Pattern Recognition 81 (2018) 357–369

Contents lists available at ScienceDirect

Pattern Recognition

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

Deep CNN based binary hash video representations for face retrieval

Zhen Dong, Chenchen Jing, Mingtao Pei*, Yunde Jia

Beijing Laboratory of Intelligent Information Technology, Beijing Institute of Technology, Beijing 100081, PR China

ARTICLE INFO

Article history:
Received 9 August 2017
Revised 29 March 2018
Accepted 11 April 2018
Available online 13 April 2018

Keywords:
Face video retrieval
Cross-domain face retrieval
Deep CNN
Hash learning

ABSTRACT

In this paper, a novel deep convolutional neural network is proposed to learn discriminative binary hash video representations for face retrieval. The network integrates face feature extractor and hash functions into a unified optimization framework to make the two components be as compatible as possible. In order to achieve better initializations for the optimization, the low-rank discriminative binary hashing method is introduced to pre-learn the hash functions of the network during the training procedure. The input to the network is a face frame, and the output is the corresponding binary hash frame representation. Frame representations of a face video shot are fused by hard voting to generate the binary hash video representation. Each bit in the binary representation of frame/video describes the presence or absence of a face attribute, which makes it possible to retrieve faces among both the image and video domains. Extensive experiments are conducted on two challenging TV-Series datasets, and the excellent performance demonstrates the effectiveness of the proposed network.

1. Introduction

Face video retrieval aims to search a video database to find the videos containing a particular person, with a face image/video of the same person as the query. It is attracting more and more attentions in recent years owing to its extensive applications, such as searching large amounts of long videos on the Internet to annotate face data for vision researchers, locating and tracking a specific criminal suspect in the mass of surveillance videos, and the intelligent fast-forward and fast-backward of movies.

The challenging problems of face video retrieval are the large intra-class variations of faces and the strong demands of time/pace saving. The faces in Fig. 1 show the dramatic intra-class variations caused by poses, lighting conditions, expressions, clothes, background interferences, and the orientation of the actor in TV-Series. Good representations of faces should be robust to these variations and discriminative between classes. Moreover, the representations have to be compact for fast retrieval and space saving. In this paper, we design a deep convolutional neural network (CNN) to learn discriminative and compact representations for face video retrieval.

Deep neural networks recently have been successfully applied on many face-related tasks, such as face recognition [1–5], face alignment [6–8], face detection [9,10], and face attribute prediction [11], which manifests their powerful abilities on learning appropriate high-level representations of faces. Despite the discriminative power, the CNN features in the form of high dimensional vectors of floating point numbers lead to tremendous time and space cost of the retrieval procedure. Hashing methods [12,13],which project high dimensional features into a binary space with relatively low dimensions, are widely utilized in retrieval tasks.

The above mentioned methods concentrate on either hash function learning or feature learning. The two procedures are independent with each other, resulting that the learned features might be incompatible with the learned hash functions. Therefore, we integrate hash functions into our deep CNN to accomplish end-to-end training, by which the face feature extraction and hash function learning procedures can be optimized jointly to learn discriminative and compact representations of faces. Fig. 1 depicts the proposed deep CNN for face retrieval. Even though faces of a same person have dramatically various appearances caused by expressions, lighting conditions, orientations and poses, the deep CNN can represent these various facial appearances by similar compact binary hash codes.

Our deep CNN contains two components: face feature extractor and hash functions. Fig. 2 shows that the training of our CNN follows a general-to-specific deep transfer scheme and includes three steps: learning face feature extractor, learning hash functions, and fine-tuning. In the first step, we retrain the well-known AlexNet [14] which is trained on the ImageNet dataset [15], with the large-scale face identity dataset, CASIA-WebFace [16], to adapt the network to the face domain and simultaneously enhance the discriminative power of face features. The bottom seven layers of the retrained AlexNet are cut out and utilized as the feature extractor. In

* Corresponding author.
E-mail address: peimt@bit.edu.cn (M. Pei).

https://doi.org/10.1016/j.patcog.2018.04.014
0031-3203/

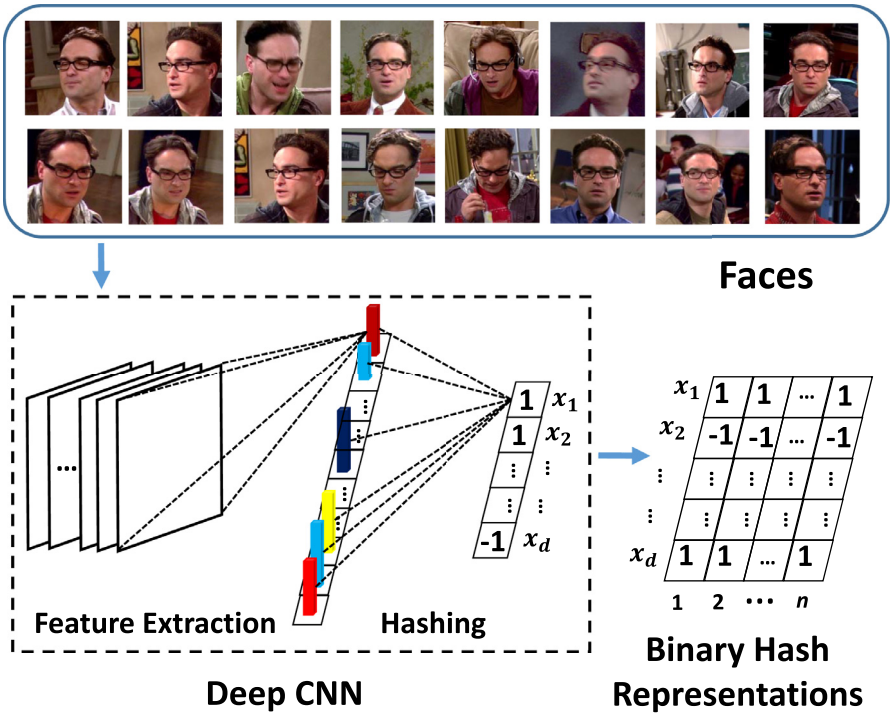


Fig. 1. Illustration of an ideal face retrieval system. The facial appearances of Leonard Hofstadter have dramatic variations caused by expressions, lighting conditions, orientations, and poses, the deep CNN in the system can still output similar binary representations for these faces. Besides, the hash functions are integrated into the deep CNN to accomplish the end-to-end learning.

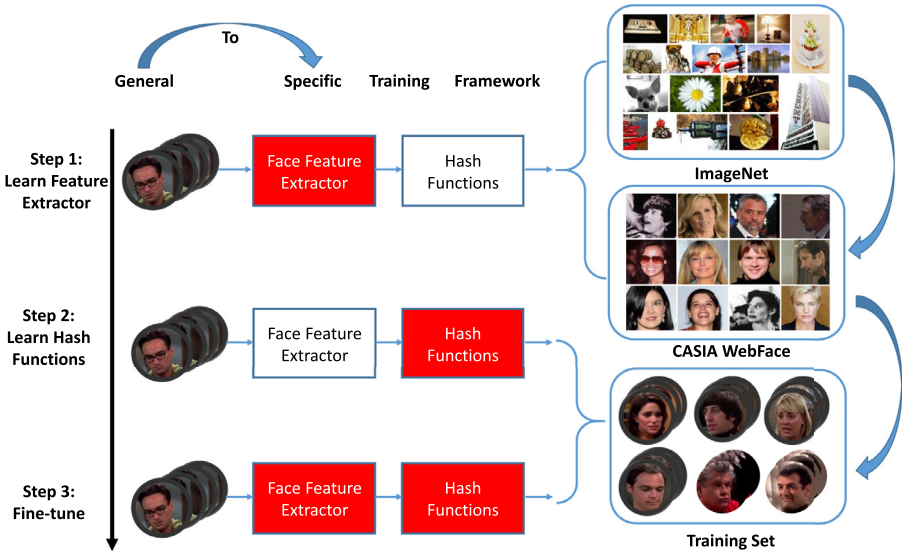


Fig. 2. The training procedure of our deep CNN follows a general-to-specific deep transfer scheme and includes three steps: learning face feature extractor, learning hash functions, and fine-tuning. The red rectangle represents the parameters of the component are learned in the corresponding step, the bold blue arrow presents the deep transfer scheme, and the black arrow shows the three steps. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the second step, the hash functions are learned with the extracted features of the training set of face videos by the low-rank discriminative binary hashing method. The hashing method is guided by the supervisory information of the training set to make faces of a same person have similar hash representations. In the third step, the entire CNN, including face feature extractor and hash functions, is specifically fine-tuned with the training set. The triplet ranking loss [17–20] is used to separate positive sample pairs and negative pairs by setting a distance margin.

The input to the network is a face frame, and the output is the corresponding binary representation. Each bit in the representation can be considered as the presences or absences of visual attributes of the face. For a face video shot which is actually a set of face frames, we get the corresponding set of binary hash representations through the trained network. Since the bits of representations describe the presences or absences of face attributes, it is easy to fuse the set of frame representations through hard voting to generate the representation of the video. The bits of the video representation still have the same meanings as frame represen-

tations. Therefore, cross-domain face retrieval can be conducted, *i.e.* retrieving face videos given the face image, and retrieving face images given the face video. We conduct experiments of cross-domain face retrieval on two challenging TV-series datasets, and achieve excellent performances on both datasets.

The contributions of our work are threefold: (1) our deep CNN, which simultaneously learns face feature extractor and hash functions, can learn discriminative and compact hash representations of face videos for retrieval; (2) the low-rank discriminative binary hashing method, which encourages the discriminative power, low-rank property and stability of hash codes, is proposed to initialize the hash functions in our deep CNN, and outperforms other state-of-the-art traditional hashing methods; and (3) our approach achieves excellent performances for various face retrieval tasks on two challenging TV-series datasets.

This paper is an extension of our previous work [21]. The extensions include: (1) we employ the proposed face video representation method to the cross-domain face retrieval task and achieve good performances; (2) we train the face feature extractor of our network on the CASIA-WebFace dataset and the ImageNet dataset following a general-to-specific deep transferring scheme to prevent over-fitting; and (3) more details of the implementations are described in this manuscript, including the fusion of frame representations and the selection of triplets.

