

Chapter 2

Feature Learning for Facial Kinship Verification

Abstract In this chapter, we discuss feature learning techniques for facial kinship verification. We first review two well-known hand-crafted facial descriptors including local binary patterns (LBP) and the Gabor feature. Then, we introduce a compact binary face descriptor (CBFD) method which learns face descriptors directly from raw pixels. Unlike LBP which samples small-size neighboring pixels and computes binary codes with a fixed coding strategy, CBFD samples large-size neighboring pixels and learn a feature filter to obtain binary codes automatically. Subsequently, we present a prototype-based discriminative feature learning(PDFL) method to learn mid-level discriminative features with low-level descriptor for kinship verification. Unlike most existing prototype-based feature learning methods which learn the model with a strongly labeled training set, this approach works on a large unsupervised generic set combined with a small labeled training set. To better use multiple low-level features for mid-level feature learning, a multiview PDFL (MPDFL) method is further proposed to learn multiple mid-level features to improve the verification performance.

2.1 Conventional Face Descriptors

2.1.1 Local Binary Patterns

Local binary pattern (LBP) is a popular texture descriptor for feature representation. Inspired by its success in texture classification, Ahonen et al. [1] proposed a novel LBP feature representation method for face recognition. The basic idea is as follows: for each pixel, its 8-neighborhood pixels are thresholded into 1 or 0 by comparing them with the center pixel. Then the binary sequence of the 8-neighborhoods is transferred into a decimal number (bit pattern states with upper left corner moving clockwise around center pixel), and the histogram with 256 bins of the processed image is used as the texture descriptor. To capture the dominant features, Ahonen et al. [1] extended LBP to a parametric form (P, R), which indicates that P gray-scale values are equally distributed in a circle with radius R to form circularly symmetric neighbor sets. To better characterize the property of texture information, uniform pattern is

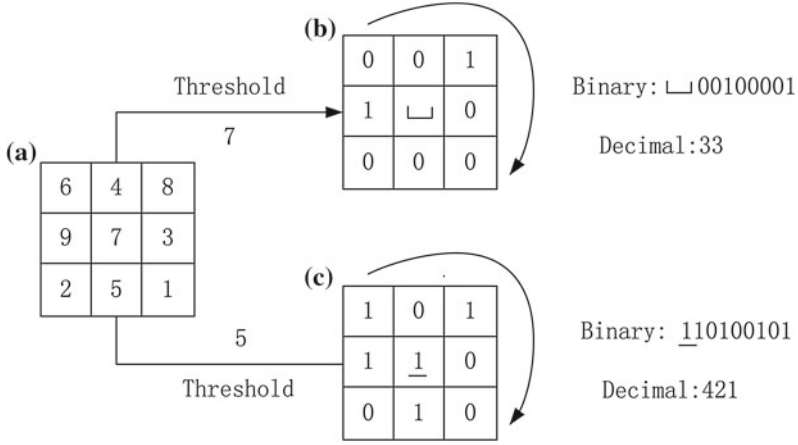


Fig. 2.1 The LBP and ILBP operators where **a** Original image patch **b** Results of LBP operator **c** Results of ILBP operator

defined according to the number of spatial transitions that are bitwise 0/1 changes in the calculated binary values. If there are at most two bitwise transitions from 0 to 1 or vice versa, the binary value is called a uniform pattern.

Recently, Jin et al. [22] developed an improved LBP (ILBP) method. Different from LBP, ILBP makes better use of the central pixel in the original LBP and takes the mean of all gray values of elements as the threshold value. For the newly generated binary value of the central pixel, it was added to the most left position of the original binary string. The corresponding decimal value range would be changed from [0, 255] to [0, 510] for a 3×3 operator. Figure 2.1 shows the basic ideas of LBP and ILBP. Since LBP and ILBP are robust to illumination changes, they have been widely used in many semantic understanding tasks such as face recognition and facial expression recognition.

2.1.2 Gabor Feature Representation

Gabor wavelet is a popular feature extraction method for visual signal representation, and discriminative information is extracted by convoluting the original image with a set of Gabor kernels with different scales and orientations. A 2-D Gabor wavelet kernel is the product of an elliptical Gaussian envelope and a complex plane wave, defined as [31]:

$$\psi_{\mu,v}(z) = \frac{\|k_{\mu,v}\|^2}{\sigma^2} e^{-\frac{\|k_{\mu,v}\|^2 \|z\|^2}{2\sigma^2}} [e^{ik_{\mu,v}z} - e^{-\frac{\sigma^2}{2}}] \quad (2.1)$$

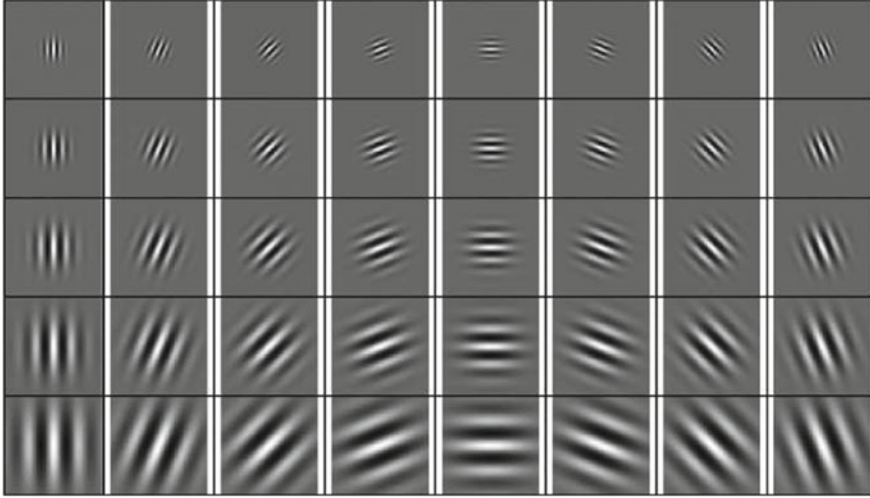


Fig. 2.2 Real part of Gabor kernels at eight orientations and five scales [31]

where μ and ν define the orientation and scale of the Gabor kernels, $z = z(x, y)$ is the variable in a complex spatial domain, $\|\cdot\|$ denotes the norm operator, and the wave vector $k_{\mu,\nu}$ is defined as follows:

$$k_{\mu,\nu} = k_\nu e^{j\phi_\mu} \quad (2.2)$$

where $k_\nu = k_{\max}/f^\nu$, $\phi_\mu = \pi\mu/8$, k_{\max} is the maximum frequency, f is the spacing factor between kernels in the frequency domain, and σ is the standard deviation of Gaussian envelope determining the number of oscillations. For a given image I , the convolution of I and a Gabor kernel $\psi_{\mu,\nu}$ is defined as

$$O_{\mu,\nu}(z) = I(z) * \psi_{\mu,\nu}(z) \quad (2.3)$$

where $O_{\mu,\nu}(z)$ is the convolution result corresponding to the Gabor kernel at orientation μ and scale ν . Figure 2.2 shows the the real part of the Gabor kernels.

Usually, five spatial frequencies and eight orientations are used for Gabor feature representation, and there will be a total of 40 Gabor kernel functions employed on each pixel of an image. Its computational cost is generally expensive. Moreover, only the magnitudes of Gabor wavelet coefficients are used as features because the phase information are sensitive to inaccurate alignment.

2.2 Feature Learning

There have been a number of feature learning methods proposed in recent years [3, 16, 18, 20, 26, 38]. Representative feature learning methods include sparse auto-encoders [3], denoising auto-encoders [38], restricted Boltzmann machine [16], convolutional neural networks [18], independent subspace analysis [20], and reconstruction independent component analysis [26]. Recently, there have also been some work on feature learning-based face representation, and some of them have achieved reasonably good performance in face recognition. For example, Lei et al. [29] proposed a discriminant face descriptor (DFD) method by learning an image filter using the LDA criterion to obtain LBP-like features. Cao et al. [7] presented a learning-based (LE) feature representation method by applying the bag-of-word (BoW) framework. Hussain et al. [19] proposed a local quantized pattern (LQP) method by modifying the LBP method with a learned coding strategy. Compared with hand-crafted feature descriptors, learning-based feature representation methods usually show better recognition performance because more data-adaptive information can be exploited in the learned features.

Compared with real-valued feature descriptors, there are three advantages for binary codes: (1) they save memory, (2) they have faster computational speed, and (3) they are robust to local variations. Recently, there has been an increasing interest in binary code learning in computer vision [14, 40, 43, 44]. For example, Weiss et al. [44] proposed an efficient binary coding learning method by preserving the similarity of original features for image search. Norouzi et al. [36] learned binary codes by minimizing a triplet ranking loss for similar pairs. Wang et al. [43] presented a binary code learning method by maximizing the similarity of neighboring pairs and minimizing the similarity of non-neighboring pairs for image retrieval. Trzcinski et al. [40] obtained binary descriptors from patches by learning several linear projections based on pre-defined filters during training. However, most existing binary code learning methods are developed for similarity search [14, 40] and visual tracking [30]. While binary features such as LBP and Haar-like descriptor have been used in face recognition and achieved encouraging performance, most of them are hand-crafted. In this chapter, we first review the compact binary face descriptor method which learns binary features directly from raw pixels for face representation. Then, we introduce a feature learning approach to learn mid-level discriminative features with low-level descriptor for kinship verification.

