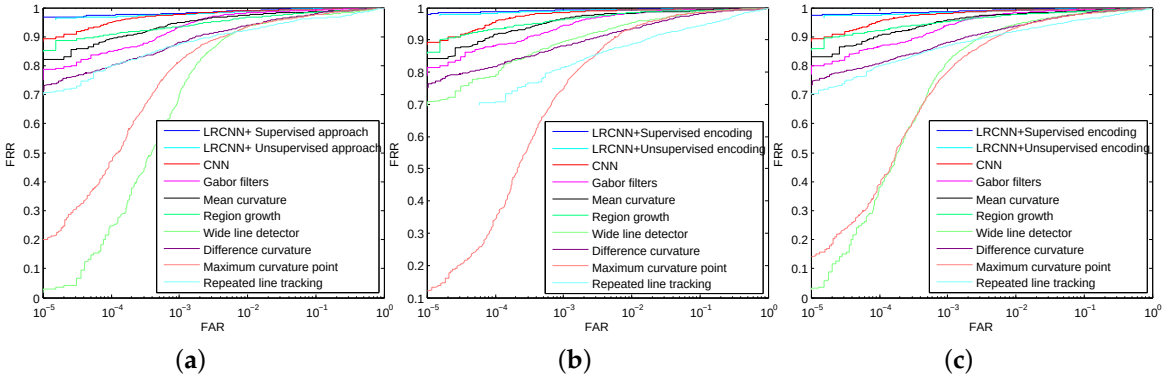


Figure 10. Receiver operating characteristics from image data collected at (**a**) first session, (**b**) second session, and (**c**) two mixed sessions.

3.6. Verification Results Based on Image Dataset from Two Sessions

This experiment aims at estimating the effectiveness and robustness of various algorithms on the finger-vein image data from both sessions. In the testing dataset, there are 1260 (105 fingers $\times$ 6 images $\times$ 2 sessions) images, acquired at two sessions. For each finger, we select the 6 images captured at the first session as enrollment samples and the remaining 6 images captured at the second session as testing samples. The genuine matching scores are produced by matching samples from same finger, while the impostor scores are produced by matching samples from different fingers. This results in a total of 630 (105 $\times$ 6) genuine scores and (105$\times$ 104 $\times$ 6/2) impostor scores, based on which we the compute FRR and FAR. In addition, we computed the sensitive index(d') [48] by $d' = Z(hit \quad rate) - Z(false \quad alarm \quad rate)$ to estimate the performance of various approaches.

The experimental results from various approaches are summarized in Table 2. The ROC curves for the corresponding performances are illustrated in Figure 11. The experimental results summarized in Table 2 show consistent trends with the those from experiments in each session. The proposed SCNN-LSTM-based approaches (e.g., SCNN-LSTM + Unsupervised encoding and SCNN-LSTM + Supervised encoding) get the best results, especially at the lower FAR. The lowest EER of 2.38% is achieved using the supervised encoding approach. Similarly, the proposed method achieves higher d' (e.g., 3.89 and 3.95) compared to existing approaches, which implies that the lowest verification error is achieved using our SCNN-LSTM model.

Table 2. EER of various approaches on image dataset from two different sessions.

Methods	EER (%)	Sensitive Index (d')
Repeated line tracking [24]	12.85	2.26
Maximum curvature point [15]	8.30	2.76
Region growth [27]	5.71	3.15
Wide line detector [13]	7.62	2.86
Gabor filters [3]	5.08	2.97
Mean curvature [14]	4.20	3.45
Difference curvature [43]	7.90	2.92
CNN [33]	3.02	3.72
SCNN-LSTM+Unsupervised encoding	2.59	3.89
SCNN-LSTM+Supervised encoding	2.38	3.95

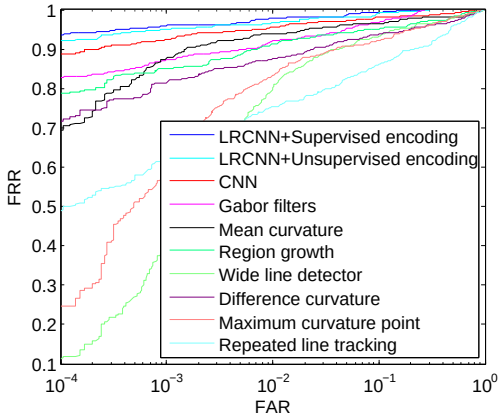


Figure 11. Receiver operating characteristics on image data from two separate sessions.

