Pattern Recognition Letters 119 (2019) 148–156

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

# Finger vein identification using Convolutional Neural Network and supervised discrete hashing

Cihui Xie, Ajay Kumar*

*Department of Computing, The Hong Kong Polytechnic University, Hung Hum, Hong Kong*

ARTICLE INFO

*Article history:*
Available online 31 January 2018

ABSTRACT

Automated personal identification using vascular biometrics, such as from the finger vein images, is highly desirable as it helps to protect the personal privacy and anonymity in automated personal identification. The Convolutional Neural Network (CNN) has shown remarkable capability for learning biometric features that can offer robust and accurate matching. This paper introduces a new approach for the finger vein authentication using the CNN and supervised discrete hashing. We also systematically investigate comparative performance using several popular CNN architectures in other domains, i.e., Light CNN, VGG-16, Siamese and the CNN with Bayesian inference based matching. The experimental results are presented using a publicly available two-session finger-vein database. Most accurate performance is achieved by incorporating supervised discrete hashing from a CNN trained using the triplet-based loss function. The proposed approach not only achieves outperforming results over other considered CNN architecture available in the literature but also offers significantly reduced template size as compared with those over the other finger-vein images matching methods available in the literature.



## 1. Introduction

Automated personal identification using unique anatomical characteristics of humans is widely employed for e-governance, border crossing security and a range of *e*-business applications. There has been significant increase in the detection of surgically altered fingerprints, fake iris stamps, or the usage of sophisticated face masks, during the last decade. Vascular biometrics identification, like using finger vein images, can help to preserve the integrity of biometrics system as it's extremely difficult to surgically alter vascular biometrics. Another advantage with the usage of finger vein image based identification lies in the enhanced anonymity during personal authentication as the subsurface vascular patterns are largely hidden underneath and difficult to steal or imaged under visible illumination.

The possibility of personal identification using vascular patterns imaged by the light transmitted through hands was indicated in 1992 [18] but was not demonstrated until 2000 [19]. Such earliest work demonstrated feasibility of finger vein identification using the normalized-cross correlation. Miura et al. [9] later introduced repeated line tracking approach to improve the performance of finger vein identification, and they further enhanced the performance with maximum curvature [2]. Kumar and Zhou [4] introduced earliest publicly accessible finger-vein images database and comparatively evaluated range the effectiveness of hand-crafted features for the finger-vein identification. The method introduced in [4] using Gabor filter based enhancement and morphological operations is still regarded the best performing methods for matching finger-vein images. A range of other hand-crafted finger vein features [2,4,9,11–15,20], primarily obtained from the careful evaluation of the registered images, have been attempted in the literature with very promising results. Multiple features acquired from the two cameras [13] or using multiple feature extractors [17] can be combined to significantly improve the performance for the vascular biometrics matching. One of the limitations of finger-vein identification methods introduced in the literature is related to their large template size. Smaller template size is highly desirable to reduce storage and/or enhance the matching speed for the online applications. There have also been successful attempts to reduce the finger-vein template size, like in [11,14] or recently in [12] using sparse representation of enhanced finger-vein images using the Gabor filters. Table 1 presents summary of some promising methods for finger vein matching in the literature. This table also presents the template size in the respective reference (several of these have been estimated from details provided in respective reference), performance in terms of EER and the database used for the performance evaluation. The last two rows in this table summarize best performing results from our investigation detailed in this work.

* Corresponding author.
*E-mail address:* csajaykr@comp.polyu.edu.hk (A. Kumar).

https://doi.org/10.1016/j.patrec.2017.12.001
0167-8655/

**Table 1**
Summary of hand-crafted feature based methods for finger vein matching in the literature with this work.

| Ref. | Feature | Database | Two session | Template size (bytes) | EER | No. of subjects | No. of genuine scores | No. of impostor scores |
|---|---|---|---|---|---|---|---|---|
| [4] | Hand Crafted (Even Gabor) | Public | Yes | 106,384 | 4.61% | 105 | 1260 | 263,340 |
| [4] | Hand Crafted (Morphological) | Public | Yes | 6710 | 4.99% | 105 | 1260 | 263,340 |
| [9] | Hand Crafted (Repeated Line Tracking) | Proprietary | No | 43,200* | 0.145% | 678 | 678* | 458,328* |
| [2] | Hand Crafted (Maximum Curvature) | Proprietary | No | 43,200* | 0.0009% | 678 | 678* | 458,328* |
| [11] | Hand Crafted (Local Binary Pattern) | Public | No | $\leq$ 260* | 3.53% | 156 | 624 | 194,064 |
| [12] | Hand Crafted (Sparse Representation using l1-norm) | Proprietary | No | 630* | Unknown | 17 | Unknown | Unknown |
| [14] | Hand Crafted (Extended Local Binary Pattern) | Public | Yes | 131,328* | 7.22% | 105 | 1260 | 263,340 |
| [15] | Hand Crafted (Unknown Algorithm) | Proprietary | Yes | 20,480* | 0.77% | Unknown | 10,000 | 499,500 |
| [20] | Hand Crafted (Histogram of Salient Edge Orientation Map) | Public | No | $\leq$3496* | 0.9% | 100 | 3000 | 1,797,000 |
| Ours | CNN with triplet similarity loss | Public | Yes | 1024 | 13.16% | 105 | 1260 | 263,340 |
| Ours | Supervised discrete hashing with CNN | Public | Yes | 250 | 9.77% | 105 | 1260 | 263,340 |

* Computed by us from the details available in the respective reference.

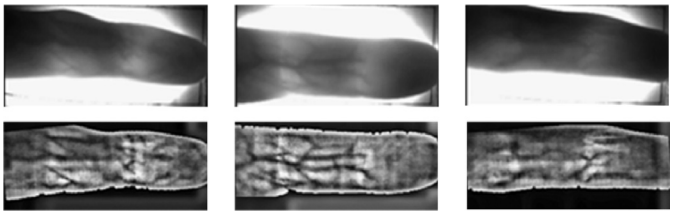

**Fig. 1.** Finger vein image samples before (first row and after preprocessing (second row) steps that automatically extracts and enhances the region of interest images.

The key objective of this work is to investigate and develop advanced capabilities for matching finger-vein images using deep learning. Section 2 details preprocessing and image enhancement steps incorporated in the experiments for evaluating effectiveness of various CNN architectures which are detailed in Section 3. The supervised hashing is introduced in Section 4 while the experimental protocols and results [21] are discussed in Section 5. The discussion on some of our findings appears is Section 6. The key conclusions from this paper are summarized in Section 7.

## 2. Image normalization for finger vein images

### 2.1. Preprocessing

Finger vein image acquisition can introduce translational and rotational changes among the different images from the same finger or subject. Therefore, automated extraction of fixed region of interest (ROI) that can minimize such intra-class variations is highly desirable. The method of ROI localization considered in our work is same as detailed in [4]. Fig. 1 illustrates acquired image sample, extracted ROI and the image sample after enhancement to improve the image contrast (as detailed in [4].

### 2.2. Image enhancement

The vascular patterns in the normalized image samples can be further enhanced by spatial filtering from orientation selective band pass filters, similar as employed for the enhancement of fingerprint ridges. We also attempted to ascertain usefulness of such enhanced finger vein images using the Gabor filters. These filters from the twelve different orientations are selected to generate enhanced finger vein images as shown in Fig. 2. Such enhanced images using Gabor filters are effective in accentuating the vascular features and therefore its possible usage in automatically learning features from CNN was also investigated in the experiments.