2.2.1 Learning Compact Binary Face Descriptor

While binary features have been proven to be very successful in face recognition [1], most existing binary face descriptors are all hand-crafted (e.g., LBP and its extensions) and they suffer from the following limitations:

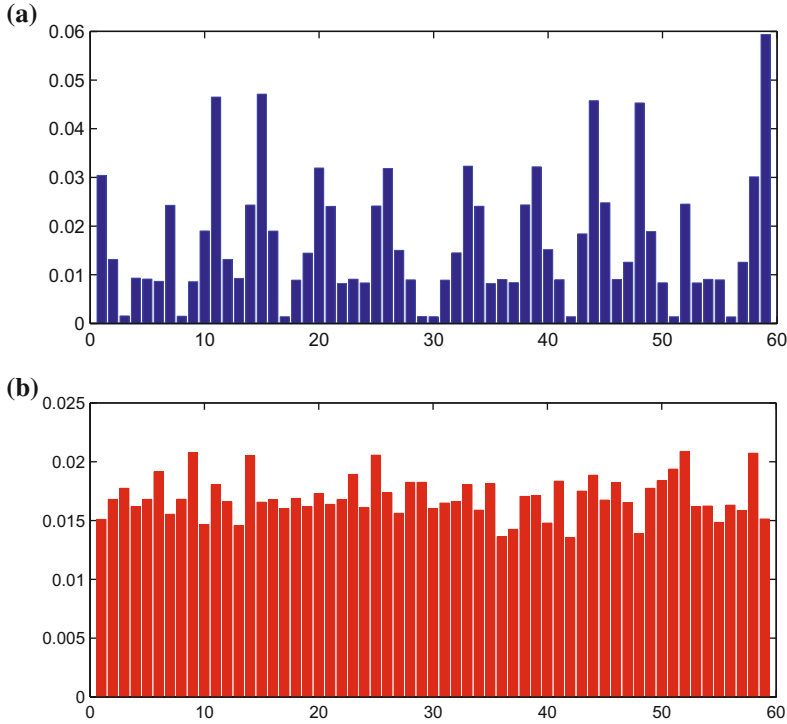


Fig. 2.3 The bin distributions of the **a** LBP and **b** our CBFD methods. We computed the bin distributions in the LBP histogram and our method in the FERET training set, which consists of 1002 images from 429 subjects. For a fair comparison, both of them adopted the same number of bins for feature representation, which was set to 59 in this figure. We clearly see from this figure that the histogram distribution is uneven for LBP and is more uniform for our CBFD method. © [2015] IEEE. Reprinted, with permission, from Ref. [35]

1. It is generally impossible to sample large size neighborhoods for hand-crafted binary descriptors in feature encoding due to the high computational burden. However, a large sample size is more desirable because more discriminative information can be exploited in large neighborhoods.
2. It is difficult to manually design an optimal encoding method for hand-crafted binary descriptors. For example, the conventional LBP adopts a hand-crafted codebook for feature encoding, which is simple but not discriminative enough because the hand-crafted codebook cannot well exploit more contextual information.
3. Hand-crafted binary codes such as those in LBP are usually unevenly distributed, as shown in Fig. 2.3a. Some codes appear less than others in many real-life face images, which means that some bins in the LBP histogram are less informative and compact. Therefore, these bins make LBP less discriminative.

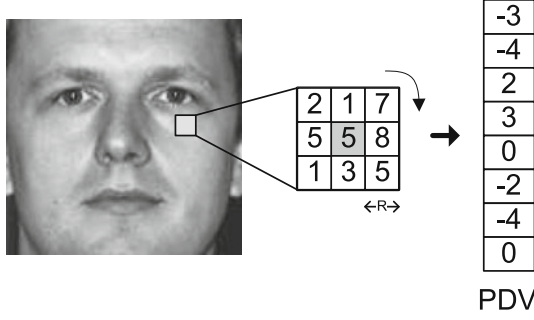


Fig. 2.4 One example to show how to extract a pixel difference vectors (PDV) from the original face image. For any pixel in the image, we first identify its neighbors in a $(2R + 1) \times (2R + 1)$ space, where R is a parameter to define the neighborhood size and it is selected as 1 in this figure for easy illustration. Then, the difference between the center point and neighboring pixels is computed as the PDV. © [2015] IEEE. Reprinted, with permission, from Ref. [35]

To address these limitations, the compact binary local feature learning method [35] was proposed to learn face descriptors directly from raw pixels. Unlike LBP which samples small-size neighboring pixels and computes binary codes with a fixed coding strategy, we sample large-size neighboring pixels and learn a feature filter to obtain binary codes automatically. Let $X = [x_1, x_2, \dots, x_N]$ be the training set containing N samples, where $x_n \in \mathbb{R}^d$ is the n th pixel difference vector (PDV), and $1 \leq n \leq N$. Unlike most previous feature learning methods [18, 26] which use the original raw pixel patch to learn the feature filters, we use PDVs for feature learning because PDV measures the difference between the center point and neighboring pixels within a patch so that it can better describe how pixel values change and implicitly encode important visual patterns such as edges and lines in face images. Moreover, PDV has been widely used in many local face feature descriptors, such as hand-crafted LBP [1] and learning-based DFD [29]. Figure 2.4 illustrates how to extract one PDV from the original face image. For any pixel in the image, we first identify its neighbors in a $(2R + 1) \times (2R + 1)$ space, where R is a parameter to define the neighborhood size. Then, the difference between the center point and neighboring pixels is computed as the PDV. In our experiments, R is set as 3 so that each PDV is a 48-dimensional feature vector.

Our CBFD method aims to learn K hash functions to map and quantize each x_n into a binary vector $b_n = [b_{n1}, \dots, b_{nK}] \in \{0, 1\}^{1 \times K}$, which encodes more compact and discriminative information. Let $w_k \in \mathbb{R}^d$ be the projection vector for the k th function, the k th binary code b_{nk} of x_n can be computed as

$$b_{nk} = 0.5 \times (\text{sgn}(w_k^T x_n) + 1) \quad (2.4)$$

where $\text{sgn}(v)$ equals to 1 if $v \geq 0$ and -1 otherwise.

To make b_n discriminative and compact, we enforce three important criterions to learn these binary codes:

1. The learned binary codes are compact. Since large-size neighboring pixels are sampled, there are some redundancy in the sampled vectors, making them compact can reduce these redundancy.
2. The learned binary codes well preserve the energy of the original sample vectors, so that less information is missed in the binary codes learning step.
3. The learned binary codes evenly distribute so that each bin in the histogram conveys more discriminative information, as shown in Fig. 2.3b.

To achieve these objectives, we formulate the following optimization objective function:

$$\begin{aligned}
\min_{w_k} J(w_k) &= J_1(w_k) + \lambda_1 J_2(w_k) + \lambda_2 J_3(w_k) \\
&= - \sum_{n=1}^N \|b_{nk} - \mu_k\|^2 \\
&\quad + \lambda_1 \sum_{n=1}^N \|(b_{nk} - 0.5) - w_k^T x_n\|^2 \\
&\quad + \lambda_2 \sum_{n=1}^N \|b_{nk} - 0.5\|^2
\end{aligned} \tag{2.5}$$

where N is the number of PDVs extracted from the whole training set, μ_k is the mean of the k th binary code of all the PDVs in the training set, which is recomputed and updated in each iteration in our method, λ_1 and λ_2 are two parameters to balance the effects of different terms to make a good trade-off among these terms in the objective function.

The physical meanings of different terms in (2.5) are as follows:

1. The first term J_1 is to ensure that the variance of the learned binary codes are maximized so that we only need to select a few bins to represent the original PDVs in the learned binary codes.
2. The second term J_2 is to ensure that the quantization loss between the original feature and the encoded binary codes is minimized, which minimizes the information loss in the learning process.
3. The third term J_3 is to ensure that feature bins in the learned binary codes evenly distribute as much as possible, so that they are more compact and informative to enhance the discriminative power.

Let $W = [w_1, w_2, \dots, w_K] \in \mathbb{R}^{d \times K}$ be the projection matrix. We map each sample x_n into a binary vector as follows:

$$b_n = 0.5 \times (\text{sgn}(W^T x_n) + 1). \tag{2.6}$$

Then, (2.5) can be re-written as:

$$\begin{aligned}
\min_W J(W) &= J_1(W) + \lambda_1 J_2(W) + \lambda_2 J_3(W) \\
&= -\frac{1}{N} \times \text{tr}((B - U)^T (B - U)) \\
&\quad + \lambda_1 \| (B - 0.5) - W^T X \|_F^2 \\
&\quad + \lambda_2 \| (B - 0.5) \times \mathbf{1}^{N \times 1} \|_F^2
\end{aligned} \tag{2.7}$$

where $B = 0.5 \times (\text{sgn}(W^T X) + 1) \in \{0, 1\}^{N \times K}$ is the binary code matrix and $U \in \mathbb{R}^{N \times K}$ is the mean matrix which is repeated column vector of the mean of all binary bits in the training set, respectively.

To our knowledge, (2.7) is an NP-hard problem due to the non-linear $\text{sgn}(\cdot)$ function. To address this, we relax the $\text{sgn}(\cdot)$ function as its signed magnitude [14, 43] and rewrite $J_1(W)$ as follows:

$$\begin{aligned}
J_1(W) &= -\frac{1}{N} \times (\text{tr}(W^T X X^T W)) \\
&\quad - 2 \times \text{tr}(W^T X M^T W) \\
&\quad + \text{tr}(W^T M M^T W)
\end{aligned} \tag{2.8}$$

where $M \in \mathbb{R}^{N \times d}$ is the mean matrix which is repeated column vector of the mean of all PDVs in the training set.

