Pattern Recognition 113 (2021) 107754

Contents lists available at ScienceDirect

Pattern Recognition

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

Deep video code for efficient face video retrieval

Shishi Qiao[a,b], Ruiping Wang[a,b,*], Shiguang Shan[a,b], Xilin Chen[a,b]

[a] Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China

[b] University of Chinese Academy of Sciences, Beijing 100049, China

ARTICLE INFO

Article history:
Received 5 November 2019
Revised 3 November 2020
Accepted 8 November 2020
Available online 12 November 2020

Keywords:
Face video retrieval
Temporal feature pooling
Bounded triplet loss
Deep video code
Hash learning

ABSTRACT

In this paper, we address one specific video retrieval problem in terms of human face. Given one query in forms of either a frame or a sequence from a person, we search the database and return the most relevant face videos, i.e., ones have the same class label with the query. Such problem is very challenging due to the large intra-class variations and the high request on the efficiency of video representations in terms of both time and space. To handle such challenges, this paper proposes a novel Deep Video Code (*DVC*) method which encodes video faces into compact binary codes. Specifically, we devise an end-to-end convolutional neural network (CNN) framework that takes face videos as training inputs, models each of them as a unified representation by temporal feature pooling operation, and finally projects the high-dimensional representations of both frames and videos into Hamming space to generate binary codes. In such Hamming space, distance of dissimilar pairs is larger than that of similar pairs by a margin. To this end, a novel bounded triplet hashing loss is elaborately designed, which takes all dissimilar pairs into consideration for each anchor point in a mini-batch, and the optimization of the loss function is smoother and more stable. Extensive experiments on challenging video face databases and general image/video datasets with comparison to the state-of-the-arts verify the effectiveness of our method in different kinds of retrieval scenarios.

1. Introduction

In the last decades, with the great popularization of video recording devices and rapid development of the Internet techniques, massive video data are being created, stored, shared and transmitted every day. Meanwhile, the explosive increase of such data also leads to many potential applications in the fields of computer vision and pattern recognition, such as semantic video indexing and retrieval [1–3], film character identification [4], video action recognition [5], person re-identification [6,7], near-duplicate or similar video retrieval [8,9], etc. Amongst the huge amount of videos, a substantial portion of them are human-centered. Therefore, a typical scenario in our daily life could be that given one query of a particular person, retrieve the shots containing him or her [10]. Fig. 1 shows a case of the face video retrieval problem, we input one face frame or sequence as query, inspect each sequence in the database and return the ones which have the same class label with the query. Such research area is very promising that we can find many real-world applications, such as 'intelligent fast-forwards' - where the video jumps to the next shot containing the specific actor, retrieval of all the shots containing a particular family member from thousands of short videos, and locating and tracking criminal suspects from masses of surveillance videos [11].

While face video retrieval is in great demand, there still exists many challenges to be handled. As shown in Fig. 1, the video data usually have large intra-class variations and low image quality influenced by illumination, pose, expressions, resolution and occlusion. Fortunately, we can obtain multiple frames of one subject simultaneously owing to the sequential characteristics of video data. In this way, complementary information from different frames is provided. Therefore, dealing with each video as a whole and trying to aggregate a comprehensive representation for the video would be more favorable. To address this issue, a typical class of video-based face recognition methods [12–21] put the sequential dynamic information of video aside, and simply treat the video as a set of images (*i.e.* frames) and then formulate the problem as image set classification. Given hand-crafted image features, these methods model image set as linear/affine subspace [12–15], nonlinear manifold [16,17], or second-order statistic [18,19]. While these methods can capture the intrinsic geometrical structure of video representations from different perspectives, the hand-crafted fea-

* Corresponding author at: Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China.
E-mail address: wangruiping@ict.ac.cn (R. Wang).

https://doi.org/10.1016/j.patcog.2020.107754
0031-3203/

Fig. 1. Illustration of face video retrieval problem and the motivation of our method. The symbols Q, A, B and C denote four face clips, and colors surrounded them mean different subjects. Given the query Q (Q can be a sequence here or only a frame), we search the database to find the most relevant face clips according to their similarity to Q. Large variations of the video data can be found here, e.g. variation of illumination in B, pose in Q, expression in C and occlusion in A. Our method aims to project them into a discriminative Hamming space where distance of dissimilar pairs is larger than that of similar pairs by a margin.

tures used in most of them cannot well cope with the challenging intra-class variations. Besides, these methods usually involve complicated matrix operations (e.g. Gaussian kernel, matrix logarithm, etc), which are time and memory consuming and thus not be quite qualified for large scale and real-time retrieval task. To overcome such limitations, our conference work [22] makes an early attempt to devise an end-to-end CNN architecture to learn image features and effective video modeling jointly for video-to-video retrieval task. In this paper, we further improve the previous video modeling module and extend the framework to more retrieval scenarios.

To be specific, in the video modeling procedure, we propose two temporal feature pooling schemes, i.e. the temporal max-pooling and the weighted temporal average-pooling scheme, considering the characteristics of face videos from two perspectives. On one hand, each frame of one video may capture only partial but complementary information (e.g., faces with different pose). Since the activations of the convolution filter can be regarded as the confidence measurements of some local concepts, when these concepts only exist on one or a few frames, large activations will only appear on some of these frames. Therefore, the complementary information can be extracted via preserving only the maximum activation for each local concept among frames within a video. On the other hand, different faces within a video can have different image quality (e.g., faces with different illumination), and thus lead to different contributions to the identification of the face. In this way, the video representation can be obtained via weighted average-pooling across frames according to their "quality" (e.g., their similarities to the class centers in this paper).

Another challenge of current retrieval task is the high time and memory cost resulting from high-dimensional real-valued representations of deep features, especially in the large scale data scenarios. To handle such problem, we note that the popular hashing method for solving approximate nearest neighbor (ANN) search problem is qualified for current task. In general, hashing aims to transform the data in high-dimensional space to a low-dimensional and binary representations, or equivalently a short code consisting of a sequence of bits [23]. Therefore, after the video modeling module, we further resort to the hash learning technique to obtain the compact binary codes for videos and frames, which we name as Deep Video Code (*DVC*).

Fig. 2 shows the framework of our method. It takes face videos as training inputs and extracts convolutional features for each frame simultaneously. In order to aggregate useful information from frames and obtain a unified real-valued representation for each video firstly, we utilize either of the two aforementioned video pooling schemes. With such representations, the hashing module finally generates compact binary codes for both videos and frames. To guide the training of such deep hashing network, various supervised signals have been proposed, [24–26]. Among them, triplet loss [27] introduces the relative local ordinal embedding constraints, which is more tally with the retrieval task (i.e. the goal of ranking). However, due to randomly sampling scheme in mini-batch construction, the gradients are not stable and smooth. To address this issue, we turn to optimize a smooth upper bound on the loss function inspired by a recent metric learning method [28]. Specifically, instead of sampling particular triplets (e.g. hard or semi-hard), each positive pair compares against all negative pairs in the batch weighted by the margin violation degree. Extensive evaluations of video modeling, loss function, training methodologies and comparison with the state-of-the-arts on several challenging video (face) and image datasets verify the effectiveness of the proposed framework for (face) video retrieval tasks in scenarios of (cross-scene) video-to-video, image-to-video, video-to-image and image-to-image.

This journal paper builds on the earlier conference work [22]. Compared with it, this paper has made three major extensions. **First**, we improve the inferior temporal average-pooling for *video modeling* by weighted average pooling, which is expected to achieve better performance for this task. **Second**, we generalize the framework to be compatible with not only the previous video-to-video retrieval but also the more practical *image-to-video, video-to-image* and *image-to-image* scenario. With such new framework, additional experiments in different retrieval scenarios on the more challenging *YouTube Celebrities* and *UMDFaces* face video datasets, *JHMDB* action video dataset, and *CIFAR-10* image dataset compared with *more state-of-the-arts* are conducted. **Third**, we provide more detailed comparisons and discussions regarding different *loss functions, inference time* and *training methodologies* including the second-order sampling scheme to alleviate the problem of imbalanced datasets, fine-tuning pre-trained single image model and short code length model to speed up convergence and improve the performance of long code length video model.