4. Discussion

The experiments depicted in Tables 1 and 2, Figures 10 and 11 show that the proposed SCNN-LSTM-based models achieve best performance among the all approaches considered in our work, including seven baselines and the CNN-based model. For example, the EER achieved by the best one (CNN) among existing approaches is reduced to 2.53% using the proposed SCNN-LSTM model with unsupervised encoding scheme on the data set acquired from two sessions. The verification accuracy may be further improved by combing the features of sequences along more directions or enlarging the training set. The good performance can be explained by the following fact. The existing handcrafted approaches (seven baselines) explicitly extract some features by image processing method, which might discard relevant information about finger-vein pattern. Also, they do not get any prior knowledge from the different images as they segment each image independently from the others. In addition, all approaches, including CNN, independently process each pixel based on a predefined neighborhood region or cross-sectional profile during the segmentation procedure, and ignore the spatial dependencies among different vein pixels. By contrast, the proposed approach uncovers hierarchical features for vein texture representation by training its CNN module and harnesses rich dependency information by training its LSTM module on a huge sequence set from different images. Therefore, it is capable of predicting the probability of a pixel belonging to a vein pattern.

We can also observe from the experimental results (Tables 1 and 2, Figures 10 and 11), the performance is improved after adopting a supervised encoding scheme. For instance, the EER is reduced to 0.95% (about 6% relative error reduction) on the data from two mixed sessions. When we employ the images in the first session as templates and the remaining images captured at the second session as testing samples, a EER, namely 2.38% (about 8.1% relative error reduction) is achieved by the SCNN-LSTM + Supervised encoding. The experimental results are explained by this fact. The existing finger-vein encoding approaches do not infer any prior knowledge from the different images because they compute the threshold from each image independently from others or employ some empirical threshold

values such as 0.5 and 0. By contrast, the proposed encoding approach harnesses a rich prior knowledge acquired by maximizing the distance between the genuine score set and impostor score set (as shown in Equation (14)) and the resulting threshold is directly related to verification error reduction. Therefore, our approach can extract the discriminative vein texture for verification. Also, the experimental results show that the supervised encoding shows more significant improvement on the data acquired in two sessions. The reason is that there is not large room for improvement because it is easier to distinguish the images from one session compared to those from two sessions. Actually, the 2-sessions scenario is more realistic so the supervised encoding scheme is effective to reduce the verification error.

Compared to the experimental results in Section 3.5 (Table 1 and Figure 10) and in Section 3.6 (Table 2 and Figure 11), we see that all approaches achieve significant improvement in terms of verification accuracy on image datasets acquired in one session. Such a good performance can be attributed to the fact that there exist smaller within-class variations in the images captured at the same session because the imaging environment is similar and the subjects increase familiarity in the finger presentations during finger-vein image acquisition within a short duration. On the contrary, there are the larger within-class variations for the data acquired in two different sessions, which causes more mismatching errors.

In addition, we also compare our approach with existing approaches with respect to the computational cost. All experiments are carried out in Matlab 2014a and conducted on a high performance computer with 8 Core E3-1270v3 3.5 GHz processor, 16 GB of RAM, and a NVIDIA Quadro GTX1070 graphics cards. For our approach and CNN [33], they are trained with Caffe package [49] on the graphics cards, and tested with Matlab on the central processing unit (CPU). To improve the time cost, we optimize SCNN-LSTM to extract the vein feature of a test image. First, as described in Section 3.2, a test image with size of 39×146 is divided into 39×146 overlapping patches, based on which 39×146 sequences are generated for all pixels along a given orientation using the scheme in Section 2.1.2. Therefore, there are same patches in the sequences of adjacent pixels. If we input the sequence for each pixel into SCNN-LSTM for feature extraction, it results in a lot of repeated feature extraction operations in the CNN model. To further reduce the computation time, 39×146 patches from an image are separately input into CNN model of SCNN-LSTM and we take its output (a 100 dimensional vector) as the feature vector of the input. Then, for each pixel, we arrange resulting vectors along a given orientation to form a sequence, which is forwarded to the LSTM model to extract its spatial dependence feature. Therefore, as each patch is only subject to one feature extraction operation using CNN model, the computational time for our model is significantly reduced in this way. Second, the four SCNN-LSTMs (shown in Figure 5) for four orientations are implemented in parallel to further reduce time cost. For the remaining approaches mentioned in our work, all experiments are implemented in Matlab on CPU. The average verification time of an image using various methods is listed in Table 3. We can see from Table 1 that the proposed method, CNN, and Repeated line tracking approach require more than two seconds to verify a finger-vein image, e.g., 3.25 s, 2.13 s, and 2.53 s, respectively, which are more than those achieved by the remaining approaches. This can be explained by the following fact. The proposed approach and CNN process the patch centered on each pixel and predict its probability of belonging to a vein pattern. When the size of test image is large, it is computationally expensive. The Repeated line tracking approach starts at a seed point and then tracks all vein patterns pixel by pixel by detecting the local dark line. When a dark line is not detectable, a new tracking operation starts at another position. The local line tracking operation is repeatedly performed and the tracking number for each pixel is recorded in a tracking matrix for segmentation. The larger tracking number will enhance the vein pattern and result in high verification accuracy, but the computational cost increases. Overall, our approach shows high time cost, but it can achieve best performance for finger-vein verification (as shown in experimental results in Tables 1 and 2 and Figures 10 and 11). Moreover, these time costs are expected to be significantly reduced after code optimization. For example, implementing these algorithms in C++ can also improve the computation speed. With development of parallel computing technologies