Similarly, $J_3(W)$ can be re-written as:

$$\begin{aligned}
J_3(W) &= \| (W^T X - 0.5) \times \mathbf{1}^{N \times 1} \|_2^2 \\
&= \| W^T X \times \mathbf{1}^{N \times 1} \|_2^2 \\
&\quad - N \times \text{tr}(\mathbf{1}^{1 \times K} \times W^T X \times \mathbf{1}^{N \times 1}) \\
&\quad + 0.5 \times \mathbf{1}^{1 \times N} \times \mathbf{1}^{N \times 1} \times 0.5 \\
&= \text{tr}(W^T X \mathbf{1}^{N \times 1} \mathbf{1}^{1 \times N} X^T W) \\
&\quad - N \times \text{tr}(\mathbf{1}^{1 \times K} W^T X \mathbf{1}^{N \times 1}) \\
&\quad + H
\end{aligned} \tag{2.9}$$

where $H = 0.5 \times \mathbf{1}^{1 \times N} \times \mathbf{1}^{N \times 1} \times 0.5$, which is a constant and is not influenced by W .

Combining (2.7)–(2.9), we have the following objective function for our Cbfd model:

$$\begin{aligned}
\min_W J(W) &= \text{tr}(W^T Q W) + \lambda_1 \| (B - 0.5) - W^T X \|_2^2 \\
&\quad - \lambda_2 \times N \times \text{tr}(\mathbf{1}^{1 \times K} W^T X \mathbf{1}^{N \times 1}) \\
\text{subject to: } &W^T W = I
\end{aligned} \tag{2.10}$$

where

Algorithm 2.1: CBFD

Input: Training set $X = [x_1, x_2, \dots, x_N]$, iteration number T , parameters λ_1 and λ_2 , binary code length K , and convergence parameter ϵ .

Output: Feature projection matrix W .

Step 1 (Initialization):

Initialize W to be the top K eigenvectors of XX^T corresponding to the K largest eigenvalues.

Step 2 (Optimization):

For $t = 1, 2, \dots, T$, repeat

2.1. Fix W and update B using (2.13).

2.2. Fix B and update W using (2.14).

2.3. If $|W^t - W^{t-1}| < \epsilon$ and $t > 2$, go to Step 3.

Step 3 (Output):

Output the matrix W .

$$Q \triangleq -\frac{1}{N} \times (XX^T - 2XM^T + MM^T) + \lambda_2 X \mathbf{1}^{N \times 1} \mathbf{1}^{1 \times N} X^T \quad (2.11)$$

and the columns of W are constrained to be orthogonal.

While (2.10) is not convex for W and B simultaneously, it is convex to one of them when the other is fixed. Following the work in [14], we iteratively optimize W and B by using the following two-stage method.

Update B with a fixed W : when W is fixed, (2.10) can be re-written as:

$$\min_B J(B) = \|(B - 0.5) - W^T X\|_F^2. \quad (2.12)$$

The solution to (2.12) is $(B - 0.5) = W^T X$ if there is no constraint to B . Since B is a binary matrix, this solution is relaxed as:

$$B = 0.5 \times (\text{sgn}(W^T X) + 1). \quad (2.13)$$

Update W with a fixed B : when B is fixed, (2.10) can be re-written as:

$$\begin{aligned} \min_W J(W) &= \text{tr}(W^T Q W) + \lambda_1 (\text{tr}(W^T X X^T W)) \\ &\quad - 2 \times \text{tr}((B - 0.5) \times X^T W) \\ &\quad - \lambda_2 \times N \times \text{tr}(\mathbf{1}^{1 \times K} W^T X \mathbf{1}^{N \times 1}) \\ \text{subject to} \quad &W^T W = I. \end{aligned} \quad (2.14)$$

We use the gradient descent method with the curvilinear search algorithm in [45] to solve W . Algorithm 2.1 summarizes the detailed procedure of proposed CBFD method.

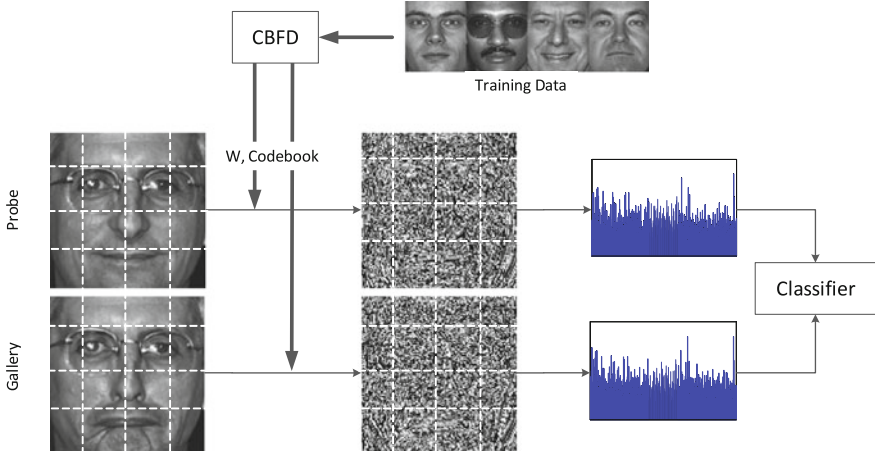


Fig. 2.5 The flow-chart of the CBFD-based face representation and recognition method. For each training face, we first divide it into several non-overlapped regions and learn the feature filter and dictionary for each region, individually. Then, we apply the learned filter and dictionary to extract histogram feature for each block and concatenate them into a longer feature vector for face representation. Finally, the nearest neighbor classifier is used to measure the sample similarity. © [2015] IEEE. Reprinted, with permission, from Ref. [35]

Having obtained the learned feature projection matrix W , we first project each PDV into a low-dimensional feature vector. Unlike many previous feature learning methods [26, 27] which usually perform feature pooling on the learned features directly, we apply an unsupervised clustering method to learn a codebook from the training set so that the learned codes are more data-adaptive. In our implementations, the conventional K -means method is applied to learn the codebook due to its simplicity. Then, each learned binary code feature is pooled as a bin and all PDVs within the same face image are represented as a histogram feature for face representation. Previous studies have shown different face regions have different structural information and it is desirable to learn position-specific features for face representation. Motivated by this finding, we divide each face image into many non-overlapped local regions and learn a CBFD feature descriptor for each local region. Finally, features extracted from different regions are combined to form the final representation for the whole face image. Figure 2.5 illustrates how to use the CBFD for face representation.

2.2.2 Prototype-Based Discriminative Feature Learning

Recently, there has been growing interest in unsupervised feature learning and deep learning in computer vision and machine learning, and a variety of feature learning methods have been proposed in the literature [10, 16, 18, 21, 23, 25, 28, 33, 38, 41,

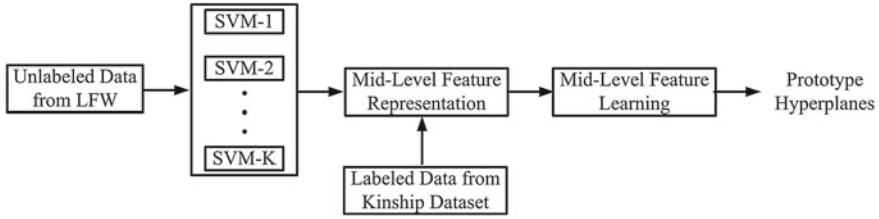


Fig. 2.6 Pipeline of our proposed kinship verification approach. First, we construct a set of face samples from the labelled face in the wild (LFW) dataset as the prototypes and represent each face image from the kinship dataset as a combination of these prototypes in the hyperplane space. Then, we use the labeled kinship information and learn mid-level features in the hyperplane space to extract more semantic information for feature representation. Finally, the learned hyperplane parameters are used to represent face images in both the training and testing sets as a discriminative mid-level feature for kinship verification. ©[2015] IEEE. Reprinted, with permission, from Ref. [51]

[49, 52] to learn feature representations from raw pixels. Generally, feature learning methods exploit some prior knowledge such as smoothness, sparsity, and temporal and spatial coherence [4]. Representative feature learning methods include sparse auto-encoder [38, 41], restricted Boltzmann machines [16], independent subspace analysis [8, 28], and convolutional neural networks [25]. These methods have been successfully applied in many computer vision tasks such as image classification [25], human action recognition [21], face recognition [18], and visual tracking [42].

Unlike previous feature learning methods which learn features directly from raw pixels, we propose learning mid-level discriminative features with low-level descriptor [51], where each entry in the mid-level feature vector is the corresponding decision value from one support vector machine (SVM) hyperplane. We formulate an optimization objective function on the learned features so that face samples with a kin relation are expected to have similar decision values from these hyperplanes. Hence, our method is complementary to the exiting feature learning methods. Figure 2.6 shows the pipeline of our proposed approach.

Low-level feature descriptors such as LBP [2] and scale-invariant feature transform (SIFT) [32] are usually ambiguous, which are not discriminative enough for kinship verification, especially when face images were captured in the wild because there are large variations on face images captured in such scenarios. Unlike most previous kinship verification work where low-level hand-crafted feature descriptors [12, 13, 15, 24, 34, 39, 47, 48, 50, 53, 54] such as local binary pattern (LBP) [2, 9] and Gabor features [31, 54] are employed for face representation, we expect to extract more semantic information from low-level features to better characterize the relation of face images for kinship verification.

To achieve this, we learn mid-level feature representations by using a large unsupervised dataset and a small supervised dataset because it is difficult to obtain a large number of labeled face images with kinship labels for discriminative feature learning. First, we construct a set of face samples with unlabeled kinship relation

from the labelled face in the wild (LFW) dataset [17] as the reference set. Then, each sample in the training set with a labeled kin relation is represented as a mid-level feature vector. Finally, we formulate an objective function by minimizing the intra-class samples (with a kinship relation) and maximizing the inter-class neighboring samples with the mid-level features. Unlike most existing prototype-based feature learning methods which learn the model with a strongly labeled training set, our method works on a large unsupervised generic set combined with a small labeled training set because unlabeled data can be easily collected. The following details the proposed approach.