2. Related work

Our method aims to model video as a whole and reduce the dimension of deep feature embedding via learning to hash. Hence, in this section, we review works including face video retrieval using

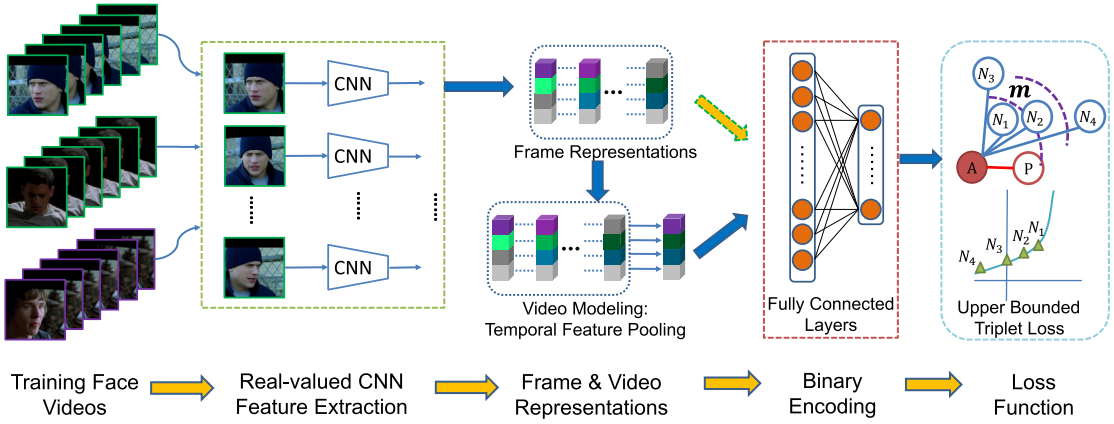


Fig. 2. Framework of the proposed DVC method. Taking face videos with their class labels as training inputs, DVC first extracts convolutional features for each frame and then utilizes temporal feature pooling on all frames belonging to the same video to produce video-level representation. Finally, the fully connected layers project the features of both videos and frames from the same high-dimensional Euclidean space into a much lower-dimensional Hamming space, using the elaborately designed upper bounded triplet loss function where each positive pair compares against all negative pairs in the batch weighted by the margin violation degree.

real-valued representations, video based face recognition, hashing methods and deep feature embedding.

2.1. Face video retrieval

Retrieving videos of a particular person has many important applications. Recently it has been attracting more and more attentions on the study of such problem [10,11,29–37]. [29,30] built an end-to-end video shots retrieval system using several feature-length films. To overcome the large variations within a video, they proposed a cascade of processing steps to obtain the signature image as the representative of the video. However, the rich information involved in different frames were not used sufficiently. As we have discussed in Section 1, videos provide multiple complementary frames with different information, mining such information helps to learn more comprehensive representation of the video. [11] thus made a progress to this research direction, which models each face video as distributions of histograms and measures them by chi-square distance. [32] further used the popular Fisher Vector (FV) [19] as the video representation and achieved better retrieval performance. These pioneer works built fundamental processing pipelines and retrieval task definition on the face video data. However, they are based on real-valued video representations on simple film data, which are not time and space efficient, especially in the large scale scenario nowadays. Instead, we mainly focus on the deephash learning framework, which is qualified for the efficient retrieval task, and is expected to have potential wide applications in larger scale retrieval tasks.

2.2. Video-based face recognition

Video-based face recognition is closely related to our current task considering the common form of data (i.e. face videos) they are dealing with. Recent years have witnessed increasing works on video-based face recognition. Among them, a typical class of methods simply treated the problem as image set (formed by frames) classification and focused on modeling image set with different representations and measuring their similarities. Based on image set modeling manners, these methods can be briefly divided into three categories: subspace based [12–15], manifold based [16,17] and statistic based methods [18,19]. Linear/affine subspace based methods assume image sets are spanned by a linear or affine subspace. When the size and variations of image sets are large, it would be hard for subspace to model such nonlinear structure. To address this limitation, manifold based methods use nonlinear manifold to model image sets and geodesic distance to compare sets. Statistic of image sets is another alternative for nonlinear modeling, which employ covariance [18] or Gaussian Mixture model (GMM) [19] to characterize the second-order variations among images within a set.

These early works mainly focused on effective video modeling while made little effort to frame-level feature extraction by merely relying on hand-crafted features, which are unfavorable of handling challenging image variations. [20,21,38,39] thus made attempts to integrate deep image feature learning and video modeling in an end-to-end neural network, and the performance are improved largely, verifying the usefulness of joint learning of image representation and feature aggregation. However, most of these methods including both shallow and most deep ones usually involve complicated matrix operations (e.g. Gaussian kernel, matrix logarithm, etc), which are time and memory consuming and thus not be quite qualified for large scale and real-time retrieval task in this paper.

2.3. Hashing methods

Benefiting from its compact representations and efficient distance measurements compared with real-valued methods such as [40–42], hashing has been widely applied in the retrieval area especially for large-scale approximate nearest neighbor (ANN) search problem. The early hashing studies aimed to learn hashing functions in data-independent manner, such as the pioneering work Locality Sensitive Hashing (LSH) [43]. However, the performance of these methods is not well when the code length is short. To overcome such limitations, data-dependent methods were proposed to learn similarity-preserving and compact binary codes using training data. Such method can be further categorized into unsupervised [44–47] and (semi-)supervised ones [24–26,46,48–64]. Generally speaking, supervised methods achieve better performance than unsupervised ones owing to strong semantic constraints. Besides, the end-to-end trained deep hashing [24–26,56–62,64] usually outperform shallow ones due to the powerful nonlinear ability of deep neural networks. These deep hashing methods can be briefly classified into three categories according to their supervised signals, i.e. point-wise [24,58](i.e. softmax based), pairwise [25,56,59,64] and triplet ones [26,57,60–62]. Review of the three embedding constraints can be found in Section 2.4.

Recently increasing hashing methods have been proposed to handle the (face) video retrieval problem [1–3,22,33–37]. [33–35] were the early works which proposed to compress face video

into compact binary code by means of learning to hash. However, the isolation of fixed feature representation and hash coding in these works limits their performance. [36,37] thus made an early attempt to learn the CNN features and hashing codes jointly for each frame as similarly done in the deep hashing methods for image retrieval. Since their leaning did not make full use of correlation information provided by multiple frames, the ordinary CNN network framework seems not an optimal solution to efficient face video hashing. To overcome such limitations in face video retrieval problem, our conference work [22] proposed to devise an end-to-end framework to learn feature representations, video modeling and binary codes simultaneously. In the following, [1–3] further studied general video retrieval task with similar framework. Besides, they proposed more complicated video modeling schemes, i.e subspace clustering in [3], vector of locally aggregated descriptors in [1] and temporal feature fusion in [2]. Though the temporal feature pooling schemes used in our framework is simple, they are efficient and effective enough for the specific face video retrieval problem. In addition to the proposed video modeling schemes, to ensure binary codes discriminative, we also propose to optimize a smooth upper bound on the triplet loss function, which is expected to be more stable and achieve better performance.

2.4. Deep feature embedding

Deep feature embedding directly learns non-linear mapping function from the input data to a real-valued embedding space with the powerful deep neural networks and elaborately designed supervised signals, the final objective of which is to push dissimilar data far away and pull similar data close. One of the most successful supervised loss is the softmax classification loss [65–68]. The softmax loss aims at minimizing the cross entropy between images and its belonging category label. On the one hand, traditional softmax loss only considers to make features of different classes be separable, ignoring the generalization ability of embedding for unseen data. On the other hand, it measures similarities using Euclidean distance, ignoring the effects of feature norm, which may be not accurate. To address the first issue, [67] combines softmax loss with center loss which additionally contraints intra-class compactness. As for the second issue, [68] and [66] constraint learned features to be discriminative on a hypersphere manifold using the cosine distance. Though these softmax loss based methods have achieved remarkable success, they suffer from large scale of class numbers, i.e. complexity of classifiers are positively related to the number of categories.

Alternative to learn discriminative embedding is the research of deep metric learning based methods [27,28,69–72]. The most prominent loss functions are the siamese networks with contrastive loss [69] and triplet loss [27]. The contrastive loss directly pulls samples of the same class as close as possible, and pushes samples of different classes far away than a margin. The intra-class constraint of the contrastive loss may be too strict to be satisfied, especially in the case of large scale of data. The triplet loss introduces the relative distance to relax the contrastive formulation, which allows samples to move more freely once the relative margin is kept. Though triplet loss is more ideal for large scale discriminative space learning, the success of which is largely dependent on an effective triplet sampling strategy due to the mini-batch training manner. Generally, samples are randomly selected to construct a mini-batch, leading to slow and even failed convergence since few effective triplets in the batch can provide gradients with large magnitude. To address this issue, [70] adopts the semi-hard negative mining scheme with the goal of finding negative pairs that are within the margin. [71] learns the proxies as part of the network parameters to reduce the generated triplets number within mini-batch and thus speed up the convergence of models. [72] proposes an adaptive controller which automatically adjusts the sampling hyper-parameters by monitoring training performance. Different from designing an elaborately sampling scheme, [28] introduces the structured triplet loss, where each positive pair compares against all negative pairs in the batch weighted by the margin violation degree. By doing so, the gradients would become smoother and the model would converge more easily. In light of this, we introduce such structured triplet loss to the hash learning area which is expected to mitigate the slow convergence problem limited by the binary constraints in deep hashing [25].