such as CUDA, the computing performance can be dramatically improved by harnessing the power of the graphics processing unit (GPU). Therefore, our approach can achieve computational requirement for practical application after accelerating using GPU.

Table 3. Average computational time of various approaches.

Methods	Time (s)
Repeated line tracking [24]	2.53
Maximum curvature point [15]	1.01
Region growth [27]	0.54
Wide line detector [13]	0.04
Gabor filters [3]	1.96
Mean curvature [14]	0.14
Difference curvature [43]	1.16
CNN [33]	2.13
The proposed approach	3.25

5. Conclusions

In this paper, we proposed an approach to extract the finger-vein pattern for verification. First, a SCNN-LSTM is proposed to predict the probability of a vein pixel belonging to a vein patten. As SCNN-LSTM combines recurrent models such as LSTMs with deep convolutional networks, it can be jointly trained to learn the complex spatial dependencies and convolutional perceptual representations. Second, to improve the performance, we proposed a supervised scheme to encode the vein patterns. As the threshold for encoding is related to verification performance, it can extract robust vein texture features for verification. Experimental results show that the proposed approach extracts robust vein features and significantly improves the verification error rate with respect to state of the art.

As our model can learn the complex spatial dependencies, it extract continuous vein network for verification. Also, our approach is employed to extract the hand-vein and palm-vein for recognition. In medical image analysis, some images such as retinal image, brain segmentation, and neuronal membranes contain continuous texture patterns, so the proposed approach can be applied to segment such texture patterns for disease diagnosis. In addition, if the patterns in vision image show the similar connectivity to vein pattern (as shown in Figure 1), our approach can be used to process vision image. In future work, we will extend the application of our approach to further verify its generalization.

Author Contributions: Conceptualization, H.Q.; methodology, H.Q.; software, P.W.; validation, P.W. and P.W.; formal analysis, H.Q.; investigation, H.Q.; resources, P.W.; data curation, P.W.; writing—original draft preparation, P.W.; writing—review and editing, H.Q.; visualization, P.W.; supervision, H.Q.; project administration, H.Q.; funding acquisition, H.Q.

Funding: This work is supported by the National Natural Science Foundation of China (Grant No. 61402063), the Natural Science Foundation Project of Chongqing (Grant No.cstc2017jcyjAX0002, Grant No.cstc2018jcyjAX0095, Grant No.cstc2013kjrc-qnrc40013,Grant No.cstc2017zdcy-zdyfX0067).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jain, A.; Hong, L.; Bolle, R. On-line fingerprint verification. *IEEE Trans. Pattern Anal. Mach. Intell.* **1997**, *19*, 302–314. [CrossRef]
2. Zhang, D.D.; Kong, W.; You, J.; Wong, M. Online palmprint identification. *IEEE Trans. Pattern Anal. Mach. Intell.* **2003**, *25*, 1041–1050. [CrossRef]
3. Kumar, A.; Zhou, Y. Human identification using finger images. *IEEE Trans. Image Process.* **2012**, *21*, 2228–2244. [CrossRef] [PubMed]
4. Yang, L.; Yang, G.; Xi, X.; Su, K.; Chen, Q.; Yin, Y. Finger Vein Code: From Indexing to Matching. *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 1210–1223. [CrossRef]