**Fig. 2.** Enhanced finger vein images to emphasize on vascular features and suppress associated noise. These Gabor filter based enhanced images are respective to ROI in Fig. 1.

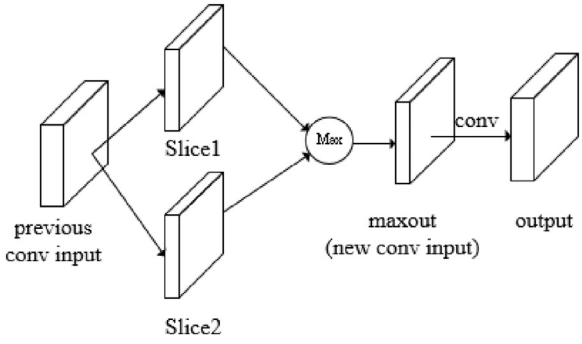


**Fig. 3.** Illustration for computing the Max feature map in LCNN.

## 3. Convolutional Neural Network architectures

Several successful models for the deep learning have been developed to learn useful feature representation but largely for the face biometric image patterns. A variety of such models using CNN have been introduced in the literature and were investigated to ascertain performance for the finger vein image matching. A brief introduction to various CNN architectures considered in this work is provided in the following sections.

### 3.1. Light CNN

The light CNN (LCNN) framework introduces a Max-Feature-Map (MFM) operation [3] between convolutional layers which establishes a competitive relationship for superior generalization capability and reduces parameter space (compact feature representation). Such *maxout* activation function (Fig. 3) significantly reduces the complexity and makes CNN lighter.

The architecture of LCNN employed in our experiments is shown in Fig. 4 (MFM part is excluded to maintain the clarity). The network contains 9 convolutional layers (conv), 4 pooling layers (pooling) and 2 fully connected layers (fc) and some assistant layers.

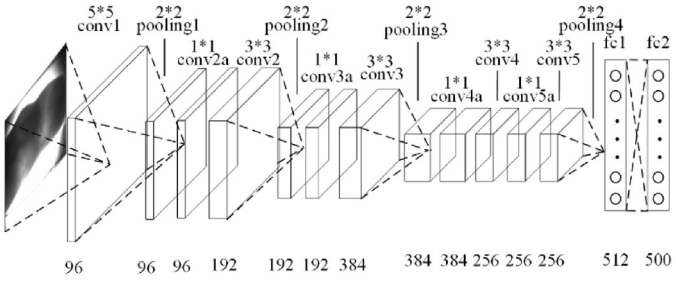


**Fig. 4.** The architecture for LCNN investigated in our experiments.

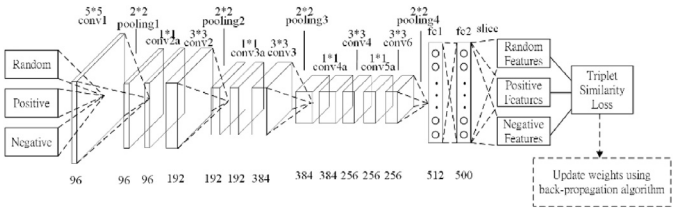


**Fig. 5.** The LCNN architecture with triplet similarity loss function.

### 3.2. LCNN with triplet similarity loss function

Deep Siamese networks have been successfully incorporated in the literature to learn a similarity metric between a pair of images. We incorporated triplet similarity loss function as detailed in [6] for LCNN to learn the similarity metric. We randomly select an image $x^r$ from training set as random sample in Fig. 5. Then we choose image $x^p$ which is from the same class referred to as positive sample and image $x^n$ which is from a different class referred to as negative. After LCNN, we get the features $f(x^r)$, $f(x^n)$ and $f(x^p)$. Our objective is to decrease the similarity distance between random and positive features, and increase it between random and negative features, which indicates why it's named as triplet similarity loss. At the same time, we need to ensure there is a sufficient *margin* between them.

Suppose we have a random set $\mathbf{X^r} = \{x_i^r\}_{i=1}^{\boldsymbol{N}}$ and its corresponding positive set $\mathbf{X^p} = \{x_i^p\}_{i=1}^{\boldsymbol{N}}$ and negative set $\mathbf{X^n} = \{x_i^n\}_{i=1}^{\boldsymbol{N}}$. Considering these notations, we can write our loss function as follows:

$$\sum_{i=1}^{N}\left[\left\| f\left(x_i^r\right) - f\left(x_i^p\right)\right\|^2 - \left\| f\left(x_i^r\right) - f\left(x_i^n\right)\right\|^2 + margin\right]_+ \tag{1}$$

where $[]_+$ represents that we maintain positive values and change others to zero. The detailed architecture of LCNN with such triplet loss function is shown in Fig. 5 and Table 2. When the set of input consists of $n$ random samples, $n$ positives and $n$ negatives, we generate $3n \times 500$ features. These pairs were split into three parts, each with the size of $n \times 500$, and used as the input for computing triplet loss for updating the neuron weights during the network training.

### 3.3. Modified VGG-16

The Visual Geometry Group architecture with 16 layers (VGG-16) [7] was modified for the CNN to directly recover the match scores, instead of the feature vectors, in our experiment. Our modification was motivated to fit the rectangular finger vein ROI images without introducing the distortions. We used pair of images rather than single image as the input in conventional VGG-16 since we wanted to compare the similarity of two finger vein images. The input image size is also different from conventional VGG-16, which is $224 \times 224$, while its $128 \times 488$ pixels for our finger vein ROI images. The training phase utilized the cross-entropy loss function

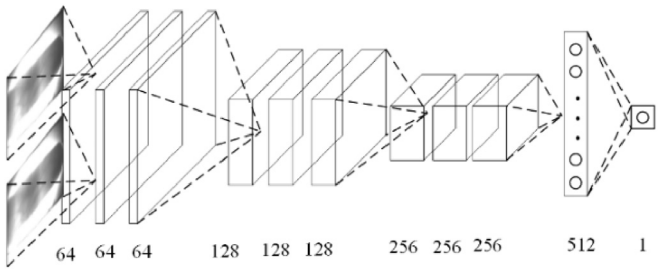


**Fig. 6.** Architecture of modified VGG-16 for finger-vein matching.

which can be written as follows:

$$-\frac{1}{n}\sum_{i=1}^{n}[y_i \log(\widehat{\boldsymbol{y}}_i) + (1 - y_i)\log(1 - \widehat{\boldsymbol{y}}_i)] \tag{2}$$

where $\widehat{\boldsymbol{y}}_i = g(\boldsymbol{w^T x_i})$, $g(\cdot)$ is the logistic function, $\boldsymbol{x_i}$ is the extracted feature and w is the weight that needs optimized during training. The architecture of Modified VGG-16 (MVGG) is illustrated in the following Fig. 6.

## 4. Supervised discrete hashing

One of the key challenges for the successful usage of biometrics technologies are related to efficient search speed (fast retrieval) and template storage/size. Hashing is one of the most effective approaches to address such challenges and can efficiently encode the biometrics templates using binary numbers (2000 in our experiments) that closely reflect the similarity with input data/templates. With such strategy we can only store the corresponding short/compact binary codes, instead of original feature templates, and significantly improve the search or the matching speed by highly efficient pairwise comparison using the Hamming distance.