Softmax based and metric learning based methods are two alternative supervised signals for deep feature embedding, advantages and drawbacks of them are compared and analyzed in [73], readers can make a reference to it. In the experimental section, we also compare some alternative designs of them in our hashing framework.

3. Approach

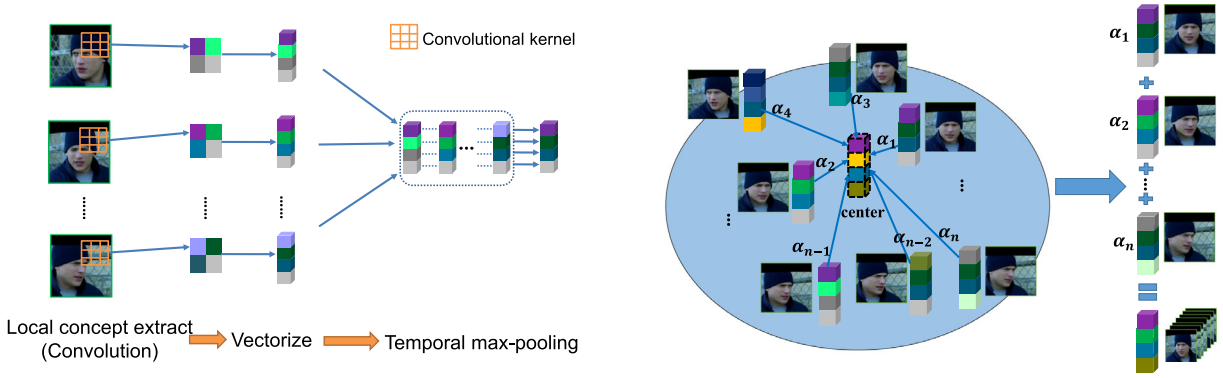
Our goal is to learn compact binary codes for face videos and frames such that: (a) each video should be treated as a whole, i.e., learn a single binary code for each video; (b) the binary codes should be similarity-preserving, i.e., the Hamming distance between similar faces (videos or frames) should be smaller than that between dissimilar ones by a margin. To fulfill the task, as demonstrated in Fig. 2, our method mainly involves two steps: 1) video modeling, which extracts frame-level features and aggregates them for video-level representation, and 2) binary encoding, which learns the optimal binary codes for videos and frames under the designed upper bounded triplet loss function.

3.1. Video modeling

In this step, our goal is to learn a comprehensive real-valued representation for each face video firstly, using the frame-level CNN features. One straightforward method is to extract frame-level features using a *pre-trained* CNN model first, and then pooling them *offline* (e.g., by average-pooling) to obtain a unified representation. Processing in such *non-end-to-end* manner mainly has two disadvantages. First, it separates the successive steps of frame-level feature extraction and video-level representation, which would lead to poor coupling between the two steps and thus poor performance. Second, it gives equal treatment to all frames, which does not take full consideration of the complementary properties of different frames.

To cope with video data, another widely used solution is the 3D Convolution Neural Network (3D CNN). As introduced in [74], the basic idea of 3D CNN is stacking fixed length of video frames as a cube and then convolving the 3D kernels on the cube. Though 3D CNN has the power of making use of information from multiple frames in videos, it is still not suitable for the face video modeling. The reason is that the main goal of 3D CNN is to capture motion variation in temporal dimension and appearance information in spatial dimension simultaneously, which has high request on the capacity of the network. One embarrassed consequence would be that neither of the learned motion or appearance information is satisfactory. Besides, the temporal motion information among frames makes little sense to the identification of one face video, while the appearance information contributes more to such problem. Therefore, as for the face video retrieval task, it cares more about learning a powerful appearance information extractor and together with an effective video modeling scheme.

To fulfill the purpose of extracting appearance information and modeling video jointly, we devise the end-to-end video representation framework shown in Fig. 2. Let $F = [f_1, f_2, \ldots, f_n]$ be a face



(a) Temporal max-pooling (b) Weighted temporal average-pooling

Fig. 3. Illustrations of (a) temporal max-pooling and (b) weighted temporal average-pooling. In (a), given frame-level convolutional features, the maximum activation of each dimension is computed. Darker colors mean larger activations for corresponding dimensions. In (b), quality of frames within a video is measured according to their distance to the learned class center in the feature space. The final video representation is the weighted summation of frame-level features, where weight of each frame is the computed image quality.

video with n frames, where f_i denotes the ith frame. We first obtain the frame-level CNN features $\mathbf{d}_i \in \mathbb{R}^p$ for the ith frame by propagating the frame through the frontal CNN module, i.e. the stacked operations of convolution, pooling and ReLU non-linear activation. Next, these frame-level features should be aggregated to model the video as a whole. To this end, we mainly design two temporal feature pooling schemes to model face videos.

The first scheme is the temporal max-pooling across frames within a video shown in Fig. 3a. Due to large variations within a video, each frame may carry only partial but complementary information. In addition, the convolution kernel can be regarded as a local concept detector. Therefore when some concepts only exist on one or a few faces in the video, the detectors will only have large responses on some of these faces. Taking this into consideration, we preserve only the maximum activation for each local concept among frames within a video. To be specific, let each dimension of $\mathbf{d}_i$ denote the activation of one local concept, the temporal max-pooling is defined as:

$$\mathbf{v}_j = \max(\mathbf{d}_{1,j}, \mathbf{d}_{2,j}, \ldots, \mathbf{d}_{n,j}). \tag{1}$$

where $\mathbf{v} \in \mathbb{R}^p$ is the representation of face video F, i.e., the result of temporal max-pooling on $[\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_n]$. $\mathbf{v}_j$ is the jth dimension of $\mathbf{v}$.

The second scheme is the weighted temporal average-pooling shown in Fig. 3b. Since different faces within a video can have different image quality and high-quality faces usually contribute more to the identification of the video than the low-quality ones, we should involve more information from those high-quality frames when aggregating frame-level features of a video [20,21]. To this end, we utilize the *cosine* similarity between $\mathbf{d_i}$ and the class center $\mathbf{c}_{\mathbf{y_F}}$ (y_F is the class label of the video F) to measure each frame's quality:

$$s_i = \frac{\mathbf{d_i^T c}_{\mathbf{y_F}}}{||\mathbf{d_i}||_2 \cdot ||\mathbf{c}_{\mathbf{y_F}}||_2}. \tag{2}$$

where s_i can be used as the weight for frame i. To ensure the summation of all frame weights is 1, we further normalize the weight as:

$$\alpha_i = \frac{exp(s_i)}{\sum_{j=1}^{n} exp(s_j)}. \tag{3}$$

Then we can obtain the aggregated representation of video F via the weighted temporal average-pooling, defined as:

$$\mathbf{v} = \sum_{i=1}^{n} \alpha_i \mathbf{d}_i. \tag{4}$$

It is noted that in Eq. (2), the class center $\mathbf{c}_{\mathbf{y_F}}$ needs to be updated during training. Due to the mini-batch training manner in deep learning, it is inefficient and impractical to take the entire training set to update the class centers in each iteration. We thus resort to a recent work for classification [67], and the update of kth class center $\mathbf{c_k}$ in the tth iteration is as:

$$\Delta \mathbf{c}_\mathbf{k}^\mathbf{t} = \frac{\sum_{i=1}^{n_b} \delta(y_i = k) \cdot (\mathbf{c}_\mathbf{k}^\mathbf{t} - \mathbf{d_i})}{1 + \sum_{i=1}^{n_b} \delta(y_i = k)},$$
$$\mathbf{c}_\mathbf{k}^{\mathbf{t+1}} = \mathbf{c}_\mathbf{k}^\mathbf{t} - \beta \Delta \mathbf{c}_\mathbf{k}^\mathbf{t}. \tag{5}$$

where n_b is the total number of frames in a mini-batch, y_i is the class label of the ith frame and the scalar β controls the updating step of the centers.