The remainder of the paper are organized as follows. Section 2 reviews the related work including face video retrieval methods, traditional hashing methods, and deep hashing methods. We show the overview of our approach in Section 3, and elaborate the details of the training of our CNN in Section 4. Exhaustive experiments and discussions on two TV-Series datasets are shown in Section 5, and Section 6 concludes this paper.

2. Related work

2.1. Face video retrieval

Arandjelovic and Zisserman [22,23] presented a face shot retrieval system where each face shot is represented by a variation-robust signature image. Sivic et al. [24] characterized face shots via the probability distributions of its face frames for face shot retrieval. These works aim to accomplish complete retrieval system of face videos, and the key points are the implementations of each procedure including shot boundary detection, face detection, face tracking, etc. In these retrieval systems, face videos are represented by high dimensional vectors of real-valued numbers rather than compact binary hash code, which still has large time and space complexities. In contrast, we concentrate on learning compact and binary hash codes of face videos for retrieval.

Arandjelovic [25] proposed quasi-transitive similarity for identity-based retrieval of face sets from large unlabelled collections acquired in uncontrolled environments. His work is a meta-algorithm and focuses on leveraging the structure of the data to make the best use of an available baseline. Different from this method, we focus on learn discriminative and compact representations of face video.

Recently, Li et al. [26] exploited the covariance matrices of DCT features of frames to represent face videos, and introduced the compact video code (CVC) to encoded covariance matrices to get the binary hash codes of face videos for retrieval. They further utilized the Fisher vector features instead of the DCT features and extended the CVC method via kernel tricks in [27]. At the same time, they released two large scale TV-Series face video datasets which can be used to evaluate the performance of face video retrieval methods. They proposed a hashing method across the Euclidean Space and the Riemannian Manifold to measure the similarity of face images and videos for cross-domain face retrieval in [13], and represented face videos as spatial pyramid covariance matrices for face retrieval in TV-series in [28]. Although the above methods achieve good performances, the feature extracting (DCT, Fisher vector, *etc.*) and hash function learning are still two independent procedures. In other words, the features are not extracted for the retrieval task, which might limit the improvement of the retrieval performance. Different from these methods, our method integrates the face feature extraction procedure and the hash function learning procedure into a unified end-to-end training framework via a deep CNN.

2.2. Traditional hashing methods

Taking advantages of the compactness and binary property of the hash codes, hashing methods are commonly exploited in retrieval tasks for fast retrieval. Existing hashing methods can be roughly classified into two categories: data-independent and data-dependent. The data-independent hashing methods project high-dimensional features into low-dimensional space via hash functions which have nothing to do with training data. For example, the locality sensitive hashing (LSH) [29] and the kernelized locality-sensitive hashing (KLSH) [30] exploit random hash functions, and the shift-invariant kernel hashing (SIKH) [31] uses a shifted cosine function as the hash function. In real applications, the hash codes of data-independent methods are usually very long to guarantee satisfactory retrieval performances.

Different from data-independent hashing methods, data-dependent methods learn hash functions to make sure that the hash codes are semantically similar or structurally similar through fully discovering the supervision information or the structure information of the training data. The data-dependent methods are thus learning-based methods, and can be categorized as unsupervised, semi-supervised, and supervised according to whether the supervision information are used or not. Unsupervised methods, including the spectral hashing (SH) [32], the iterative quantization hashing (ITQ) [33], the anchor graph hashing (AGH) [34], the termed evolutionary compact embedding (ECE) [35], *etc.*, exploit only the training data without label information to learn hash functions. The SH calculates the hash codes by thresholding the eigenvalues of the Laplacian matrix of the similarity graph, and the ITQ iteratively optimizes the projection from the original high-dimensional feature space to the target low-dimensional Hamming Space via minimizing the quantization error of each iteration. Semi-supervised and supervised methods use both the training data and the corresponding label information to obtain high quality hash codes, and the representatives are the supervised iterative quantization hashing (SITQ) [33], the semi-supervised hashing (SSH) [36], the minimal loss hashing (MLH) [37], the kernel-based supervised hashing (KSH) [12], the discriminative binary coding (DBC) [38], the locality-sensitive two-step hashing (LS-TSH) [39], the supervised discrete hashing with relaxation (SDHR) [40], the robust discrete code modeling (RDCM) [41], and the predictable hash code learning (PHCL) [42], *etc*. Specifically, the SITQ utilizes the canonical correlation analysis instead of the principal component analysis used in ITQ. The SSH learns hash functions by using both labeled data and unlabeled data, *i.e.*, minimizing the quantization error of the labeled data, and maximizing the variance and the independence of hash representations on all the data simultaneously. The KSH employs the algebraic equivalence of the binary Hamming distance and the inner product of hash codes, which provides the feasibility of hashing in kernel spaces. The DBC jointly optimizes the discriminability and predictability of hash representations to discover visual attributes. The LS-TSH first applies the LSH on label vectors to generate hash codes, and then learns hash functions by using the generated hash codes, which provides a more fast hash learning method. The RDCM learns high-quality discrete binary codes

and hash functions by restraining the influence of unreliable binary codes and potentially noisily-labeled samples. The PHCL learns the predictable hash code by minimizing the distance between codes for samples from the same class and maximizing the distance between codes for samples from different classes.

2.3. Deep hashing methods

Recently, more and more hashing methods based on deep neural networks are proposed with significant performance improvements than traditional hashing methods. Xia et al. [43] proposed a CNN based supervised hashing method (CNNH) which contains two stages, *i.e.*, the first stage is to learn approximate hash codes by minimizing reconstruction errors, and the second stage is to learn image representations and hash functions simultaneously by using a deep CNN and the approximate hash codes obtained in the first stage. The CNNH is a pioneer work of hashing by introducing deep networks, but the learned image representation in the latter stage is not able to guide the former stage to learn better approximate hash codes. After CNNH, more literatures devoted to building end-to-end networks which contain all the stages of the hash learning so that every stage can be optimized for the final retrieval task. Lai et al. [18] proposed a "network in network" [44] based deep hashing model which projects the input images into the Hamming space for image retrieval. Zhao et al. [19] presented a multi-label image retrieval method, and the core component of the method is a deep CNN which is trained under the guidance of multilevel semantic ranking information. Lin et al. [45,46] modified the AlexNet by adding a new latent layer to learn hash codes of clothing images for the clothing image retrieval system. Liong et al. [47] learned compact binary representations via the proposed deep neural networks based hashing methods. Zhang et al. [17] designed an element-wise layer for the deep CNN to weight the bits of hashing codes, and proposed the bit-scalable hashing method based on the designed layer. Zhuang et al. [48] addressed that deep hashing method with triplet ranking loss needs a extremely large amount of triplets, and formulated the hash learning procedure as a multi-label classification problem. Lin et al. [49] presented an unsupervised deep hashing method via imposing three constrains on binary codes as the guidance of the network training, and outperformed the state-of-the features on the tasks of image matching, image retrieval, and object recognition. Liu et al. [50] addressed that the widely used non-linear approximation functions, sigmoid or tanh, inevitably slow down the convergence of the network, and imposed a constraint enforcing the values of the network outputs around ± 1 instead of the non-linear activation functions. Zhang et al. [51] trained a very deep neural network for hashing by introducing auxiliary variables and updating parameters layer by layer. Li et al. [52] learned hash representations for image data with pairwise labels based on multiple deep CNNs. Tang et al. [53] proposed a supervised deep hashing method for scalable face image retrieval based on Classification and Quantization errors. Since the deep learning framework is able to simultaneously learn compatible features and hash functions, the above hashing methods achieve encouraging performances on the image retrieval task. Motivated by these methods, we propose a deep learning based hashing method to learn compact binary codes of face videos for retrieval.

3. Approach overview

Fig. 3 depicts the four procedures of our approach. Firstly, a deep CNN is trained with large amounts of face images. The input to the network is a face frame, and the output is the corresponding compact binary hash representation of the face frame. Secondly, a face video is regarded as a collection of face frames, and a set of binary frame representations of the face video are computed by the trained deep CNN. Thirdly, all the frame representations are fused into a unified binary representation for the face video. Each bit of the video representation has the same meaning with the frame representation, which is convenient for measuring the similarity between face images and videos. Finally, the retrieval procedure is executed by calculating the distances between the binary representation of the query and ones of face videos in the database.

Fig. 4 shows the architecture of our deep CNN. The CNN is based on the well-known AlexNet [14], and we make the following modifications to implement hash functions by CNNs. The "FC8" and "Softmax" layers for classification of the AlexNet are removed, and four layers are added after the "FC7" layer: l_2 normalization layer (L2N), fully connected layer (FC8), tanh layer (TH), and the loss layer. The "L2N" layer executes the l_2 normalization on the output features of the "FC7" layer for the convenience of learning hash functions. After the "L2N" layer, a fully connected layer named "FC8" is added to act as the hash functions, and the number of the output neurons is the same as the number of the bits of the hash representation. The values of the outputs of the "FC8" layer are consecutive and range over the real number domain, so a tanh layer named "TH" is added after the "FC8" layer to quantify the consecutive features to the space of $(-1,+1)^S$, where S is the number of the bits of the hash representation. Finally, a triplet loss is inserted at the end of the network for the fine-tuning. The details of the training are elaborated in Section 4.

Ideally, the complex non-linear mapping implemented by the CNN should be $h:\mathcal{F}\rightarrow\{-1,+1\}^S$ where $\mathcal{F}$ denotes the face image space, but the CNN shown in Fig. 4 now only describes the mapping $h_f:\mathcal{F}\rightarrow(-1,+1)^S$. In order to obtain the binary representation of a face frame $\boldsymbol{f_i}$, a sign function is used as

$$\begin{aligned}\boldsymbol{p_i} &= h_f(\boldsymbol{f_i}),\\ \boldsymbol{c_i} &= h(\boldsymbol{f_i}) = sgn(\boldsymbol{p_i}),\end{aligned} \tag{1}$$

where $\boldsymbol{p_i}$ represents the output of the "TH" layer, $\boldsymbol{c_i}$ denotes the corresponding binary representation of $\boldsymbol{f_i}$, and the $sgn(\cdot)$ is the sign function, *i.e.*, for each element of the matrix inside the function, the function returns $+1$ if the element is greater than zero, and -1 if it is less than or equal to zero. For a face video clip $\boldsymbol{V}=\{\boldsymbol{f_1},\boldsymbol{f_2},\ldots,\boldsymbol{f_n}\}$ with n frames where $\boldsymbol{f_i}$ denotes the ith frame of $\boldsymbol{V}$, we compute the binary representations of its frames $\boldsymbol{C}=[\boldsymbol{c_1},\boldsymbol{c_2},\ldots,\boldsymbol{c_n}]\in\{-1,+1\}^{S\times n}$ by the trained deep CNN. A fused representation of the video is required for the face retrieval task, and the mean vector of frame representations, $\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{c_i}$, is a straightforward choice. Note that each element in $\boldsymbol{c_i}$ is either $+1$ or -1, which can be viewed as the presence or absence of a face attribute. The mean vector of these frame representations reports the average degree of presence of attributes in the face video.