This framework for an effective supervised hashing scheme is introduced in [1] and the objective in the learning phase is to generate binary codes for the linear classification. We firstly define the problem and assume that we have $n$ samples/features $\mathbf{X} = [\mathbf{x}_1\ \mathbf{x}_2 \ldots\ \mathbf{x}_n]$, and our goal is to recover corresponding binary codes $\mathbf{B} = [\mathbf{b}_1\ \mathbf{b}_2 \ldots\ \mathbf{b}_n]$, where $\mathbf{b}_i \in \{-1,\ 1\}$, $i = 1,\ 2,\ \ldots,\ n$. Since we have labels, in order to make good use of these information, we define a multi-class classification function:

$$\mathbf{y} = \mathbf{W}^{\mathrm{T}}\mathbf{b} \text{ where } \mathbf{W} = [\mathbf{w}_1\ \ \mathbf{w}_2\ \ \ldots\ \ \mathbf{w}_C], \tag{3}$$

where $C$ is the total number of classes, and $\mathbf{y} \in \mathbb{R}^{C\times 1}$ is the label vector, where the maximum one indicates its class of input $\mathbf{x}$. Now we can formulate the hashing problem as follows:

$$\min_{\boldsymbol{B},\boldsymbol{W},F}\sum_{i=1}^{n} L(\boldsymbol{y}_i,\ \ \mathbf{W^T b}_i) + \ \lambda\|\mathbf{W}\|^2, \quad \text{s.t. } \mathbf{b}_i = \mathrm{sgn}(F(\mathbf{x}_i)) \tag{4}$$

where $L(\cdot)$ represents the loss function used by us which is the L2-norm in our experiments, $\lambda$ is the regularization parameter, and at the same time, $\mathbf{b}_i$ is generated by the hash function $\mathrm{sgn}(F(\mathbf{x}_i))$, where $\mathrm{sgn}(\cdot)$ is the sign function. With the help of Lagrange Multiplier, we then can rewrite (4) as:

$$\min_{\boldsymbol{B},\boldsymbol{W},F}\sum_{i=1}^{n} L(\mathrm{y_i},\ \ \boldsymbol{W^T b}_i) + \ \lambda\|\boldsymbol{W}\|^2 + \mu\ \sum_{i=1}^{n}\|\boldsymbol{b}_i - F(\boldsymbol{x}_i)\|^2 \tag{5}$$

where $\mu$ is the Lagrange multiplier. We further select a non-linear form of function for $F(\mathbf{x})$:

$$F(\mathbf{x}) = \ \mathbf{U}^{\mathrm{T}}\phi(\mathbf{x}) \tag{6}$$

**Table 2**
Details of layer information of LCNN with triplet loss function.

| Index | Type | Input index | Output index | Filter size | Output size | Stride | Pad |
|---|---|---|---|---|---|---|---|
| 1 | DataR | – | 4 | – | $N^*W^*H$ | – | – |
| 2 | DataP | – | 4 | – | $N^*W^*H$ | – | – |
| 3 | DataN | – | 4 | – | $N^*W^*H$ | – | – |
| 4 | Data | 1,2,3 | 5 | – | $3N^*W^*H$ | – | – |
| 5 | conv1 | 4 | 6 | 5 | $3N^*96$ | 1 | 2 |
| 6 | MFM1 | 5 | 7 | – | $3N^*48$ | – | – |
| 7 | pooling1 | 6 | 8 | 2 | – | 2 | 0 |
| 8 | conv2a | 7 | 9 | 1 | $3N^*96$ | 1 | 0 |
| 9 | MFM2a | 8 | 10 | – | $3N^*48$ | – | – |
| 10 | conv2 | 9 | 11 | 3 | $3N^*192$ | 1 | 1 |
| 11 | MFM2 | 10 | 12 | – | $3N^*96$ | – | – |
| 12 | pooling2 | 11 | 13 | 2 | – | 2 | 0 |
| 13 | conv3a | 12 | 14 | 1 | $3N^*192$ | 1 | 1 |
| 14 | MFM3a | 13 | 15 | – | $3N^*96$ | – | – |
| 15 | conv3 | 14 | 16 | 3 | $3N^*384$ | 1 | 1 |
| 16 | MFM3 | 15 | 17 | – | $3N^*192$ | – | – |
| 17 | pooling3 | 16 | 18 | 2 | – | 2 | 0 |
| 18 | conv4a | 17 | 19 | 1 | $3N^*384$ | 1 | 1 |
| 19 | MFM4a | 18 | 20 | – | $3N^*192$ | – | – |
| 20 | conv4 | 19 | 21 | 3 | $3N^*256$ | 1 | 1 |
| 21 | MFM4 | 20 | 22 | – | $3N^*128$ | – | – |
| 22 | conv5a | 21 | 23 | 1 | $3N^*256$ | 1 | 1 |
| 23 | MFM5a | 22 | 24 | – | $3N^*128$ | – | – |
| 24 | conv5 | 23 | 25 | 3 | $3N^*256$ | 1 | 1 |
| 25 | MFM5 | 24 | 26 | – | $3N^*128$ | – | – |
| 26 | pooling4 | 25 | 27 | 2 | – | 2 | 0 |
| 27 | fc1 | 26 | 28 | – | $3N^*512$ | – | – |
| 28 | MFMfc | 27 | 29 | – | $3N^*256$ | – | – |
| 29 | fc2 | 28 | 30,31,32 | – | $3N^*500$ | – | – |
| 30 | SliceR | 29 | 33 | – | $N^*500$ | – | – |
| 31 | SliceP | 29 | 33 | – | $N^*500$ | – | – |
| 32 | SliceN | 29 | 33 | – | $N^*500$ | – | – |
| 33 | Loss | 30,31,32 | – | – | – | – | – |

where $\mathbf{U}$ is the parameter matrix and $\phi(\mathbf{x})$ is a k-dimensional kernel that

$$\phi(\mathbf{x}) = \begin{bmatrix} \exp\left(\frac{\|\mathbf{x}-\boldsymbol{a}_1\|^2}{\sigma}\right) \\ \cdots \\ \exp\left(\frac{\|\mathbf{x}-\boldsymbol{a}_k\|^2}{\sigma}\right) \end{bmatrix} \tag{7}$$

$\boldsymbol{a}_j,\ j = 1,\ 2,\ \ldots,\ k$, are randomly selected anchor vectors from input. In order to compute $\boldsymbol{U}$ in the function, we can rewrite (4) as following format:

$$\min_{\mathbf{U}} \sum_{i=1}^{n} \|\mathbf{b}_i - F(\mathbf{x}_i)\|^2 = \min_{\mathbf{U}} \left\|\mathbf{U}^{\mathrm{T}}\Phi(\mathbf{X}) - \mathbf{B}\right\|^2 \tag{8}$$

where $\Phi(\mathbf{X}) = \{\phi(\mathbf{x}_i)\}_{i=1}^{n}$ and our purpose is to set the gradient to zero, which is

$$\nabla_{\mathbf{U}}\left(\left\|\mathbf{U}^{\mathrm{T}}\Phi(\mathbf{X}) - \mathbf{B}\right\|^2\right) = 2\left(\mathbf{U}^{\mathrm{T}}\Phi(\mathbf{X}) - \mathbf{B}\right)\Phi(\mathbf{X})^{\mathrm{T}} = 0 \tag{9}$$