After training, class centers are fixed. Given a new video without label, we compute the similarities between frames and class centers, and take the most similar center via hard-voting to measure frames' quality in Eq. (2).

3.2. Binary encoding

Due to the demand for time and space efficiency of learned representation for large scale retrieval task, the high-dimensional real-valued features introduced above are not qualified. To address such problem, we propose a hashing method with improved triplet loss function. By doing so, the high-dimensional representation is further projected into a much lower-dimensional Hamming space.

To guarantee the similarity-preserving power of learned hashing functions, several kinds of objectives for deep hashing have been proposed in [25,26,58,64]. Among them, the triplet ranking loss aims to preserve relative rank order among datums, which is desirably consistent with the objective of retrieval task. The triplet constraints can be described as the form: "image i is more similar to image j than to image k" [26]. When the dataset is challenging, such constraints are easier to be satisfied than point-wise [58] or pairwise constraints [25,64]. Apart from that, such form of triplet-based relative similarities are easier to be constructed without request of detailed category-level labels (e.g., for two images with multiple attributes/tags, simply count the number of common attributes/tags as the similarity metric between them). For better understanding of the triplet ranking loss in hash learning, let i, j, k be three samples and i is more similar to j than to k, our goal is to learn a Hamming space where the relative similarity among samples should be preserved, as illustrated in the loss module of Fig. 2.

Otherwise, penalty should be imposed on them, defined by:

$$J_{i,j,k} = \max(0, m + D_h(\mathbf{b}_i, \mathbf{b}_j) - D_h(\mathbf{b}_i, \mathbf{b}_k)).$$
$$s.t. \quad \mathbf{b}_i, \mathbf{b}_j, \mathbf{b}_k \in \{0,1\}^r \tag{6}$$

where $D_h(\cdot)$ denotes the Hamming distance between two binary vectors and $m > 0$ is a margin threshold parameter. $\mathbf{b}_i$, $\mathbf{b}_j$ and $\mathbf{b}_k$ are the r-bit binary codes of sample i, j and k, respectively.

In general, the whole framework is trained with the SGD algorithm in a mini-batch. Due to the random construction of the triplets in the batch, a substantial portion of them contribute little to the convergence of the network during each iteration as they already satisfy the triplet constraint as described in Eq. (6) or their loss is quite small. To cope with this issue, it is better to make use of "difficult" triplets, i.e., given a pair of similar samples, we actively find the dissimilar neighbor closest to them in current Hamming space. Based on this, we rewrite Eq. (6) and give the overall loss function per batch as[1]:

$$J = \frac{1}{2|\widehat{\mathcal{P}}|} \sum_{(i,j)\in\widehat{\mathcal{P}}} \max(0, J_{i,j}),$$
$$J_{i,j} = \max\Big(\max_{(i,k)\in\widehat{\mathcal{N}}} \{m - D_h(\mathbf{b}_i, \mathbf{b}_k)\},$$
$$\max_{(j,l)\in\widehat{\mathcal{N}}} \{m - D_h(\mathbf{b}_j, \mathbf{b}_l)\}\Big) + D_h(\mathbf{b}_i, \mathbf{b}_j).$$
$$s.t. \quad \mathbf{b}_i, \mathbf{b}_j, \mathbf{b}_k, \mathbf{b}_l \in \{0,1\}^r \tag{7}$$

where $\widehat{\mathcal{P}}$ and $\widehat{\mathcal{N}}$ are the set of positive and negative pairs (i.e., similar and dissimilar pairs) in the training mini-batch, respectively. Note that here we allow both sample i and j play the role of anchor point in the triplet structure in order to make full use of samples in the batch.

However, such loss function is non-smooth when optimized with mini-batch sampling caused by the maximum function, which would lead to unstable convergence and bad local optimum. To make more use of the "difficult" triplets and ensure the stability of gradients update, we turn to optimize a smooth upper bound on Eq. (7) inspired by a recent metric learning work [28], defined as:

$$\tilde{J} = \frac{1}{2|\widehat{\mathcal{P}}|} \sum_{(i,j)\in\widehat{\mathcal{P}}} \max(0, \tilde{J}_{i,j}),$$
$$\tilde{J}_{i,j} = \log\Big(\sum_{(i,k)\in\widehat{\mathcal{N}}} \exp\{m - D_h(\mathbf{b}_i, \mathbf{b}_k)\}$$
$$+ \sum_{(j,l)\in\widehat{\mathcal{N}}} \exp\{m - D_h(\mathbf{b}_j, \mathbf{b}_l)\}\Big) + D_h(\mathbf{b}_i, \mathbf{b}_j).$$
$$s.t. \quad \mathbf{b}_i, \mathbf{b}_j, \mathbf{b}_k, \mathbf{b}_l \in \{0,1\}^r \tag{8}$$

From Eq. (8), it can be seen that the triplet loss $\tilde{J}_{i,j}$ takes all dissimilar pairs into consideration rather than the scarce most "difficult" ones in Eq. (7), which makes the overall loss $\tilde{J}$ for each batch smoother. Meanwhile, the exp operator strengthens the contributions of those more "difficult" triplets while weakens other "easier" ones in the summation terms.

It is noted that there exists the binary constraints in Eq. (8). Such constraints require discretizing the real-valued output (e.g., with signum function). Unfortunately, it is intractable to optimize the network via the SGD algorithm with such discrete activation function. For ease of optimization, we replace the Hamming distance $D_h(\cdot)$ with square of Euclidean distance and relax the binary constraints on $\mathbf{b}$ to range constraints. We formulate the relaxed $\tilde{J}$ as follows:

$$\tilde{J} = \frac{1}{2|\widehat{\mathcal{P}}|} \sum_{(i,j)\in\widehat{\mathcal{P}}} \max(0, \tilde{J}_{i,j}),$$
$$\tilde{J}_{i,j} = \log\Big(\sum_{(i,k)\in\widehat{\mathcal{N}}} \exp\{m - D_e^2(\mathbf{b}_i, \mathbf{b}_k)\}$$
$$+ \sum_{(j,l)\in\widehat{\mathcal{N}}} \exp\{m - D_e^2(\mathbf{b}_j, \mathbf{b}_l)\}\Big) + D_e^2(\mathbf{b}_i, \mathbf{b}_j).$$
$$s.t. \quad \mathbf{b}_i, \mathbf{b}_j, \mathbf{b}_k, \mathbf{b}_l \in [0,1]^r \tag{9}$$

where D_e denotes the Euclidean distance and the binary constraints on $\mathbf{b}_i$, $\mathbf{b}_j$, $\mathbf{b}_k$ and $\mathbf{b}_l$ are relaxed to range constraints of 0 to 1.

With Eq. (9), back-propagation algorithm with mini-batch gradient descent method is applied to train the network. Specifically, we give the gradients of Eq. (9) with respect to the relaxed binary vectors as follows:

$$\frac{\partial \tilde{J}}{\partial D_e^2(\mathbf{b}_i, \mathbf{b}_j)} = \frac{1}{2|\widehat{\mathcal{P}}|} \mathbb{1}[\tilde{J}_{i,j} > 0],$$
$$\frac{\partial \tilde{J}}{\partial D_e^2(\mathbf{b}_i, \mathbf{b}_k)} = \frac{1}{2|\widehat{\mathcal{P}}|} \mathbb{1}[\tilde{J}_{i,j} > 0] \frac{-\exp\{m - D_e^2(\mathbf{b}_i, \mathbf{b}_k)\}}{\exp\{\tilde{J}_{i,j} - D_e^2(\mathbf{b}_i, \mathbf{b}_j)\}},$$
$$\frac{\partial \tilde{J}}{\partial D_e^2(\mathbf{b}_j, \mathbf{b}_l)} = \frac{1}{2|\widehat{\mathcal{P}}|} \mathbb{1}[\tilde{J}_{i,j} > 0] \frac{-\exp\{m - D_e^2(\mathbf{b}_j, \mathbf{b}_l)\}}{\exp\{\tilde{J}_{i,j} - D_e^2(\mathbf{b}_i, \mathbf{b}_j)\}},$$
$$\frac{\partial D_e^2(\mathbf{b}_i, \mathbf{b}_j)}{\partial \mathbf{b}_i} = 2(\mathbf{b}_i - \mathbf{b}_j), \quad \frac{\partial D_e^2(\mathbf{b}_i, \mathbf{b}_k)}{\partial \mathbf{b}_i} = 2(\mathbf{b}_i - \mathbf{b}_k),$$
$$\frac{\partial D_e^2(\mathbf{b}_i, \mathbf{b}_j)}{\partial \mathbf{b}_j} = 2(\mathbf{b}_j - \mathbf{b}_i), \quad \frac{\partial D_e^2(\mathbf{b}_j, \mathbf{b}_l)}{\partial \mathbf{b}_j} = 2(\mathbf{b}_j - \mathbf{b}_l),$$
$$\frac{\partial D_e^2(\mathbf{b}_i, \mathbf{b}_k)}{\partial \mathbf{b}_k} = 2(\mathbf{b}_k - \mathbf{b}_i), \quad \frac{\partial D_e^2(\mathbf{b}_j, \mathbf{b}_l)}{\partial \mathbf{b}_l} = 2(\mathbf{b}_l - \mathbf{b}_j). \tag{10}$$

where, $\mathbb{1}[\cdot]$ is the indicator function which equals 1 if the expression in the bracket is true and 0 otherwise. As shown in Eq. (10), the gradients of each iteration contain all negative pairs' information which makes the optimization more stable.

