Deep learning-based Region of Interest Extraction for Finger Vein Images

Kenan Yang, Peiyu Fang, Jin Wu

College of Computer Science and Technology, Beijing University of Posts and Telecommunications, China

*Corresponding author e-mail: conlan7@163.com, #fangpeiyu@bupt.edu.cn

Abstract. Biometric technology has played an important role in our lives. Finger vein recognition has many advantages as a new biometric technology. Different from iris, fingerprint or face, finger vein recognition is cheap, Non-contact, living. However, finger vein recognition is susceptible to many factors such as light environment, finger moving, etc. In order to solve these problems, in this paper we first propose a method for extracting region of interest for finger vein images based on a convolutional neural network. Secondly, we have integrated several common finger vein databases and manually labeled the region of interest. We also extended the dataset using data augmentation methods. Finally, for the transformed images, we calculate the IOU of the prediction box and the theoretical box to measure the robustness of the method and Experiments show that our method improves the performance of the recognition system.

1. Introduction

Biometric technology has made significant progress in both technical and application levels in recent years. It is widely used in many fields such as security, e-commerce, and electronic equipment. [1] Therefore, finger vein recognition has gained more and more attention as an emerging biometric technology. Compared with traditional biometrics such as fingerprint recognition, face recognition, and iris recognition, finger vein recognition has many advantages such as living detection, non-contact, difficult to forge, and low price, which makes it have good application prospects in various fields. [1, 2]

Although finger vein recognition has made good progress in practice, there are still many problems waiting to be overcome. Finger vein recognition system generally has four steps: image acquisition, image preprocessing, feature extraction and matching. [2, 3] To improve the recognition accuracy, an effective and stable image preprocessing result is extremely important. Typical image preprocessing steps include region of interest (ROI) localization, vein enhancement, and vein thinning. ROI extraction is the most important step. It has two main aims: the first is to determine which part of the image is the finger, and the other is to extract the informative part from the image [3]. Traditional ROI extraction methods generally consist of edge extraction, angle correction, joint detection, and region extraction. [4] In the past decade, there have been many traditional finger vein ROI extraction methods, such as Bakhtiar et al. [5], Yang et al. [6], Yang et al [7], and Lu et al [8]. Figure 1 shows the pipeline of the traditional method. However, these steps are easily affected by finger translation, lighting conditions, rotation, and acquisition equipment, making it difficult to obtain high-quality ROI [3].
Another significant problem is that finger vein recognition is an emerging biometric technology with only a few small-scale public databases. Moreover, the equipment and methods used to obtain data vary from database to database. Well-known public finger vein databases so far include Shandong University’s dataset, Tsinghua University’s dataset, Twente University’s dataset, Chonbuk National University’s dataset, etc. Considering the above difficulties and the importance of ROI extraction in the finger vein recognition system, we need a new and more robust ROI extraction method. Since the convolutional neural network has proved its effectiveness and generalization in many fields, it has entered our vision.

Figure 1. Pipeline of traditional method

Convolutional neural networks first showed excellent performance in image classification [9, 10]. Inspired by this, many CNN-based methods have emerged to solve various tasks in the field of computer vision, such as object detection [11, 12], image segmentation, [13] object tracking [14] and so on. Finger vein ROI extraction is an object localization task, which can be regarded as a sub task of object detection. Using CNN is a natural idea. In general, the outstanding performance of CNN comes from the large amount of well-labeled data and its strong learning ability. In order to obtain a large amount of well-labeled data, we manually labeled images in multiple databases. Further, we have made data augmentation in these databases and get enough images to be learned.

The main contributions of this paper are as follows:

- We integrate several public datasets and carried out manual label and data augmentation.
- This paper presents a method for extracting finger vein ROI using VGG16, [19] and compares the effects of using different convolutional layer outputs.
- We compared the stability and effectiveness of the ROI extracted by our method and traditional methods.

