Finger Vein Recognition Algorithm Based on Lightweight Deep Convolutional Neural Network

Jiaquan Shen, Ningzhong Liu, Chenglu Xu, Han Sun, Yushun Xiao, Deguang Li, *Member, IEEE*, and Yongxin Zhang, *Member, IEEE*

***Abstract*— Even though the deep neural networks have strong feature representation capability and high recognition accuracy in finger vein recognition, the deep models are computationally intensive and poor in timeliness. To address these issues, this article proposes a lightweight algorithm for finger vein image recognition and matching. The proposed algorithm uses a lightweight convolutional model in the backbone network and employs a triplet loss function to train the model, which not only improves the matching accuracy, but also satisfies the real-time matching requirements. In addition, the Mini-region of interest (RoI) and finger vein pattern feature extraction also effectively solve the problems of large amounts of calculation and background noise. Moreover, the present model recognizes new categories based on the feature vector space constructed by the finger vein recognition system, so that new categories can be recognized without retraining the model. The results show that the finger vein recognition and matching algorithm proposed in this article achieves 99.3% and 99.6% in recognition accuracy and 14.2 and 16.5 ms in matching time for the dataset Shandong University Machine Learning and Applications Laboratory-Homologous Multimodal Biometric Traits (SDUMLA-HMT) and Peking University Finger Vein Dataset (PKU-FVD), respectively. These metrics show that our approach is time-saving and more effective than previous algorithms. Compared with the state-of-the-art finger vein recognition algorithm, the proposed algorithm improves 1.45% in recognition accuracy while saving 45.7% in recognition time.**

***Index Terms*— Deep learning, finger vein recognition, lightweight convolution network, Mini-region of interest (RoI) extraction, triplet loss.**

Manuscript received November 9, 2021; accepted November 18, 2021. Date of publication December 3, 2021; date of current version February 21, 2022. This work was supported in part by the Science and Technology Innovation Team of Henan University under Grant 22IRTSTHN016; in part by the Key Scientific Research Project of Higher Education of Henan Province under Grant 22A520006, Grant 22A110014, and Grant 22A120009; in part by the National Natural Science Foundation of China under Grant 61375021 and Grant 61802162; in part by the Fundamental Research Funds for the Central Universities under Grant NS2016091; in part by the Science and Technology Key Project of Henan province under Grant 202102210370; in part by the Collaborative Innovation Center of Novel Software Technology and Industrialization; and in part by two finger vein databases, Peking University Finger Vein Dataset (PKU-FVD) [47] and Shandong University Machine Learning and Applications Laboratory-Homologous Multimodal Biometric Traits (SDUMLA-HMT) [48], respectively, provided by the Artificial Intelligence Laboratory of Peking University and Shandong University. The Associate Editor coordinating the review process was Dr. Wenqiang Liu. *(Corresponding author: Ningzhong Liu.)*

Jiaquan Shen, Deguang Li, and Yongxin Zhang are with the School of Information Science, Luoyang Normal University, Luoyang 471022, China.

Ningzhong Liu, Chenglu Xu, Han Sun, and Yushun Xiao are with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China (e-mail: lnz_nuaa@163.com).

Digital Object Identifier 10.1109/TIM.2021.3132332

I. Introduction

With the higher requirements for authentication, the traditional authentication technology can no longer meet the needs of the current stage. Relative to traditional authentication methods (e.g., keys, cards, and passwords), biometric technology can more effectively protect personal identity information, and its security and effectiveness are significantly better than traditional authentication methods [1]–[5]. Recently, finger vein biometric technology has attracted the attention of many research groups and has shown great market potential [6], [7]. Finger vein patterns are inside the body and have biological activity, which makes them virtually impossible to replicate [8]–[10]. Compared with other biometric technologies, a finger vein technique has its feature: 1) finger vein belongs to human bodies' internal characteristics and does not change with time, which has higher uniqueness and stability; 2) finger vein images must be captured in the living body, and they are difficult to forge, steal, or be affected by external environments such as skin, pollution, and temperature; and 3) the finger vein collection device is smaller and non-contact, which is more acceptable and user-friendly.

Traditional finger vein recognition algorithms use a hand-designed approach for feature extraction [11]–[13], which are usually divided into local binary-based methods and vein-pattern-based methods [14], [15]. The traditional finger vein recognition algorithm suffers from shortcomings that may severely limit their further applications. These algorithms extract finger vein features by filters and morphological methods, but these methods tend to produce irregular shadows and loss of vein feature image information, which makes the matching accuracy low. The traditional methods use complex steps to learn a feature transformation matrix, and the feature transformation matrix needs to be retrained when a new user joins, which limits the recognition ability of the model. In addition, some traditional finger vein recognition algorithms need to be designed with specific sensors in the process of finger vein image acquisition, which makes the algorithm less capable of generalization.

The deep convolutional neural networks use convolution kernels of different sizes to extract the features of the finger vein image, which can completely obtain the detailed features in the vein image. Moreover, the use of a lightweight convolutional neural network structure makes the vein recognition algorithm faster and ensures matching accuracy [16], [17]. Some finger vein recognition algorithms with a

neural network show that their works perform much better than traditional finger vein recognition algorithms in recognition accuracy [18]. In addition, deep learning methods can replace manual extraction features with unsupervised or semi-supervised feature learning [19]. Similar to face recognition, finger veins are recognized by first selecting those regions of interest with distinctive vein features and then performing feature extraction and matching [20], [21]. Massive public face databases have been widely used in researches of face recognition. However, the available public dataset of finger veins is insufficient for effective training on deep models. In addition, some deep networks such as Google-Net [22] and ResNet [23] cannot effectively match and recognize finger veins in real-time due to a large number of parameters.

In [24], the authors propose an end-to-end model to extract vein patterns through integrating the fully convolutional neural network with the conditional random field, and this method achieves accurate and effective segmentation in vein patterns. Zeng *et al.* [25] propose a lightweight real-time segmentation method for finger veins based on the embedded terminal, and this algorithm has a significant improvement in matching efficiency, but its detection accuracy is not high. In [26], the authors use a multi-receptive field bilinear convolutional neural network to identify finger veins, and the experimental results show that this method not only improves the accuracy of finger vein recognition, but also reduces the training time and model parameters.

Aiming at the above problems, we propose a lightweight convolution neural network with triplet loss for finger-vein recognition, and the feature extraction network includes a Stem Block and a Stage Block. The Stem Block uses a multi-scale approach for initial feature extraction, and the Stage Block uses a channel stacking approach to gradually acquire the detailed features of the finger vein images, which makes the model improve the matching accuracy while reducing the computational effort. In addition, in our method, new categories can be identified by calculating the spatial distance of vectors, which can avoid repetitive training. The main contributions of this article are as follows.

1) We propose a fast and accurate finger vein feature extraction and recognition algorithm based on a lightweight deep convolutional neural network, which is better than the existing depth model algorithms in recognition accuracy and matching efficiency.
2) In the process of deep model training, we use a triplet loss function for training. The loss function can effectively extract and recognize the features of fine-grained images, and it can construct the 3-D vector space of finger veins to avoid multiple training of the model.
3) A maximum curvature finger vein pattern feature extraction and optimization algorithm based on Gaussian filtering optimization is proposed in this work, which can effectively extract the pattern features of finger veins and reduce the effect of noise on recognition accuracy.

The remainder of this article is organized as follows. Section II discusses the literature review. Section III explains the proposed method, and Section IV shows the experiment and results. Section V provides the conclusions of the article.

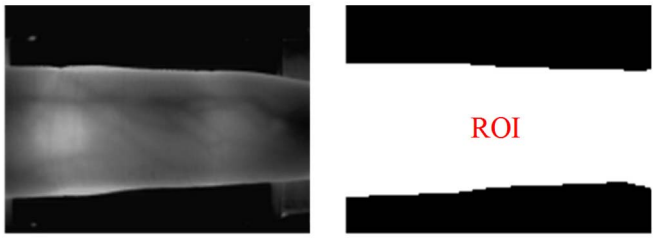


Fig. 1. Example of RoI.

II. Literature Review

A. Finger Vein Image Preprocessing Algorithms

The preprocessing of finger vein images mainly includes the steps of a region of interest (RoI) extraction, finger region correction, and image enhancement. The extraction of RoI in the finger vein image retains the main information in the vein features and removes the irrelevant background noise (as shown in Fig. 1). However, most of the current studies focus on finger vein feature extraction, finger vein image enhancement, and finger vein feature characterization methods, and there are fewer studies on optimized extraction algorithms for finger vein RoI regions. In [27], the authors propose a method for finger vein region-of-interest localization and vein crest enhancement, which combines finger structure and optical knowledge to localize the RoI regions of finger vein features. In [28], the authors use a flexible segmentation method for localizing the region of interest of finger veins, but this algorithm cannot effectively deal with the rotation problem arising from finger veins during sample collection. Kumar and Zhou [29] extract the region of interest by adding an edge detector. However, the method ignores the effect of angle and orientation on feature extraction during image acquisition, which makes the method invalid for some finger vein images. In addition, there is still more irrelevant background noise in the region of interest extracted by this method.