Let $Z = [z_1, z_2, \dots, z_N] \in R^{d \times N}$ be a unlabeled reference image set, where N and d are the number of samples and feature dimension of each sample, respectively. Assume $S = (x_1, y_1), \dots, (x_i, y_i), \dots, (x_M, y_M)$ be the training set containing M pairs of face images with kinship relation (positive image pairs), where x_i and y_i are face images of the i th pair, and $x_i \in R^d$ and $y_i \in R^d$. Different from most existing feature learning methods which learn feature representations from raw pixels, we aim to learn a set of mid-level features which are obtained from a set of prototype hyperplanes. For each training sample in S , we apply the linear SVM to learn the weight vector w to represent it as follows:

$$w = \sum_{n=1}^N \alpha_n l_n z_j = \sum_{n=1}^N \beta_n z_j = Z\beta \quad (2.15)$$

where l_n is the label of the unlabeled data z_n , $\beta_n = \alpha_n l_n$ is the combination coefficient, α_n is the dual variable, and $\beta = [\beta_1, \beta_2, \dots, \beta_N] \in R^{N \times 1}$ is the coefficient vector. Specifically, if β_n is non-zero, it means that the sample z_k in the unlabeled reference set is selected as a support vector of the learned SVM model, and $l_n = 1$ if β_n is positive. Otherwise, $l_n = -1$. Motivated by the maximal margin principal of SVM, we only need to select a sparse set of support vectors to learn the SVM hyperplane. Hence, β should be a sparse vector, where $\|\beta\|_1 \leq \gamma$, and γ is a parameter to control the sparsity of β .

Having learned the SVM hyperplane, each training sample x_i and y_i can be represented as

$$f(x_i) = w^T x_i = x_i^T w = x_i^T Z\beta \quad (2.16)$$

$$f(y_i) = w^T y_i = y_i^T w = y_i^T Z\beta. \quad (2.17)$$

Assume we have learned K linear SVM hyperplanes, then the mid-level feature representations of x_i and y_i can be represented as

$$f(x_i) = [x_i^T Z\beta_1, x_i^T Z\beta_2, \dots, x_i^T Z\beta_K] = B^T Z^T x_i \quad (2.18)$$

$$f(y_i) = [y_i^T Z\beta_1, y_i^T Z\beta_2, \dots, y_i^T Z\beta_K] = B^T Z^T y_i \quad (2.19)$$

where $B = [\beta_1, \beta_2, \dots, \beta_K]$ is the coefficient matrix.

Now, we propose the following optimization criterion to learn the coefficient matrix B with the sparsity constraint:

$$\begin{aligned}
\max H(B) &= H_1(B) + H_2(B) - H_3(B) \\
&= \frac{1}{Mk} \sum_{i=1}^M \sum_{t_1=1}^k \|f(x_i) - f(y_{it_1})\|_2^2 \\
&\quad + \frac{1}{Mk} \sum_{i=1}^M \sum_{t_2=1}^k \|f(x_{it_2}) - f(y_i)\|_2^2 \\
&\quad - \frac{1}{M} \sum_{i=1}^M \|f(x_i) - f(y_i)\|_2^2 \\
s.t. \quad &\|\beta_k\|_1 \leq \gamma, k = 1, 2, \dots, K
\end{aligned} \tag{2.20}$$

where y_{it_1} represents the t_1 th k -nearest neighbor of y_i and x_{it_2} denotes the t_2 th k -nearest neighbor of x_i , respectively. The objectives of H_1 and H_2 in (2.20) is to make the mid-level feature representations of y_{it_1} and x_i , and x_{it_2} and y_i as far as possible if they are originally near to each other in the low-level feature space. The physical meaning of H_3 in (2.20) is to expect that x_i and y_i are close to each other in the mid-level feature space. We enforce the sparsity constraint on β_k such that only a sparse set of support vectors from the unlabeled reference dataset are selected to learn the hyperplane because we assume each sample can be sparsely reconstructed the reference set, which is inspired by the work in [23]. In our work, we apply the same parameter γ to reduce the number of parameters in our proposed model so that the complexity of the proposed approach is reduced.

Combining (2.18)–(2.20), we simplify $H_1(B)$ to the following form

$$\begin{aligned}
H_1(B) &= \frac{1}{Mk} \sum_{i=1}^M \sum_{t_1=1}^k \|B^T Z^T x_i - B^T Z^T y_{it_1}\|_2^2 \\
&= \frac{1}{Mk} \text{tr} \left(\sum_{i=1}^M \sum_{t_1=1}^k B^T Z^T (x_i - y_{it_1})(x_i - y_{it_1})^T Z B \right) \\
&= \text{tr}(B^T F_1 B)
\end{aligned} \tag{2.21}$$

where

$$F_1 = \frac{1}{Mk} \sum_{i=1}^M \sum_{t_1=1}^k Z^T (x_i - y_{it_1})(x_i - y_{it_1})^T Z. \tag{2.22}$$

Similarly, $H_2(B)$ and $H_3(B)$ can be simplified as

$$H_2(B) = \text{tr}(B^T F_2 B), H_3(B) = \text{tr}(B^T F_3 B) \tag{2.23}$$

where

$$F_2 = \frac{1}{Mk} \sum_{i=1}^M \sum_{2=1}^k Z^T (x_{it_2} - y_i)(x_{it_2} - y_i)^T Z \quad (2.24)$$

$$F_3 = \sum_{i=1}^M Z^T (x_i - y_i)(x_i - y_i)^T Z. \quad (2.25)$$

Based on (2.21)–(2.25), the proposed PDFL model can be formulated as follows:

$$\begin{aligned} \max H(B) &= \text{tr}[B^T(F_1 + F_2 - F_3)B] \\ \text{s.t.} \quad &B^T B = I, \\ &\| \beta_k \|_1 \leq \gamma, k = 1, 2, \dots, K. \end{aligned} \quad (2.26)$$

where $B^T B = I$ is a constraint to control the scale of B so that the optimization problem in (2.26) is well-posed with respect to B .

Since there is a sparsity constraint for each β_k , we cannot obtain B by solving a standard eigenvalue equation. To address this, we propose an alternating optimization method in [37] by reformulating the optimization problem as a regression problem. Let $F = F_1 + F_2 - F_3$. We perform singular value decomposition (SVD) on $F = G^T G$, where $G \in R^{N \times N}$. Following [37], we reformulate a regression problem by using an intermediate matrix $A = [a_1, a_2, \dots, a_K] \in R^{N \times K}$ (Please see *Theorem 1* in [37] for more details on this reformulation)

$$\begin{aligned} \min H(A, B) &= \sum_{k=1}^K \|Ga_k - G\beta_k\|^2 + \lambda \sum_{k=1}^K \beta_k^T \beta_k \\ \text{s.t.} \quad &A^T A = I_{K \times K}, \\ &\| \beta_k \|_1 \leq \gamma, k = 1, 2, \dots, K. \end{aligned} \quad (2.27)$$

We employ an alternating optimization method [37] to optimize A and B iteratively.

Fix A , optimize B : For a given A , we solve the following problem to obtain B :

$$\begin{aligned} \min H(B) &= \sum_{k=1}^K \|Ga_k - G\beta_k\|^2 + \lambda \sum_{k=1}^K \beta_k^T \beta_k \\ \text{s.t.} \quad &\| \beta_k \|_1 \leq \gamma, k = 1, 2, \dots, K. \end{aligned} \quad (2.28)$$

Considering that β_k are independent in (2.28), we individually obtain β_k by solving the following optimization problem:

Algorithm 2.2: PDFL

Input: Reference set: $Z = [z_1, z_2, \dots, z_N] \in \mathbb{R}^{d \times N}$, training set:

$S = \{(x_i, y_i) | i = 1, 2, \dots, M\}, x_i \in \mathbb{R}^d \text{ and } y_i \in \mathbb{R}^d.$

Output: Coefficient matrix $B = [\beta_1, \beta_2, \dots, \beta_K].$

Step 1 (Initialization):

Initialize $A \in \mathbb{R}^{N \times K}$ and $B \in \mathbb{R}^{N \times K}$ where each entry of them is set as 1.

Step 2 (Local optimization):

For $t = 1, 2, \dots, T$, repeat

2.1. Compute B according to (2.28).

2.2. Compute A according to (2.30)–(2.31).

2.3. If $t > 2$ and $\|B_t - B_{t-1}\|_F \leq \epsilon$ (ϵ is set as 0.001 in our experiments), go to Step 3.

Step 3 (Output coefficient matrix):

Output coefficient matrix $B = B_t.$

$$\begin{aligned} \min H(\beta_k) &= \|h_k - G\beta_k\|^2 + \lambda\beta_k^T \beta_k \\ &= \|g_k - P\beta_k\|^2 \\ \text{s.t.} \quad &\|\beta_k\|_1 \leq \gamma. \end{aligned} \quad (2.29)$$

where $h_k = Ga_k$, $g_k = [h_k^T, \mathbf{0}_N^T]^T$, $P = [G^T, \sqrt{\lambda}\mathbf{1}_N^T]^T$, and β_k can be easily obtained by using the conventional least angle regression solver [11].