It is simpler to achieve the final computation for $\mathbf{U}$ as follows.

$$\mathbf{U} = \left(\Phi(\mathbf{X})\Phi(\mathbf{X})^{\mathrm{T}}\right)^{-1}\Phi(\mathbf{X})\mathbf{B}^{\mathrm{T}} \tag{10}$$

In order to solve for $\mathbf{W}$, we make use of the same method, first simplify (4) to

$$\min_{\mathbf{W}} \sum_{i=1}^{n} L(\boldsymbol{y}_i, \mathbf{W}^{\mathrm{T}}\mathbf{b}_i) + \lambda\|\mathbf{W}\|^2 = \min_{\mathbf{W}} \left\|\mathbf{Y} - \mathbf{W}^{\mathrm{T}}\mathbf{B}\right\|^2 + \lambda\|\mathbf{W}\|^2 \tag{11}$$

and then calculate its gradient based on $\mathbf{W}$

$$\nabla_{\mathbf{W}}\left(\left\|\mathbf{Y} - \mathbf{W}^{\mathrm{T}}\mathbf{B}\right\|^2 + \lambda\|\mathbf{W}\|^2\right) = 2\mathbf{B}\left(\mathbf{B}^{\mathrm{T}}\mathbf{W} - \mathbf{Y}^{\mathrm{T}}\right) + 2\lambda\mathbf{W}, \tag{12}$$

which can be set as zero and we get

$$\mathbf{W} = \left(\mathbf{B}\mathbf{B}^{\mathrm{T}} + \lambda\mathbf{I}\right)^{-1}\mathbf{B}\mathbf{Y}^{\mathrm{T}} \tag{13}$$

Finally we can solve for $\mathbf{B}$, we exclude those variables which have no relation to $\mathbf{B}$ and then rewrite (4) as follows.

$$\begin{aligned} &\min_{\mathbf{B}} \left\|\mathbf{Y} - \mathbf{W}^{\mathrm{T}}\mathbf{B}\right\|^2 + \mu\|\mathbf{B} - F(\mathbf{X})\|^2 \\ &= \min_{\mathbf{B}} \|\mathbf{Y}\|^2 - 2\mathrm{tr}\left(\mathbf{Y}^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\mathbf{B}\right) + \left\|\mathbf{W}^{\mathrm{T}}\mathbf{B}\right\|^2 \\ &+\mu\left(\|\mathbf{B}\|^2 + 2\mathrm{tr}\left(\mathbf{B}^{\mathrm{T}}F(\mathbf{X})\right) + F(\mathbf{X})^2\right) \end{aligned} \tag{14}$$

or can be further simplified as follows.

$$\min_{\mathbf{B}} \left\|\mathbf{W}^{\mathrm{T}}\mathbf{B}\right\|^2 - 2\mathrm{tr}\left(\mathbf{B}^{\mathrm{T}}(F(\mathbf{X}) + \mathbf{W}\mathbf{Y})\right) \tag{15}$$

$\|\mathbf{B}\|^2$ is excluded here because $\mathbf{b}_i \in \{-1,\ 1\},\ i = 1,\ 2,\ \ldots,\ n$, indicating that $\|\mathbf{B}\|^2$ is some constant.

We can now solve this problem *bit-by-bit*. Let $\mathbf{p}^{\mathrm{T}}$ represent the $l$th row of $\mathbf{B}$, and $\mathbf{B}'$ is the matrix without $\mathbf{p}^{\mathrm{T}}$. Similarly let $\mathbf{v}^{\mathrm{T}}$ be the $l$th row of $\mathbf{W}$, and let $\mathbf{q}^{\mathrm{T}}$ be the $l$th row of $\mathbf{Q}$, where $\mathbf{Q} = F(\mathbf{X}) + \mathbf{W}\mathbf{Y}$, then we can ignore $\mathbf{W}'$ and $\mathbf{Q}'$. While moving a row to the end for all matrices would not cause problems, to better understand the problem. In order to enhance clarity of the problem, we can move all the $l$th row to the end and rewrite $\mathbf{B} = [\mathbf{B}'\quad \mathbf{p}^{\mathrm{T}}]^T$, and the same for $\mathbf{W}$ and $\mathbf{Q}$. We can then rewrite first term in (15) as follows.

$$\begin{aligned} \left\|\mathbf{W}^{\mathrm{T}}\mathbf{B}\right\|^2 &= \left\|\begin{bmatrix}\mathbf{W}'^{\mathrm{T}} & \mathbf{v}\end{bmatrix}\begin{bmatrix}\mathbf{B}' \\ \mathbf{p}^{\mathrm{T}}\end{bmatrix}\right\|^2 \\ &= \left\|\mathbf{W}'^{\mathrm{T}}\mathbf{B}'\right\|^2 + \left\|\mathbf{v}\mathbf{p}^{\mathrm{T}}\right\|^2 + 2\mathrm{tr}\left(\mathbf{B}'^{\mathrm{T}}\mathbf{W}'\mathbf{v}\mathbf{p}^{\mathrm{T}}\right) \\ &= \left\|\mathbf{W}'^{\mathrm{T}}\mathbf{B}'\right\|^2 + \mathrm{tr}\left(\mathbf{v}\mathbf{p}^{\mathrm{T}}\mathbf{p}\mathbf{v}^{\mathrm{T}}\right) + 2\left(\mathbf{W}'\mathbf{v}\right)^{\mathrm{T}}\mathbf{B}'\mathbf{p} \end{aligned} \tag{16}$$

While $\|\mathbf{W}'^{\mathrm{T}}\mathbf{B}'\|^2 + \mathrm{tr}(\mathbf{v}\mathbf{p}^{\mathrm{T}}\mathbf{p}\mathbf{v}^{\mathrm{T}})$ is equal to some constant, and because our goal is to solve for $\mathbf{p}$, we can regard other

parts as constant. $\|\mathbf{v}\mathbf{p}^{\mathrm{T}}\|^2$ is omitted because $\|\mathbf{v}\mathbf{p}^{\mathrm{T}}\|^2 = \|\mathbf{p}\mathbf{v}^{\mathrm{T}}\|^2 = \mathrm{tr}(\mathbf{v}\mathbf{p}^{\mathrm{T}}\mathbf{p}\mathbf{v}^{\mathrm{T}}) = n\ \mathrm{tr}(\mathbf{v}\mathbf{v}^{\mathrm{T}})$, where $\mathbf{p}^{\mathrm{T}}\mathbf{p} = n$. And the other part can be simplified as follows.

$$\begin{aligned}\mathrm{tr}(\mathbf{B}^{\mathbf{T}}\mathbf{Q}) &= \mathrm{tr}(\mathbf{B}'^{\mathbf{T}}\mathbf{Q}' + \mathbf{p}\mathbf{q}^{\mathrm{T}}) = \mathrm{tr}(\mathbf{B}'^{\mathbf{T}}\mathbf{Q}') + \mathrm{tr}(\mathbf{p}\mathbf{q}^{\mathrm{T}}) \\ &= \mathrm{tr}(\mathbf{B}'^{\mathbf{T}}\mathbf{Q}') + \mathbf{q}^{T}\boldsymbol{p} \end{aligned} \tag{17}$$

Combining these terms, we can rewrite (15) as follows.