The Hamming distance of the binary space is introduced to measure the similarity of hash representations, and can be written as

$$d(\boldsymbol{c}_i,\boldsymbol{c}_j)=\frac{S-\boldsymbol{c}_i^{\top}\boldsymbol{c}_j}{2}. \tag{2}$$

In the face video retrieval task, we need to compute the distance between two video representations to decide whether the face videos are similar or not. The mean vector of frame representations can be used as the video representation.

However, since the mean vector is unlikely to be binary, it needs to be quantized. For each bit in the mean vector, $+1$ is used if the value of bit is larger than 0, otherwise -1 is used, which is formulated as

$$\boldsymbol{c}=sgn(\frac{1}{n}\sum_{i=1}^{n}\boldsymbol{c_i}). \tag{3}$$

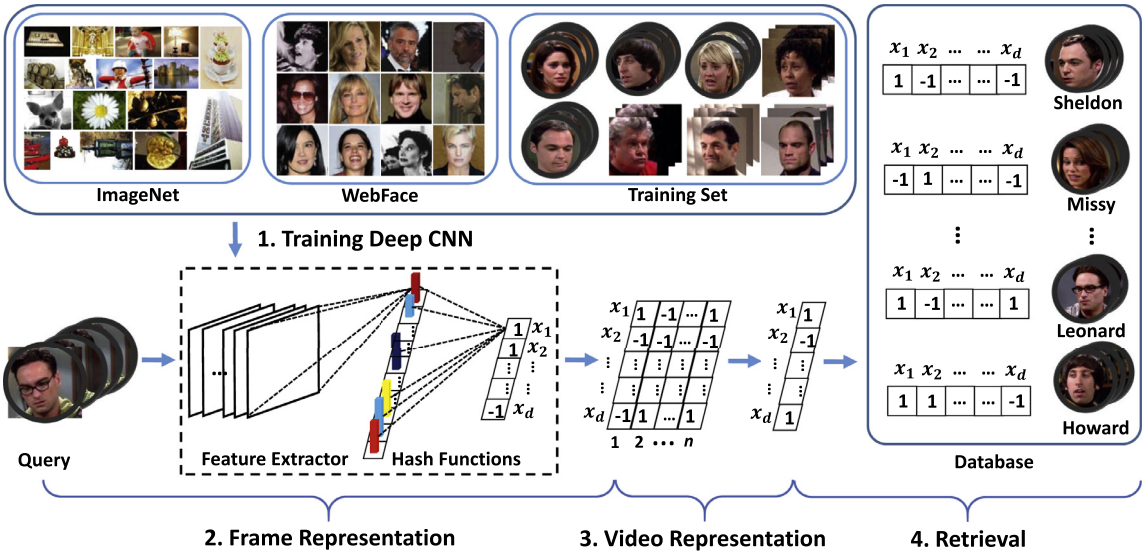


Fig. 3. The overview of our face video retrieval approach.

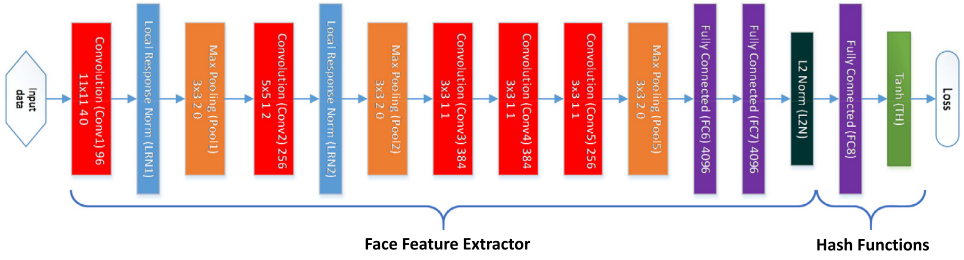


Fig. 4. The architecture of our deep CNN.

where $\boldsymbol{c}$ is the binary representation of face video V. The representations after quantization is equivalent to the hard-voting on each bit over all the frame representations.

The representations of all the face videos in the database are obtained off-line and stored for retrieval. For a query video, the corresponding representation is calculated on-line. Both of the query and database video representations are binary, so the similarities between the query video and all the videos in the database are calculated by Eq. (2) which represents counting the number of different bits of the two representations. The videos in the database are returned according to the calculated distances. The entire retrieval procedure takes less computational resources and can be accomplished in short time owing to the learned binary video representations.

4. Training deep CNN

The CNN depicted in Fig. 4 has two components: layers before "L2N" as the face feature extractor, and the newly added "FC8" and "TH" layers as the hash functions. Fig. 2 shows that the training of the deep CNN contains three steps: learning face feature extractor, learning hash functions, and fine-tuning.

4.1. Learning face feature extractor

The face feature extractor in our network is based on the well-known AlexNet. The released AlexNet is learned by using the large scale image dataset, ImageNet [15], which contains more than 1.2 million images of 1,000 categories. The AlexNet has five types of layers: convolutional layer, max-pooling layer, local contrast normalization layer, fully connected layer and the non-linear ReLU activation layer. The "FC7" layer is followed by a fully connected layer with 1,000 output neurons and a softmatx layer to compute the probability distribution over the categories. Since previous studies such as [14,54] presented better performances of the 4096-dimensional features of the "FC7" layer than a large amount of hand-crafted features, we thus use layers before "FC7" of the AlexNet as the initialization of the face feature extractor.

Considering that the AlexNet is trained with natural images, we learn the face feature extractor by using the large-scale face identity dataset, CASIA-WebFace [16], to transfer the network from natural image domain to the face image domain. The CASIA-WebFace dataset has 494,414 face images of 10,575 individuals in total. To learn the face feature extractor, we add a fully connected layer with 10,575 output neurons and a softmax layer after the "L2N" layer. The learning procedure is implemented by using the open source Caffe tool [55]. We learn the face feature extractor via the stochastic gradient descent method where the momentum and the weight decay are set as 0.9 and 0.0001, respectively. The learning rate of the optimization is initialized as 0.001 and decreased according to the polynomial policy with power value of 0.5. Besides, the size of the mini-batch of the training samples is 256, and the total number of the iterations is 210,000.

4.2. Learning hash functions

In order to effectively initialize the hash functions of our deep CNN, we introduce the Low-rank Discriminative Binary Hashing (LDBH) method to represent the face videos via compact binary representations.

Assume that we have N training face frames, and use the learned face feature extractor to generate their corresponding face

features represented as $\boldsymbol{P}=[\boldsymbol{p}_1,\boldsymbol{p}_2,\ldots,\boldsymbol{p}_N]\in\mathbb{R}^{T\times N}$ where $\boldsymbol{p}_i$ represents the face feature of the ith training sample, and T represents the dimension of the face feature. The hash functions $\boldsymbol{W}$ are to be learned to map these face features to the compact binary hash codes, which is formulated as

$$\boldsymbol{B}=sgn(\boldsymbol{W}^\top\boldsymbol{P}), \tag{4}$$

where $\boldsymbol{B}=[\boldsymbol{b}_1,\boldsymbol{b}_2,\ldots,\boldsymbol{b}_N]\in\{+1,-1\}^{S\times N}$ is the binary hash code matrix of $\boldsymbol{P}$, $\boldsymbol{b}_i$ represents the corresponding compact binary hash code of the face feature $\boldsymbol{p}_i$, and $\boldsymbol{W}=[\boldsymbol{w}_1,\boldsymbol{w}_2,\ldots,\boldsymbol{w}_S]\in\mathbb{R}^{T\times S}$ has S hash functions in total. Note that the dimension of the hash code satisfies $S\ll T$ to ensure the compactness of the hash codes.

To clearly elaborate the learning details of the LDBH method, we reformulate $\boldsymbol{P}$ and $\boldsymbol{B}$ as $\boldsymbol{P}=[\boldsymbol{P}_1,\boldsymbol{P}_2,\ldots,\boldsymbol{P}_C]$ and $\boldsymbol{B}=[\boldsymbol{B}_1,\boldsymbol{B}_2,\ldots,\boldsymbol{B}_C]$, respectively, where C is the number of classes, $\boldsymbol{P}_i$ is the face feature set of the training samples of the ith class, and $\boldsymbol{B}_i$ is the binary hash code set of $\boldsymbol{P}_i$. To guarantee the qualities of the generated binary codes, the LDBH encourages binary codes to be discriminative, low-rank and stable. The discriminations of the hash codes make that face features of a same person have similar hash codes and the hash codes of face features of different individuals are as dissimilar as possible, which is implemented by minimizing

$$g(\boldsymbol{B})=\sum_{\substack{p=1,\\ \boldsymbol{b}_i,\boldsymbol{b}_j\in\boldsymbol{B}_p}}^{C} d(\boldsymbol{b}_i,\boldsymbol{b}_j)-\lambda\sum_{\substack{p=1,\\ \boldsymbol{b}_i\in\boldsymbol{B}_p}}^{C}\sum_{\substack{q=1,q\neq p,\\ \boldsymbol{b}_j\in\boldsymbol{B}_q}}^{C} d(\boldsymbol{b}_i,\boldsymbol{b}_j), \tag{5}$$

where $d(\cdot,\cdot)$ is the Hamming distance between binary codes, and λ is a parameter to balance the two terms. In addition, the binary code matrix $\boldsymbol{B}$ are expected to be low-rank so that hash codes belonging to a same individual are well correlated and the redundancy of hash codes is significantly decreased. It is NP-hard to solve the problem of rank function minimization, so minimizing its convex envelope, the nuclear norm $\|\boldsymbol{B}\|_*$, is exploited instead. Since hash functions $\boldsymbol{w}_i$ actually act as hyperplanes splitting the face feature space, the statistical learning theory successfully utilized in SVM is able to guide the optimization. Specifically, the hyperplanes with the largest margin should be used to ensure the stabilities of the hash codes. Overall, the LDBH is modeled as

$$\begin{aligned}\min_{\boldsymbol{W},\xi,\boldsymbol{B}}\ & g(\boldsymbol{B})+\eta\|\boldsymbol{B}\|_*+\frac{1}{2}\|\boldsymbol{W}\|_F^2+\mu\sum_{i=1}^{d}\sum_{j=1}^{n}\xi_{ij}\\ s.t.\ & \boldsymbol{B}_{ij}(\boldsymbol{w}_i^\top\boldsymbol{p}_j)\geq 1-\xi_{ij},\\ & \xi_{ij}\geq 0,\\ & \boldsymbol{B}=sgn(\boldsymbol{W}^\top\boldsymbol{P}),\end{aligned} \tag{6}$$

where η is a parameter to control the weight of low-rank term, $\|\boldsymbol{W}\|_F$ represents the matrix Frobenius norm of $\boldsymbol{W}$ and aims to seek the hyperlanes with largest margins, and μ is set to take a trade-off between the splitting error and the capacity like SVM.