Fix B , optimize A : For a given B , we solve the following problem to obtain A :

$$\begin{aligned} \min H(A) &= \|GA - GB\|^2 \\ \text{s.t.} \quad &A^T A = I_{K \times K}. \end{aligned} \quad (2.30)$$

And A can be obtained by using SVD, namely

$$G^T GB = USV^T, \quad \text{and} \quad A = \tilde{U}V^T \quad (2.31)$$

where $\tilde{U} = [u_1, u_2, \dots, u_K]$ be the top K leading eigenvectors of $U = [u_1, u_2, \dots, u_N]$.

We repeat the above two steps until the algorithm meets a certain convergence condition. The proposed PDFL algorithm is summarized in Algorithm 2.2.

2.2.3 Multiview Prototype-Based Discriminative Feature Learning

Different feature descriptors usually capture complementary information to describe face images from different aspects [6] and it is helpful for us to improve the kinship verification performance with multiple feature descriptors. A nature solution for

feature learning with multiview data is concatenating multiple features first and then applying existing feature learning methods on the concatenated features. However, it is not physically meaningful to directly combine different features because they usually show different statistical characteristics and such a concatenation cannot well exploit the feature diversity. In this work, we introduce a multiview PDFL (MPDFL) method to learn a common coefficient matrix with multiple low-level descriptors for mid-level feature representation for kinship verification.

Given the training set S , we first extract L feature descriptors denoted as S^1, \dots, S^L , where $S^l = (x_1^l, y_1^l), \dots, (x_i^l, y_i^l), \dots, (x_M^l, y_M^l)$ is the l th feature representation, $1 \leq l \leq L$, $x_i^l \in R^d$, and $y_i^l \in R^d$ are the i th parent and child faces in the l th feature space, $l = 1, 2, \dots, L$. MPDFL aims to learn a shared coefficient matrix B with the sparsity constraint so that the intra-class variations are minimized and the inter-class variations are maximized in the mid-level feature spaces.

To exploit complemental information from facial images, we introduce a nonnegative weighted vector $\alpha = [\alpha_1, \alpha_2, \dots, \alpha_L]$ to weight each feature space of PDFL. Generally, the larger α_l , the more contribution it is to learn the sparse coefficient matrix. MPDFL is formulated as the following objective function by using an intermediate matrix $A = [a_1, a_2, \dots, a_K] \in R^{N \times K}$

$$\begin{aligned}
 \max_{B, \alpha} \quad & \sum_{l=1}^L \alpha_l \text{tr}[B^T (F_1^l + F_2^l - F_3^l) B] \\
 \text{s.t.} \quad & B^T B = I, \\
 & \|\beta_k\|_1 \leq \gamma, k = 1, 2, \dots, K. \\
 & \sum_{l=1}^L \alpha_l = 1, \alpha_l \geq 0
 \end{aligned} \tag{2.32}$$

where F_1^l, F_2^l and F_3^l are the expressions of F_1, F_2 and F_3 in the l th feature space, and $1 \leq l \leq L$.

Since the solution to (2.32) is $\alpha_l = 1$, which corresponds to the maximal $\text{tr}[B^T (F_1^l + F_2^l - F_3^l) B]$ over different feature descriptors, and $\alpha_p = 0$ otherwise. To address this, we revisit α_l as α_l^r ($r > 1$) and re-define the following optimization function as

$$\begin{aligned}
 \max_{B, \alpha} \quad & \sum_{l=1}^L \alpha_l^r \text{tr}[B^T (F_1^l + F_2^l - F_3^l) B] \\
 \text{s.t.} \quad & B^T B = I, \|\beta_k\|_1 \leq \gamma, k = 1, 2, \dots, K. \\
 & \sum_{l=1}^L \alpha_l = 1, \alpha_l \geq 0.
 \end{aligned} \tag{2.33}$$

Similar to PDFL, we also reformulate MPDFL as the following regression problem:

$$\begin{aligned}
\min_{A, B, \alpha} \quad & \sum_{l=1}^L \sum_{k=1}^K \alpha_l^r \|G_l a_k - G_l \beta_k\|^2 + \lambda \sum_{k=1}^K \beta_k^T \beta_k \\
s.t. \quad & A^T A = I_{K \times K}, \\
& \|\beta_k\|_1 \leq \gamma, k = 1, 2, \dots, K.
\end{aligned} \tag{2.34}$$

where $F_l = G_l^T G_l$, and $F_l = F_1^l + F_2^l - F_3^l$.

Since (2.34) is non-convex with respect to A , B , and α , we solve it iteratively similar to PDFL by using an alternating optimization method.

Fix A and B , optimize α : For the given A and B , we construct a Lagrange function

$$\begin{aligned}
L(\alpha, \eta) = \sum_{l=1}^L \alpha_l^r \text{tr}[B^T (F_1^l + F_2^l - F_3^l) B] \\
- \zeta \left(\sum_{l=1}^L \alpha_l - 1 \right).
\end{aligned} \tag{2.35}$$

Let $\frac{\partial L(\alpha, \eta)}{\partial \alpha_l} = 0$ and $\frac{\partial L(\alpha, \eta)}{\partial \zeta} = 0$, we have

$$r \alpha_l^{r-1} \text{tr}[B^T (F_1^l + F_2^l - F_3^l) B] - \zeta = 0 \tag{2.36}$$

$$\sum_{l=1}^L \alpha_l - 1 = 0. \tag{2.37}$$

Combining (2.36) and (2.37), we can obtain α_l as follows:

$$\alpha_l = \frac{(1/\text{tr}[B^T (F_1^l + F_2^l - F_3^l) B])^{1/(r-1)}}{\sum_{l=1}^L (1/\text{tr}[B^T (F_1^l + F_2^l - F_3^l) B])^{1/(r-1)}} \tag{2.38}$$

Fix A and α , optimize B : For the given A and α , we solve the following problem to obtain B :

$$\begin{aligned}
\min H(B) = \sum_{l=1}^L \sum_{k=1}^K \alpha_l^r \|G_l a_k - G_l \beta_k\|^2 + \lambda \sum_{k=1}^K \beta_k^T \beta_k \\
s.t. \quad \|\beta_k\|_1 \leq \gamma, k = 1, 2, \dots, K.
\end{aligned} \tag{2.39}$$

Similar to PDFL, we individually obtain β_k by solving the following optimization problem:

$$\begin{aligned}
\min H(\beta_k) &= \sum_{l=1}^L \alpha_l^r \|G_l a_k - G_l \beta_k\|^2 + \lambda \beta_k^T \beta_k \\
&= \|h_k - G \beta_k\|^2 + \lambda \beta_k^T \beta_k \\
&= \|g_k - P \beta_k\|^2 \\
s.t. \quad &\|\beta_k\|_1 \leq \gamma.
\end{aligned} \tag{2.40}$$

where $h_k = \sum_{l=1}^L \alpha_l^r G_l a_k$, $g_k = [h_k^T, \mathbf{0}_N^T]^T$, $P = [\sum_{l=1}^L \alpha_l^r G_l, \sqrt{\lambda} \mathbf{1}_N^T]^T$, and β_k can be obtained by using the conventional least angle regression solver [11].

Fix B and α , optimize A : For the given B and α , we solve the following problem to obtain A :

$$\begin{aligned}
\min H(A) &= \sum_{l=1}^L \alpha_l^r \|G_l A - G_l B\|^2 \\
s.t. \quad &A^T A = I_{K \times K}.
\end{aligned} \tag{2.41}$$

And A can be obtained by using SVD, namely

$$\left(\sum_{l=1}^L \alpha_l^r G_l^T G_l \right) B = U S V^T, \quad \text{and} \quad A = \tilde{U} V^T \tag{2.42}$$

where $\tilde{U} = [u_1, u_2, \dots, u_K]$ be the top K leading eigenvectors of $U = [u_1, u_2, \dots, u_N]$.

We repeat the above three steps until the algorithm converges to a local optimum. Algorithm 2.3 summarizes the proposed MPDFL algorithm.

2.3 Evaluation

In this section, we conduct kinship verification experiments on four benchmark kinship datasets to show the efficacy of feature learning methods for kinship verification applications. The following details the results.

Algorithm 2.3: MPDFL

Input: $Z^l = [z_1^l, z_2^l, \dots, z_N^l]$ is the l th feature representation of the reference set,
 $S^l = \{(x_i^l, y_i^l) | i = 1, 2, \dots, M\}$ is the l th feature representation of the training set.
Output: Coefficient matrix $B = [\beta_1, \beta_2, \dots, \beta_K]$.
Step 1 (Initialization):
 1.1. Initialize $A \in R^{N \times K}$ and $B \in R^{N \times K}$ where each entry is set as 1.
 1.2. Initialize $\alpha = [1/K, 1/K, \dots, 1/K]$.
Step 2 (Local optimization):
 For $t = 1, 2, \dots, T$, repeat
 2.1. Compute α according to (2.38).
 2.2. Compute B according to (2.39).
 2.3. Compute A according to (2.41)–(2.42).
 2.4. If $t > 2$ and $\|B_t - B_{t-1}\|_F \leq \epsilon$ (ϵ is set as 0.001 in our experiments), go to Step 3.
Step 3 (Output coefficient matrix):
 Output coefficient matrix $B = B_t$.

2.3.1 Data Sets

Four publicly available face kinship datasets, namely KinFaceW-I [34],¹ KinFaceW-II [34],² Cornell KinFace [12]³, and UB KinFace [46],⁴ were used for our evaluation. Facial images from all these datasets were collected from the internet online search. Figure 2.7 shows some sample positive pairs from these four datasets.

There are four kin relations in both the KinFaceW-I and KinFaceW-II datasets: Father-Son (F-S), Father-Daughter (F-D), Mother-Son (M-S), and Mother-Daughter (M-D). For KinFaceW-I, these four relations contain 134, 156, 127, and 116 pairs, respectively. For KinFaceW-II, each relation contains 250 pairs.