$$\min_{\mathbf{B}} \mathbf{v}^{\mathrm{T}}\ \mathbf{W}'^{\mathrm{T}}\ \mathbf{B}'\mathbf{p} - \mathbf{q}^{\mathrm{T}}\mathbf{p} = \min_{\mathbf{B}} (\mathbf{v}^{\mathrm{T}}\ \mathbf{W}'^{\mathrm{T}}\ \mathbf{B}' - \mathbf{q}^{\mathrm{T}})\mathbf{p} \tag{18}$$

This is an optimization problem, and $\mathbf{p} \in \{-1,\ 1\}^n$, therefore we just need to incorporate the opposite sign of its first argument.

$$\mathbf{p} = -\mathrm{sgn}(\mathbf{v}^{\mathrm{T}}\ \mathbf{W}'^{\mathrm{T}}\ \mathbf{B}' - \mathbf{q}^{\mathrm{T}}) = \mathrm{sgn}(\mathbf{q}^{\mathrm{T}} - \mathbf{v}^{\mathrm{T}}\ \mathbf{W}'^{\mathrm{T}}\ \mathbf{B}') \tag{19}$$

We can now explicitly *outline* computed parts in the following.

$$\mathbf{W} = (\mathbf{B}\mathbf{B}^{\mathrm{T}} + \lambda\mathbf{I})^{-1}\mathbf{B}\mathbf{Y}^{\mathrm{T}}, \tag{13}$$
$$\mathbf{U} = (\Phi(\mathbf{X})\Phi(\mathbf{X})^{\mathrm{T}})^{-1}\Phi(\mathbf{X})\mathbf{B}^{\mathrm{T}} \tag{10}$$
$$\mathbf{p} = \mathrm{sgn}(\mathbf{q}^{\mathrm{T}} - \mathbf{v}^{\mathrm{T}}\ \mathbf{W}'^{\mathrm{T}}\ \mathbf{B}') \tag{19}$$

Shen et al. [1] have provided another computation based on the hinge loss. However for the simplicity, we incorporated L2-norm in our experiments and therefore this part has been excluded here. We now have the required equations here and can summarize our algorithm as follows.

**Algorithm:** Supervised discrete hashing

Input: Training data $\{\mathbf{X},\ \mathbf{Y}\}$
Output: Binary codes $\mathbf{B}$
1. Randomly select k anchors $\boldsymbol{a}_j,\ j = 1,\ 2,\ \ldots,\ k$, from $\mathbf{X}$ and calculate $\Phi(\mathbf{X})$
2. Randomly initiate $\mathbf{B}$
3. Loop until converge or reach maximum iterations
   * Calculate $\mathbf{W}$ and $\mathbf{U}$, which are describe in (13) and (10)
   * Learn $\mathbf{B}$ bit by bit, with the help of (19)

## 5. Experiments and results

This section provides details on the experiments performed using various CNN architectures discussed in previous sections.

### 5.1. Database and evaluation protocol

In order to ensure reproducibility of experimental results, we utilized publicly available two session database [5]. This database of 6264 images has been acquired from 156 different subjects and includes finger-vein images from two fingers for each subject. However, the second session images are only from 105 different subjects. In our experiments we only used first session images to train different network architectures discussed in previous section, and excluded 51 subjects without second session. The experimental results are presented using second session test data. Therefore each of the receiver operating characteristics uses 1260 ($210 \times 6$) genuine scores and 263,340 ($210 \times 209 \times 6$) impostor scores.

We experimented on ROI images, enhanced ROI images and even Gabor filtered images separately. ROI images have $256 \times 513$ pixels, enhanced ROI images have $218 \times 488$ pixels and even Gabor filtered images have $218 \times 488$ pixels. The experimental results using ROC and CMC from the respective CNN architecture are presented in the following.

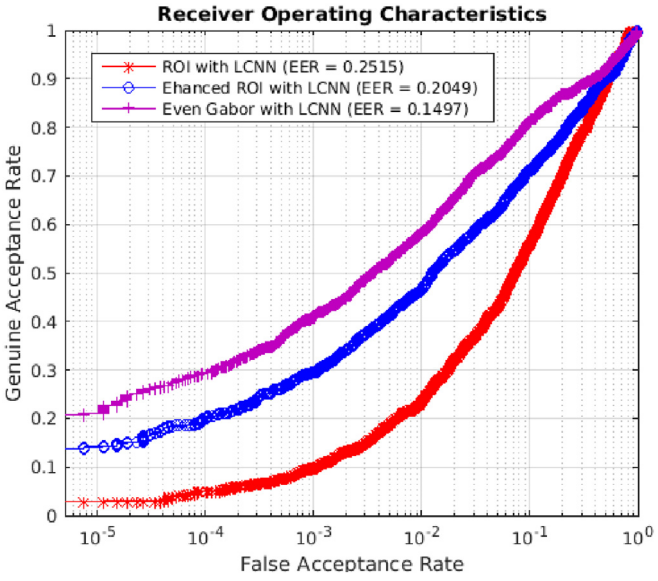


**Fig. 7.** Comparative ROC performance using LCNN in Fig. 4.

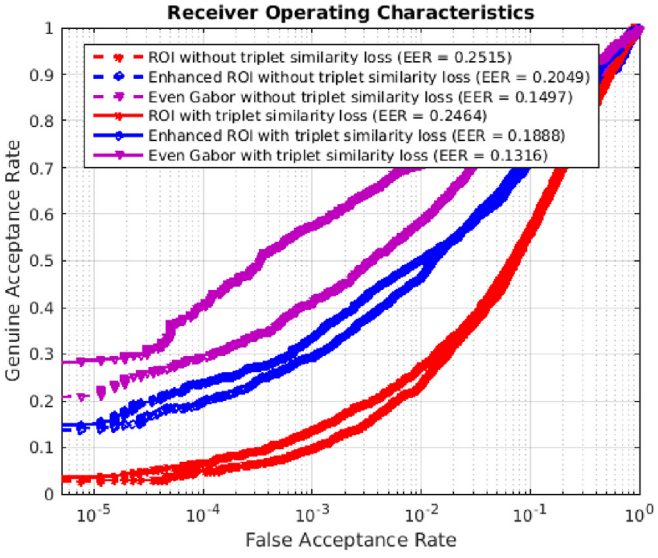


**Fig. 8.** Comparative ROC performance using triplet similarity loss based LCNN in Fig. 5.

### 5.2. Results using LCNN

We firstly performed the experiments for the CNN trained using the ROI images, enhanced ROI images and the enhanced images using Gabor filters (Fig. 1 and 2). The experimental results using respective second session dataset are shown in Fig. 7. The respective ROC in this figure illustrates that *enhanced* images can achieve superior matching performance than those from using ROI images. The enhancement of ROI images using Gabor filters significantly helps to suppress the noisy pixels and accentuate the vascular regions and is the plausible reason for superior accuracy.