With such computed gradients over mini-batches, the rest of back-propagation can be run in standard manner.

3.3. Implementation details

The proposed DVC method is implemented on Caffe platform.[2] The CNN feature extraction module in Fig. 2 can be any stacked convolutional blocks, considering the difficulty of different datasets, we design two baseline architectures, i.e., the *shallow* one and the *deep* one (detailed architectures can be found in the supplementary materials). During training, we set momentum to 0.9 and adopt the fixed learning rate policy 10^{-4}. The margin m in Eq. (9) is empirically set to 1. Other hyper-parameters setting is shown in Table 1. The pre-trained model for initializing the *deep* architecture is pre-trained for face recognition task on CASIA Webface dataset [75]. To speed up the training and to handle the imbalanced categories distribution, we leverage some training methodologies which are introduced and verified in the supplementary materials.

4. Experiments

In this section, we first give ablation studies of the framework including different video modeling schemes discussed in Section 3.1. Next, comparisons with state-of-the-art methods on

[1] Here either of $\mathbf{b_i}$ or $\mathbf{b_j}$ could be the anchor point.

[2] The source code of DVC is available at https://github.com/greatmanqss/DVC.

Table 1
Training hyper-parameters setting for different CNN architectures.

Network	Batch size (videos #)	Weight decay	Initialization	Iterations
shallow	100	0.004	"Xavier"[76]	30,000
deep	30	0.0005	pre-trained model	50,000

different retrieval scenarios are conducted to illustrate the advantages of the proposed method. Besides, alternative loss designs discussed in Section 2.4 and the effects of frame selection during inference are also studied in the supplementary materials.

4.1. Datasets and evaluation protocols

Datasets. The following face video retrieval experiments in this paper are conducted on the challenging ICT-TV dataset [34] and YouTube Celebrities (YTC) dataset [77]. ICT-TV dataset contains two face video collections clipped from two popular American TV-Series, i.e., the Big Bang Theory (BBT) and Prison Break (PB). These two TV-Series are quite different in their filming style and therefore pose different challenges. BBT is a sitcom and most stories take place indoors. Each episode contains 5∼8 characters. By contrast, many scenes of PB are taken outdoors with a main cast list around 19 characters. Consequently face videos from PB have larger variation of illumination. All the face videos are extracted from the whole first season of each TV-Series, i.e., 17 episodes of BBT and 22 episodes of PB. The numbers of face videos of these two datasets are 4667 and 9435 respectively. YTC is a widely studied and challenging benchmark containing 1,910 video clips of 47 celebrities collected from YouTube. Specifically, these clips are parsed from three raw videos of each celebrity which have large variations recorded in different scenes. Besides, the video frames are mostly highly compressed, with low resolution and large intra-class variations. Some video frames of the three datasets can be found in our supplementary materials. Considering the difficulty of datasets, we use the *shallow* network for BBT and PB, and the *deep* one for YTC in the following.

Training and test protocols. For BBT and PB, with the "unknown" class abandoned, we randomly select $\frac{2}{3}$ of face videos for each character for training and leave the rest for test. For YTC, we treat samples from two of the three raw videos of each celebrity as training set, and leave the remaining ones as test set. To ensure enough videos in a mini-batch with limited memory space, we divide each long video clips into multiple sub-video clips with fixed length of 10 frames for BBT and PB, and varied length but less than 30 frames for YTC. By doing this, we obtain 25,590, 40,941 and 7,190 video clips as training set, and 12,735, 20,357 and 3,101 video clips as test set for BBT, PB and YTC respectively.

Measurements. We evaluate methods via utilizing videos (or middle frame for retrieval across image and video[3]) in the test set as query to retrieve the other videos in the test set as database. For quantitative evaluation, mean Average Precision (mAP) and precision recall curves are computed on three datasets.

4.2. Ablation study

Evaluation of video modeling. In this part, we validate the effectiveness of the proposed temporal feature pooling for video modeling on the three datasets. We use the same hashing objective function in Eq. (9) and mainly evaluate different video modeling schemes discussed in Section 3.1, i.e. 1) **DVC-s**: single-frame model which treats each frame as an individual sample and does not involve any video modeling operation; 2) **DVC-3D**: 3D CNN model which conducts 3D convolution across the videos; 3) **DVC-a**: temporal average pooling model (a special case of the weighted temporal average pooling with equal attention for each frame); 4) our **DVC-wa**: weighted temporal average-pooling model formulated in Eq. (4); and 5) our **DVC-m**: temporal max-pooling model defined in Eq. (1). The network architectures of **DVC-s, DVC-a, DVC-wa**, and **DVC-m** have been discussed in Section 3.3. **DVC-3D**'s network configuration is similar to them except that the convolution kernels of the first convolution layer are $3 \times 3 \times 3T$ where T is the number of frames of a face video, each frame has 3 channels and totally $3T$ channels in temporal dimension.[4] To obtain a unified binary code for each face video, we simply average the codes of the segmented sub-videos for **DVC-3D, DVC-a, DVC-wa** and **DVC-m**, or all frames for **DVC-s** belonging to that face video.

Table 2
mAP comparison of video modeling schemes with 12-bit binary codes.

	DVC-3D	DVC-s	DVC-a	DVC-wa	DVC-m
BBT	0.9759	0.9853	0.9791	**0.9876**	0.9808
PB	0.8573	0.9547	0.9552	**0.9605**	0.9590
YTC	–	0.6423	0.6557	0.6609	**0.6719**

The mAP results are listed in Table 2. There are three conclusions we can reach: (1). **DVC-3D** does not perform well on both BBT and PB. The reason is that **DVC-3D** is tailored to action recognition task [74]. The main goal of **DVC-3D** is to learn both spatial and temporal information, which has high request on the network capacity. Besides, it abstracts the spatial appearance information of multiple frames together in the first convolution layer which causes the network to fail to capture the appearance information of each frame in the latter layers. Therefore, the appearance information of the face is not learned very well. By contrast, the **DVC-s** only aims to learn the appearance information for face images, and the performance is better than **DVC-3D** for this task, demonstrates the significance of appearance information for face video retrieval task. (2) Our proposed **DVC-wa** and **DVC-m** achieve the best performance on all the three datasets, which demonstrates the effectiveness of video modeling via either extracting complementary local concepts from frames or paying different attentions on frames with different quality. Moreover, the advantages of **DVC-wa** and **DVC-m** over other alternatives become larger with datasets being more challenging. The reason is that from BBT to PB and then to YTC, samples tend to have larger intra-class variations and worse imaging condition (the more frequent case in real world videos), leading to information carried by different frames more complementary and the decrease of quality of some frames. Since the proposed temporal feature pooling operations are just designed to handle such problems, and they achieve expected performance. (3) **DVC-wa** performs better than **DVC-a**. To further analyze the proposed weighted temporal average-pooling, we show some frames with different attention weights of some video clips in YTC dataset in Fig. 4. It is observed frames with large attention weights usually have better image quality while those with small attention weights present bad image quality caused by large pose, motion blur, etc.

[3] Unless particularly stated, binary codes for frames are not optimized and experimental results are reported for video-to-video retrieval task.

[4] Since length of processed video clips of YTC is varied as introduced in Section 4.1, **DVC-3D** cannot be applied on YTC.

Fig. 4. Video frames with different attention weights on YTC dataset. The left and right two rows belong to one video clip. The first row shows frames with large attentions and second row shows those with small attentions. It is observed the attention weights are assigned reasonably according to different image quality cased by pose, motion blur, etc.

This demonstrates the advantage of involving frames information with different attentions according to their image quality, i.e. their distance to the class centers.