There are 150 pairs of parents and children images in the Cornell KinFace dataset, where 40, 22, 13, and 25% of them are with the F-S, F-D, M-S, and M-D relation, respectively.

There are 600 face images of 400 persons in the UB KinFace dataset. These images are categorized into 200 groups, and each group has three images, which correspond to facial images of the child, young parent and old parent, respectively. For each group, we constructed two kinship face pairs: child and young parent, and child and old parent. Therefore, we constructed two subsets from the UB KinFace dataset: Set 1 (200 child and 200 young parent face images) and Set 2 (200 child and 200 old parent face images). Since there are large imbalances of the different kinship relations of the UB Kinface database (nearly 80% of them are the F-S relation), we have not separated different kinship relations on this dataset.

¹<http://www.kinfacew.com>.

²<http://www.kinfacew.com>.

³<http://chenlab.ece.cornell.edu/projects/KinshipVerification>.

⁴<http://www.ece.neu.edu/~yunfu/research/Kinface/Kinface.htm>.



Fig. 2.7 Some sample positive pairs (with kinship relation) from different face kinship databases. Face images from the first to fourth row are from the KinFaceW-I [34], KinFaceW-II [34], Cornell KinFace [12], and UB KinFace [48] databases, respectively. ©[2015] IEEE. Reprinted, with permission, from Ref. [51]

2.3.2 Experimental Settings

We randomly selected 4000 face images from the LFW dataset to construct the reference set, which was used for all the four kinship face datasets to learn the mid-level feature representations. We aligned each face image in all datasets into 64×64 pixels using the provided eyes positions and converted it into gray-scale image. We applied three different feature descriptors including LBP [2], Spatial Pyramid Learning (SPLE) [53] and SIFT [32] to extract different and complementary information from each face image. The reason we selected these three features is

that they have shown reasonably good performance in recent kinship verification studies [34, 53]. We followed the same parameter settings for these features in [34] so that a fair comparison can be obtained.

For the LBP feature, 256 bins were used to describe each face image because this setting yields better performance. For the SPLE method, we first constructed a sequence of grids at three different resolution (0, 1, and 2), such that we have 21 cells in total. Then, each local feature in each cell was quantized into 200 bins and each face image was represented by a 4200-dimensional long feature vector. For the SIFT feature, we densely sampled and computed one 128-dimensional feature over each 16×16 patch, where the overlap between two neighboring patches is 8 pixels. Then, each SIFT descriptor was concatenated into a long feature vector. For these features, we applied principal component analysis (PCA) to reduce each feature into 100 dimensions to remove some noise components.

The fivefold cross-validation strategy was used in our experiments. We tuned the parameters of our PDFL and MPDFL methods on the KinFaceW-II dataset because this dataset is the largest one such that it is more effective to tune parameters on this dataset than others. We divided the KinFaceW-II dataset into fivefolds with an equal size, and applied fourfolds to learn the coefficient matrix and the remaining one for testing. For the training samples, we used three of them to learn our models and the other one fold to tune the parameters of our methods. In our implementations, the parameters r , λ , γ , and K were empirically set as 5, 1, 0.5, and 500, respectively. Finally, the support vector machine (SVM) classifier with the RBF kernel is applied for verification.

2.3.3 Results and Analysis

2.3.3.1 Comparison with Existing Low-Level Feature Descriptors

We compared our PDFL and MPDFL methods with the existing low-level feature descriptors. The difference between our methods and the existing feature representations is that we use the mid-level features rather than the original low-level features for verification. Tables 2.1, 2.2, 2.2, and 2.4 tabulate the verification rate of different feature descriptors on the KinFaceW-I, KinFaceW-II, Cornell KinFace, and UB KinFace kinship databases, respectively. From these tables, we see that our proposed PDFL and MPDFL outperform the best existing methods with the lowest gain in mean verification accuracy of 2.6 and 7.1%, 6.2 and 6.8%, 1.0 and 2.4%, 0.9 and 4.6% on the KinFaceW-I, KinFaceW-II, Cornell KinFace, and UB KinFace kinship datasets, respectively.

Table 2.1 Verification rate (%) on the KinFaceW-I dataset. ©[2015] IEEE. Reprinted, with permission, from Ref. [51]

Feature	F-S	F-D	M-S	M-D	Mean
LBP	62.7	60.2	54.4	61.4	59.7
LBP+PDFL	65.7	65.5	60.4	67.4	64.8
LE	66.1	59.1	58.9	68.0	63.0
LE+PDFL	68.2	63.5	61.3	69.5	65.6
SIFT	65.5	59.0	55.5	55.4	58.9
SIFT+PDFL	67.5	62.0	58.8	57.9	61.6
MPDFL (All)	73.5	67.5	66.1	73.1	70.1

Table 2.2 Verification rate (%) on the KinFaceW-II dataset. ©[2015] IEEE. Reprinted, with permission, from Ref. [51]

Feature	F-S	F-D	M-S	M-D	Mean
LBP	64.0	63.5	62.8	63.0	63.3
LBP+PDFL	69.5	69.8	70.6	69.5	69.9
LE	69.8	66.1	72.8	72.0	70.2
LE+PDFL	77.0	74.3	77.0	77.2	76.4
SIFT	60.0	56.9	54.8	55.4	56.8
SIFT+PDFL	69.0	62.4	62.4	62.0	64.0
MPDFL (All)	77.3	74.7	77.8	78.0	77.0

Table 2.3 Verification rate (%) on the Cornell KinFace dataset. ©[2015] IEEE. Reprinted, with permission, from Ref. [51]

Feature	F-S	F-D	M-S	M-D	Mean
LBP	67.1	63.8	75.0	60.0	66.5
LBP+PDFL	67.9	64.2	77.0	60.8	67.5
LE	72.7	66.8	75.4	63.2	69.5
LE+PDFL	73.7	67.8	76.4	64.2	70.5
SIFT	64.5	67.3	68.4	61.8	65.5
SIFT+PDFL	66.5	69.3	69.4	62.8	67.0
MPDFL (All)	74.8	69.1	77.5	66.1	71.9

2.3.3.2 Comparison with State-of-the-Art Kinship Verification Methods

Table 2.5 compares our PDFL and MPDFL methods with the state-of-the-art kinship verification methods presented in the past several years. To further investigate the performance differences between our feature learning approach and the other compared methods, we evaluate the verification results by using the null hypothesis statistical test based on Bernoulli model [5] to check whether the differences between

Table 2.4 Verification rate (%) on the UB KinFace dataset. © [2015] IEEE. Reprinted, with permission, from Ref. [51]

Feature	Set 1	Set 2	Mean
LBP	63.4	61.2	62.3
LBP+PDFL	64.0	62.2	63.1
LE	61.9	61.3	61.6
LE+PDFL	62.8	63.5	63.2
SIFT	62.5	62.8	62.7
SIFT+PDFL	63.8	63.4	63.6
MPDFL (All)	67.5	67.0	67.3

Table 2.5 Verification accuracy (%) on different kinship datasets. © [2015] IEEE. Reprinted, with permission, from Ref. [51]

Method	KinFaceW-I	KinFaceW-II	Cornell KinFace	UB KinFace
Method in [12]	N.A.	N.A.	70.7 (0, 1)	N.A.
Method in [48]	N.A.	N.A.	N.A.	56.5 (1, 1)
NRML [34]	64.3 (0, 1)	75.7 (0, 1)	69.5 (0, 1)	65.6 (0, 1)
MNRML [34]	69.9 (0, 0)	76.5 (0, 1)	71.6 (0, 0)	67.1 (0, 0)
PDFL (best)	64.8	70.2	70.5	63.6
MPDFL	70.1	77.0	71.9	67.3

the results of our approach and those of other methods are statistically significant. The results of the p -tests of PDFL and MPDFL are given in the brackets right after the verification rate of each method in each table, where the number “1” represents significant difference and “0” represents otherwise. There are two numbers in each bracket, where the first represents the significant difference of PDFL and the second represents that of MPDFL over previous methods. We see that PDFL achieves comparable accuracy with the existing state-of-the-art methods, and MPDFL obtains better performance than the existing kinship verification methods when the same kinship dataset was used for evaluation. Moreover, the improvement of MPDFL is significant for most comparisons.

Since our feature learning approach and previous metric learning methods exploit discriminative information in the feature extraction and similarity measure stages, respectively, we also conduct kinship verification experiments when both of them are used for our verification task. Table 2.6 tabulates the verification performance when such discriminative information is exploited in different manners. We see that the performance of our feature learning approach can be further improved when the discriminative metric learning methods are applied.

Table 2.6 Verification accuracy (%) of extracting discriminative information in different stages on different kinship datasets. © [2015] IEEE. Reprinted, with permission, from Ref. [51]

Method	KinFaceW-I	KinFaceW-II	Cornell KinFace	UB KinFace
NRML	64.3	75.7	69.5	65.6
PDFL (best)	64.8	70.2	70.5	63.6
PDFL (best) + NRML	67.4	77.5	73.4	67.8
MNRML	69.9	76.5	71.6	67.1
MPDFL	70.1	77.0	71.9	67.3
MPDFL + MNRML	72.3	78.5	75.4	69.8

Table 2.7 Verification accuracy (%) of PDFL with the best single feature and different classifiers. © [2015] IEEE. Reprinted, with permission, from Ref. [51]

Method	KinFaceW-I	KinFaceW-II	Cornell KinFace	UB KinFace
NN	63.9	69.3	70.1	63.1
SVM	64.8	70.2	70.5	63.6

Table 2.8 Verification accuracy (%) of MPDFL with different classifiers. © [2015] IEEE. Reprinted, with permission, from Ref. [51]

Method	KinFaceW-I	KinFaceW-II	Cornell KinFace	UB KinFace
NN	69.6	76.0	70.4	66.2
SVM	70.1	77.0	71.9	67.3

2.3.3.3 Comparison with Different Classifiers

We investigated the performance of our PDFL (best single feature) and MPDFL with different classifiers. In our experiments, we evaluated two classifiers: (1) SVM and (2) NN. For the NN classifier, the cosine similarity of two face images is used. Tables 2.7 and 2.8 tabulate the mean verification rate of our PDFL and MPDFL when different classifiers were used for verification. We see that our feature learning methods are not sensitive to the selection of the classifier.