### 5.3. Results using LCNN with triplet similarity loss function

The experimental results using LCNN trained with Siamese triplet similarity loss function are presented in Fig. 8. These results consistently illustrate superior performance using this architecture than the LCNN. The performance from the ROC of enhanced ROI with Gabor filters is superior and this observation is in line with

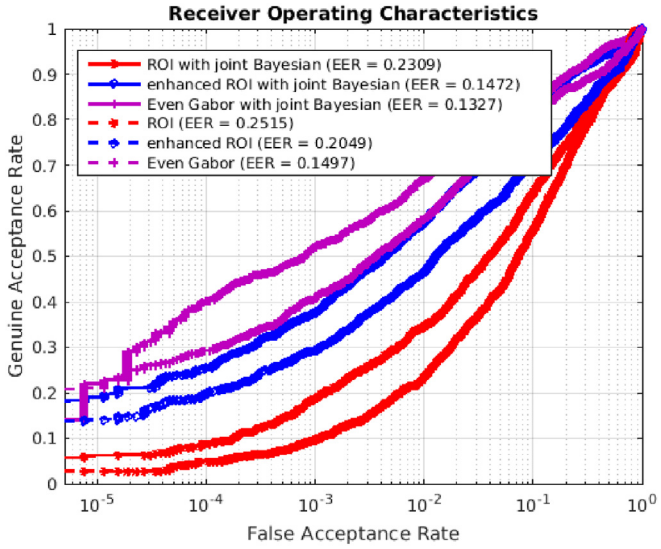


**Fig. 9.** Comparative ROC using LCNN with Bayesian approach.

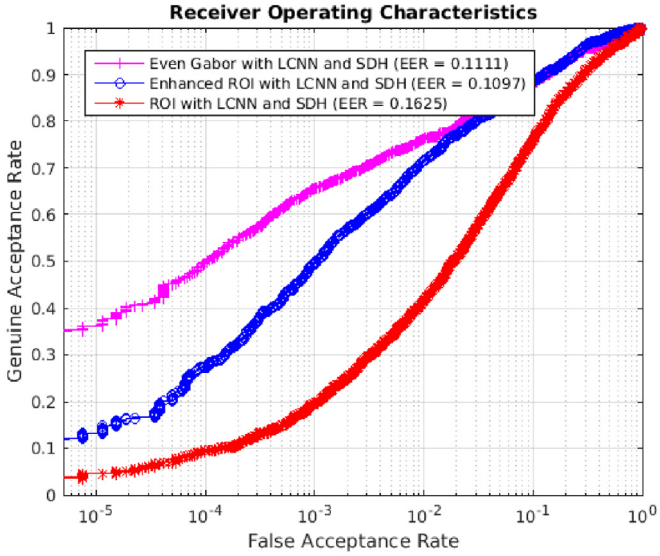


**Fig. 10.** Comparative ROC using LCNN and SDH.

the trends observed from results using LCNN in the Fig. 7. LCNN without triplet similarity loss tries to match a sample with its label, while LCNN with the similarity loss focuses on similarities between the images, this is the likely reason for the superior ROC performance.

### 5.4. Results using CNN and joint Bayesian formulation

Another scheme that has recently shown superior performance for the ocular identification in [10] uses joint Bayesian [8], instead of L2 norm, as the metrics for the similarity. The LCNN with the joint Bayesian classification scheme was also attempted to ascertain the performance. The ROC using this approach is illustrated in Fig. 9 and indicates notable performance improvement over LCNN.

### 5.5. Comparisons and results using supervised discrete hashing

The supervised discrete hashing (SDH) scheme detailed in Section 4 was also investigated for the performance improvement. Only *first* session data was employed for the training part and employed for generating the binarized bits that were used for computing match score using the Hamming distance. The resulting ROC in Fig. 10 illustrates consistent performance improvement with the usage of SDH and the trends in the usage of enhanced ROI images are also consistent with our earlier observations.

The LCNN trained with triplet similarity loss function was also employed used with the SDH to evaluate the performance. We attempted to ascertain the performance with different number of bits. Higher number of bits for SDH can be generally expected to offer superior results but requires more training time. It should be noted that this method is actually a second-step training, and tries to map features from the Euclidian space to the binary space. The training phase and hamming distance metrics can contribute to its superior performance. The combination of CNN with the hashing to reduce for the faster and real-time identification has also been attempted earlier [16] but for the face recognition problem. Authors in [16] incorporated Boosted Hashing Forest (BHF) for the hashing and therefore we also attempted to incorporate such BHF scheme to comparatively evaluate the performance. However, our results illustrated superiority of SDH over BHF and the template size using BHF was also fixed to 2000 bits. Although our results did not achieve significant improvement in the performance using BHF, its usage can help in remarkably reducing the template size.

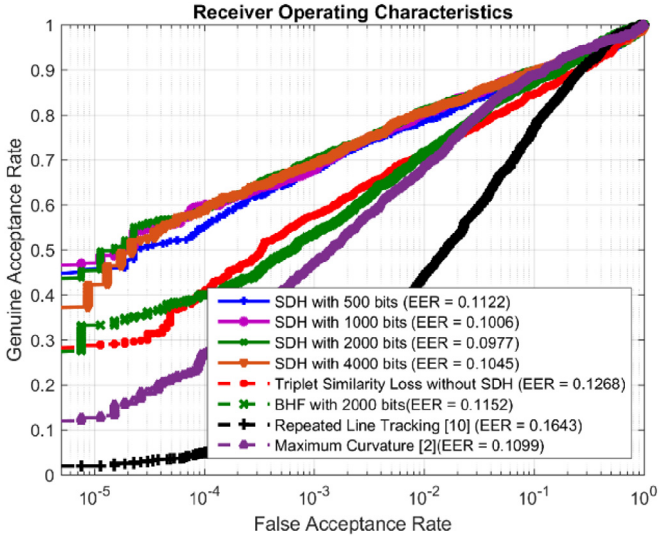


**Fig. 11.** Comparative ROC using triplet loss based LCNN with SDH and previous work.

In order to as-certain comparative performance for matching finger vein images using the hand-crafted features, we also performed additional experiments. The ROC from the same test images and matching protocols but using Repeated line tracking [8] (also used as baseline method in [15]) and maximum curvatures [2] method is illustrated in Fig. 11. We can observe from these experimental results that using SDH and LCNN can offer superior performance and significantly reduce the template size. Our results over the method using [4] are can be considered as competing and not yet superior but offers significantly reduced template size (∼26 times smaller) over the best of the methods in [4].

### 5.6. Results using modified VGG-16

The experimental results using modified VGG-16, as detailed in Section 3.3 are presented in Fig. 12. It should be noted that this CNN architecture generates single match score and therefore prohibits use from using the SDH scheme to the infer features. We can infer from the ROCs that the modified VGG-16 architecture generates superior performance for matching finger-vein images as com-