4.3. Comparison with the state-of-the-art

Comparative methods: We compare our DVC with several state-of-the-art hashing methods in recent years, including LSH [43], SH [44], BRE [48], ITQ [46], CCA-ITQ [46], MLH [50], KSH [51], CVC [33], DLBHC [58], DNNH [26], DSH [25], HashNet [64] and SSDH [24]. Strictly speaking, these compared methods except CVC are not specifically designed for face video retrieval task. To conduct face video retrieval experiments with them, as similarly done in **DVC-s**, we trained such methods by treating each frame as a sample and finally all the frame-level binary representations are fused by hard-voting method as the representation of the face video. For fair comparison, DLBHC, DNNH, DSH, HashNet and SSDH used the same network architecture as the **DVC-s**. Please note that in this case, the DNNH actually is the fully connected version described in [26]. For evaluating DVC comprehensively, we test **DVC-s, DVC-m** and **DVC-wa** on all code lengths and datasets.

Training settings: Due to the limitation of memory, we cannot feed all training data to all compared methods. Hence we have to randomly select 5K and 10K frames from all training face videos for MLH and KSH respectively, which costs more than 8GB of memory. Parameters of the compared methods are all set based on the authors' suggestions in their original publications. On BBT and PB datasets, we use the extracted block discrete cosine transformation features of face images as used in [33] for all non-deep methods, *i.e*, methods using hand-crafted features. On YTC dataset, all non-deep methods utilize the image representations of the last pooling layer (pool5) in of the pre-trained face recognition model that we used for initializing DVC. Besides, all deep hashing methods including DLBHC, DNNH, DSH, HashNet, SSDH and our DVC use the finetuning trick that has been proved effective in the supplementary materials.

Results: Table 3 shows the retrieval performance comparison on the three datasets with different code lengths. The precision recall curves can be found in the supplementary materials. In general, supervised hashing methods perform better than unsupervised methods, validating the importance of label information for learning similarity-preserving Hamming space. In addition, those deep hashing methods outperform non-deep ones with hand-crafted features by a large margin on BBT and PB datasets, while the gap becomes small when the non-deep methods are equipped with deep CNN features on YTC dataset, demonstrating the advantage of CNN for image feature learning. We also train some conventional hashing methods with CNN features on BBT and PB, their performance is improved significantly, but still inferior to our DVC. Details are introduced in the supplementary materials.

Apart from above observations, we can see that performance of compared deep hashing methods on BBT is very close to ours because face videos in BBT have small intra-class variations. However, when it comes to PB and YTC which are more challenging than BBT as described in Section 4.1, the performance gap between our method and others becomes larger. Such advantages benefit from the optimized smooth loss function and video modeling schemes. On one hand, DVC leverages the triplet ranking constraints to optimize the local rank among samples, which is more suitable to the preservation of the semantic similarity on datasets with larger intra-class variations. Meanwhile, the smooth upper bound on the loss function which takes all negative pairs into consideration and biases towards triplets with large loss also leads the network to converge stably. By contrast, DLBHC and SSDH focuses on the final classification loss which neglects the relative similarity among samples, thus encoding dissimilar images to similar codes would not be punished as long as the classification accuracy is unaffected. DSH and HashNet are based on pairwise discriminative analysis which may not work well in large scale settings. Though the objective of DNNH is similar to ours, it gives equal treatment to all the triplets in the batch and overwhelms the contributions of tirplets that violate the constraints. As a result, it may converge to an undesired local optimal. Benefitting from the designed hashing objective, our **DVC-s** outperforms these competing deep hashing methods. On the other hand, **DVC-m** and **DVC-wa** utilize the temporal pooling network to represent face video which has the ability of making use of more complementary information and paying more attention on frames close to the class center, while compared five deep methods and **DVC-s** simply average all frame-level binary codes, the final video code is unreliable when faces have large variations in the video. Therefore, **DVC-m** and **DVC-wa** achieve the expected better performance on the challenging PB and YTC datasets.

4.4. More retrieval scenarios

Retrieval across image and video. Apart from the video-to-video retrieval task evaluated above, using video frame to retrieve video clips, and using video clips to retrieve images, i.e. the retrieval across image and video, is another popular retrieval application. Therefore, we further compared our DVC with several competitive hashing methods standing out from Table 3 for such task on the more challenging PB and YTC datasets. To be specific, binary codes of both video frames and videos for DVC are optimized in a common Hamming space, as shown in Fig. 2. Results are shown in Table 4 (video-to-image task can be found in the supplementary materials). It is observed our method significantly outperforms other methods on such task, validating pretty expansibility of our method on different retrieval tasks.

Retrieval across video scenes. In aforementioned experiments, the query and gallery face clips may come from the same raw videos. Another practical retrieval scenario is the cross scene linking, where potentially video of one scene is matched with the ones of other scenes. As we have introduced in Section 4.1, samples of YTC in the training set and test set are from different

Table 3
Comparison of retrieval mAP between our DVC method and the other hashing methods on different datasets.

Method	BBT			PB			YTC		
	12-bit	24-bit	48-bit	12-bit	24-bit	48-bit	12-bit	24-bit	48-bit
LSH [43]	0.2778	0.3062	0.3679	0.1259	0.1360	0.1412	0.4024	0.6131	0.7813
SH [44]	0.3745	0.3652	0.3329	0.1403	0.1496	0.1504	0.4514	0.6380	0.7547
ITQ [46]	0.4771	0.4928	0.4968	0.1414	0.1525	0.1608	0.5665	0.7701	0.8460
CCA-ITQ [46]	0.7159	0.8141	0.8547	0.1819	0.2312	0.2814	0.5604	0.7703	0.8481
BRE [48]	0.4275	0.4810	0.4860	0.1423	0.1468	0.1510	0.5242	0.7356	0.8363
MLH [50]	0.7670	0.8058	0.8402	0.2294	0.2325	0.2783	0.5153	0.7352	0.8586
KSH [51]	0.8819	0.8830	0.8856	0.3405	0.3840	0.4086	0.5840	0.7605	0.8635
CVC [33]	0.7784	0.8121	0.8166	0.2767	0.3314	0.3648	0.5345	0.6679	0.7697
DLBHC [58]	0.9870	0.9914	0.9922	0.9476	0.9498	0.9602	0.6096	0.7244	0.7684
DNNH [26]	**0.9878**	0.9884	0.9909	0.9262	0.9306	0.9262	0.6273	0.7501	0.8356
DSH [25]	0.9858	0.9850	0.9845	0.9414	0.9618	0.9652	0.6368	0.7700	0.8358
HashNet [64]	0.9746	0.9849	0.9846	0.9234	0.9588	0.9658	0.5874	0.7742	0.8300
SSDH [24]	0.9849	0.9877	0.9909	0.9327	0.9474	0.9590	0.5962	0.7097	0.8364
DVC-s	0.9853	**0.9933**	**0.9941**	0.9547	0.9663	**0.9788**	0.6423	0.7842	0.8535
DVC-wa	0.9876	0.9916	0.9909	**0.9605**	0.9657	0.9748	0.6609	**0.7868**	0.8605
DVC-m	0.9808	0.9926	0.9915	0.9590	**0.9707**	0.9727	**0.6950**	0.7828	**0.8701**

Table 4
Comparison of image-to-video retrieval mAP between our DVC method and competitive hashing methods on different datasets.

Method	PB			YTC		
	12-bit	24-bit	48-bit	12-bit	24-bit	48-bit
CCA-ITQ [46]	0.8037	0.8991	0.9093	0.4643	0.6697	0.7810
KSH [51]	0.9019	0.9239	0.9371	0.4957	0.6493	0.7844
DLBHC [58]	0.9378	0.9419	0.9522	0.5389	0.6486	0.6935
DNNH [26]	0.9100	0.9161	0.9116	0.5504	0.6669	0.7693
DSH [25]	0.9272	0.9484	0.9593	0.5535	0.6960	0.7792
HashNet [64]	0.9119	0.9429	0.9556	0.5181	0.6974	0.7687
SSDH [24]	0.9220	0.9355	0.9489	0.5402	0.6476	0.7857
DVC	**0.9451**	**0.9569**	**0.9703**	**0.6046**	**0.7214**	**0.8060**

Table 5
mAP comparison of video-to-video retrieval across shooting scenes on different datasets.