2.3.3.4 Parameter Analysis

We took the KinFaceW-I dataset as an example to investigate the verification performance and training cost of our MPDFL versus varying values of K and γ . Figures 2.8 and 2.9 show the mean verification accuracy and the training time of MPDFL versus different K and γ . We see that K and γ were set to 500 and 0.5 are good tradeoffs between the efficiency and effectiveness of our proposed method.

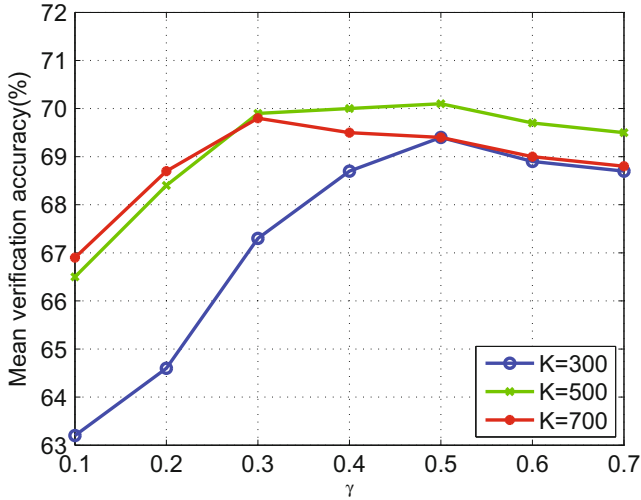


Fig. 2.8 Mean verification rate of our PDFL and MPDFL versus different values of K and γ . © [2015] IEEE. Reprinted, with permission, from Ref. [51]

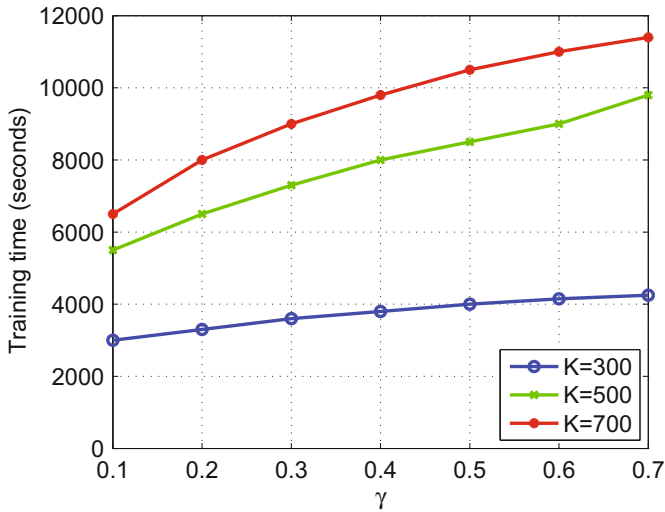


Fig. 2.9 Training time of our MPDFL versus different values of K and γ . © [2015] IEEE. Reprinted, with permission, from Ref. [51]

Figure 2.10 shows the mean verification rates of PDFL and MPDFL versus different number of iteration on the KinFaceW-I dataset. We see that PDFL and MPDFL achieve stable verification performance in several iterations.

We investigated the effect of the parameter r in MPDFL. Figure 2.11 shows the verification rate of MPDFL versus different number of r on different kinship datasets.

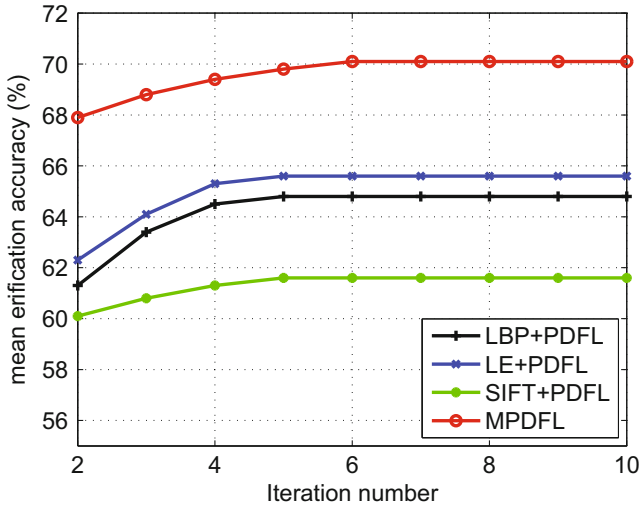


Fig. 2.10 Mean verification rate of our PDFL and MPDFL versus different number of iterations, on the KinFaceW-I dataset. © [2015] IEEE. Reprinted, with permission, from Ref. [51]

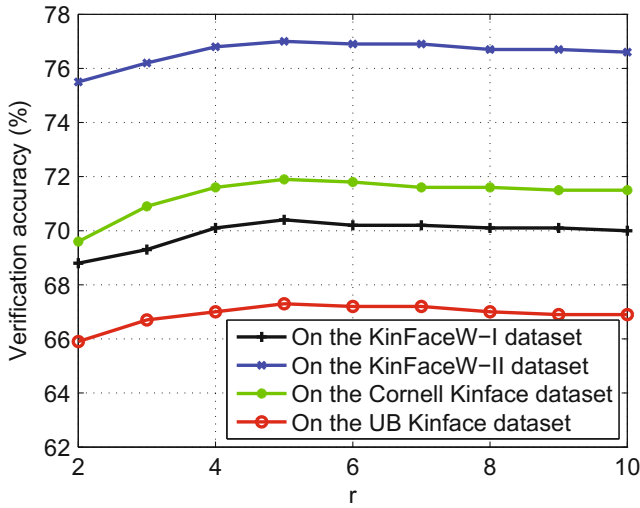


Fig. 2.11 Mean verification rate of our MPDFL versus different values of r on different kinship face datasets. © [2015] IEEE. Reprinted, with permission, from Ref. [51]

We observe that our MPDFL method is in general robust to the parameter r , and the best verification performance can be obtained when r was set to 5.

2.3.3.5 Computational Time

We compare the computational time of the proposed PDFL and MPDFL methods with state-of-the-art metric learning-based kinship verification methods including NRML and MNRML. Our hardware consists of a 2.4-GHz CPU and a 6GB RAM. Table 2.9 shows the time spent on the training and the testing stages of different methods, where the Matlab software, the KinFaceW-I database, and the SVM classifier were used. We see that the computational time of our feature learning methods are comparable to those of NRML and MNRML.

2.3.3.6 Comparison with Human Observers in Kinship Verification

Human ability in kinship verification was evaluated in [34]. We also compared our method with humans on the KinFaceW-I and KinFaceW-II datasets. For a fair comparison between human ability and our proposed approach, the training samples as well as their kin labels used in our approach were selected and presented to 10 human observers (5 males and 5 females) who are 20–30 years old [34] to provide the prior knowledge to learn the kin relation from human face images. Then, the testing samples used in our experiments were presented to these to evaluate the performance of human ability in kinship verification. There are two evaluations for humans: HumanA and HumanB in [34], where the only face region and the whole original face were presented to human observers, respectively. Table 2.10 shows the

Table 2.9 CPU time (in second) used by different kinship verification methods on the KinFaceW-I database. © [2015] IEEE. Reprinted, with permission, from Ref. [51]

Method	Training	Testing
NRML	18.55	0.95
MNRML	22.35	0.95
PDFL	18.25	0.95
MPDFL	21.45	0.95

Table 2.10 Performance comparison (%) between our methods and humans on kinship verification on the KinFaceW-I and KinFaceW-II datasets, respectively. © [2015] IEEE. Reprinted, with permission, from Ref. [51]

Method	KinFaceW-I				KinFaceW-II			
	F-S	F-D	M-S	M-D	F-S	F-D	M-S	M-D
HumanA	62.0	60.0	68.0	72.0	63.0	63.0	71.0	75.0
HumanB	68.0	66.5	74.0	75.0	72.0	72.5	77.0	80.0
PDFL (LE)	68.2	63.5	61.3	69.5	77.0	74.3	77.0	77.2
MPDFL	73.5	67.5	66.1	73.1	77.3	74.7	77.8	78.0

performance of these observers and our approach. We see that our proposed methods achieve even better kinship verification performance than human observers on most subsets of these two kinship datasets.

We make the following four observations from the above experimental results listed in Tables 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9 and 2.10, and Figs. 2.8, 2.9, 2.10, and 2.11:

1. Learning discriminative mid-level feature achieves better verification performance than the original low-level feature. This is because the learned mid-level feature exploits discriminative information while the original low-level feature cannot.
2. MPDFL achieves better performance than PDFL, which indicates that combining multiple local-level descriptors to learn mid-level features is better than using a single one because multiple features can provide complementary information for feature learning.
3. PDFL achieves comparable performance and MPDFL achieves better performance than existing kinship verification methods. The reason is that most existing kinship verification methods used low-level hand-crafted features for face representation, which is not discriminative enough to characterize the kin relation of face images.
4. Both PDFL and MPDFL achieve better kinship verification performance than human observers, which further shows the potentials of our computational face-based kinship verification models for practical applications.