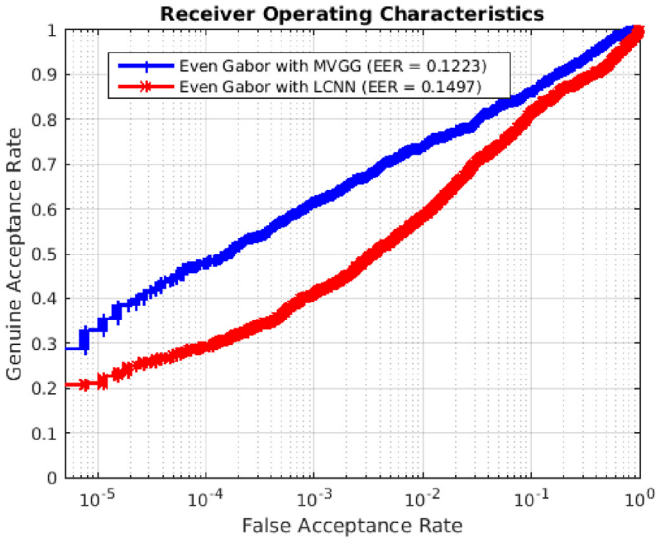


**Fig. 12.** Comparative ROC performance using modified VGG-16.

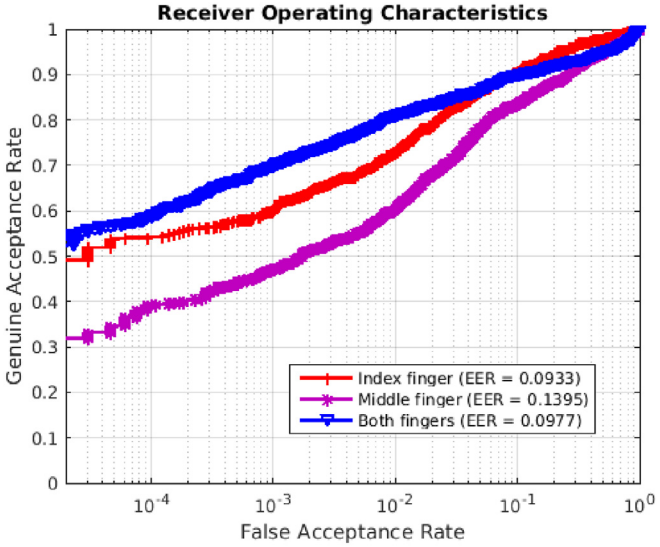


**Fig. 13.** Comparative ROC performance using triplet loss based LCNN and SDH using the finger vein images from a single finger.

pared with the network trained using LCNN. This structure generates matching scores directly, the problem of feature space is excluded, which may be a convincible reason for its superior results.

### *5.7. Results using single fingers and unregistered fingers*

Since we used both finger-vein images from two fingers to form larger dataset (as in [4]) for above experiments, it is judicious to ascertain the performance when only one, *i.e.*, index or middle, finger vein images are used in the training and test phase. The results using respective finger-vein images from 105 different subjects are comparatively shown in Fig. 13 using the ROC. The performance using the images from both fingers (using 210 class formed by combination of index and middle finger for 105 different subjects) is superior to single finger, and index finger shows better performance than middle finger. Similar trends are also observed while using hand crafted features in [4] and can be attributed to the nature of imaging or the dataset.

In earlier experiments, the first session data had images acquired from the same subjects who were providing their images

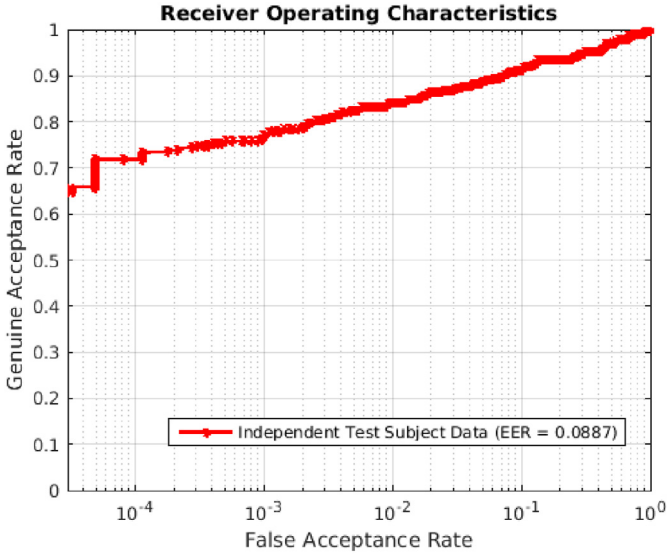


**Fig. 14.** The performance using independent test subjects in [5] for matching finger-vein images.

during the second session and were used as test set for the performance. In order to ascertain robustness of self-learned features using the best scheme so far, we also evaluated the performance from the independent 51 subjects in this dataset which did not have any two-session finger-vein images. Therefore images from none of these subjects images were employed during the training for CNN in any of the earlier experiments. The six images from these 51 subjects were used to ascertain performance using challenging protocol, *i.e.*, all-to-all, so that we generated a total of 1530 genuine scores and 185,436 impostor scores to ascertain such performance. The ROC corresponding to this independent test subjects finger vein data is shown in Fig. 14 and the results indicate promising performance from the self-learned features using a model trained (in Section 3.2 and SDH) for matching finger-vein images from unknown subjects.

## 6. Discussion

This paper has investigated finger vein matching performance using Convolutional Neural Network architectures. Unlike earlier work on finger vein image matching which largely employed hand crafted features, our emphasis has been to investigate automatically learned features using the capabilities of deep learning. We systematically investigated the performance improvement using just the ROI images and the systematically enhanced images that mainly emphasizes on subsurface vascular network. Our results consistently indicate superior performance from the CNN that are trained with images which have such enhanced vascular features.

According to our experimental results in Section 5.6, modified VGG-16 (MVGG) achieves superior performance than LCNN. However MVGG requires significantly higher time for the training (also for the test phase). This can be largely attributed to the fact that it *directly* generates the match scores and therefore the loss function outputs propagate iteratively through the whole network to ascertain the similarity between a pair of finger vein images. At the same time, we cannot incorporate SDH (hashing scheme) with the MVGG, due to non-availability of intermediate features, while the usage of SDH has shown to offer remarkable improvement in the performance.

It should be noted that the triplet similarity loss function helps to significantly improve the experimental performance using the LCNN. However, this approach cannot adequately make use of the

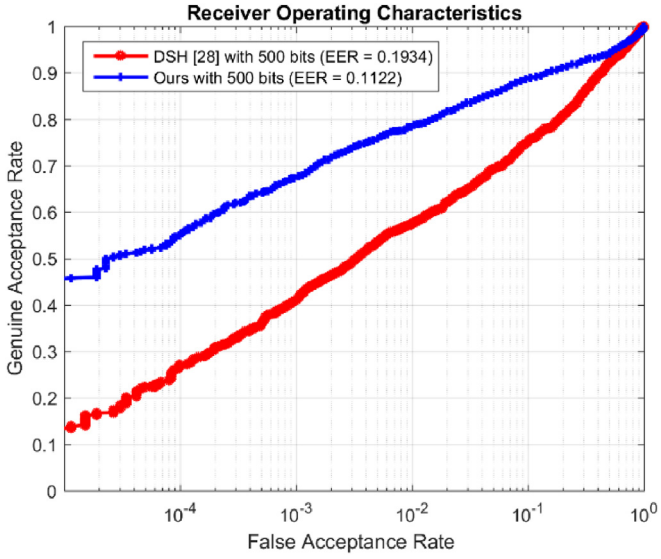


**Fig. 15.** Comparative performance using deep supervised hashing scheme.

label information, because it attempts to decrease the feature similarity between the pairwise images from the same subject, but cannot accurately locate the labels, *i.e.*, identity of the subjects they are associated with. Supervised discrete hashing approach further improves the performance and retrieval speed, and decrease the storage which requires only 250 bytes (2000 bits) for the one template (feature vector). However, it should also be noted that this method needs a separate training phase and training time rapidly increases when the bit length or number of features are increased. In this context, it is possible to incorporate recently introduced end-to-end learning schemes that incorporate deep neural networks to hash the features for fast image retrieval or matching [24–26]. Therefore, we also attempted to ascertain the performance from such deep supervised hashing scheme detailed in [25]. In order to ensure fairness in the comparison with SDH or two-step training model, same training data was used for training and test phase. Fig. 15 illustrates comparative performance with the CNN based deep supervised hashing and our SDH scheme detailed in Section 4. These results indicate that two step training model using SDH can achieve superior performance for matching the finger vein images. The plausible reason for poor performance from deep supervised hashing scheme lies in the lack of sufficient training data for our problem as we are attempting to directly map 210 class data, with high intra-class variations, into 500 or more bits. However with our two-step approach, the first step from CNN helps to extract useful information and therefore the second step with SDH received better input to consolidate the output into smaller number of bits.