Method	UMDFaces			YTC		
	12-bit	24-bit	48-bit	12-bit	24-bit	48-bit
DLBHC [58]	0.3632	0.5194	0.6155	0.5531	0.6370	0.5674
DNNH [26]	0.3429	0.4722	0.5606	0.5189	0.5869	0.6251
DSH [25]	0.3426	0.4252	0.4394	0.5101	0.5832	0.5766
HashNet [64]	0.3178	0.4587	0.5571	0.4340	0.5637	0.6446
SSDH [24]	0.4720	0.4771	0.5571	0.5486	0.5639	0.6475
DVC-s	0.3867	0.6252	0.7176	0.5608	0.6488	0.6822
DVC-m	**0.5134**	**0.6838**	**0.7329**	**0.6237**	**0.6512**	**0.6852**

raw videos which are shot in different scenes. Besides, YTC is highly compressed with low resolution and synthetically deteriorated. Apart from YTC, we notice another recently released large scale face video dataset called UMDFaces [78] whose face clips are also parsed from several raw video scenes. We randomly select 200 subjects for experimental evaluation, where 70% raw videos of each subject are treated as training set and the remaining ones are left as test set. To simulate the cross-scene use case, we use the video clips in test set as query to retrieve the training set as gallery. Comparison results of ours with competitive deep hashing methods are show in Table 5. It is observed that this retrieval scenario is more difficult than that in Table 3 (e.g. on YTC, the mAP drops almost 20 percentages under 48-bit code length). Even so, our DVC still achieves the best results on all bit lengths in such use case.

General video retrieval. Our DVC method is designed for mining complementary information from videos, which is not limited in face video data. Moreover, the proposed loss function in hash learning is also a general one. Therefore, our framework can be transferred to general image and video retrieval without many efforts. In this part, we validate the effectiveness of the whole framework on action video retrieval task. Specifically, for action video retrieval task, experiments are conducted on JHMDB dataset [79], adopting the same network backbone and training/evaluation settings with DVH [2]. The mAP results are shown in Table 6.[5] We can see that our method achieves significant advantages over state-of-the-arts hashing methods on such retrieval task, validating the effectiveness of the proposed loss function and video modeling scheme. We also verify the proposed hash learning objective on image retrieval task which is introduced in our supplementary materials.

Table 6
Retrieval mAP on JHMDB dataset for general video-to-video retrieval task.

Method	16-bit	32-bit	64-bit
PCAH [49]	0.1689	0.1764	0.1830
ITQ [46]	0.1380	0.1416	0.1444
AGH [45]	0.1374	0.1430	0.1690
CCA-ITQ [46]	0.2720	0.3196	0.3144
KSH [51]	0.2750	0.2832	0.3351
FastHash [62]	0.3119	0.3372	0.3663
SSDH [24]	0.3487	0.3744	0.3803
DVC-s	0.3651	0.3721	0.3727
DVH [2]	0.3519	0.3743	0.3795
DVC-a	**0.4611**	**0.4718**	**0.4672**

5. Conclusions

In this paper, we propose a deep video code (DVC) framework via the end-to-end CNN architecture, which takes face videos as inputs and outputs compact binary codes. The learned DVC achieves promising performance in comparison to state-of-the-art hashing methods on three challenging face video datasets and two general image/video datasets for different imagen and video retrieval task. We owe it to three aspects: **First**, the integration of frame-level feature learning, video-level modeling and hash coding into a unified framework, which makes the three stages compatible to each other. **Second**, the optimization of a smooth upper bound on triplet loss function for hash learning avoids model converging unstably or falling into bad local optimal. **Third**, the

[5] mAP of DVH and methods above SSDH in the table is referred to DVH, and others are reproduced by ourselves using public released source codes.

designed video modeling schemes, i.e., the temporal max-pooling can mine complementary information from different frames and the weighted temporal average pooling aggregates video information via the elaborately devised attention mechanism. Currently, the video modeling in our method only involves the first-order statistic of the image set. To further fuse higher-order statistic like the second-order pooling would be one of our future research directions. Besides, we believe that constructing a larger and more challenging face video dataset to study and evaluate more complicated end-to-end frameworks is also in great demand.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work is partially supported by Natural Science Foundation of China under contracts Nos. 61922080, U19B2036, 61772500, CAS Frontier Science Key Research Project No. QYZDJ-SSWJSC009, and Beijing Academy of Artificial Intelligence No. BAAI2020ZJ0201.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2020.107754.