The global optimum solution is not easy to find due to the non-convex property of the objective function of the LDBH, so we iteratively optimize each term to obtain an effective local optimum solution instead. Firstly, we fix $\boldsymbol{B}$ and utilize the rows of $\boldsymbol{B}$ as labels and $\boldsymbol{P}$ as training data to train S linear SVMs to update $\boldsymbol{W}$ and ξ. Secondly, the binary code matrix $\boldsymbol{B}$ is updated by using the new $\boldsymbol{W}$ as Eq. (4). Thirdly, with the fixed $\boldsymbol{W}$ and ξ, an auxiliary variable $\boldsymbol{Z}$ is introduced to deal with the discriminative and low-rank constraints simultaneously, and Eq. (6) is written as

$$\begin{aligned}\min_{\boldsymbol{B},\boldsymbol{Z}}\ & g(\boldsymbol{B})+\eta\|\boldsymbol{Z}\|_*\\ s.t.\ & \boldsymbol{B}=\boldsymbol{Z},\boldsymbol{B}\in\{+1,-1\}^{S\times N}.\end{aligned} \tag{7}$$

The alternating direction method is exploited to solve Eq. (7), and the augmented Lagrangian function is given by

$$g(\boldsymbol{B})+\eta\|\boldsymbol{Z}\|_*+tr(\boldsymbol{H}(\boldsymbol{Z}-\boldsymbol{B})^\top)+\frac{\alpha}{2}\|\boldsymbol{Z}-\boldsymbol{B}\|_F^2. \tag{8}$$

where $\boldsymbol{H}$ is the Lagrangian multiplier matrix. The $\boldsymbol{Z}$ has a closed analytical form solution [56] to be optimum with fixed $\boldsymbol{B}$, and $\boldsymbol{B}$ is optimized by the subgradient descent method [38] with fixed $\boldsymbol{Z}$. The algorithm of the LDBH method is summarized in Algorithm 1 where $\otimes$ represents the element-wise multiplication operator.

Algorithm 1: Low-rank discriminative binary hashing.

Input: Feature set $\boldsymbol{P}\in\mathbb{R}^{T\times N}$ and the corresponding labels.
Output: Hash functions $\boldsymbol{W}\in\mathbb{R}^{T\times S}$.

repeat
  Train S linear SVMs with $\boldsymbol{B}$ as labels to update $\boldsymbol{W}$;
  $\boldsymbol{B}=sgn(\boldsymbol{W}^\top\boldsymbol{P})$;
  repeat
    SVD decomposition: $\boldsymbol{B}-\boldsymbol{H}/\beta=\boldsymbol{U}\boldsymbol{Q}\boldsymbol{V}^\top$;
    $\boldsymbol{Z}=\boldsymbol{U}\big(\max(|\boldsymbol{S}-\eta/\alpha|,0)\otimes sgn(\boldsymbol{Q})\big)\boldsymbol{V}^\top$;
    Update $\boldsymbol{B}$ by using subgradient descend method [38];
    $\boldsymbol{H}=\boldsymbol{H}+\beta(\boldsymbol{Z}-\boldsymbol{B})$;
    $\beta=\gamma\beta$;
  until *Convergence*;
until *Convergence*;

4.3. Fine-tuning

With the effective initializations of the face feature extractor and the hash functions, this procedure aims to simultaneously fine-tune them in a unified optimization framework. The framework makes the two components interact with each other for the optimal compatibility, *i.e.*, the extracted face features are utilized to fine-tune the hash functions for good hashing results, and the hashing results can inversely guide the fine-tuning of the face feature extractor. After fine-tuning, the face feature extractor and hash functions are firmly combined to form a unified hashing network for face video retrieval. To get outstanding hash representations for retrieval, the object of the fine-tuning procedure is to hold a large margin between the distances of positive and negative pairs of hash representations. To this end, the triplet ranking loss is exploited to fine-tune the entire network.

The triplet ranking loss describes the relative similarities of the hash representations in the form like "face frame $\boldsymbol{f}$ is more similar to $\widetilde{\boldsymbol{f}}$ than $\widehat{\boldsymbol{f}}$". Consistent with Eq. (1), the $(\boldsymbol{p},\widetilde{\boldsymbol{p}},\widehat{\boldsymbol{p}})$ is used to represent $(h_f(\boldsymbol{f}),h_f(\widetilde{\boldsymbol{f}}),h_f(\widehat{\boldsymbol{f}}))$, and the $(\boldsymbol{c},\widetilde{\boldsymbol{c}},\widehat{\boldsymbol{c}})$ is used to represent corresponding binary hash representations $(h(\boldsymbol{f}),h(\widetilde{\boldsymbol{f}}),h(\widehat{\boldsymbol{f}}))$. Since the sign function $sgn(\cdot)$ is non-smooth and non-differentiable, the $(\boldsymbol{p},\widetilde{\boldsymbol{p}},\widehat{\boldsymbol{p}})$ is used instead of the $(\boldsymbol{c},\widetilde{\boldsymbol{c}},\widehat{\boldsymbol{c}})$ during the fine-tuning procedure. Similar to Norouzi et al. [20], the triplet ranking loss for $(\boldsymbol{p},\widetilde{\boldsymbol{p}},\widehat{\boldsymbol{p}})$ is defined as

$$t(\boldsymbol{f},\widetilde{\boldsymbol{f}},\widehat{\boldsymbol{f}})=\max\Big(d\big(\boldsymbol{p},\widetilde{\boldsymbol{p}}\big)-d\big(\boldsymbol{p},\widehat{\boldsymbol{p}}\big)+\zeta,0\Big), \tag{9}$$

where $d(\cdot,\cdot)$ is the Hamming distance of binary space described in Eq. (2), and $\zeta\geq 0$ represents the margin of the distance differences between positive and negative pairs. Supposing that the training face frame set of C classes is $\boldsymbol{F}=[\boldsymbol{F}_1,\boldsymbol{F}_2,\ldots,\boldsymbol{F}_C]$, we model the fine-tuning procedure of our deep CNN as

$$\min_{\boldsymbol{W}_f,\boldsymbol{W}}\sum_{i=1}^{C}\sum_{\substack{\boldsymbol{q},\widetilde{\boldsymbol{q}}\in\boldsymbol{F}_i,\\ \boldsymbol{q}\neq\widetilde{\boldsymbol{q}}}}\sum_{\substack{j\neq i,\\ \widehat{\boldsymbol{q}}\in\boldsymbol{F}_j}} t(\boldsymbol{f},\widetilde{\boldsymbol{f}},\widehat{\boldsymbol{f}}), \tag{10}$$

where $\boldsymbol{W}_f$ and $\boldsymbol{W}$ represent the parameters of the face feature extractor and the hash functions, respectively.

To solve Eq. (10), we need to compute the gradients of Eq. (9) w.r.t. $(\boldsymbol{p}, \widetilde{\boldsymbol{p}}, \widehat{\boldsymbol{p}})$:

$$\frac{\partial t}{\partial \boldsymbol{p}} = \frac{1}{2}(\widehat{\boldsymbol{p}} - \widetilde{\boldsymbol{p}}) \times \mathbf{1}\big(\Lambda\big),$$
$$\frac{\partial t}{\partial \widetilde{\boldsymbol{p}}} = -\frac{1}{2}\boldsymbol{p} \times \mathbf{1}\big(\Lambda\big), \quad \frac{\partial t}{\partial \widehat{\boldsymbol{p}}} = \frac{1}{2}\boldsymbol{p} \times \mathbf{1}\big(\Lambda\big),$$
$$\Lambda \triangleq d(\boldsymbol{p}, \widetilde{\boldsymbol{p}}) - d(\boldsymbol{p}, \widehat{\boldsymbol{p}}) + \zeta > 0, \tag{11}$$

where $\mathbf{1}(\cdot)$ is the indicator function which returns 1 if the condition inside is true and 0 for other occasions.

Another issue is how to select the triplets, since many triplets whose loss equals to 0 would take much memory and computation cost resulting in slower convergence. In order to ensure fast convergence and good optimization simultaneously, it is crucial to generate triplets which have contributions to the training. We simplify the problem of generating triplets as selecting the negative sample $\widehat{\boldsymbol{p}}$ for the similar sample pair $(\boldsymbol{p}, \widetilde{\boldsymbol{p}})$ from the whole batch. The class information is used to measure the similarity, *i.e.*, $\boldsymbol{p}$ and $\widetilde{\boldsymbol{p}}$ are face frames of the same person, and $\widehat{\boldsymbol{p}}$ belongs to another individual. Specifically, we organize the training set in the form of similar sample pairs to let a batch with L samples have $L/2$ similar sample pairs. For a similar sample pair $(\boldsymbol{p}, \widetilde{\boldsymbol{p}})$, we select negative samples only from the left $L-2$ samples of the batch. The negative sample $\widehat{\boldsymbol{p}}$ needs to meet two conditions: $\widehat{\boldsymbol{p}}$ belongs to a different individual from $\boldsymbol{p}$ and $\widetilde{\boldsymbol{p}}$, and the consisted triplet $(\boldsymbol{p}, \widetilde{\boldsymbol{p}}, \widehat{\boldsymbol{p}})$ has positive loss, *i.e.* $d(\boldsymbol{p}, \widetilde{\boldsymbol{p}}) - d(\boldsymbol{p}, \widehat{\boldsymbol{p}}) + \zeta > 0$. Let $\mathbb{N}$ be the negative sample set for pair $(\boldsymbol{p}, \widetilde{\boldsymbol{p}})$, we select M negative samples from $\mathbb{N}$ in two ways: hard negative selecting and random negative selecting.