The work detailed in this paper also had several constraints and therefore should be considered only preliminary. The database employed, although one of the largest two session finger vein database available in public domain, is still of smaller size for the deep learning based algorithms. There are several references in the literature that have shown promising performance but yet to demonstrate superior matching performance over the method in [4] using fair comparison or the same matching protocols. Therefore, we are justified in using the performance from [4], for this publicly available dataset, as the reference.

## 7. Conclusions and further work

This paper has investigated finger vein matching performance using various Convolutional Neural Network architectures. Unlike earlier work on finger vein matching which employed hand crafted features, our emphasis has been to investigate performance from the automatically learned features using the capabilities of deep learning. We systematically investigated the performance improvement using the ROI finger vein images, and the enhances images and consistently observed that the usage of ROI images with enhanced vascular features and attenuation of background (noise) can significantly improve the performance. The factors that most influence the accuracy of matching finger vein images is the depth of the network, the pre-training and the data augmentation in terms of random crops and rotations.

The usage of supervised discrete hashing with a Siamese network trained using the triplet loss function achieves most accurate performance among the all architectures considered in this work. The usage of hashing also significantly reduces the template size, to 2000 bits, for every subject and is the key advantage of this approach over other methods for finger vein matching available in the literature. The matching accuracy using this scheme also achieves outperforming results as compared with baseline methods considered in this work.

The work detailed in this work also had several constraints and therefore should be considered only preliminary. The database employed, although the largest two session finger vein images database available in public domain, is still of smaller size for the deep learning algorithms. Larger size database is expected to result in better learning of the representative features. One possibility is to use synthesized finger vein images, such as those generated in [22], to enrich the learning phase and should be carefully considered in further work. Further work should also be directed to develop deep learning architectures for finger vein images that can directly recover binary codes, *e.g.* [23], and achieve efficient matching. More baseline methods, instead of just two in this work, should be selected for the comparison with the performance using the hand crafted methods and is part of our ongoing work in this area.

## References

[1] F. Shen, C. Shen, W. Liu, H.T. Shen, Supervised discrete hashing, in: Proc. CVPR, 2015, June, Boston, 2015, pp. 37–45.
[2] N. Miura, A. Nagasaka, T. Miyatake, Extraction of finger-vein patterns using maximum curvature points in image profiles, in: Proc. IAPR Conf. Machine Vis. & Appl., May, Tsukuba Science City, 2005, pp. 347–350.
[3] X. Wu, R. He, Z. Sun and T. Tan, A light CNN for deep face representation with noisy labels, arXiv:1151.02683v2, 2016.
[4] A. Kumar, Y. Zhou, Human identification using finger images, IEEE Trans. Image Process. 21 (April) (2012) 2228–2244.
[5] The Hong Kong Polytechnic University Finger Image Database (Version 1.0), http://www.comp.polyu.edu.hk/~csajaykr/fvdatavase.htm, 2012.
[6] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. arXiv:1503.03832, 2015.
[7] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv:1409.1556, 2014
[8] D. Chen, X. Cao, L. Wang, F. Wen, J. Sun, Bayesian face revisited: A joint formulation, in: Proc. ECCV 2012, 2012.
[9] N. Miura, A. Nagasaka, T. Miyatake, Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification, Mach. Vis. Appl. (July) (2004) 194–203.
[10] Z. Zhao, A. Kumar, Accurate periocular recognition under less constrained environment using semantics-assisted convolutional neural network, IEEE Trans. Info. Forensics Secur. (May) (2017).
[11] L. Dong, G. Yang, Y. Yin, F. Liu, X. Xi, Finger vein verification based on a personalized best patches map, in: Proc. 2nd IJCB 2014, Sep.-Oct., Tampa, 2014.
[12] L. Chen, J. Wang, S. Yang, H. He, A finger vein image-based personal identification system with self-adaptive illuminance control, IEEE Trans. Instrum. Meas. (2017).
[13] Y. Lu, S. Yoon, D.S. Park, Finger vein identification system using two cameras, Electron. Lett. (2014).
[14] C. Liu, Y.H. Kim, An efficient finger-vein extraction algorithm based on random forest regression with efficient local binary patterns, in: Proc. ICIP 2016, 2016.
[15] Y. Ye, L. Ni, H. Zheng, S. Liu, Y. Zhu, D. Zhang, W. Xiang, FVRC2016: The 2nd finger vein recognition competition, in: Proc. ICB 2016, June, Halmstad, Sweden, 2016.

[16] Y. Vizilter, V. Gorbatsevich, A. Vorotnikov, N. Kostromov, Real-time face identification via CNN and boosted hashing forest, in: Proc. CVPR 2016 Biometrics Workshop, CVPR'W 2016, June, Las Vegas, 2016.
[17] B. Bhanu, A. Kumar, Deep Learning for Biometrics, Springer, 2017.
[18] M. Kono, H. Ueki, S. Umemura, A new method for the identification of individuals by using of vein pattern matching of a finger, in: Proc. 5th Symp. Pattern Measurement, Yamaguchi, Japan, 2000, pp. 9–12. (in Japanese).
[19] M. Kono, H. Ueki, S. Umemura, Near-infrared finger vein patterns for personal identification, Appl. Opt. 41 (35(December)) (2002) 7429–7436.
[20] Y. Lu, S. Yoon, S.J. Xie, J. Yang, Z. Wang, D.S. Park, Efficient descriptor of histogram of salient edge orientation map for finger vein recognition, Opt. Soc. Am. 53 (20(July)) (2014) 4585–4593.
[21] Weblink to download codes for the finger vein matching schemes detailed in this paper, 2017, https://polyuit-my.sharepoint.com/personal/csajaykr_polyu_edu_hk/_layouts/15/guestaccess.aspx?docid=11d706337994749759a2c2c64cb70b604a&authkey=AcDq15geZ7942-7owNQfmYQ
[22] F. Hillerstorm, A. Kumar, R. Veldhuis, Generating and analyzing synthetic finger-vein images, in: Proc. BIOSIG 2014, Sep., Darmstadt, Germany, 2014.
[23] M. Zhang, R. He, D. Cao, Z. Sun, T. Tan, Simultaneous feature and sample reduction for image-set classification, in: Proc. AAAI 2016, 2016, pp. 1401–1407.
[24] W.-J. Li, S. Wang, W.-C. Kang, Feature learning based deep supervised hashing with pairwise labels, arXiv:1511.03855, 12 Nov, 2015,
[25] H Liu, R Wang, S Shan, X Chen, Deep supervised hashing for fast image retrieval, in: Proc. CVPR 2016, 2016, pp. 2064–2072.
[26] H. Zhu, M. Long, J. Wang, Y. Cao, Deep hashing network for efficient similarity retrieval, in: Proc. AAAI 2016, 2016, pp. 2415–2421.