5. Kumar, A.; Prathyusha, K.V. Personal authentication using hand vein triangulation and knuckle shape. *IEEE Trans. Image Process.* **2009**, *18*, 2127–2136. [CrossRef]
6. Zhou, Y.; Kumar, A. Human identification using palm-vein images. *IEEE Trans. Inf. Forensics Secur.* **2011**, *6*, 1259–1274. [CrossRef]
7. Turk, M.A.; Pentland, A.P. Face recognition using eigenfaces. In Proceedings of the 1991 IEEE Computer Society Conference on Computer Vision and Pattern Recognitio, Maui, HI, USA, 3–6 June 1991; pp. 586–591.
8. Daugman, J. How iris recognition works. *IEEE Trans. Circuits Syst. Video Technol.* **2004**, *14*, 21–30. [CrossRef]
9. Ramírez, J.; Segura, J.C.; Górriz, J.M.; García, L. Improved voice activity detection using contextual multiple hypothesis testing for robust speech recognition. *IEEE Trans. Audio Speech Lang. Process.* **2007**, *15*, 2177–2189. [CrossRef]
10. El-Yacoubi, M.A.; Gilloux, M.; Bertille, J.M. A statistical approach for phrase location and recognition within a text line: An application to street name recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2002**, *24*, 172–188. [CrossRef]
11. Lu, Y.; Xie, S.J.; Yoon, S.; Yang, J.; Park, D.S. Robust finger vein ROI localization based on flexible segmentation. *Sensors* **2013**, *13*, 14339–14366. [CrossRef] [PubMed]
12. Hashimoto, J. Finger vein authentication technology and its future. In Proceedings of the 2006 Symposium on VLSI Circuits, Honolulu, HI, USA, 15–17 June 2006; pp. 5–8.
13. Huang, B.; Dai, Y.; Li, R.; Tang, D.; Li, W. Finger-vein authentication based on wide line detector and pattern normalization. In Proceedings of the 2010 20th International Conference on Pattern Recognition, Istanbul, Turkey, 23–26 August 2010; pp. 1269–1272.
14. Song, W.; Kim, T.; Kim, H.C.; Choi, J.H.; Kong, H.J.; Lee, S.R. A finger-vein verification system using mean curvature. *Pattern Recognit. Lett.* **2011**, *32*, 1541–1547. [CrossRef]
15. Miura, N.; Nagasaka, A.; Miyatake, T. Extraction of finger-vein patterns using maximum curvature points in image profiles. *IEICE Trans. Inf. Syst.* **2007**, *90*, 1185–1194. [CrossRef]
16. Yang, J.; Shi, Y. Towards finger-vein image restoration and enhancement for finger-vein recognition. *Inf. Sci.* **2014**, *268*, 33–52. [CrossRef]
17. Lee, E.C.; Park, K.R. Image restoration of skin scattering and optical blurring for finger vein recognition. *Opt. Lasers Eng.* **2011**, *49*, 816–828. [CrossRef]
18. Yang, W.; Huang, X.; Zhou, F.; Liao, Q. Comparative competitive coding for personal identification by using finger vein and finger dorsal texture fusion. *Inf. Sci.* **2014**, *268*, 20–32. [CrossRef]
19. Yang, J.; Shi, Y. Finger–vein ROI localization and vein ridge enhancement. *Pattern Recognit. Lett.* **2012**, *33*, 1569–1579. [CrossRef]
20. Yang, J.; Shi, Y.; Yang, J. Finger-vein recognition based on a bank of Gabor filters. In Proceedings of the Asian Conference on Computer Vision, Xi'an, China, 23–27 September 2009; pp. 374–383.
21. Yu, C.B.; Qin, H.F.; Cui, Y.Z.; Hu, X.Q. Finger-vein image recognition combining modified hausdorff distance with minutiae feature matching. *Interdiscip. Sci. Comput. Life Sci.* **2009**, *1*, 280–289. [CrossRef]
22. Chaudhuri, S.; Chatterjee, S.; Katz, N.; Nelson, M.; Goldbaum, M. Detection of blood vessels in retinal images using two-dimensional matched filters. *IEEE Trans. Med. Imaging* **1989**, *8*, 263–269. [CrossRef]
23. Qin, H.; He, X.; Yao, X.; Li, H. Finger-vein verification based on the curvature in Radon space. *Expert Syst. Appl.* **2017**, *82*, 151–161. [CrossRef]
24. Miura, N.; Nagasaka, A.; Miyatake, T. Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification. *Mach. Vis. Appl.* **2004**, *15*, 194–203. [CrossRef]
25. Liu, T.; Xie, J.; Yan, W.; Li, P.; Lu, H. An algorithm for finger-vein segmentation based on modified repeated line tracking. *Imaging Sci. J.* **2013**, *61*, 491–502. [CrossRef]
26. Gupta, P.; Gupta, P. An accurate finger vein based verification system. *Digit. Signal Process.* **2015**, *38*, 43–52. [CrossRef]
27. Qin, H.; Qin, L.; Yu, C. Region growth–based feature extraction method for finger-vein recognition. *Opt. Eng.* **2011**, *50*, 057208. [CrossRef]
28. Yang, L.; Yang, G.; Yin, Y.; Xi, X. Finger vein recognition with anatomy structure analysis. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *28*, 1892–1905. [CrossRef]
29. Ciresan, D.; Giusti, A.; Gambardella, L.M.; Juergen, S. Deep neural networks segment neuronal membranes in electron microscopy images. In Proceedings of the 25th International Conference on Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 2843–2851.