- **Hard Negative Selecting:** Hard negative samples mean that they are much closer from $\boldsymbol{p}$ than other negative samples. Let $\mathbb{H}$ be the set of hard negative samples, we have

$$\max_{\boldsymbol{h}\in\mathbb{H}} d(\boldsymbol{p}, \boldsymbol{h}) < \min_{\boldsymbol{h}\in\mathbb{N}-\mathbb{H}} d(\boldsymbol{p}, \boldsymbol{h}). \tag{12}$$

  We enforce that $|\mathbb{H}| = K$.
- **Random Negative Selecting:** As for other $M-K$ negative samples, we randomly select them from $\mathbb{N}-\mathbb{H}$.

The percentage of hard negative samples, $\eta = K/M$, is set as 0.5 in our experiments. For a selected negative sample $\widehat{\boldsymbol{p}}$, two triplets are generated: $(\boldsymbol{p}, \widetilde{\boldsymbol{p}}, \widehat{\boldsymbol{p}})$ and $(\widetilde{\boldsymbol{p}}, \boldsymbol{p}, \widehat{\boldsymbol{p}})$, thus providing $2M$ triplets for each similar sample pair. Note that we shuffle the training set at the beginning of each epoch to generate appropriate triplets as many as possible.

After obtaining the selected triplets, we execute the fine-tuning procedure of the network through the back-propagation (BP) algorithm. The BP algorithm is implemented in the form of the stochastic gradient descent where the momentum and the weight decay are set as 0.9 and 0.0005, respectively. The learning rate of the optimization is initialized as 0.001 and decreased according to the polynomial policy with power value of 0.6. Besides, the size of the mini-batch of the training samples is set as 64, and the total number of the iterations is 50,000.

Overall, the training procedure follows a general-to-specific deep transferring scheme to reduce the risk of over-fitting. We use three types of image and video data during the entire training procedure: ImageNet, CASIA-WebFace, and the training video set of specific individuals. As shown in Fig. 2, the deep CNN is first trained by the ImageNet dataset to achieve good initializations than random values. The large-scale face identity dataset, CASIA-WebFace, is then used to further improve the robustness of the face feature extractor. Finally, the training set of specific domain face video shots is exploited to learn hash functions and fine-tune the network. In this way, we adapt the deep CNN from the general nature image domain to a specific face domain to reduce the uncertainty and the diversity of representations.

5. Experiments

5.1. Dataset

The ICT-TV dataset [27] which has two large-scale face video shot collections is utilized to test the performance of the proposed method. All the face video shots are collected from the whole first season of two popular American shows: the Big Bang Theory (BBT) and Prison Break (PB). The filming styles of the two TV-series are quite different. The BBT is an indoor melodrama with only 5 main characters, and each episode lasts about just 20 min. In contrast, the PB mostly takes place outside, and the average length of all the episodes is around 42 minutes, which leads to large illumination variations. The total number of face video shots of the two collections are 4,667 and 9,435, respectively. This dataset provides original images of face frames rather than extracted features in previous TV-series datasets, and each face frame is stored with size of 150×150.

5.2. Comparison methods and evaluation criterions

We compare our approach with three types of the state-of-the-art methods to evaluate the performance:

1. Hashing methods: LSH [29], SH [32], ITQ [33], SITQ [33], RR [33], SSH [36] and KSH [12];
2. Face video retrieval methods: Hierarchical Hybrid Statistic based Video Binary Code (HHSVBC) [27] and Spatial Pyramid Covariance-based Compact Video Code (SPC-CVC) [28];
3. Cross-domain face retrieval method: Hashing across Euclidean space and Riemannian manifold (HER) [13].

For fair comparisons, all the comparison methods together with our method use the same training and testing sets, and the details of splitting training and testing sets are elaborated in Section 5.3. The length of hash codes ranges from 8 to 256 to show the performances of all these methods versus code lengths, and the performance improvement is not obvious when the length of hash codes is larger than 256. The parameters of the comparison methods are carefully set for fair performance based on the suggestions in their original publications.

To evaluate the retrieval performance, four evaluation criterions are used: Precision Recall curve (PR curve), Precision curve w.r.t. Number of top returned samples (PN curve), Recall curve w.r.t. Number of top returned samples (RN curve) and mean Average Precision (mAP). In addition, the mAP curve w.r.t. hash code length (mAP curve) is presented to show the influence of hash code length to the mAP. All the reported results are the average of 300 round of tests. It should be noted that the PR, PN and RN curves only with hash code length of 128 as representative results are presented in the main body of this paper for space limitations, and please find the complete experimental results in the supplementary material.

5.3. Results and discussions

We conduct three experiments. The proposed Low-rank Discriminative Binary Hashing (LDBH) method is first evaluated and compared with other the state-of-the-art hashing methods. Then, we compare the performance of our method on the face video retrieval task with other methods and report the comparison results. Finally, the cross-domain face retrieval experiment is conducted.

Table 1
Comparison mAPs of LDBH and other hashing methods.

Methods	*the Big Bang Theory*						*Prison Break*					
	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits
LSH [29]	0.29	0.39	0.58	0.70	0.78	0.85	0.09	0.12	0.16	0.24	0.29	0.38
RR [33]	0.78	0.79	0.84	0.84	0.87	0.87	0.24	0.32	0.37	0.42	0.44	0.46
ITQ [33]	0.84	0.85	0.87	0.88	0.89	0.90	0.31	0.39	0.45	0.48	0.49	0.50
SH [32]	0.59	0.50	0.53	0.50	0.47	0.42	0.22	0.27	0.29	0.29	0.27	0.24
SSH [36]	0.75	0.78	0.63	0.56	0.51	0.48	0.28	0.37	0.29	0.25	0.24	0.23
KSH [12]	0.72	0.91	0.93	0.95	0.95	0.94	0.55	0.71	0.75	0.80	0.82	0.81
SITQ [33]	0.80	0.88	0.92	0.92	0.93	0.93	0.39	0.56	0.62	0.65	0.65	0.62
LDBH	**0.97**	**0.96**	**0.97**	**0.99**	**0.99**	**0.99**	**0.58**	**0.73**	**0.79**	**0.83**	**0.85**	**0.85**

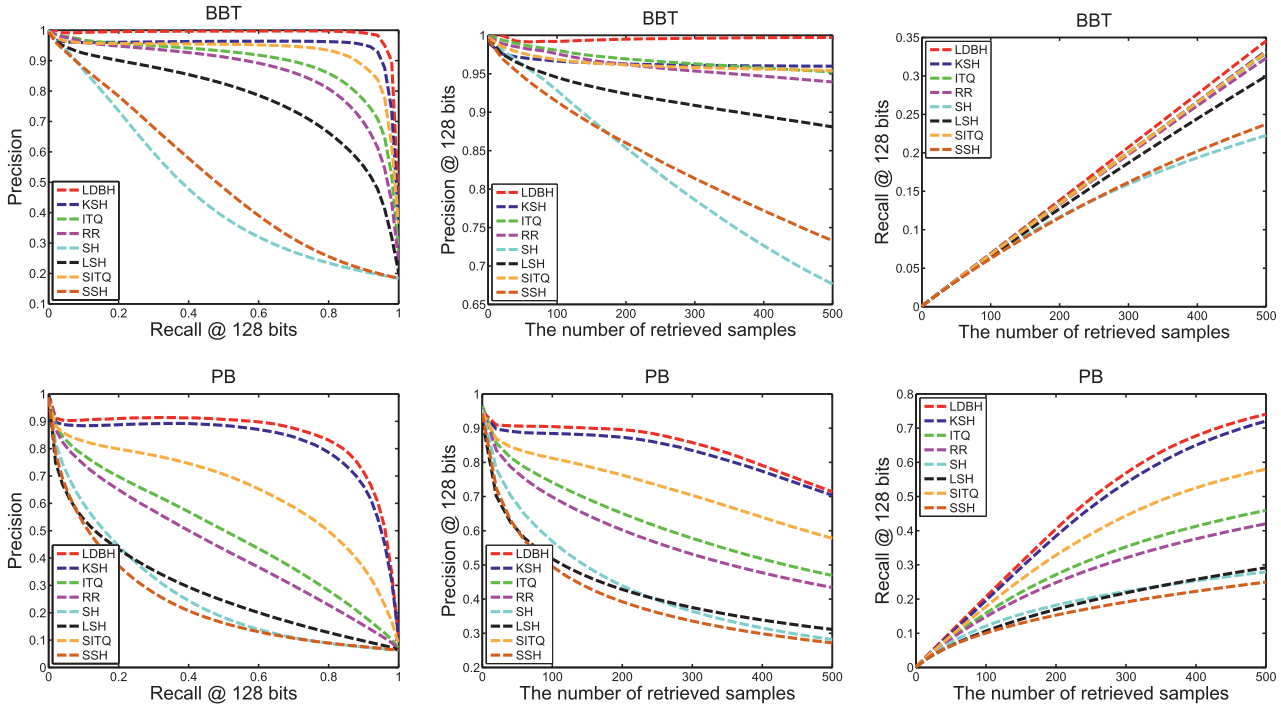


Fig. 5. Comparisons of PR, PN and RN curves between the LDBH and other hashing methods on two TV-series datasets.

5.3.1. LDBH performance

Even though the LDBH is proposed to initialize the hash functions of our network, it can be executed independently as a hashing method. Similar to other hashing methods, the LDBH takes feature vectors as its input and outputs the corresponding hash codes to keep the semantic similarities of the feature vectors. The performance of the LDBH method is evaluated on both BBT and PB datasets. To eliminate the influence of the frame representation fusion, we compare the LDBH and other comparison methods on the task of face image retrieval, *i.e.*, "image query & image database".