The finger vein images need to be image-enhanced before the feature extraction because the images produce problems such as inconspicuous contrast or noise during the acquisition process. In recent years, researches related to finger vein image enhancement have also been developing [30], [31]. In [32], the authors use histogram equalization combined with high-frequency filtering to enhance the contrast of finger vein images. Yang and Shi [33] use different directional filtering strategies to enhance the contrast of vein images and highlight the pattern features of finger veins, but the adaptability of this method can be limited when faced with poor-quality images. The above histogram equalization-related image enhancement algorithm is relatively simple to implement. However, this method may result in inexact edge detection.

B. Traditional Finger Vein Feature Extraction and Recognition Algorithm

Traditional finger vein feature extraction and recognition algorithms mainly include dimensionality reduction, local binary, and finger vein patterns recognition methods. The recognition method based on finger vein patterns has been widely used and achieved a good matching performance. The method based on dimensionality reduction is a typical

recognition method, which can transform the image into a low-dimensional space before classification. During the dimensional transformation, the information is continuously discriminated and de-noised [34], [35]. In [36], a finger vein recognition algorithm based on metric learning is proposed. The algorithm uses the two-dimensional principal component analysis (2DPCA) method to extract finger vein features and then uses the k-nearest neighbor (KNN) classifier to classify vein images, which improves the recognition effect. However, the dimensionality reduction-based method is time-consuming, it needs to be retrained and relearned each time when a new user joins the model, and it is sensitive to sample parameters.

The local binary pattern (LBP) algorithm is widely used in vein feature extraction and recognition. This algorithm obtains the pattern features of finger veins by comparing the gray value of the current pixel point with the surrounding point pixels and matches the finger veins based on the pattern features [37], [38]. In [39], the authors propose a hybrid parallel finger vein recognition matching method based on local linear binary patterns and Hamming distance, which reduces the computation time of finger veins in performing individual recognition. In [40], the authors propose a finger vein feature extraction and recognition algorithm based on the fusion of two-dimensional Gabor filter (2DGF) and LBP, which further improves the matching accuracy. These algorithms can effectively extract the local features of the vein pattern in binary format and show significant advantages in finger vein recognition. However, these algorithms cannot effectively express the correlation between multiple domains due to the dimension curse.

The recognition algorithm based on vein pattern features extracts the topology from the grayscale map in the feature extraction process and uses the obtained topology and geometry to perform the corresponding matching operation. Song *et al.* [41] invent a mean curvature-based vein pattern feature extraction method, which can extract the topology of finger vein patterns. The region growth method [42] extracts the vein pattern from a cross-sectional view of the finger vein image, the cross-sectional view of the finger vein striation looks like a valley, and the algorithm uses this property in the finger vein image for the extraction of the features of the vein pattern. The repetitive line tracking method [43] performs pixel-by-pixel tracking along with the movement of the feature lines to identify the local dark lines, which is much better than the traditional method based on matched filters. Miura *et al.* [44] invent a vein pattern feature extraction algorithm based on maximum curvature, which extracts the pattern features of finger veins by calculating the maximum curvature of the cross section of finger vein samples. Meanwhile, these feature extraction methods can continuously extract the centerline of the vein without the influence of pulsewidth and brightness fluctuation, and the feature pattern of the finger vein can be well preserved.

C. Finger Vein Recognition Algorithm Based on Deep Learning

In recent years, deep learning has shown superior performance in the field of image recognition, and some scholars have tried to apply deep learning neural networks to finger vein recognition, which has led to the further development of finger vein recognition algorithms. The finger vein recognition algorithm based on the depth model uses a convolutional neural network to extract features, which has better generalization and feature representation ability. Yang *et al.* [6] create a finger vein recognition system with strong privacy protection based on a deep neural network multilayer extreme learning machine, and this algorithm not only has high recognition accuracy, but also has strong security. In [7], the authors propose a lightweight finger vein recognition and anti-spoofing network (FVRAS-Net), which uses multi-task learning (MTL) approach to integrate recognition and anti-spoofing tasks into a unified convolutional neural network (CNN) model. Kuzu *et al.* [8] propose a real-time finger vein recognition algorithm based on recurrent convolutional neural networks, which is able to capture the structure of finger veins adaptively and obtain high recognition accuracy. In [9], the authors establish a 3-D vein feature and use the 3-D information to identify finger vein.

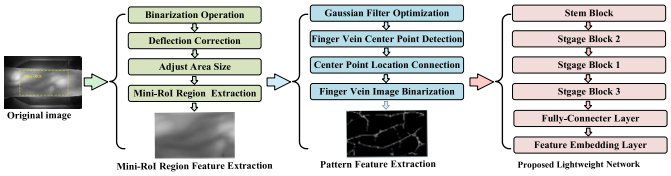


Fig. 2. Overall framework of the proposed finger vein recognition method.

In [19], the authors propose a new method for finger vein extraction and verification based on generative adversarial networks (GANs). In [45], the authors used bias field correction, spatial attention mechanism, and multistage migration learning strategy to build a deep convolutional neural network algorithm for finger vein recognition, which can remove the unbalanced bias field from the original image while enhancing the pattern information of finger veins. Hou and Yan [46] use the inverse cosine central loss function to improve the ability of a convolutional neural network for finger vein feature extraction and recognition. The experimental results confirm that the algorithm based on the deep model can effectively improve the accuracy of finger vein recognition. Although the finger vein recognition algorithm based on the deep model shows significant advantages over the traditional method, the algorithm is time-consuming and ineffective in feature extraction.

III. Proposed Method

In this article, we propose three algorithms according to the steps in the finger vein recognition process, which are: 1) finger vein Min-RoI region feature extraction algorithm; 2) finger vein pattern feature extraction algorithm based on Gaussian filter optimization; and 3) finger vein recognition and matching algorithm based on the lightweight depth model. Fig. 2 illustrates the overall framework of the proposed finger vein recognition method. In detail, the finger vein Min-RoI region feature extraction algorithm can effectively remove the interference information and background noise from the vein image. The vein pattern feature extraction algorithm based on Gaussian filtering can effectively extract the pattern feature. This algorithm is not affected by the fluctuation of image brightness and finger vein width, so the image after the vein feature extraction is beneficial to the training and recognition

of a deep neural network. The finger vein recognition and matching algorithm based on the lightweight depth model can effectively train and match finger vein pattern images, while introducing triplet loss in the depth model can directly make similarity judgments by predicting feature vectors when new categories are added without retraining the model, which performs better than many traditional and deep learning finger vein algorithms.

A. Finger Vein Mini-RoI Region Feature Extraction

The finger vein image in the process of acquisition not only collects the effective finger vein image information, but also contains background noise. In order to reduce the computational complexity of finger vein recognition algorithm, reduce the inter-class gap of the finger vein features, and improve the recognition accuracy of the algorithm, it is necessary to extract the region of interest in the finger vein image. In the process of finger vein RoI region feature extraction, the finger vein feature region of interest contains the feature information as large as possible, and the area of the core finger vein feature region is as small as possible. We call this algorithm as Mini-RoI region feature extraction algorithm. This algorithm maximizes the preservation of the feature areas of the finger veins while minimizing the complexity of subsequent calculations.

When finger vein images are captured, the captured fingers may inevitably show up-and-down offset and angle rotation, which affects the effect of the finger vein recognition algorithm. Therefore, before performing Mini-RoI feature region extraction, we first need to perform feature deflection correction on the finger vein images. In the vein image feature deflection correction, the binarization operation is first used to obtain the edge information of the finger vein image, and the middle point of the upper and lower edges is used to calculate the orientation angle of the image. The deflection degree of the finger is obtained by finding the tilt angle of the orientation angle.

The detailed implementation process is to first select the middle point y_i corresponding to the upper and lower edges of the finger vein binary image according to (1). In (1), up and low are the coordinate points of the upper and lower boundary areas of the finger vein, respectively. In addition, i represents the point where the edge region differs from left to right. We use (2) and (3) to fit the selected centroids and obtain the centerline of the finger vein image feature region. After obtaining the centerline of the finger vein feature region, we can find the corresponding deflection angle θ by calculating the slope k of this centerline, as shown in (4). Finally, the finger vein feature area is rotated horizontally according to the obtained deflection angle. Fig. 3 illustrates the process of finger vein image rotation offset correction

$$y_i = \frac{\text{up}_i + \text{low}_i}{2}, \quad i = 1, 2, 3, \ldots, n \tag{1}$$

$$k = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \tag{2}$$

$$b = \bar{y} - k\bar{x} \tag{3}$$

$$\theta = \begin{cases} -\tan^{-1}(k), & k < 0 \\ \tan^{-1}(k), & k >= 0. \end{cases} \tag{4}$$

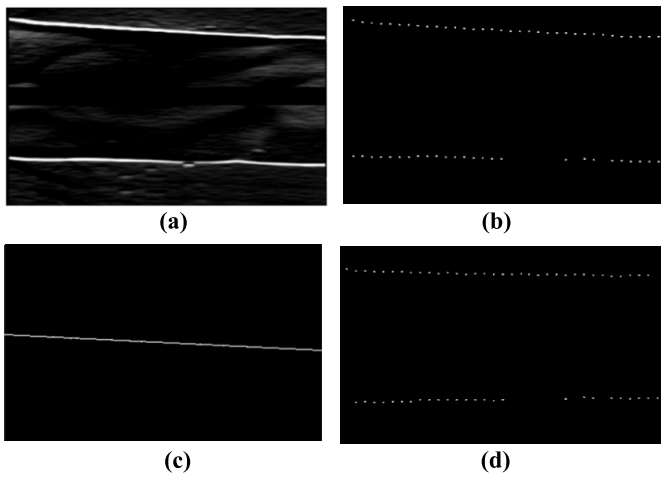


Fig. 3. Rotation correction of finger vein patterns. (a) Finger edge detection result and (b) parts of the edge points chosen from the edge information of (a). (c) Center line obtained by fitting and (d) result of deflection angle correction.