The remaining chapters of this paper are as follows: Section 2 details the proposed finger vein ROI extraction method. Section 3 describes our experiments in detail, and discusses the experimental results. Finally, Section 4 concludes this paper.

2. Our Method

For a finger vein image, there will be a lot of background areas and noise on the image. Therefore, ROI extraction is an important and effective step in finger vein recognition systems or encryption systems. Figure 2 shows the pipeline of our method. The method mainly includes (1) Transform an input image and calculate the transformed ground truth. (2) Input the transformed image into the VGG16 network and extract the feature map of a convolutional layer. (3) Feed the output into MLP and calculate the MSE of coordinates. Figure 3 is the architecture of the VGG16 network.
2.1. Data Augmentation

Finger vein public databases are relatively rare and small, and in order to train an effective network, data augmentation is essential. Performing various transformations on images can not only increase the number of samples in the training set for better training, but also improve the generalization ability of the network and make it pay more attention to the effective area of the finger vein. The strong generalization ability of the network can counteract the effects of finger image offset, tilt, occlusion, etc.

- When taking pictures of finger veins, it is common for fingers to pan, tilt, and change in size. Different devices will also take pictures of different size proportions, and even take different parts of the finger. Although the convolutional neural network itself has some shifting invariance. But it is not adaptive to the movement of large objects or rotation. Therefore, affine transformation of images is a common data augmentation method. For an image \( I \in \mathbb{R}^{m \times n} \) i.e. the size is \( m \times n \), its affine transformation can be expressed by the following formula:

\[
\begin{bmatrix}
    x' \\
    y'
\end{bmatrix} = \begin{bmatrix}
    a_1 & a_2 & t_x \\
    a_3 & a_4 & t_y \\
    0 & 0 & 1
\end{bmatrix} \begin{bmatrix}
    x \\
    y \\
    1
\end{bmatrix}
\]

(1)

Where \((t_x, t_y)\) represents the amount of translation, \(a_i\) represents the image rotation and scaling. Through these parameters, we can get the affine transformation of the images.

- When finger veins are extracted at different times and scenes, the average brightness of the image may change due to the surrounding lighting environment or camera exposure. Also, noise often appears on the image. Therefore, we also use the method of brightness transformation and adding Gaussian noise to expand the data. The probability density function of Gaussian noise follows the following distribution:
\[ p(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-u)^2}{2\sigma^2}} \]  

- For possible occlusion or stain factors, we use cutout to simulate. For an image \( I \in \mathbb{R}^{m \times n} \), we create a mask \( m \in \mathbb{R}^{m \times n} \) which has a small square area \( A \) with 1. It can be described by the following formula:

\[
m_{i,j} = \begin{cases} 
  1, & \text{where } m_{i,j} \in A \\
  0, & \text{else}
\end{cases}
\]  

Multiply the original image with the mask to get the blocked image \( B \).

\[ B = I \ast m \]  

Each image has a half probability to remain as it is, and a half probability to do a transformation such as rotation in affine transformation. Figure 4 is some examples of affine transformation. Figure 5 is some examples of noise, brightness changes and cutout.

**Figure 4.** Examples of affine transformation. (a) Origin image; (b) Scale image; (c) Flip image; (d) Rotate90° image; (e) Rotate image; (f) Shift image

**Figure 5.** Examples of noise and light change. (a) Origin image; (b) Cutout image; (c) Light change image; (d) Noise image
2.2. Enrollment
The task is to create a feature vector from input image. Normally, enrollment includes normalization and feature extraction from the image. Before Enrollment, we normalized the input image to size normalization and grayscale normalization. In particular, we resize the image to the size of 224 * 224. We use VGG16 network as our backbone and extract feature maps from different convolutional layers. We show details of these layers in Table 1. We feed normalized image to the network and get the feature map $F \in \mathbb{R}^{C \times H \times W}$ and reshape $F$ to $C \times H \times W$ dimensions vector. After two fully connected layers, we can get the prediction label. The last fully connected layer uses sigmoid as the activation function to ensure that the result is between $(0, 1)$