The 4096-dim features of face frames extracted through our face feature extractor trained in the first step, *i.e.*, the AlexNet retrained on the WebFace dataset, are used for all these hashing methods. On both BBT and PB, frames of 10 randomly selected face shots per actor or actress are used as the training set, 10 frames per main actor or actress are randomly selected to form the query set, and 10,000 frames are randomly selected from the left frames as the database for retrieval. The main actors and actresses of BBT are Howard Wolowitz, Leonard Hofstadter, Penny, Raj Koothrappali and Sheldon Cooper, and ones of PB are Benjamin Miles 'C-Note' Franklin, Brad Bellick, Fernando Sucre, Henry Pope, John Abruzzi, Lincoln Burrows, Michael Scofield, Paul Kellerman, Sara Tancredi, Theodore 'T-Bag' Bagwell and Veronica Donovan.

Table 1 shows the comparison mAPs between the LDBH and other hashing methods, and Fig. 5 shows the comparisons of PR, PN, RN and mAP curves. Among these hashing methods, LSH, RR, ITQ and SH are unsupervised methods, and other comparison hashing methods and our LDBH are supervised methods. From the table and figure, we find that supervised hashing methods outperforms unsupervised methods in most cases since the supervised methods take full advantages of the label information. The main reason of the phenomenon that the LDBH outperforms all the other comparison methods is that the LDBH simultaneously takes the low-rank property, the discriminative power and the stability of hash codes into account, and fuses them into a unified optimization framework for hashing performance improvements. By the way, the reported PR, PN and RN curves are the average of the retrieval results of randomly selected $10 \times C$ queries where C represents the total number of the individuals.

5.3.2. Face video retrieval

In this part, our method is tested on the task of face video retrieval whose query set and database are both consisted of face video shots, *i.e.*, "video query & video database". This experiment is performed on both BBT and PB TV-series datasets.

On both TV-series datasets, 10 random selected face shots per actor or actress are exploited as the training set, and the remaining

Table 2
Comparison mAPs of the face video retrieval experiment.

Methods	*the Big Bang Theory*						*Prison Break*					
	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits
LSH [29]	0.43	0.53	0.69	0.75	0.85	0.88	0.13	0.13	0.19	0.27	0.35	0.43
RR [33]	0.83	0.87	0.84	0.86	0.89	0.91	0.28	0.38	0.42	0.46	0.49	0.51
ITQ [33]	0.84	0.90	0.89	0.91	0.93	0.93	0.36	0.45	0.51	0.53	0.54	0.53
SH [32]	0.64	0.54	0.56	0.53	0.49	0.44	0.26	0.31	0.33	0.33	0.29	0.27
SSH [36]	0.81	0.82	0.68	0.60	0.56	0.53	0.34	0.44	0.33	0.28	0.27	0.26
KSH [12]	0.83	0.91	0.94	0.94	0.94	0.94	0.50	0.62	0.63	0.70	0.72	0.75
SITQ [33]	0.85	0.94	0.95	0.95	0.95	0.95	0.48	0.60	0.67	0.70	0.69	0.67
HHSVBC [27]	0.51	0.59	0.67	0.68	0.72	0.74	0.14	0.15	0.16	0.16	0.18	0.20
SPC-CVC [28]	0.52	0.65	0.73	0.75	0.77	0.79	0.14	0.15	0.17	0.19	0.21	0.23
Ours (r.h.)	0.94	0.96	0.94	0.96	0.96	0.96	0.49	0.53	0.59	0.60	0.62	0.64
Ours (p.h.)	**0.95**	**0.97**	**0.98**	**0.98**	**0.97**	**0.98**	**0.75**	**0.78**	**0.76**	**0.79**	**0.83**	**0.83**

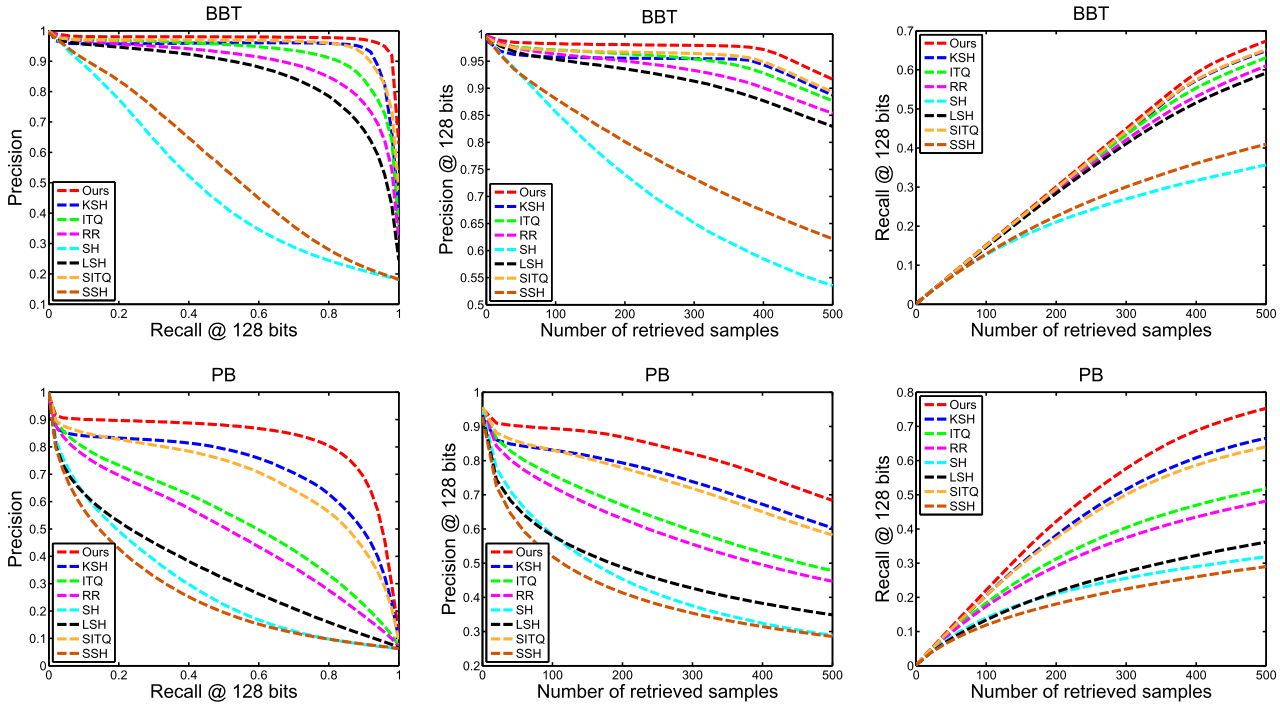


Fig. 6. Comparisons of PR, PN and RN curves of the face video retrieval experiment on two TV-series datasets.

Table 3
Comparison mAPs of the cross-domain face retrieval experiment.

Type	Methods	*the Big Bang Theory*						*Prison Break*					
		8 bits	16 bits	32 bits	64 bits	128 bits	256 bits	8 bits	16 bits	32 bits	64 bits	128 bits	256 bits
image query & video database	LSH [29]	0.38	0.48	0.60	0.78	0.84	0.88	0.10	0.12	0.20	0.24	0.33	0.41
	RR [33]	0.86	0.84	0.57	0.59	0.56	0.54	0.28	0.36	0.41	0.47	0.48	0.48
	ITQ [33]	0.88	0.88	0.90	0.91	0.92	0.92	0.36	0.43	0.49	0.52	0.53	0.53
	SH [32]	0.67	0.57	0.59	0.56	0.54	0.49	0.26	0.32	0.35	0.34	0.30	0.27
	SSH [36]	0.80	0.83	0.70	0.62	0.56	0.54	0.34	0.43	0.33	0.28	0.27	0.25
	KSH [12]	0.84	0.90	0.93	0.93	0.94	0.94	0.50	0.59	0.60	0.68	0.69	0.71
	SITQ [33]	0.82	0.93	0.95	0.94	0.95	0.94	0.46	0.60	0.65	0.67	0.65	0.64
	HER [13]	0.80	0.88	0.90	0.91	0.93	–	0.35	0.45	0.56	0.65	0.68	–
	Ours	**0.94**	**0.97**	**0.97**	**0.97**	**0.98**	**0.98**	**0.65**	**0.71**	**0.71**	**0.75**	**0.80**	**0.81**
video query & image database	LSH [29]	0.39	0.55	0.62	0.74	0.85	0.88	0.11	0.13	0.17	0.25	0.34	0.41
	RR [33]	0.86	0.87	0.88	0.88	0.90	0.90	0.30	0.37	0.41	0.47	0.49	0.48
	ITQ [33]	0.90	0.90	0.91	0.91	0.92	0.92	0.34	0.43	0.49	0.51	0.52	0.54
	SH [32]	0.69	0.59	0.61	0.58	0.53	0.49	0.27	0.32	0.34	0.33	0.30	0.28
	SSH [36]	0.82	0.84	0.72	0.64	0.58	0.54	0.34	0.44	0.33	0.29	0.27	0.26
	KSH [12]	0.85	0.92	0.93	0.94	0.95	0.94	0.49	0.59	0.60	0.68	0.69	0.72
	SITQ [33]	0.88	0.93	0.93	0.95	0.95	0.95	0.45	0.61	0.66	0.67	0.66	0.65
	HER [13]	0.84	0.87	0.90	0.91	0.91	–	0.39	0.50	0.57	0.64	0.66	–
	Ours	**0.94**	**0.97**	**0.98**	**0.98**	**0.98**	**0.98**	**0.65**	**0.72**	**0.70**	**0.75**	**0.80**	**0.80**