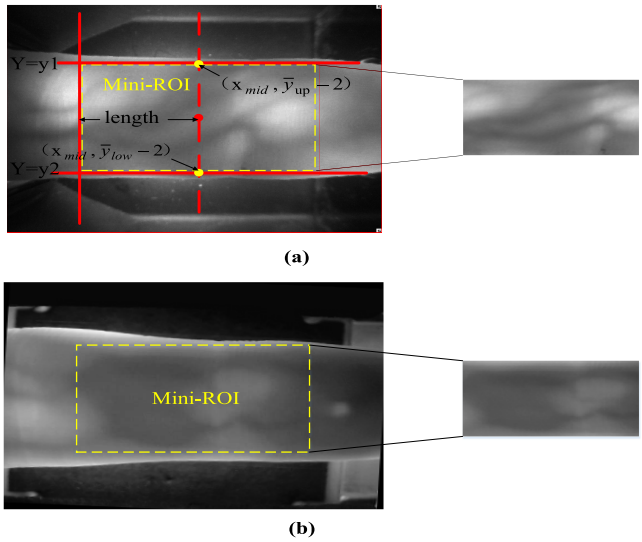


Fig. 4. Schematic of Mini-RoI region extraction for finger vein images. (a) Sample Mini-RoI extraction in PKU-FVD [47] and (b) sample Mini-RoI extraction in Shandong University Machine Learning and Applications Laboratory-Homologous Multimodal Biometric Traits (SDUMLA-HMT) [48].

According to the finger vein image after rotation offset correction, we can extract the features of the Mini-RoI region from the finger vein image. The procedure is as follows. First, we select the vertical coordinate of the upper edge point in the binary image of the finger vein after rotation correction and find the average value $\bar{y}_{\text{up}}$, and the vertical coordinate value is randomly selected at a fixed and reasonable interval. In order to reduce the bias, we have obtained a more accurate edge point when the average value of the edge points $\bar{y}_{\text{up}}$ minus 2 pixels after a large number of experiments. Therefore, we take the average of the upper edge points $\bar{y}_{\text{up}} - 2$ as the final edge value. As shown in Fig. 4(a), we choose the value of $\bar{y}_{\text{up}} - 2$ as the upper edge vertical coordinate value and noted as $y1$. Similarly, the vertical coordinate of the lower edge point is selected and the average $\bar{y}_{\text{low}}$ is found, and the value of $\bar{y}_{\text{low}} - 2$ is chosen as the value of the lower edge vertical coordinate, which is noted as $y2$.

After getting the upper and lower edges of the target area, we adjusted the width of the RoI area. As shown in Fig. 4,

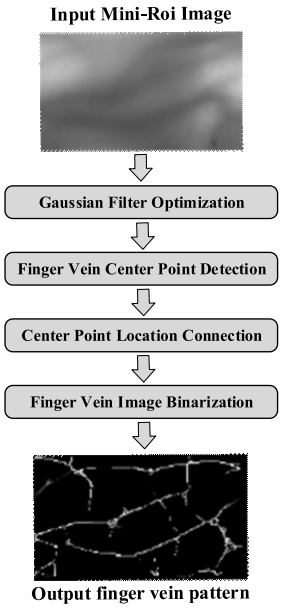


Fig. 5. Finger vein pattern feature extraction.

the core feature of the finger vein is generally located in the center of the finger and its longitudinal center axis is easily determined according to the length of the image, so we select the appropriate parameter length and then move the length from the fixed center axis to the left and right edges, respectively. Finally, the cropped vein feature region images are uniformly resized to the same size, and the final finger vein Mini-RoI region of interest images are obtained. An example of the extracted Mini-RoI regions from the processed finger vein samples is shown in Fig. 4(b).

B. Maximum Curvature Finger Vein Pattern Feature Extraction Algorithm Based on Gaussian Filter Optimization

In Section III-B1, we obtain the core region of the finger vein image by feature extraction from the Mini-RoI region of the finger vein image, which removes the effect of background noise and reduces the computational consumption. In this section, we extract the finger vein pattern features on the Mini-RoI region and propose an algorithm for extracting finger vein pattern features with maximum curvature based on Gaussian filter optimization. The method mainly consists of four steps: Gaussian filter optimization, finger vein center point detection, center point location connection, and finger vein image binarization. Fig. 5 illustrates the flowchart of maximum curvature finger vein pattern feature extraction based on Gaussian filtering optimization.

Gaussian filtering can smooth the finger vein image, thus effectively reducing the influence of interference information such as noise points and burrs on recognition accuracy. The expression of the Gaussian function is shown in the following equation:

$$h(x, y) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{5}$$

where σ is the standard deviation, which is the estimated parameter of the Gaussian blur kernel function. The higher the standard deviation of the function, the higher the blurring of the image. x and y represent the values of the horizontal and vertical coordinates, respectively.

We calculate the local maximum curvature of the finger vein pattern on the finger vein image, which can enhance the feature extraction accuracy and describe the vein features in more detail. Meanwhile, the method can effectively extract the centerline of the vein independent of the fluctuation of vein brightness and width, so the image after feature extraction has high accuracy when used for matching. This method mainly consists of three steps: finger vein center point detection, center point location connection, and finger vein image binarization.

1) Finger Vein Center Point Detection: The curvature of the finger vein image in cross-sectional view is calculated as follows:

$$F(x, y) = T_{\mathrm{rs}}(P_f(z)) \tag{6}$$

where $F(x, y)$ denotes the pixel value at the point (x, y) on the image. We define $P_{f(z)}$ to denote a transverse profile intercepted in the image $F(x, y)$, and T_{rs} is a mapping function from the point $Z(x, y)$ to $F(x, y)$ [as shown in (6)], and use $K(z)$ to denote the curvature value at z. The detailed calculation method is shown in the following equation:

$$k(z) = \frac{\frac{d^2 p_{f(z)}}{dz^2}}{\left(1 + \left(\frac{dp_{f(z)}}{dz}\right)^2\right)^{\frac{3}{2}}}. \tag{7}$$

Detection of the Center Position of the Finger Vein Pattern: We classify the intercepted contours into concave or convex according to the value of curvature $K(z)$. If $K(z)$ is positive, we consider the profile as a concave surface, and we can calculate the local maximum value of $K(z)$ for each concave surface in the profile, the point with the local maximum curvature value can indicate the center of the dent, which is the center of the vein.

Assuming that the number of local maximum curvature values k in the profile obtained from the previous step is N, we define the set of locations of these possible centroids as Z_i', where i is an integer from 1 to 4. After obtaining the set of centroid locations Z_i' in the previous step, we use S_{cr} to express the probability that these local maximum points are located on the vein, which is defined as shown in the following equation:

$$S_{\mathrm{cr}}(z_i') = k(z_i') * W_r(i) \tag{8}$$

where $W(i)$ is the width of the region with positive curvature value containing the local maximum curvature point z_i'. If the value of $W(i)$ is large, the probability of the point is located on the vein feature is also large. In addition, when the point is clearly present, its curvature value is large, and the probability of the point is located in the center of the vein is also high. Therefore, when calculating the score, both the width of the region and the curvature value of the corresponding point need to be considered. As shown in the following equation, we assign the statistical scores to a plane. This function value is an important reference for judging veins

$$V(x_i', y_i') = V(x_i', y_i') + S_{\mathrm{cr}}(z_i') \tag{9}$$

where x_i', y_i' represents the points defined by the following equation:

$$F\left(x_i', y_i'\right) = T_{\mathrm{rs}}\left(P_f\left(z_i'\right)\right). \tag{10}$$

The above process realizes the contour in the profile of the vein image in one direction, and in order to obtain the pattern features of the finger vein features on the whole sample image in more detail, we need to analyze the contour in the cross sections of the vein image in multiple directions. So, we sequentially analyze the vein pattern in the cross-sectional profiles in four directions (horizontal, vertical, and two diagonal directions) of the finger vein image, and finally detect the center of the finger vein feature by calculating the local maximum curvature value in each plane direction. We use $Vd1$, $Vd2$, $Vd3$, and $Vd4$ to represent the contour images of the finger veins in four directions.

2) Center Point Location Connection: After getting the center point location of the finger veins in the previous step, we need to connect them to get the vein pattern $G(x, y)$. When the point (x, y) is smaller than the pixel value on both sides, then the line will have a gap at the point. For the continuity of the finger vein feature, we increase the value at the point to connect the broken vein feature curves. Similarly, when the value of the point (x, y) is larger than the value of the pixels on both sides, the point will be recognized as a noise point, in which case the value at the point (x, y) should be removed to eliminate some of the noise. The specific implementation is described in the following equation:

$$\begin{aligned} C_{d1}(x, y) = \min\{&\max(V(x+1, y), V(x+2, y)) \\ &+ \max(V(x-1, y), V(x-2, y))\}. \end{aligned} \tag{11}$$

We eliminate the noise points from the contour map Vd_i $(i = 1, 2, 3, 4)$ according to (11) and connect the finger vein feature dispersion points. Finally, the contour maps of the veins in four directions, horizontal, vertical, 45°, and 135°, which obtained and represented by Cd_i, where i is an integer from 1 to 4. Then, we traverse the four contour maps of Cd_i and select the maximum value of each pixel in the four vein contour maps to obtain the final vein pattern extraction image $G(x, y)$, as shown in the following equation:

$$G(x, y) = \max(Cd_1, Cd_2, Cd_3, Cd_4). \tag{12}$$

3) Finger Vein Image Binarization: We use a suitable threshold to binarize the image $G(x, y)$ and obtain a binary image of the finger vein pattern features. The specific method is based on the obtained vein pattern $G(x, y)$, traverse all vein centroids on the pattern $G(x, y)$, and select the pixel value located in the median feature point as the threshold for image binarization, and the specific finger vein pattern feature extraction results are shown in Fig. 6.