30. Guo, Y.; Gao, Y.; Shen, D. Deformable MR Prostate Segmentation via Deep Feature Learning and Sparse Patch Matching. *IEEE Trans. Med. Imaging* **2016**, *35*, 1077–1089. [CrossRef] [PubMed]
31. Liskowski, P.; Krawiec, K. Segmenting Retinal Blood Vessels with Deep Neural Networks. *IEEE Trans. Med. Imaging* **2016**, *35*, 2369–2380. [CrossRef]
32. Zhang, W.; Li, R.; Deng, H.; Wang, L.; Lin, W.; Ji, S.; Shen, D. Deep convolutional neural networks for multi-modality isointense infant brain image segmentation. *Neuro Image* **2015**, *108*, 214–224. [CrossRef]
33. Qin, H.; El-Yacoubi, M.A. Deep representation-based feature extraction and recovering for finger-vein verification. *IEEE Trans. Inf. Forensics Secur.* **2017**, *12*, 1816–1829. [CrossRef]
34. Han, T.Z.; Yu, X. Anatomical study and clinical application of superficial palmar digital veins in finger replantation. *Chin. J. Clin. Anat.* **1997**, *15*, 39–41.
35. Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015. pp. 2625–2634.
36. Du, Y.; Wang, W.; Wang, L. Hierarchical recurrent neural network for skeleton based action recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1110–1118.
37. Graves, A.; Mohamed, A.R.; Hinton, G. Speech recognition with deep recurrent neural networks. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, BC, Canada, 26–31 May 2013; pp. 6645–6649.
38. Graves, A.; Schmidhuber, J. Offline handwriting recognition with multidimensional recurrent neural networks. In Proceedings of the 21st International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 7–10 December 2009; pp. 545–552.
39. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
40. Zhu, W.; Lan, C.; Xing, J.; Zeng, W.; Li, Y.; Shen, L.; Xie, X. Co-Occurrence Feature Learning for Skeleton Based Action Recognition Using Regularized Deep LSTM Networks. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; Volume 2, p. 6.
41. Byeon, W.; Breuel, T.M.; Raue, F.; Liwicki, M. Scene labeling with lstm recurrent neural networks. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 3547–3555.
42. Jain, A.; Zamir, A.R.; Savarese, S.; Saxena, A. Structural-RNN: Deep learning on spatio-temporal graphs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 5308–5317.
43. Qin, H.; Qin, L.; Xue, L.; He, X.; Yu, C.; Liang, X. Finger-vein verification based on multi-features fusion. *Sensors* **2013**, *13*, 15048–15067. [CrossRef]
44. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *NIPS* **2012**, *25*, 1097–1105. [CrossRef]
45. Platt, J. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Adv. Large Margin Classif.* **1999**, *10*, 61–74.
46. Taigman, Y.; Yang, M.; Ranzato, M.; Wolf, L. Deepface: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1701–1708.
47. Hamming, R.W. Error detecting and error correcting codes. *Bell Syst. Tech. J.* **1950**, *29*, 147–160. [CrossRef]
48. Macmillan, N.; Creelman, C. *Detection Theory: A User's Guide*; Lawrence Erlbaum: Mahwah, NJ, USA, 2005.
49. Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Girshick, R.; Guadarrama, S.; Darrell, T. Caffe: Convolutional architecture for fast feature embedding. In Proceedings of the 22nd ACM International Conference on Multimedia, Orlando, FL, USA, 3–7 November 2014; pp. 675–678.