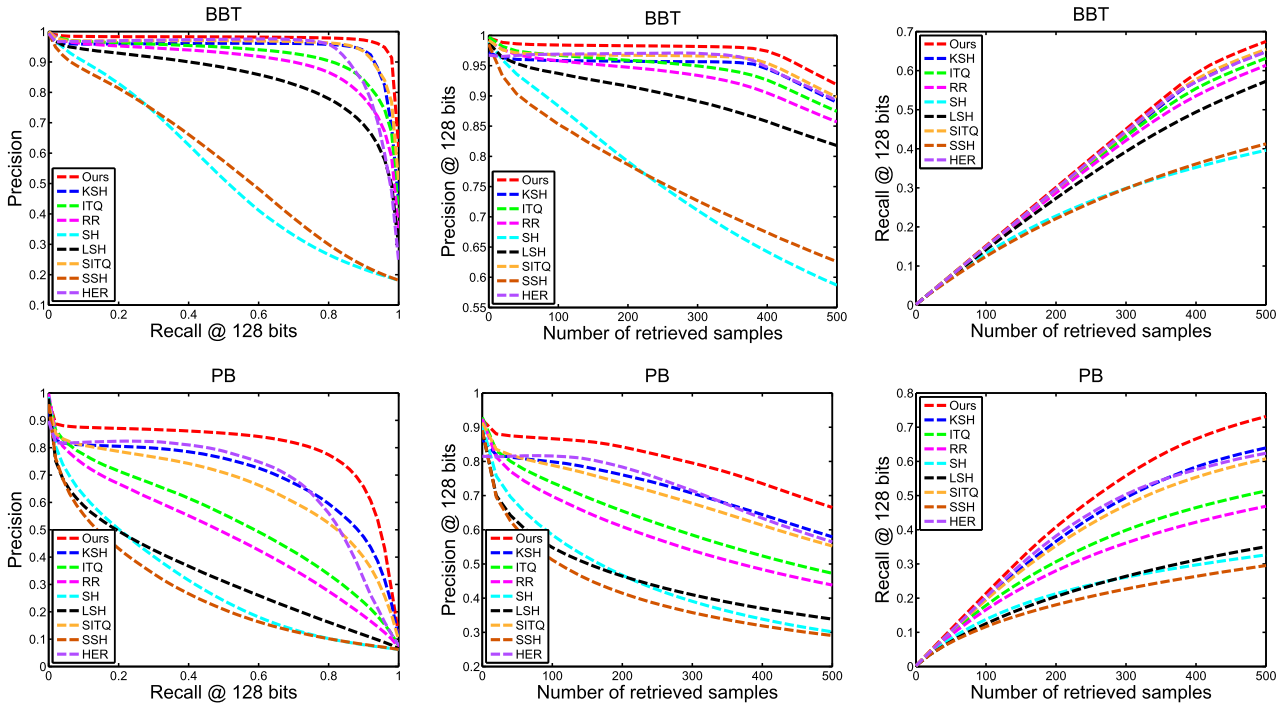


Fig. 7. Comparisons of PR, PN and RN curves of the cross-domain face retrieval experiment ("image query & video database") on two TV-series datasets.

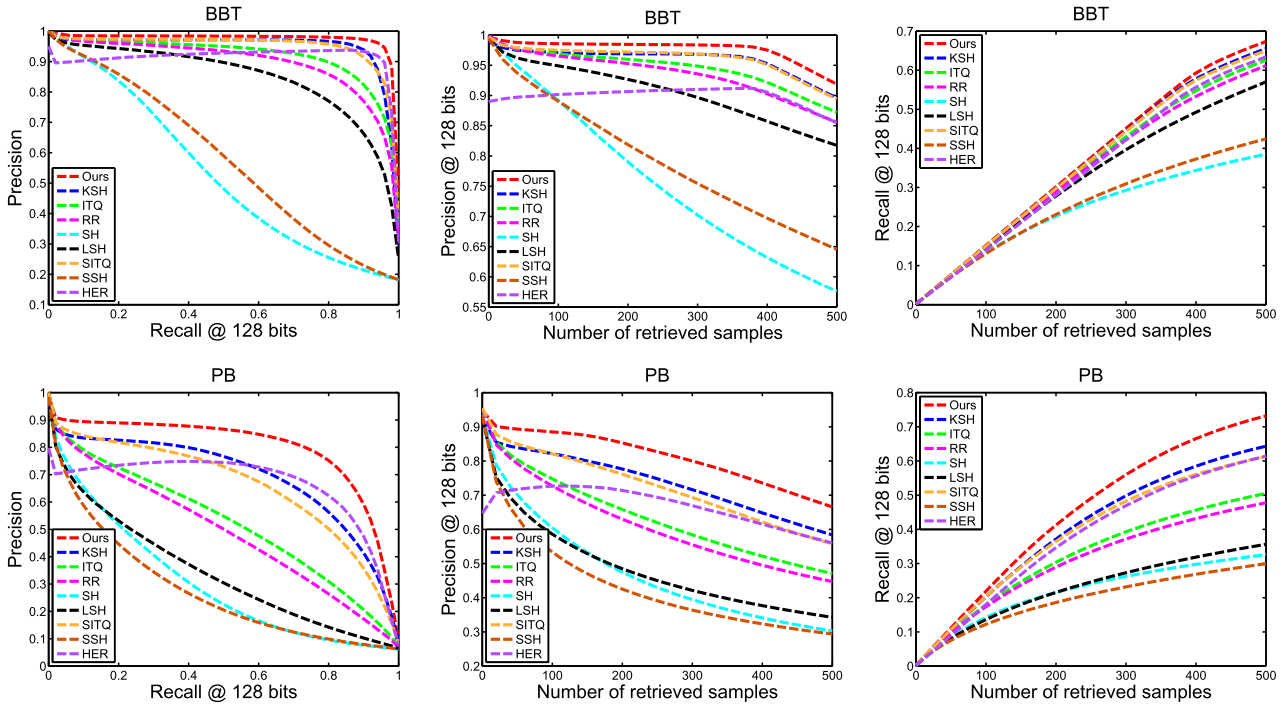


Fig. 8. Comparisons of PR, PN and RN curves of the cross-domain face retrieval experiment ("video query & image database") on two TV-series datasets.

face video shots are used for testing. Same as Li et al. [28], we further select 10 face shots per main actor or actress from the testing set randomly to form the query set, and the database is consisted of the left face shots in the testing set. All the frames of the face video shots in the training set are utilized to train our deep CNN, and the representations of face shots in the testing set can be obtained with the learned CNN for performance evaluation.

The comparison methods consist of seven hashing methods and two face video retrieval methods: HHSVBC [27] and SPC-CVC [28]. Table 2 lists the mAPs of our method and the comparison methods, and Fig. 6 depicts the comparisons of curves.

For the seven hashing methods, the same fusion method as ours described in Section 3 is utilized to obtain the final video hash representation. For fair comparisons, these hashing methods use the 4096-dim input features generated by our face feature extractor trained in the first step, *i.e.*, the AlexNet retrained on the WebFace dataset. As shown in Table 2 and Fig. 6, the proposed method significantly outperforms other comparison methods, especially in

the case of small hash code length. The reasons mainly lie in two aspects: the initialization through the LDBH method which has superior performances, and the fine-tuning mechanism to make the face feature extractor and hash functions optimally compatible for the retrieval task.

To verify the effectiveness of the pre-learned hash functions, we exclude the second step of the training procedure, and use random initializations of hash functions for the fine-tuning procedure instead of ones pre-learned by the LDBH method. The mAPs of our method with random initializations of hash functions in Table 2 is the line of "Ours (r.h.)", and the mAPs of our method with pre-learned hash functions is the line of "Ours (p.h.)". The performance difference between "Ours (p.h.)" and "Ours (r.h.)" demonstrates the positive effect of pre-learned hash functions as initializations of the fine-tuning procedure. It should be pointed out that our method even without pre-learned hash functions can achieve comparable results with other methods.

The face video retrieval methods, HHSVBC and SPC-CVC, use multiple size-variant covariance matrices calculated from fisher vectors and raw intensities as video features, respectively, and learn video hash representations from these covariance matrices. We keep the experimental setting of our method same with them and report the results published in [28]. Our method outperforms the HHSVBC and SPC-CVC methods. The main reason is that our method simultaneously optimizes the face feature extractor and hash functions for optimal compatibility, rather than uses fixed features which has nothing to do with the hashing procedure as input.

The HHSVBC and SPC-CVC methods extract features from the 20×16 gray images, which may have influence to the performance. However, it takes so large time and space for running them on larger size face frames that the comparison experiment cannot be conducted under current hardware conditions.

5.3.3. Cross-domain face retrieval

As described above, the input to our deep CNN is a face frame, and the output is the corresponding compact binary hash representation. Each bit of the binary frame representation characterizes the presence or absence of a face attribute, and has the same meaning with the corresponding bit of the fused video representation. The frame binary representation and the video binary representation thus lie in the same space, which provides the practicability of face retrieval across the image domain and the video domain. We conduct the cross-domain face retrieval experiment in this section, *i.e.* retrieving face videos given the image of this person, and its inverse task, retrieving face images given the video of this person. These two cases are respectively called "image query & video database" and "video query & image database" for simplicity.

Similar to Li et al. [13], the face image is acquired by extracting the medium frame of a face video. We compare our method with seven hashing methods and a cross-domain face retrieval method, HER [13]. The experimental setting and the training procedure are consistent with the face video retrieval experiment. All the comparison methods extract features for face frames by using our trained face feature extractor. For the HER method, a face video shot is characterized by the covariance matrix of extracted features of its face frames. For comparison hashing methods, we first calculate the distances between the query face image and each frame of the face video in database, and then integrate these distances by taking the average. The comparison mAPs are depicted in Table 3. Fig. 7 presents the comparison curves of the "image query & video database" task, and Fig. 8 presents the comparison curves of the "video query & image database" task. The initialization of the HER method picks several columns of kernel matrix, and the hash code length equals to the number of picked columns. However, the total number of the columns of the kernel matrix keeps consistent with the number of training samples, *i.e.*, 140 for the BBT dataset and 190 for the PB dataset, which is smaller than 256. The results of the HER method with hash code length of 256 thus cannot be reported. Our method significantly outperforms other methods on both "image query & video database" and "video query & image database" tasks.

6. Conclusion

This paper presented a deep CNN for face video retrieval. The network integrates feature extraction and hash learning into a unified optimization framework, which can guarantee that the feature extractor is optimally compatible with the followed hashing. The proposed low-rank discriminative binary hashing achieved better initialization of the network for hash function learning. The fine-tuning of the network after the initialization can further improve the performance of face video retrieval. Extensive experiments conducted on two challenging TV-Series datasets indicate the effectiveness of the proposed network.

Acknowledgments

This work was supported in part by the Natural Science Foundation of China (NSFC) under Grant No.61472038 and No.61375044.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2018.04.014.