Fig. 6(a) shows three different finger vein Mini-RoI feature region images, and the finger vein pattern feature extraction results achieved according to the above algorithm steps are shown in Fig. 6(b). In Fig. 6(b), we can see that there are still some burrs and noise points in the finger vein pattern feature images. Before extracting the finger vein pattern feature, we perform Gaussian filtering and smoothing on the finger vein image, which can effectively reduce the number of noise points and burrs. The experimental results after optimization by Gaussian filtering are shown in Fig. 6(c). After adding a Gaussian smoothing operation, the noise and burrs in the finger vein pattern image are effectively reduced and suppressed.

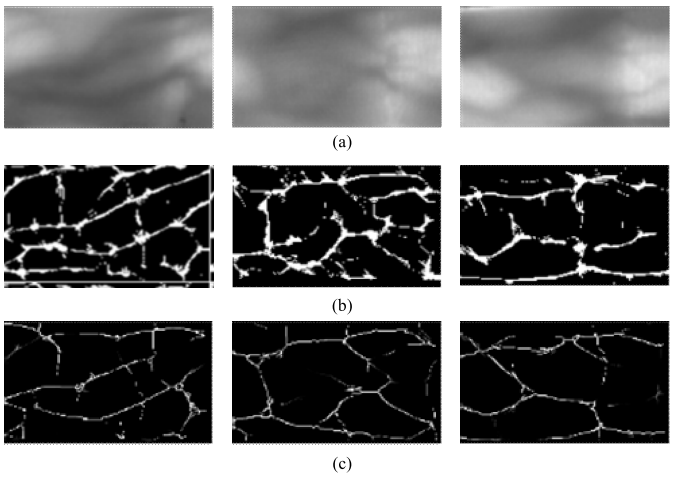


Fig. 6. Example of finger vein feature extraction. (a) Finger vein Mini-ROI region. (b) Finger vein texture features without Gaussian filtering. (c) Finger vein features optimized with Gaussian filtering (ours).

C. Lightweight Deep Network for Finger Vein Image Recognition

In this section, we build a lightweight finger vein recognition network based on channel stack, which includes two feature extraction and recognition modules, namely Stem Block and Stage Block. Stem Block is used to quickly convolve the input image information in a multi-scale manner and obtain more accurate position information of the features. The Stage Block performs a more refined stepwise stacking operation on the input features extracted by the Stem Block, which allows the deep model to obtain more detailed features with less computational effort. In this lightweight deep network, feature extraction and propagation are performed by stacking on the feature channels. In addition, we use a small convolutional kernel of 1×1 in the recognition network to improve the extraction of detail features, a uniform number of channels for internal propagation of features to reduce the computational cost of the model, and a three-channel feature concat approach for feature fusion.

1) Overall Architecture: Inspired by the network structures of Inception-v4 [51] and PeleeNet [52], we built a lightweight finger vein feature extraction and recognition network model consisting of Stem Block and Stage Block. Stem Block extracts and fuses the features of the input image at two different feature scales, where one way performs the Max pooling operation, while the other way performs the convolution operation, and finally performs the Concat feature fusion operation. Stage Block performs more refined processing on the features extracted from the Stem Block. In the propagation of feature channels, a stack method is used to gradually increase the number of feature channels, and the three-channel feature extraction and fusion method is

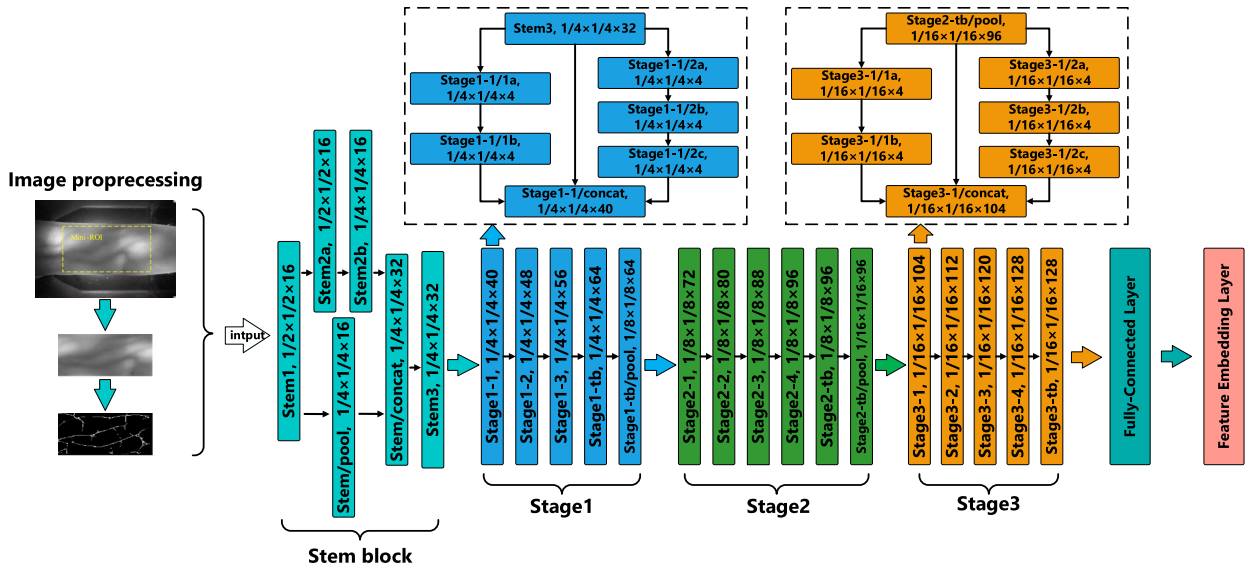


Fig. 7. Finger vein recognition network architecture based on lightweight depth model.

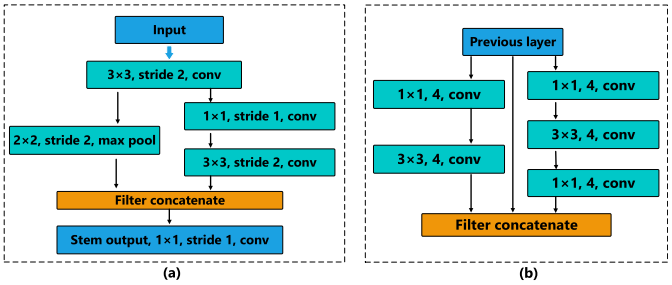


Fig. 8. Feature structure inside (a) stem block and (b) stage block.

used for feature extraction. Finally, the features extracted by Stage Block are input into the fully connected layer and feature vector layer, and the finger vein images are finally recognized and matched. Fig. 7 shows the finger vein recognition network architecture.

2) Stem Block: The stem block is divided into three small stem layers, namely Stem 1, Stem 2, and Stem 3. The function of Stem 1 is to perform a preliminary convolution operation on the image of the input channel, and the output is half the length and width of the input image. As shown in Fig. 8(a), the size of the input image is $H \times W \times 3$, and the size of the output feature map is $H/2 \times W/2 \times 16$ after inputting the image to the Stem 1 layer. The features extracted from the Stem Block are divided into two channels for feature extraction at different scales, one of which is Stem 2 and the convolution operation is performed while the size of the feature map is reduced to half of the original size, and the number of channels of features remains unchanged. The other way performs a maximum pooling operation with a convolution size of 2×2, and the output is the same size as Stem 2, the two blocks are fused with Concat features and the fused results are passed into the Stem 3 layer. Fig. 8(a) illustrates the internal structure of the Stem block.

Stem Block extracts the input image with two different feature scales and fuses the extracted features, one of which is used for Max pooling and the other for convolution, while the two extracted features are fused in Concat feature fusion. The Stem Block reduces the information loss in the original input image and performs the initial feature extraction in a multi-scale manner, which allows the model to improve its expressiveness without adding much computational cost. The results of the experiments show that the Stem Block effectively improves the recognition performance of the model. Different from PeleeNet, we unify the number of internal channels of the features in Stem Block, which speeds up the extraction and propagation of the model while significantly reducing the computational cost of the model.