References

1. Ahonen, T., Hadid, A., Pietikäinen, M.: Face recognition with local binary patterns. In: European Conference on Computer Vision, pp. 469–481 (2004)
2. Ahonen, T., Hadid, A., et al.: Face description with local binary patterns: application to face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **28**(12), 2037–2041 (2006)
3. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H.: Greedy layer-wise training of deep networks. In: NIPS, pp. 153–160 (2007)
4. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(8), 1798–1828 (2013)
5. Beveridge, J.R., She, K., Draper, B., Givens, G.H.: Parametric and nonparametric methods for the statistical evaluation of human id algorithms. In: International Workshop on the Empirical Evaluation of Computer Vision Systems (2001)
6. Bickel, S., Scheffer, T.: Multi-view clustering. In: IEEE International Conference on Data Mining, pp. 19–26 (2004)
7. Cao, Z., Yin, Q., Tang, X., Sun, J.: Face recognition with learning-based descriptor. In: CVPR, pp. 2707–2714 (2010)
8. Deng, W., Liu, Y., Hu, J., Guo, J.: The small sample size problem of ICA: a comparative study and analysis. *Pattern Recognit.* **45**(12), 4438–4450 (2012)
9. Deng, W., Hu, J., Lu, J., Guo, J.: Transform-invariant pca: a unified approach to fully automatic face alignment, representation, and recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **36**(6), 1275–1284 (2014)

10. Dornaika, F., Bosaghzadeh, A.: Exponential local discriminant embedding and its application to face recognition. *IEEE Trans. Cybern.* **43**(3), 921–934 (2013)
11. Efron, B., Hastie, T., Johnstone, I., Tibshirani, R.: Least angle regression. *Ann. Stat.* **32**(2), 407–499 (2004)
12. Fang, R., Tang, K., Snaveley, N., Chen, T.: Towards computational models of kinship verification. In: *IEEE International Conference on Image Processing*, pp. 1577–1580 (2010)
13. Fang, R., Gallagher, A.C., Chen, T., Loui, A.: Kinship classification by modeling facial feature heredity, pp. 2983–2987 (2013)
14. Gong, Y., Lazebnik, S.: Iterative quantization: a procrustean approach to learning binary codes. In: *CVPR*, pp. 817–824 (2011)
15. Guo, G., Wang, X.: Kinship measurement on salient facial features. *IEEE Trans. Instrum. Meas.* **61**(8), 2322–2325 (2012)
16. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. *Neural Comput.* **18**(7), 1527–1554 (2006)
17. Huang, G.B., Ramesh, M., Berg, T., Learned-Miller, E.: Labeled faces in the wild: a database for studying face recognition in unconstrained environments. Technical Reports 07-49, University of Massachusetts, Amherst (2007)
18. Huang, G.B., Lee, H., Learned-Miller, E.: Learning hierarchical representations for face verification with convolutional deep belief networks. In: *IEEE International Conference Computer Vision and Pattern Recognition*, pp. 2518–2525 (2012)
19. Hussain, S.U., Napoléon, T., Jurie, F., et al.: Face recognition using local quantized patterns. In: *BMVC*, pp. 1–12 (2012)
20. Hyvärinen, A., Hurri, J., Hoyer, P.O.: Independent component analysis. *Natural Image Statistics*, pp. 151–175 (2009)
21. Ji, S., Xu, W., Yang, M., Yu, K.: 3D convolutional neural networks for human action recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(1), 221–231 (2013)
22. Jin, H., Liu, Q., Lu, H., Tong, X.: Face detection using improved LBP under bayesian framework. In: *International Conference on Image and Graphics*, pp. 306–309 (2004)
23. Kan, M., Xu, D., Shan, S., Li, W., Chen, X.: Learning prototype hyperplanes for face verification in the wild. *IEEE Trans. Image Process.* **22**(8), 3310–3316 (2013)
24. Kohli, N., Singh, R., Vatsa, M.: Self-similarity representation of weber faces for kinship classification. In: *IEEE International Conference on Biometrics: Theory, Applications, and Systems*, pp. 245–250 (2012)
25. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. In: *Advances in Neural Information Processing Systems*, pp. 1106–1114 (2012)
26. Le, Q.V., Karpenko, A., Ngiam, J., Ng, A.Y.: Ica with reconstruction cost for efficient over-complete feature learning. In: *NIPS*, pp. 1017–1025 (2011)
27. Le, Q.V., Zou, W.Y., Yeung, S.Y., Ng, A.Y.: Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: *CVPR*, pp. 3361–3368 (2011)
28. Le, Q.V., Zou, W.Y., Yeung, S.Y., Ng, A.Y.: Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: *IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 3361–3368 (2011)
29. Lei, Z., Pietikainen, M., Li, S.Z.: Learning discriminant face descriptor. *PAMI* **36**(2), 289–302 (2014)
30. Li, X., Shen, C., Dick, A.R., van den Hengel, A.: Learning compact binary codes for visual tracking. In: *CVPR*, pp. 2419–2426 (2013)
31. Liu, C., Wechsler, H.: Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition. *IEEE Trans. Image Process.* **11**(4), 467–476 (2002)
32. Lowe, D.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **60**(2), 91–110 (2004)
33. Lu, J., Tan, Y.P.: Regularized locality preserving projections and its extensions for face recognition. *IEEE Trans. Syst. Man Cybern. Part B: Cybern.* **40**(3), 958–963 (2010)

34. Lu, J., Zhou, X., Tan, Y.P., Shang, Y., Zhou, J.: Neighborhood repulsed metric learning for kinship verification. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(2), 331–345 (2014)
35. Lu, J., Liong, V.E., Zhou, X., Zhou, J.: Learning compact binary face descriptor for face recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **37**(10), 2041–2056 (2015)
36. Norouzi, M., Fleet, D., Salakhutdinov, R.: Hamming distance metric learning. In: *NIPS*, pp. 1070–1078 (2012)
37. Qiao, Z., Zhou, L., Huang, J.Z.: Sparse linear discriminant analysis with applications to high dimensional low sample size data. *Int. J. Appl. Math.* **39**(1), 48–60 (2009)
38. Rifai, S., Vincent, P., Muller, X., Glorot, X., Bengio, Y.: Contractive auto-encoders: Explicit invariance during feature extraction. In: *International Conference on Machine Learning*, pp. 833–840 (2011)
39. Somanath, G., Kambhamettu, C.: Can faces verify blood-relations? In: *IEEE International Conference on Biometrics: Theory, Applications and Systems*, pp. 105–112 (2012)
40. Trzcinski, T., Lepetit, V.: Efficient discriminative projections for compact binary descriptors. In: *ECCV*, pp. 228–242 (2012)
41. Vincent, P., Larochelle, H., Bengio, Y., Manzagol, P.A.: Extracting and composing robust features with denoising autoencoders. In: *International Conference on Machine Learning*, pp. 1096–1103 (2008)
42. Wang, N., Yeung, D.Y.: Learning a deep compact image representation for visual tracking. In: *Advances in Neural Information Processing Systems*, pp. 809–817 (2013)
43. Wang, J., Kumar, S., Chang, S.F.: Semi-supervised hashing for scalable image retrieval. In: *CVPR*, pp. 3424–3431 (2010)
44. Weiss, Y., Torralba, A., Fergus, R.: Spectral hashing. In: *NIPS*, pp. 1753–1760 (2008)
45. Wen, Z., Yin, W.: A feasible method for optimization with orthogonality constraints. *Math. Program.* 1–38 (2013)
46. Xia, S., Shao, M., Fu, Y.: Kinship verification through transfer learning. In: *International Joint Conference on Artificial Intelligence*, pp. 2539–2544 (2011)
47. Xia, S., Shao, M., Fu, Y.: Toward kinship verification using visual attributes. In: *IEEE International Conference on Pattern Recognition*, pp. 549–552 (2012)
48. Xia, S., Shao, M., Luo, J., Fu, Y.: Understanding kin relationships in a photo. *IEEE Trans. Multimed.* **14**(4), 1046–1056 (2012)
49. Xu, Y., Li, X., Yang, J., Lai, Z., Zhang, D.: Integrating conventional and inverse representation for face recognition. *IEEE Trans. Cybern.* **44**(10), 1738–1746 (2014)
50. Yan, H., Lu, J., Deng, W., Zhou, X.: Discriminative multimetric learning for kinship verification. *IEEE Trans. Inf. Forensics Secur.* **9**(7), 1169–1178 (2014)
51. Yan, H., Lu, J., Zhou, X.: Prototype-based discriminative feature learning for kinship verification. *IEEE Trans. Cybern.* **45**(11), 2535–2545 (2015)
52. Yang, J., Zhang, D., Yang, J.Y.: Constructing pca baseline algorithms to reevaluate ica-based face-recognition performance. *IEEE Trans. Syst. Man Cybern. Part B: Cybern.* **37**(4), 1015–1021 (2007)
53. Zhou, X., Hu, J., Lu, J., Shang, Y., Guan, Y.: Kinship verification from facial images under uncontrolled conditions. In: *ACM International Conference on Multimedia*, pp. 953–956 (2011)
54. Zhou, X., Lu, J., Hu, J., Shang, Y.: Gabor-based gradient orientation pyramid for kinship verification under uncontrolled environments. In: *ACM International Conference on Multimedia*, pp. 725–728 (2012)

Facial Kinship Verification

A Machine Learning Approach

Yan, H.; Lu, J.

2017, X, 82 p. 33 illus., 29 illus. in color., Softcover

ISBN: 978-981-10-4483-0