| Ind | Type      | Filter Size | Num  | Output |
|-----|-----------|-------------|------|--------|
| 1   | Conv      | 3           | 64   | -      |
| 2   | Conv      | 3           | 64   | -      |
| 3   | Max Pool  | 2           | -    | -      |
| 4   | Conv      | 3           | 128  | -      |
| 5   | Conv      | 3           | 128  | Conv2  |
| 6   | Max Pool  | 2           | -    | -      |
| 7   | Conv      | 3           | 256  | -      |
| 8   | Conv      | 3           | 256  | -      |
| 9   | Conv      | 3           | 256  | Conv3  |
| 10  | Max Pool  | 2           | -    | -      |
| 11  | Conv      | 3           | 256  | -      |
| 12  | Conv      | 3           | 256  | -      |
| 13  | Conv      | 3           | 256  | Conv4  |

2.3. Learning
We train the network in a strongly supervised manner. We minimize the mean squared error of bounding box’s coordinate over a training set of images using stochastic gradient descent (SGD) with a batch size of 64. MSE can be described as formula following

$$MSE = \frac{1}{M} \sum_{i=1}^{m} (y_i^2 - \hat{y}_i^2)$$

(5)

Where $y_i$ represents the labeled bounding box of the i-th image. And $\hat{y}_i$ represents the predicted bounding box of the i-th image. SGD with initial learning rate 0.001, momentum 0.9 is used to train the network. We use a learning rate schedule to speed up training. Weights are initialized randomly. We use Nvidia GTX 1080TI in Pytorch and convolution routines are taken from Nvidia cuDNN library.

3. Experiments

3.1. Datasets
For training and validation, we mix Tsinghua University database, Twente University database and Shandong University database and split them into train set, validation set and test set. In addition, we use the database of Chonbuk National University as the full test set to make the experiments fair and reliable. Train set includes more than 2,000 fingers and 30,000 images. Validation set includes 300 fingers and 2,000 images. Test set includes the whole Chonbuk National University database and 500 fingers, over 3,000 images from the other three databases.
3.2. Experiment 1
In Experiment 1, we compared the effects of different convolutional layer outputs on coordinate prediction. In Section 2.2 we say output of different convolution layers will affect the results. In this Section we compared the training speed and accuracy of the output of conv2, conv3, and conv4. Table 2 lists their speeds, IOUs for prediction bounding boxes and manually labeled bounding boxes, and network loss. We calculate the average IOU and average loss for the entire test set.

| Layer  | Speed     | IOU       | Loss     |
|--------|-----------|-----------|----------|
| Conv2  | 55.31 s/epoch | 91.15%    | 9.06 * 10^-5 |
| Conv3  | 70.19 s/epoch | 94.83%    | 8.12 * 10^-5 |
| Conv4  | 92.65 s/epoch | 93.64%    | 8.43 * 10^-5 |

The result of Experiment 1 suggests that the feature map extracted by Conv3 has faster convergence and the best performance. This may be due to the fact that the finger vein is an approximately line feature and is suitable for shallow features.

3.3. Experiment 2
In Experiment 2, we compared our method with two traditional methods. We randomly select a thousand pictures and randomly perform three transformations on each picture as a test set. We define the stability of the ROI extraction method as the IOU between the theory bounding box and the prediction box. Table 3 shows the average IOU of the three methods on the same test set, and Figure 6 shows the boxes generated by the three methods.