3) Stage Block: The function of Stage Block is to process the features passed from the Stem Block feature extraction module in a more refined way and to perform detailed feature extraction and propagation by stacking the convolutional layers step by step through channel stacking. As shown in Fig. 7, the Stage Block has three small modules: Stage 1, Stage 2, and Stage 3. The structure diagram of feature propagation inside the Stage Block is shown in Fig. 8(b). In Stage Block, we use a three-branch feature propagation and fusion for feature extraction. One branch uses a 1×1 and a 3×3 convolution, another branch uses a 1×1 convolution kernel before and after the 3×3 convolution, and the last branch is the original input features. Then, the features extracted from these three branches are fused by Concat feature fusion, the final feature map of the layer is obtained after a convolutional operation with a convolutional kernel size of $1 \times 1 \times 16$. The three-way branching feature fusion can effectively obtain feature information at different scales, and the use of 1×1 small convolution also improves the model's ability to extract small object features while reducing the computational cost of the model. Moreover, in the Stage Block feature extraction module, we also adopt the design principle of unifying the number of internal channels of features, and the number of channels in internal feature propagation is unified to 4, which can further accelerate the speed of feature extraction and propagation.

The general convolutional neural network has a crude way of increasing the number of channels during feature extraction and propagation, with the number of feature channels increasing in multiples of 2, for example, from 32 to 64 to 128, and so on. Unlike these deep convolutional neural networks, the feature channels in the proposed deep feature extraction network are gradually increased, for example, from 32 to 40 to 56, 64, . . . ,112, 120, 128. Therefore, the proposed feature extraction network has a stronger feature extraction capability compared with other algorithms.

In Stage Block, we use a 1 × 1 small convolutional kernel to extract the features of the image. This is because the 1 × 1 small convolution kernel is not only effective in improving the linear representation of the model, but also can significantly reduce the computational effort of the model. For example, if the input size of the upper layer is $n \times n \times a$, after a 3 × 3 convolution (stride = 1, pad = 1) operation with b channels, the size of the output feature map is $n \times n \times b$, where the parameter calculation amount of the convolutional layer is shown in the following equation:

$$a \times 3 \times 3 \times b = 9ab. \tag{13}$$

If a 1 × 1 convolution operation with c channels is performed on the $n \times n \times a$ feature map before the convolution operation, and then a 3 × 3 convolution layer with b output channels is performed, then the size of the output feature map is still $n \times n \times b$, but the number of parameters of the convolution operation is shown in the following equation:

$$a \times 1 \times 1 \times c + c \times 3 \times 3 \times b = ac + 9bc. \tag{14}$$

Obviously, the calculation amount of (14) is far lower than that of (13), because the 1 × 1 small convolution kernel can significantly reduce the parameter amount and dimension of the feature map.

D. Training With Triplet Loss

In the training process of the proposed neural network, we choose the approach of triplet loss to train the deep model. As is known, triplet loss can be used on fine-grained images identification, which has an excellent performance in face recognition with the deep learning method. Different from training with cross-entropy loss function based on soft-max, triplet loss can solve the issue that no fixed forehead category or the category is a variable, such as a face and a finger vein recognition which are both fine-grained images identifications.

It is suffering the problem of insufficient samples when training the finger vein images on a deep convolution model. Through our research, we find that adding triplet loss in the training model can effectively solve the problem of insufficient finger vein samples. Differing from the Siamese network [51] feeding two various inputs individually into the same CNN and comparing the similarity, the proposed training model based on triplet loss does not require manually annotating data, which consists of anchor, positive, and negative, named (A, P, N) triplets. The model based on triplet loss makes the distance between the anchor and the positive sample as small as possible during the training process while making the distance between the anchor and the negative samples as large as possible. The details of the calculation are shown in the following equations:

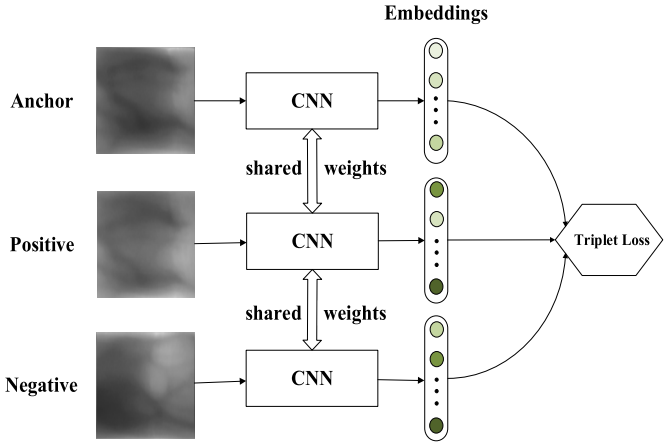


Fig. 9. Triplet loss on (A, P, N).

$$\text{triplet\_loss} = \left\| f\left(x_i^A\right) - f\left(x_i^P\right) \right\|_2^2 - \left\| f\left(x_i^A\right) - f\left(x_i^N\right) \right\|_2^2 + \alpha. \tag{15}$$

In (15) and (16), x_i^A, x_i^P, x_i^N, respectively, represent anchor, positive, and negative samples, and $f(x)$ means the feature vector of the input x. In addition, α represents a margin that can go to infinitesimal, which ensures the positive is standing for finger vein images of the same person, while the negative represents finger vein samples of different persons

$$\text{total\_loss} = \sum_i^n \left\{ \left\| f\left(x_i^A\right) - f\left(x_i^P\right) \right\|_2^2 - \left\| f\left(x_i^A\right) - f\left(x_i^N\right) \right\|_2^2 + \alpha \right\}_+. \tag{16}$$

In the process of model training, the neural network will traverse all the possible (A, P, N) triplets with gradient descent. It constantly updates the network's parameters so that the loss function can get its value going down and assure stabilized, so we can achieve an effective CNN model trained by triplet loss. Therefore, we can map the finger vein features to a 3-D vector space, where each vein pattern can obtain its feature vector as shown in Fig. 9.

Additionally, in the 3-D vector space, the Euclidean distance can be calculated by a vector acquired from the trained model. The feature vector is 128 dimensions in this article, which could be calculated by the CNN and the input images size. In this work, we use a triplet loss to train the model, the experiments get superior vein recognition results, and improve the amount of the finger vein training samples by selecting (A, P, N) triplets. When we use triplet loss to train the present models, the finger vein images are combined into different triplets, and the different combination ways represent different inputs, which can effectively solve the problem of insufficient finger vein samples.

After the model is trained over, each sample can be fed to the model to achieve its vector, then we can calculate the

distance between these vectors. During the experiment, it is not very difficult to find an appropriate threshold helping us to assure that the samples are from the same person if their distance is under the threshold. We regard that the smallest distance among vectors should be the most correct matching of the finger vein recognition.

IV. Experiment and Results

In this section, we first give a brief introduction to the dataset used in this article. Then the training method of the proposed algorithm in this article is presented in detail. Finally, the recognition performance of the proposed finger vein recognition algorithm and the results of the comparative analysis with other algorithms are shown.

A. Dataset Description

In the experiment, we use two public datasets to evaluate the proposed model. The first dataset is the SDUMLA-HMT database [48], which is captured over Shandong University, Jinan, China. The other one is Peking University Finger Vein dataset (PKU-FVD) [47], which is collected over Peking University, Beijing, China. We conduct experimental evaluations on these two public datasets to verify the performance of the proposed model. In addition, we compress the finger vein samples to a size of 160 × 160 to meet the requirements of the CNN input. We use the same method on the PKU dataset and SDUMLA dataset to expand the samples on the original image. The detailed process is that the original finger vein image is rotated at different angles to expand the sample, and the angle selection is gradually increased from $-10°$ to $10°$, so each finger vein image is expanded with 20 different samples.

1) SDMULA-HMT Database: The SDUMLA-HMT dataset contains 3816 finger vein images. Each subject owns 36 images acquired from one person's six fingers, which are the index, middle, and ring fingers of the left and right hands, respectively. Besides, there are six images per finger and they are all BMP format in gray level with the resolution of 320 × 240 pixels. Considering the limitation of the dataset, we make a classification to the SDMULA-HMT Database.

2) PKU-FV Database: PKU-FVD is collected by Peking University. In PKU-FVD, there is a public dataset part originally containing high-resolution images of 50 subjects having five images per finger, and the pixels of the images are 512 × 384, the original images are advisably rotated and translated to expand the dataset and at the same time to try to avoid over-fitting.

B. Training Details

The deep model in the experiment is implemented using the deep learning framework Tensorflow. The configuration and hardware environment of the computer in the experiment are as follows: CPU is Inter core i7-8700k, GPU is Nvidia 3090Ti (24-GB video memory), the operating system is Ubuntu 16.04 (Canonical, London, U.K.). In the training process, the stochastic gradient descent algorithm is used to train the network model. To prevent over-fitting and enhance the learning and detection capabilities of the network, a pre-training model is used to initialize the Stem Block. When initializing the newly added network layer, the initialization method of 0-mean Gaussian distribution is adopted, and the standard deviation is set to 0.01. During training, the weight decay rate is 0.0001, and the momentum is 0.9. The model has been trained for 50k iterations. In the first 30k trainings, the learning rate is set to 0.001, and in the remaining 20k iterations, the learning rate is set to 0.0001.