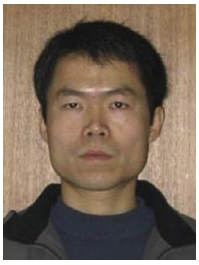
References

[1] Y. Taigman, M. Yang, M. Ranzato, L. Wolf, Web-scale training for face identification, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2476–2574.
[2] F. Schroff, D. Kalenichenko, J. Philbin, Facenet: a unified embedding for face recognition and clustering, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 815–823.
[3] Y. Sun, X. Wang, X. Tang, Sparsifying neural network connections for face recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4856–4864.
[4] M. Shao, Y. Zhang, Y. Fu, Collaborative random faces-guided encoders for pose-invariant face representation learning, IEEE Trans. Neural Netw. Learn. Syst. PP (99) (2017) 1–14.
[5] J. Yang, P. Ren, D. Chen, F. Wen, H. Li, G. Hua, Neural aggregation network for video face recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 5216–5225.
[6] J. Zhang, M. Kan, S. Shan, X. Chen, Occlusion-free face alignment: deep regression networks coupled with de-corrupt autoencoders, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3428–3437.
[7] A. Jourabloo, X. Liu, Large-pose face alignment via CNN-based dense 3d model fitting, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4188–4196.
[8] B. Shi, X. Bai, W. Liu, J. Wang, Face alignment with deep regression, IEEE Trans. Neural Netw. Learn. Syst. PP (99) (2016) 1–12.
[9] H. Li, Z. Lin, X. Shen, J. Brandt, G. Hua, A convolutional neural network cascade for face detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5325–5334.
[10] H. Qin, J. Yan, X. Li, X. Hu, Joint training of cascaded CNN for face detection, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 3456–3465.
[11] Z. Liu, P. Luo, X. Wang, X. Tang, Deep learning face attributes in the wild, in: IEEE International Conference on Computer Vision, 2014, pp. 3730–3738.
[12] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, S.-F. Chang, Supervised hashing with kernels, in: IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 2074–2081.
[13] Y. Li, R. Wang, Z. Huang, S. Shan, X. Chen, Face video retrieval with image query via hashing across euclidean space and Riemannian manifold, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 4758–4767.
[14] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Conference on Neural Information Processing Systems, 2012, pp. 1097–1105.
[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: a large-scale hierarchical image database, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248–255.
[16] D. Yi, Z. Lei, S. Liao, S.Z. Li, Learning face representation from scratch, arXiv:1411.7923 (2014).

[17] R. Zhang, L. Lin, R. Zhang, W. Zuo, L. Zhang, Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification, IEEE Trans. Image Process. 24 (12) (2015) 4766–4779.
[18] H. Lai, Y. Pan, Y. Liu, S. Yan, Simultaneous feature learning and hash coding with deep neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3270–3278.
[19] F. Zhao, Y. Huang, L. Wang, T. Tan, Deep semantic ranking based hashing for multi-label image retrieval, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1556–1564.
[20] M. Norouzi, D.J. Fleet, R. Salakhutdinov, Hamming distance metric learning, Adv. Neural Inf. Process. Syst. 2 (2012) 1061–1069.
[21] Z. Dong, S. Jia, T. Wu, M. Pei, Face video retrieval via deep learning of binary hash representations, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016, pp. 3471–3477.
[22] O. Arandjelović, A. Zisserman, Automatic face recognition for film character retrieval in feature-length films, in: IEEE Conference on Computer Vision and Pattern Recognition, 2005, pp. 860–867.
[23] O. Arandjelovi, A. Zisserman, On film character retrieval in feature-length films, in: Interactive Video, Springer, 2006, pp. 89–105.
[24] J. Sivic, M. Everingham, A. Zisserman, Person spotting: video shot retrieval for face sets, in: Image and Video Retrieval, Springer, 2005, pp. 226–236.
[25] O. Arandjelovic, Learnt quasi-transitive similarity for retrieval from large collections of faces, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4883–4892.
[26] Y. Li, R. Wang, Z. Cui, S. Shan, X. Chen, Compact video code and its application to robust face retrieval in tv-series, in: Proceedings of the British Machine Vision Conference, 2014.
[27] Y. Li, R. Wang, S. Shan, X. Chen, Hierarchical hybrid statistic based video binary code and its application to face retrieval in tv-series, in: IEEE International Conference and Workshops on Automatic Face and Gesture Recognition, 2015, pp. 1–8.
[28] Y. Li, R. Wang, Z. Cui, S. Shan, X. Chen, Spatial pyramid covariance-based compact video code for robust face retrieval in tv-series, IEEE Trans. Image Process. 25 (12) (2016) 5905–5919.
[29] A. Gionis, P. Indyk, R. Motwani, Similarity search in high dimensions via hashing, in: International Conference on Very Large Data Bases, 1999, pp. 518–529.
[30] B. Kulis, K. Grauman, Kernelized locality-sensitive hashing for scalable image search, in: IEEE International Conference on Computer Vision, 2010, pp. 2130–2137.
[31] M. Raginsky, S. Lazebnik, Locality-sensitive binary codes from shift-invariant kernels, in: Conference on Neural Information Processing Systems, 2009, pp. 1509–1517.
[32] Y. Weiss, A. Torralba, R. Fergus, Spectral hashing, in: Conference on Neural Information Processing Systems, 2008, pp. 1753–1760.
[33] Y. Gong, S. Lazebnik, Iterative quantization: a procrustean approach to learning binary codes, in: IEEE Conference on Computer Vision and Pattern Recognition, 2011, pp. 817–824.
[34] W. Liu, J. Wang, S. Kumar, S.-F. Chang, Hashing with graphs, in: International Conference on Machine Learning, 2011, pp. 1–8.
[35] L. Liu, L. Shao, Sequential compact code learning for unsupervised image hashing, IEEE Trans. Neural Netw. Learn. Syst. 27 (12) (2016) 2526–2536.
[36] J. Wang, S. Kumar, S.-F. Chang, Semi-supervised hashing for scalable image retrieval, in: IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 3424–3431.
[37] M. Norouzi, D.J. Fleet, Minimal loss hashing for compact binary codes, in: International Conference on Machine Learning, 2011, pp. 353–360.
[38] M. Rastegari, A. Farhadi, D. Forsyth, Attribute discovery via predictable discriminative binary codes, in: European Conference on Computer Vision, 2012, pp. 876–889.
[39] K. Ding, C. Huo, B. Fan, S. Xiang, C. Pan, In defense of locality-sensitive hashing, IEEE Trans. Neural Netw. Learn. Syst. PP (99) (2016) 1–17.
[40] J. Gui, T. Liu, Z. Sun, D. Tao, T. Tan, Supervised discrete hashing with relaxation, IEEE Trans Neural Netw Learn Syst PP (99) (2016) 1–10.
[41] Y. Luo, Y. Yang, F. Shen, Z. Huang, P. Zhou, H.T. Shen, Robust discrete code modeling for supervised hashing, Pattern Recognit. 75 (2018) 128–135.
[42] R. He, Y. Cai, T. Tan, L. Davis, Learning predictable binary codes for face indexing, Pattern Recognit. 48 (10) (2015) 3160–3168.
[43] R. Xia, Y. Pan, H. Lai, C. Liu, S. Yan, Supervised hashing for image retrieval via image representation learning, in: AAAI Conference on Artificial Intelligence, 2012, pp. 2156–2162.
[44] L. Min, C. Qiang, Y. Shuicheng, Network in network, in: International Conference on Learning Representations, 2014.
[45] K. Lin, H.-F. Yang, K.-H. Liu, J.-H. Hsiao, C.-S. Chen, Rapid clothing retrieval via deep learning of binary codes and hierarchical search, in: ACM International Conference on Multimedia Retrieval, 2015, pp. 499–502.
[46] K. Lin, H.-F. Yang, J.-H. Hsiao, C.-S. Chen, Deep learning of binary hash codes for fast image retrieval, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2015, pp. 27–35.
[47] V. Erin Liong, J. Lu, G. Wang, P. Moulin, J. Zhou, Deep hashing for compact binary codes learning, in: IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 2475–2483.
[48] B. Zhuang, G. Lin, C. Shen, I. Reid, Fast training of triplet-based deep binary embedding networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 5955–5964.
[49] K. Lin, J. Lu, C.-S. Chen, J. Zhou, Learning compact binary descriptors with unsupervised deep neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1183–1192.
[50] H. Liu, R. Wang, S. Shan, X. Chen, Deep supervised hashing for fast image retrieval, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2064–2072.
[51] Z. Zhang, Y. Chen, V. Saligrama, Efficient training of very deep neural networks for supervised hashing, in: IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 1487–1495.
[52] W.-J. Li, S. Wang, W.-C. Kang, Feature learning based deep supervised hashing with pairwise labels, in: International Joint Conference on Artificial Intelligence, 2016, pp. 1711–1717.
[53] J. Tang, Z. Li, X. Zhu, Supervised deep hashing for scalable face image retrieval, Pattern Recognit. 75 (2018) 25–32.
[54] M. Oquab, L. Bottou, I. Laptev, J. Sivic, Learning and transferring mid-level image representations using convolutional neural networks, in: IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1717–1724.
[55] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Caffe: convolutional architecture for fast feature embedding, in: International Conference on Multimedia, ACM, 2014, pp. 675–678.
[56] J.-F. Cai, E.J. Candès, Z. Shen, A singular value thresholding algorithm for matrix completion, SIAM J. Optim. 20 (4) (2010) 1956–1982.

Zhen Dong is currently a Ph.D. candidate in the Beijing Laboratory of Intelligent Information Technology at the School of Computer Science, Beijing Institute of Technology (BIT). He received the B.S. degree in June of 2011, then, continued to pursue a Ph.D. degree in the same department from the September of 2011, under the supervision of Prof. Yunde Jia. His research interests include computer vision, machine learning, and face recognition.

ChenChen Jing received the B.S. degree in computer science in 2016 from Beijing Institute of Technology, Beijing, China, where he is currently a Ph.D. candidate. His research interests include computer vision, pattern recognition, and face recognition.



Mingtao Pei received the Ph.D. degree in computer science from Beijing Institute of Technology, Beijing, China, in 2004. He is an Associate Professor with the School of Computer Science, Beijing Institute of Technology. From 2009 to 2011, he was a Visiting Scholar with the Center for Image and Vision Science, University of California, Los Angeles. His main research interest is computer vision with an emphasis on event recognition and machine learning. Dr. Pei is a member of China Computer Federation.

Yunde Jia is a Professor of computer science with Beijing Institute of Technology, Beijing, China. He is the Director of Beijing Laboratory of Intelligent Information Technology. He was a Visiting Scientist at Carnegie Mellon University between 1995 and 1997 and a Visiting Fellow with the Australian National University in 2011. His research interests include computer vision, media computing, and intelligent systems.