![Figure 6. ROC curves on database by different ROI localization methods.](image)

Experimental results show that our method outperforms traditional methods in a variety of transformation situations.
Table 3. IoU from three method

| Method         | IOU    |
|----------------|--------|
| Our method     | 94.95% |
| Method in 7    | 90.45% |
| Method in 8    | 90.17% |

3.4. Experiment 3
First, we will randomly transform the test images to build a test set. Then, we use our method and the method in [7, 8] to extract ROI of these images, and resize the ROI images to 96 * 64. We extract the LBP features of the ROI image and calculate the statistical histogram. We normalize the statistical histogram to get the feature vector. Calculate the Euclidean distance for every two pictures, and calculate the ROC curve based on whether they come from the same finger. It can be ascertained from Table 4 and Figure 7 that the proposed method achieves the best performance among all the approaches considered in this work. We use the LBP feature because of its poor robustness, which can make the results directly affected by the ROI extraction method. As we see, all three methods perform poorly due to weak LBP capabilities, but our method has clear advantages.

Figure 7. Boxex from different method. (blue) Standard box; (red) Our method; (green) Method in 7; (yellow) Method in 8

Table 4. Eer on of three methods

| Method         | EER    |
|----------------|--------|
| Our method     | 0.1987 |
| Method in 7    | 0.2132 |
| Method in 8    | 0.2814 |

4. Conclusion
In this paper, we present a robust ROI extraction method based on convolutional neural network. We first labeled several public databases to get ground truth. When we train, we feed the randomly transformed image into the network and calculate the meaning squared error. When predicting, we input the image into the network and obtain the 4-dimensional vector from the last layer, which is the bounding box coordinates. This method is more robust and accurate than traditional methods. And leads to higher accurate performance, as shown from the experimental results. Future work includes finding better network structures such as FPN, using key point detection instead of coordinate regression, etc.
Acknowledgments
This paper was supported by Fang Peiyu Lab of Beijing University of Posts and Telecommunications. Thanks to Mr. Fang Peiyu for his guidance and my lab classmates for their help. The authors would particularly like to thank the anonymous reviewers for their helpful suggestions.

References
[1] Wu J D, Ye S H. Driver identification using finger-vein patterns with Radon transform and neural network [J]. Expert Systems with Applications, 2009, 36 (3-part-P2): 5793 - 5799.
[2] Liu, Z.; Yin, Y.L.; Wang, H. J.; Song, S. L.; Li, Q.L. Finger vein recognition with manifold learning. J. Netw. Comput. Appl. 2010, 33, 275 – 282.
[3] Pi W.; Shin J.; Park, D. An Effective Quality Improvement Approach for Low Quality Finger Vein Image. In Proceedings of International Conference on Electronics and Information Engineering, Kyoto, Japan, 1–3 August 2010; pp. 424 – 427.
[4] Yang J , Shi Y . Finger–vein ROI localization and vein ridge enhancement[J]. Pattern Recognition Letters, 2012, 33 (12).
[5] Rosdi, B. A.; Shing, C.W.; Suandi, S.A. Finger vein recognition using local line binary pattern. Sensors 2011, 11, 11357 – 11371.
[6] Yang, J.F.; Li, X. Efficient Finger Vein Localization and Recognition. In Proceedings of 2010 International Conference on Pattern Recognition, Tianjin, China, 23–26 August 2010; pp. 1148 – 1151.
[7] Yang, Lu, Yang, et al. Sensors, Vol. 13, Pages 3799-3815: Sliding Window-Based Region of Interest Extraction for, Finger Vein Images [J]. 2013.
[8] Lu Y, Xie S, Yoon S, et al. Robust Finger Vein ROI Localization Based on Flexible Segmentation [J]. Sensors, 2013, 13 (11): 14339 - 14366.
[9] A. Krizhevsky, I. Sutskever, and G. E.Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," presented at the Advances in Neural Information Processing Systems, 2012.
[10] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 Million Image Database for Scene Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 40, no. 6, pp. 1452 - 1464, 2018.
[11] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," in 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 580 - 587.
[12] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137 - 1149, 2017.
[13] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual Tracking with Fully Convolutional Networks," in 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 3119 - 3127.
[14] H. Noh, S. Hong, and B. Han, "Learning Deconvolution Network for Semantic Segmentation," in 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1520 - 1528.