TABLE I

Time Cost in the Preprocessing of Finger Vein Images

Components	Image preprocessing					
Original image	✓	✓	✓	✓	✓	✓
Mini-RoI		✓	✓	✓	✓	✓
Gaussian Filter Optimization			✓	✓	✓	✓
Center Point Detection				✓	✓	✓
Center Point Location Connection					✓	✓
Finger Vein Image Binarization						✓
Time (SDUMLA-HMT)/per image	0ms	1.2ms	1.8ms	2.3ms	2.7ms	**3.2ms**
Time (PKU-FVD)/per image	0ms	1.4ms	2.1ms	2.5ms	2.9ms	**3.5ms**

TABLE II

Recognition Results of Our Method on the Finger Vein Databases

Components	The recognition results				
Original image	✓	✓	✓	✓	✓
Mini-RoI		✓	✓	✓	✓
Vein texture extraction			✓	✓	✓
Lightweight Network				✓	✓
Triplet Loss					✓
accuracy (SDUMLA-HMT)	86.3%	91.2%	94.3%	96.4%	**99.3%**
match time (SDUMLA-HMT)/per image	162ms	77.2ms	55.2ms	24.2ms	**14.2ms**
accuracy (PKU-FVD)	78.2%	86.3%	91.3%	97.3%	**99.6%**
match time (PKU-FVD)/per image	183ms	82.4ms	70.5ms	20.5ms	**16.5ms**

C. Results Evaluation and Comparison

The finger vein images are pre-processed with Min-Roi region feature extraction and finger vein pattern feature extraction before training in the proposed lightweight deep network. The finger vein pattern feature extraction mainly consists of four steps: Gaussian filter optimization, finger vein center point detection, center point location connection, and finger vein image binarization. The time consumed by finger vein images in each stage of preprocessing is shown in Table I. As shown in Table I, the average time consumed by each image for preprocessing on the SDUMLA dataset and PKU dataset is 3.2 and 3.5 ms, respectively. The proposed finger vein preprocessing method in this article can meet the real-time requirement of the matching algorithm. In the following comparative experimental, we consider the image preprocessing time within the total time cost of the algorithm.

Table II shows the impact of different improved components in our proposed finger vein recognition algorithm on recognition accuracy and matching time. We perform Mini-RoI region feature extraction on the original image in the finger vein recognition algorithm and perform finger vein pattern feature extraction on the Mini-RoI region. In addition, we use a lightweight deep network in the recognition and matching models, and the deep network is trained with a triplet loss

function. These innovations improve finger vein recognition accuracy to varying degrees while reducing matching time. As shown in Table II, the image preprocessing such as Mini-RoI region feature extraction and pattern feature extraction on the original finger vein image can effectively reduce the matching time of the model, while improving the recognition accuracy of the algorithm. This is because the original image contains a lot of interference information such as background noise, which not only increases the computational effort of the model, but also has a negative impact on the recognition accuracy. Therefore, image preprocessing is very important in the process of recognition and matching of finger vein images.

In addition, we introduce a lightweight channel stack deep network in the model for finger vein feature extraction and recognition, which improves the detection accuracy by 2.1% and 6.0% on the SDUMLA and PKU datasets, while the matching time is reduced by 31 and 50 ms, respectively. The lightweight depth model can effectively reduce the computational effort of the model during feature extraction and propagation, and the network can effectively extract the detailed features of the finger veins, so the introduction of this lightweight depth network not only improves the matching accuracy of the finger veins, but also reduces the computational effort of the depth model. In addition, we use a triplet loss function to train the model, and the depth model trained by this loss function can effectively realize the spatial mapping of vein features and calculate the similarity of images based on the distance while avoiding the retrain of neural network when the database changes dynamically, which finally effectively improves the accuracy and efficiency of finger vein recognition. As shown in Table II, the use of triplet loss in the model increases the detection accuracy by 2.9% and 2.3% on the SDUMLA and PKU datasets, while reducing the matching time by 41.3% and 19.5%, respectively.

Recently, a large number of backbone deep neural networks with superior performance have emerged in deep models. For example, large backbone networks such as Inception-Nets [49]–[51] and ResNet [23] have more layers and better feature extraction capability of the network, but the number of parameters of these network models is also large and the computational and memory consumption is high. To reduce the computation of deep neural networks, some scholars have developed lightweight deep neural networks, such as PeleeNet [52] and ShuffleNet [53], and this type of deep network greatly reduces the computation of models while ensuring accuracy. The lightweight deep network model has the advantages of the small number of model parameters and strong feature extraction ability. Therefore, lightweight deep neural networks have obvious advantages in detection, classification, and recognition tasks.

Table III shows the average results of different backbone networks for recognition and matching on the SDUMLA and PKU finger vein datasets. We train and test the images after finger vein pattern feature extraction on different backbone networks. As shown in Table III, the lightweight Stage Block proposed in this article has advantages in matching accuracy and time compared to other backbone networks. In addition, adding the Stage Block module on different backbone networks will also have a positive impact on the matching results. Compared with PeleeNet [52], the proposed lightweight deep neural network in this article is 2.9% higher in average recognition accuracy, while the matching time is only 28.3% of PeleeNet. Thus, the lightweight deep network proposed in this article is not only superior to other existing backbone networks in recognition accuracy, but also has obvious advantages in matching time and model size.

TABLE III

RESULTS OF DIFFERENT BACKBONE NETWORKS ON SDUMLA AND PKU DATASETS

Method	SDUMLA-acc	SDUMLA-time	PKU-acc	PKU-time
ShuffleNet-v2 [53]	94.3%	68ms	93.8%	83ms
Stem Block+ ShuffleNet-v2	95.2%	53ms	95.3%	62ms
Inception-v2 [49]	94.8%	89ms	94.6%	97ms
Stem Block+ Inception-v2	95.7%	72ms	95.9%	76ms
Inception-v3 [50]	95.3%	164ms	94.8%	173ms
Stem Block+ Inception-v3	96.4%	123ms	96.7%	136ms
Inception-v4 [51]	95.9%	183ms	95.5%	193ms
Stem Block+ Inception-v4	96.8%	142ms	97.3%	157ms
ResNet50 [23]	96.1%	131ms	95.9%	148ms
Stem Block+ResNet50	97.3%	97ms	97.6%	107ms
PeleeNet [52]	96.7%	42ms	96.4%	57ms
Stem Block+PeleeNet	97.8%	32ms	98.1%	41ms
Stage Block(the proposed network)	97.5%	21ms	97.8%	29ms
Stem Block+Stage Block(ours)	**99.3%**	**11ms**	**99.6%**	**13ms**

TABLE IV

RECOGNITION RESULTS OF OUR PROPOSED METHOD AND OTHER ALGORITHMS

Method	EER		accuracy		matching time/per image		Image preprocessing
	SDUMLA	PKU	SDUMLA	PKU	SDUMLA	PKU	
Hong et al. [5] (2017)	6.16%	13.74%	91.3%	92.6%	111.2ms	182.5ms	✓
Banerjee et al. [13] (2018)	8.30%	9.64%	90.7%	91.7%	71.2ms	115.5ms	✓
Yang et al. [6] (2019)	7.04%	6.89%	93.9%	94.5%	31.2ms	90.5ms	✓
Das et al. [14] (2019)	3.42%	4.67%	98.3%	95.1%	55.2ms	59.5ms	✓
Yang et al. [7] (2020)	1.71%	2.27%	94.7%	96.8%	16.2ms	41.5ms	✓
Rivan et al. [8] (2020)	1.68%	2.03%	95.8%	97.3%	24.2ms	45.5ms	✓
Zeng et al. [24] (2020)	1.56%	2.43%	96.4%	96.2%	23.2ms	42.5ms	✓
Zeng et al. [25] (2021)	1.63%	2.52%	96.1%	95.9%	20.2ms	30.5ms	✓
Wang et al. [26] (2021)	1.52%	1.32%	98.1%	97.9%	29.2ms	27.5ms	✓
Our method	**1.13%**	**0.67%**	**99.3%**	**99.6%**	**14.2ms**	**16.5ms**	✓

Table IV shows the results of the comparative analysis of our proposed algorithm with several other state-of-the-art finger vein recognition algorithms. As shown in Table IV, the proposed finger vein recognition algorithm has significant advantages in equal error rate (EER), accuracy rate, and detection time compared with several other recognition algorithms on SDUMLA and PKU finger vein datasets. In addition, we make a clarification in Table IV whether image preprocessing is required for the relevant comparison algorithms. In the comparison experiments, we do not directly replicate the results from the references, and all methods are replicated in our experimental environment. The same size dataset and hardware environment can ensure the fairness of different algorithms when comparing the results. In addition, we optimize the parameters of other algorithms in our hardware environment according to the relevant references and do our best to make the adopted parameters achieve the best results. Compared with Wang's algorithm in [26], the recognition accuracy of the proposed algorithm is 1.2% and 1.7% higher in SDUMLA and PKU digital vein datasets. Finally, the proposed finger vein recognition algorithm based on the lightweight depth model has an EER of 1.13% and 0.67%,

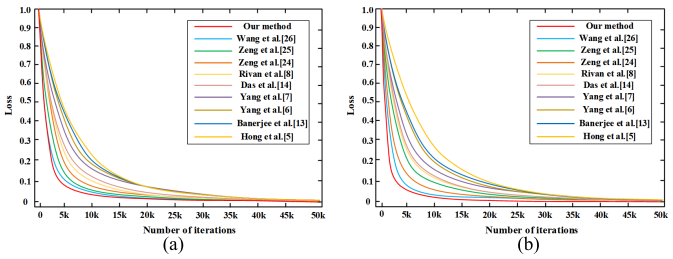


Fig. 10. Training loss curves (a) SDUMLA dataset and (b) PKU dataset.

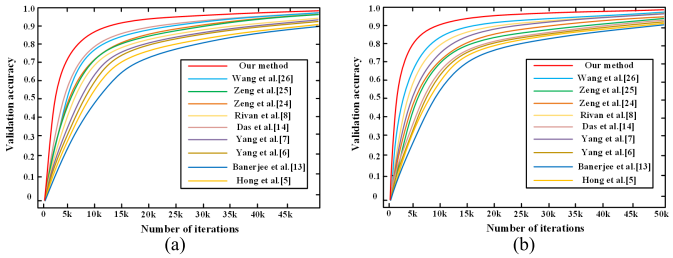


Fig. 11. Validation accuracy curves (a) SDUMLA dataset and (b) PKU dataset.

a recognition accuracy of 99.3% and 99.6%, and the matching time is 14.2 and 16.5 ms on SDUMLA and PKU finger vein datasets. Compared with the recognition results of previous finger vein recognition algorithms, the finger vein recognition algorithm proposed in this article has reached advanced results in recognition accuracy and matching efficiency.

In the proposed model, we use a triple loss function to train the deep model. This loss function can realize the spatial mapping of the fine-grained features of finger veins, which calculates the similarity between images based on the distance for recognition. Meanwhile, it avoids the retraining of the model when the database changes dynamically, thus effectively improving the recognition accuracy and efficiency. In addition, the triple loss function expands the number of datasets by combining different samples in the training model, which effectively solves the problem of over-fitting. Fig. 10 shows the results of the proposed finger vein matching algorithm compared with other algorithms in loss function curves during the training process. In the training process, the deep model performs 50 000 iterations, and output a loss function value every ten iterations, so a total of 5000 different loss function values are output. The values of the loss function are accurate to five decimal, so that a smoother training loss curve can be obtained, and the visualized loss curves of different algorithms can be better effectively distinguished. As shown in Fig. 10, the proposed matching algorithm has a faster convergence speed and better continuity than other algorithms.

Fig. 11 shows the results of the proposed detection model and other algorithms in validation accuracy on the SDUMLA dataset and PKU dataset. Similar to the training loss curves, the deep model obtains 5000 different verification accuracy values during the training process, and the value is accurate to five decimal. As a result, we also obtain a smoother curve in the verification accuracy rate. As shown in Fig. 11, the proposed model outperforms other deep recognition models in

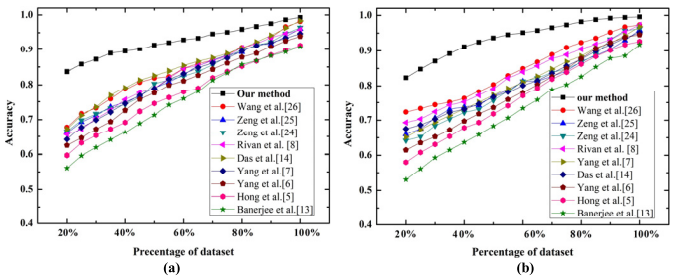


Fig. 12. Recognition accuracy of different scale datasets on each algorithm. (a) Percentage of training dataset (SDUMLA). (b) Percentage of training dataset (PKU).

verification accuracy, which achieves a higher accuracy rate in fewer iterations.

In addition, the finger vein recognition and matching algorithm proposed in this article can obtain high accuracy when the training dataset is insufficient. Fig. 12 shows the recognition accuracy on each algorithm for different scales of a dataset, where the horizontal axis represents the percentage of datasets used by the model during training and the vertical axis represents the recognition accuracy. As shown in Fig. 12, the recognition accuracy of each algorithm improves as the amount of training data increases. At the same time, the finger vein recognition and matching algorithm proposed in this article can not only obtain high recognition accuracy when the training data is insufficient, but also the detection accuracy is more obvious compared with other algorithms at this time. Therefore, the finger vein recognition and matching algorithm proposed in this article can effectively match and recognize finger vein images even on a small dataset.

Moreover, the above experiments demonstrate the superiority of the triplet loss function in the deep model-based finger vein matching task. While ensuring detection accuracy, the proposed method with a lightweight network has achieved better performance in reducing the computation cost. In addition, the previous deep learning methods in the finger vein recognition field always need to be retrained when a new category of finger vein data is added into the recognition system. In this article, we use triplet loss to train the model, when a stranger or new category comes to the system, it can be recognized by calculating the similarity of different samples through vectors' space distances without retraining models, so it is advanced than other algorithms.

V. Conclusion

Biometric identification technology can effectively protect the security of identity information and prevent the theft, loss, or forgetting of identity information, so this technology is now widely concerned. Traditional biometric technologies include fingerprint, palm-print, face, and finger vein recognition technologies. In this article, we propose a lightweight deep network finger vein recognition algorithm, which can effectively extract and match the features of finger vein images, and its recognition accuracy and matching speed are better than other previous recognition algorithms. In the finger vein image preprocessing session, this article proposes

a Mini-RoI finger vein region feature extraction algorithm, which can effectively extract the core feature regions in the finger vein image. Moreover, in the finger vein image feature extraction and recognition network, we use a lightweight deep network based on channel stacks, which can effectively reduce the computational effort of the model while acquiring detailed information features in the finger vein image. Therefore, the introduction of lightweight deep networks in the recognition algorithm can not only improve the recognition accuracy of the algorithm, but also reduce the matching time of the model.

Even though, our proposed method has insufficient generalization ability in the field of biometric recognition. The proposed lightweight deep model can run efficiently on embedded devices. Thus, the proposed algorithm has a generalization ability in the operation of hardware devices. In addition, the triplet loss maps the sample features into 3-D space and reduces the intra-class gap of the samples while expanding the inter-class gap. Therefore, the model trained by the triplet has strong robustness, and there is no need to retrain the model when a new category enters the trained model, and the samples can be judged directly according to the feature vector. Particularly, this can also reflect the robustness and generalization of the proposed algorithm. In future work, we will actively try to apply the proposed algorithm to other fields of biometric recognition.

REFERENCES

[1] S.-C. Wu, P.-L. Hung, and A. L. Swindlehurst, "ECG biometric recognition: Unlinkability, irreversibility, and security," *IEEE Internet Things J.*, vol. 8, no. 1, pp. 487–500, Jan. 2021.

[2] G. Jack, M. Ji, and C. Danny, "Matching larger image areas for unconstrained face identification," *IEEE Trans. Cybern.*, vol. 49, no. 8, pp. 3191–3202, Aug. 2019.

[3] A. Sengupta and M. Rathor, "Securing hardware accelerators for CE systems using biometric fingerprinting," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 9, pp. 1979–1992, Sep. 2020.

[4] X. Qian, Y. Fu, T. Xiang, Y.-G. Jiang, and X. Xue, "Leader-based multi-scale attention deep architecture for person re-identification," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 371–385, Feb. 2020.

[5] H. G. Hong, M. B. Lee, and K. R. Park, "Convolutional neural network-based finger-vein recognition using NIR image sensors," *Sensors*, vol. 17, no. 6, pp. 1297–1317, Jun. 2017.

[6] W. Yang, S. Wang, J. Hu, G. Zheng, J. Yang, and C. Valli, "Securing deep learning based edge finger vein biometrics with binary decision diagram," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4244–4253, Jul. 2019.

[7] W. Yang, W. Luo, W. Kang, Z. Huang, and Q. Wu, "FVRAS-net: An embedded finger-vein recognition and AntiSpoofing system using a unified CNN," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 11, pp. 8690–8701, Nov. 2020.

[8] R. S. Kuzu, E. Piciucco, E. Maiorana, and P. Campisi, "On-the-fly finger-vein-based biometric recognition using deep neural networks," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 2641–2654, 2020.

[9] W. Kang, H. Liu, W. Luo, and F. Deng, "Study of a full-view 3D finger vein verification technique," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 1175–1189, 2020.

[10] J. Zhang, Z. Lu, and M. Li, "Active contour-based method for finger-vein image segmentation," *IEEE Trans. Instrum. Meas.*, vol. 69, no. 11, pp. 8656–8665, Nov. 2020.

[11] W. Kang, Y. Lu, D. Li, and W. Jia, "From noise to feature: Exploiting intensity distribution as a novel soft biometric trait for finger vein recognition," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 4, pp. 858–869, Apr. 2019.

[12] R. Ramachandra, K. B. Raja, S. K. Venkatesh, and C. Busch, "Design and development of low-cost sensor to capture ventral and dorsal finger vein for biometric authentication," *IEEE Sensors J.*, vol. 19, no. 15, pp. 6102–6111, Aug. 2019.

[13] A. Banerjee, S. Basu, S. Basu, and M. Nasipuri, "ARTeM: A new system for human authentication using finger vein images," *Multimedia Tools Appl.*, vol. 77, no. 5, pp. 5857–5884, Mar. 2018.

[14] H. Liu, G. Yang, and Y. Yin, "Category-preserving binary feature learning and binary codebook learning for finger vein recognition," *Int. J. Mach. Learn. Cybern.*, vol. 11, no. 11, pp. 2573–2586, Jun. 2020.

[15] W. Hussain, N. Rasool, and M. Yaseen, "ADVIT: Using the potentials of deep representations incorporated with grid-based features of dorsum vein patterns for human identification," *Forensic Sci. Int.*, vol. 313, Aug. 2020, Art. no. 110345.

[16] W. Yang, C. Hui, Z. Chen, J.-H. Xue, and Q. Liao, "FV-GAN: Finger vein representation using generative adversarial networks," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 9, pp. 2512–2524, Sep. 2019.

[17] R. S. Kuzu, E. Maiorana, and P. Campisi, "Vein-based biometric verification using densely-connected convolutional autoencoder," *IEEE Signal Process. Lett.*, vol. 27, pp. 1869–1873, 2020.

[18] S. Li, B. Zhang, L. Fei, and S. Zhao, "Joint discriminative feature learning for multimodal finger recognition," *Pattern Recognit.*, vol. 111, Mar. 2021, Art. no. 107704.

[19] R. Das, E. Piciucco, E. Maiorana, and P. Campisi, "Convolutional neural network for finger-vein-based biometric identification," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 2, pp. 360–373, Feb. 2019.

[20] R. He, J. Cao, L. Song, Z. Sun, and T. Tan, "Adversarial cross-spectral face completion for NIR-VIS face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 5, pp. 1025–1037, May 2020.

[21] R. He, X. Wu, Z. Sun, and T. Tan, "Wasserstein CNN: Learning invariant features for NIR-VIS face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 7, pp. 1761–1773, Jul. 2019.

[22] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 1–9.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.

[24] J. Zeng *et al.*, "Finger vein verification algorithm based on fully convolutional neural network and conditional random field," *IEEE Access*, vol. 8, pp. 65402–65419, 2020.

[25] J. Zeng *et al.*, "Real-time segmentation method of lightweight network for finger vein using embedded terminal technique," *IEEE Access*, vol. 9, pp. 303–316, 2021.

[26] K. Wang, G. Chen, and H. Chu, "Finger vein recognition based on multi-receptive field bilinear convolutional neural network," *IEEE Signal Process. Lett.*, vol. 28, pp. 1590–1594, 2021.

[27] L. Yang, G. Yang, L. Zhou, and Y. Yin, "Superpixel based finger vein ROI extraction with sensor interoperability," in *Proc. Int. Conf. Biometrics (ICB)*, May 2015, pp. 444–451.

[28] Y. Lu, S. J. Xie, S. Yoon, Z. Wang, and D. S. Park, "An available database for the research of finger vein recognition," in *Proc. 6th Int. Congr. Image Signal Process. (CISP)*, Dec. 2013, pp. 410–415.

[29] A. Kumar and Y. Zhou, "Human identification using finger images," *IEEE Trans. Image Process.*, vol. 21, no. 4, pp. 2228–2244, Apr. 2012.

[30] X. Wen, J. Zhao, and X. Liang, "Image enhancement of finger-vein patterns based on wavelet denoising and histogram template equalization," *J. Jilin Univ. Sci. Ed.*, vol. 2, p. 026, Feb. 2008.

[31] J. Zhao, X. Xiong, and L. Zhang, "Study on enhanced algorithm of hand vein image based on CLAHE and top-hat transform," *Laser Infr.*, vol. 39, no. 2, pp. 220–222, 2009.

[32] X. Gao, M. Junshan, and W. Jia, "The research of finger-vein image enhancement algorithm," *Opt. Instrum.*, vol. 32, no. 4, pp. 29–32, 2010.

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

[34] C. He, Z. Li, L. Chen, and J. Peng, "Identification of finger vein using neural network recognition research based on PCA," in *Proc. IEEE 16th Int. Conf. Cognit. Informat. Cognit. Comput. (ICCI*CC)*, Jul. 2017, pp. 456–460.

[35] J.-D. Wu and C.-T. Liu, "Finger-vein pattern identification using SVM and neural network technique," *Expert Syst. Appl.*, vol. 38, no. 11, pp. 14284–14289, Oct. 2011.

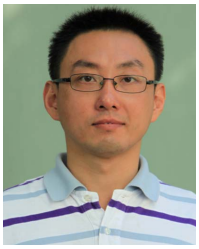
[36] N. Hu, H. Ma, and T. Zhan, "A new finger vein recognition method based on LBP and 2DPCA," in *Proc. 37th Chin. Control Conf. (CCC)*, Jul. 2018, pp. 9267–9272.

[37] C. Liu and Y.-H. Kim, "An efficient finger-vein extraction algorithm based on random forest regression with efficient local binary patterns," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 3141–3145.

[38] Z. Xiong, M. Liu, and Q. Guo, "Finger vein recognition method based on center-symmetric local binary pattern," in *Proc. IEEE Int. Conf. Artif. Intell. Comput. Appl. (ICAICA)*, Mar. 2019, pp. 262–266.
[39] R. Hernandez-Garcia, S. Guidet, R. J. Barrientos, and F. E. Frati, "Massive finger-vein identification based on local line binary pattern under parallel and distributed systems," in *Proc. 38th Int. Conf. Chilean Comput. Sci. Soc. (SCCC)*, Nov. 2019, pp. 1–7.
[40] W. Dahea and H. S. Fadewar, "Finger vein recognition system based on Multi- algorithm of fusion of Gabor filter and local binary pattern," in *Proc. 4th Int. Conf. I-SMAC (IoT Social, Mobile, Anal. Cloud) (I-SMAC)*, Oct. 2020, pp. 403–410.
[41] W. Song, T. Kim, H. Kim, J. Choi, H. Kong, and S. Lee, "A finger-vein verification system using mean curvature," *Pattern Recognit. Lett.*, vol. 32, no. 11, pp. 1541–1547, Aug. 2011.
[42] H. Qin, L. Qin, and C. Yu, "Region growth-based feature extraction method for finger-vein recognition," *Opt. Eng.*, vol. 50, no. 5, pp. 214–229, May 2011.
[43] N. Miura, A. Nagasaka, and T. Miyatake, "Feature extraction of finger-vein patterns based on repeated line tracking and its application to personal identification," *Mach. Vis. Appl.*, vol. 15, no. 4, pp. 194–203, Feb. 2004.
[44] N. Miura, A. Nagasaka, and T. Miyatake, "Extraction of finger-vein patterns using maximum curvature points in image profiles," *IEICE Trans. Inf. Syst.*, vol. 90, no. 8, pp. 1185–1194, May 2007.
[45] Z. Huang and C. Guo, "Robust finger vein recognition based on deep CNN with spatial attention and bias field correction," *Int. J. Artif. Intell. Tools*, vol. 30, no. 1, Feb. 2021, Art. no. 2140005.
[46] B. Hou and R. Yan, "ArcVein-arccosine center loss for finger vein verification," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–11, 2021.
[47] *PKU Finger Vein Database from Peking University*. Accessed: Sep. 18, 2013. [Online]. Available: https://rate.pku.edu.cn/
[48] Y. Yin, L. Liu, and X. Sun, "SDUMLA-HMT: A multimodal biometric database," in *Proc. Chin. Conf. Biometric Recognit.*, 2011, pp. 260–268.
[49] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and 0.5MB model size," 2016, *arXiv:1602.07360*.
[50] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 2818–2826.
[51] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-ResNet and the impact of residual connections on learning," 2016, *arXiv:1602.07261*.
[52] R. J. Wang, X. Li, and C. X. Ling, "Pelee: A real-time object detection system on mobile devices," 2018, *arXiv:1804.06882*.
[53] N. Ma, X. Zhang, H. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 116–131.
[54] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, Jun. 2005, pp. 539–546.

Jiaquan Shen was born in 1992. He received the M.S. degree in computer science and technology from Wenzhou University, Wenzhou, China, in 2017, and the Ph.D. degree in computer vision from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 2021.

He is currently a Lecturer with the School of Information Science, Luoyang Normal University, Luoyang, China. His research interests include deep learning and computer vision.



Ningzhong Liu was born in 1975. He received the B.S. degree in computer engineering and the Ph.D. degree in pattern recognition and intelligent systems from the Nanjing University of Science and Technology, Nanjing, China, in 1998 and 2003, respectively.

He is currently a Professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His research interests include artificial intelligence and machine learning.

Chenglu Xu was born in 1994. She received the B.S. degree in computer science and technology from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, where she is currently pursuing the M.S. degree.

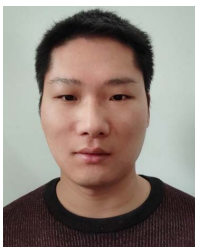
Her research interests include deep learning and computer vision.

Han Sun received the B.S. degree in computer engineering and the Ph.D. degree in pattern recognition and intelligent systems from the Nanjing University of Science and Technology, Nanjing, China, in 2000 and 2005, respectively.

He is currently an Associate Professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics. His research interests include machine learning and image processing.

Yushun Xiao is currently pursuing the doctor's degree in computer science with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, China.

His research interests include multiobjective optimization, mobile computing, and kernel method.



Deguang Li (Member, IEEE) received the B.S. degree in computer science from PLA Information Engineering University, Zhengzhou, China, in 2010, the M.S. degree in computer science from Northeastern University, Shenyang, China, in 2012, and the Ph.D. degree from Sichuan University, Chengdu, China, in 2017.

He is currently a Lecturer with the School of Information Science, Luoyang Normal University, Luoyang, China. His research interests include green computing and mobile computing.

Yongxin Zhang (Member, IEEE) received the B.E. degree in computer science and technology from Henan Polytechnic University, Jiaozuo, China, in 2003, and the M.S. and Ph.D. degrees in computer software and theory from Northwest University, Xi'an, China, in 2009 and 2014, respectively.

He is currently an Associate Professor with Luoyang Normal University, Luoyang, China. His research interest includes intelligent information processing.