References

[1] J. Feng, S. Karaman, S. Chang, Deep image set hashing, in: Proc. WACV, 2017, pp. 1241–1250.
[2] V.E. Liong, J. Lu, Y. Tan, J. Zhou, Deep video hashing, IEEE Trans. Multimed. 19 (6) (2017) 1209–1219.
[3] Z. Chen, J. Lu, J. Feng, J. Zhou, Nonlinear structural hashing for scalable video search, IEEE Trans. Circuits Syst.Video Technol. 28 (6) (2018) 1421–1433.
[4] Y. Zhang, C. Xu, H. Lu, Y. Huang, Character identification in feature-length films using global face-name matching, IEEE Trans. Multimed. 11 (7) (2009) 1276–1288.
[5] P. Wang, L. Liu, C. Shen, H.T. Shen, Order-aware convolutional pooling for video based action recognition, Pattern Recognit. 91 (2019) 357–365.
[6] X. Ma, X. Zhu, S. Gong, X. Xie, J. Hu, K. Lam, Y. Zhong, Person re-identification by unsupervised video matching, Pattern Recognit. 65 (2017) 197–210.
[7] J. Meng, A. Wu, W. Zheng, Deep asymmetric video-based person re-identification, Pattern Recognit. 93 (2019) 430–441.
[8] J. Song, Y. Yang, Z. Huang, H.T. Shen, J. Luo, Effective multiple feature hashing for large-scale near-duplicate video retrieval, IEEE Trans. Multimed. 15 (8) (2013) 1997–2008.
[9] Y. Hao, T. Mu, R. Hong, M. Wang, N. An, J.Y. Goulermas, Stochastic multiview hashing for large-scale near-duplicate video retrieval, IEEE Trans. Multimed. 19 (1) (2017) 1–14.
[10] C. Shan, Face recognition and retrieval in video, in: Video Search and Mining, Heidelberg, Germany, Springer, 2010, pp. 235–260.
[11] J. Sivic, M. Everingham, A. Zisserman, Person spotting: video shot retrieval for face sets, in: Image and Video Retrieval, 2005, pp. 226–236.
[12] O. Yamaguchi, K. Fukui, K.-i. Maeda, Face recognition using temporal image sequence, in: Proc. FG, 1998, pp. 318–323.
[13] H. Cevikalp, B. Triggs, Face recognition based on image sets, in: Proc. CVPR, 2010, pp. 2567–2573.
[14] Y. Hu, A.S. Mian, R. Owens, Sparse approximated nearest points for image set classification, in: Proc. CVPR, 2011, pp. 121–128.
[15] T.-K. Kim, J. Kittler, R. Cipolla, Discriminative learning and recognition of image set classes using canonical correlations, IEEE Trans. Pattern Anal. Mach.Intell. 29 (6) (2007) 1005–1018.
[16] R. Wang, X. Chen, Manifold discriminant analysis, in: Proc. CVPR, 2009, pp. 429–436.
[17] R. Wang, S. Shan, X. Chen, W. Gao, Manifold-manifold distance with application to face recognition based on image set, in: Proc. CVPR, 2008, pp. 1–8.
[18] R. Wang, H. Guo, L.S. Davis, Q. Dai, Covariance discriminative learning: a natural and efficient approach to image set classification, in: Proc. CVPR, 2012, pp. 2496–2503.
[19] O. Parkhi, K. Simonyan, A. Vedaldi, A. Zisserman, A compact and discriminative face track descriptor, in: Proc. CVPR, 2014, pp. 1693–1700.
[20] J. Yang, P. Ren, D. Zhang, D. Chen, F. Wen, H. Li, G. Hua, Neural aggregation network for video face recognition, in: Proc. CVPR, 2017, pp. 5216–5225.
[21] Y. Liu, J. Yan, W. Ouyang, Quality aware network for set to set recognition, in: Proc. CVPR, 2017, pp. 4694–4703.
[22] S. Qiao, R. Wang, S. Shan, X. Chen, Deep video code for efficient face video retrieval, in: Proc. ACCV, 2016, pp. 296–312.
[23] J. Wang, H.T. Shen, J. Song, J. Ji, Hashing for similarity search: a survey, arXiv preprint arXiv:1408.2927 (2014).
[24] K.L. Huei-Fang Yang, C.-S. Chen, Supervised learning of semantics-preserving hash via deep convolutional neural networks, IEEE Trans. Pattern Anal. Mach.Intell. 40 (2) (2018) 437–451.
[25] H. Liu, R. Wang, S. Shan, X. Chen, Deep supervised hashing for fast image retrieval, in: Proc. CVPR, 2016, pp. 2064–2072.
[26] H. Lai, Y. Pan, Y. Liu, S. Yan, Simultaneous feature learning and hash coding with deep neural networks, in: Proc. CVPR, 2015, pp. 3270–3278.
[27] K.Q. Weinberger, J. Blitzer, L.K. Saul, Distance metric learning for large margin nearest neighbor classification, in: NIPS, 2005, pp. 1473–1480.
[28] H.O. Song, Y. Xiang, S. Jegelka, S. Savarese, Deep metric learning via lifted structured feature embedding, in: Proc. CVPR, 2016, pp. 4004–4012.
[29] O. Arandjelović, A. Zisserman, Automatic face recognition for film character retrieval in feature-length films, in: Proc. CVPR, vol. 1, 2005, pp. 860–867.
[30] O. Arandjelović, A. Zisserman, On film character retrieval in feature-length films, in: Interactive Video, 2006, pp. 89–105.
[31] M. Everingham, J. Sivic, A. Zisserman, Hello! my name is... buffy–automatic naming of characters in tv video., in: Proc. BMVC, 2006, pp. 899–908.
[32] C. Herrmann, J. Beyerer, Face retrieval on large-scale video data, in: Proc. CRV, 2015, pp. 192–199.
[33] Y. Li, R. Wang, Z. Cui, S. Shan, X. Chen, Compact video code and its application to robust face retrieval in tv-series., in: Proc. BMVC, 2014.
[34] 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: Proc. FG, 2015, pp. 1–8.
[35] 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: Proc. CVPR, 2015, pp. 4758–4767.
[36] Z. Dong, S. Jia, T. Wu, M. Pei, Face video retrieval via deep learning of binary hash representations, in: Proc. AAAI, 2016, pp. 3471–3477.
[37] Z. Dong, C. Jing, M. Pei, Y. Jia, Deep CNN based binary hash video representations for face retrieval, Pattern Recognit. 81 (2018) 357–369.
[38] Z. Huang, L.J.V. Gool, A riemannian network for SPD matrix learning, in: Proc. AAAI, 2017, pp. 2036–2042.
[39] W. Wang, R. Wang, S. Shan, X. Chen, Discriminative covariance oriented representation learning for face recognition with image sets, in: Proc. CVPR, 2017, pp. 5749–5758.
[40] K. Chatfield, R. Arandjelović, O. Parkhi, A. Zisserman, On-the-fly learning for visual search of large-scale image and video datasets, Int. J. Multimed. Inf. Retr. 4 (2) (2015) 75–93.
[41] E.J. Crowley, O.M. Parkhi, A. Zisserman, Face painting: querying art with photos, in: Proc. BMVC, 2015.
[42] E. Ghaleb, M. Tapaswi, Z. Al-Halah, H.K. Ekenel, R. Stiefelhagen, Accio: A data set for face track retrieval in movies across age, in: Proc. ACM MM, 2015, pp. 455–458.
[43] A. Gionis, P. Indyk, R. Motwani, Similarity search in high dimensions via hashing, in: Proc. VLDB, vol. 99, 1999, pp. 518–529.
[44] Y. Weiss, A. Torralba, R. Fergus, Spectral hashing, in: Proc. NIPS, 2009, pp. 1753–1760.
[45] W. Liu, J. Wang, S. Kumar, S.-F. Chang, Hashing with graphs, in: Proc. ICML, 2011, pp. 1–8.
[46] Y. Gong, S. Lazebnik, Iterative quantization: a procrustean approach to learning binary codes, in: Proc. CVPR, 2011, pp. 817–824.
[47] P. Li, M. Wang, J. Cheng, C. Xu, H. Lu, Spectral hashing with semantically consistent graph for image indexing, IEEE Trans. Multimed. 15 (1) (2013) 141–152.
[48] B. Kulis, T. Darrell, Learning to hash with binary reconstructive embeddings, in: Proc. NIPS, 2009, pp. 1042–1050.
[49] J. Wang, S. Kumar, S.-F. Chang, Semi-supervised hashing for scalable image retrieval, in: Proc. CVPR, 2010, pp. 3424–3431.
[50] M. Norouzi, D.J. Fleet, Minimal loss hashing for compact binary codes, in: Proc. ICML, 2011, pp. 353–360.
[51] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, S.-F. Chang, Supervised hashing with kernels, in: Proc. CVPR, 2012, pp. 2074–2081.
[52] M. Rastegari, A. Farhadi, D. Forsyth, Attribute discovery via predictable discriminative binary codes, in: Proc. ECCV, 2012, pp. 876–889.
[53] J. Wang, W. Liu, A. Sun, Y.-G. Jiang, Learning hash codes with listwise supervision, in: Proc. ICCV, 2013, pp. 3032–3039.
[54] J. Wang, J. Wang, N. Yu, S. Li, Order preserving hashing for approximate nearest neighbor search, in: Proc. ACM MM, 2013, pp. 133–142.
[55] Y. Jiang, J. Wang, X. Xue, S. Chang, Query-adaptive image search with hash codes, IEEE Trans. Multimed. 15 (2) (2013) 442–453.
[56] R. Xia, Y. Pan, H. Lai, C. Liu, S. Yan, Supervised hashing for image retrieval via image representation learning., in: Proc. AAAI, 2014, pp. 2156–2162.
[57] 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.
[58] K. Lin, H.-F. Yang, J.-H. Hsiao, C.-S. Chen, Deep learning of binary hash codes for fast image retrieval, in: Proc. CVPR Workshops, 2015, pp. 27–35.
[59] V.E. Liong, J. Lu, G. Wang, P. Moulin, J. Zhou, Deep hashing for compact binary codes learning, in: Proc. CVPR, 2015, pp. 2475–2483.

[60] F. Zhao, Y. Huang, L. Wang, T. Tan, Deep semantic ranking based hashing for multi-label image retrieval, in: Proc. CVPR, 2015, pp. 1556–1564.
[61] X. Wang, Y. Shi, K.M. Kitani, Deep supervised hashing with triplet labels, in: Proc. ACCV, 2016, pp. 70–84.
[62] B. Zhuang, G. Lin, C. Shen, I.D. Reid, Fast training of triplet-based deep binary embedding networks, in: Proc. CVPR, 2016, pp. 5955–5964.
[63] G. Lin, F. Liu, C. Shen, J. Wu, H.T. Shen, Structured learning of binary codes with column generation for optimizing ranking measures, Int. J. Comput. Vis. 123 (2) (2017) 287–308.
[64] Z. Cao, M. Long, J. Wang, P.S. Yu, HashNet: Deep learning to hash by continuation, in: Proc. ICCV, 2017, pp. 5609–5618.
[65] Y. Sun, X. Wang, X. Tang, Deep learning face representation from predicting 10,000 classes, in: IEEE, CVPR, 2014, pp. 1891–1898.
[66] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, L. Song, SphereFace: Deep hypersphere embedding for face recognition, in: IEEE, CVPR, 2017, pp. 6738–6746.
[67] Y. Wen, K. Zhang, Z. Li, Y. Qiao, A discriminative feature learning approach for deep face recognition, in: Proc. ECCV, 2016, pp. 499–515.
[68] N. Wojke, A. Bewley, Deep cosine metric learning for person re-identification, in: IEEE, WACV, 2018, pp. 748–756.
[69] R. Hadsell, S. Chopra, Y. LeCun, Dimensionality reduction by learning an invariant mapping, in: IEEE, CVPR, 2006, pp. 1735–1742.
[70] F. Schroff, D. Kalenichenko, J. Philbin, FaceNet: A unified embedding for face recognition and clustering, in: Proc. CVPR, 2015, pp. 815–823.
[71] Y. Movshovitz-Attias, A. Toshev, T.K. Leung, S. Ioffe, S. Singh, No fuss distance metric learning using proxies, in: IEEE, ICCV, 2017, pp. 360–368.
[72] B. Harwood, V.K.B. G, G. Carneiro, I.D. Reid, T. Drummond, Smart mining for deep metric learning, in: IEEE, ICCV, 2017, pp. 2840–2848.
[73] S. Horiguchi, D. Ikami, K. Aizawa, Significance of softmax-based features in comparison to distance metric learning-based features, IEEE Trans. Pattern Anal. Mach.Intell. 42 (5) (2020) 1279–1285.
[74] S. Ji, W. Xu, M. Yang, K. Yu, 3d convolutional neural networks for human action recognition, IEEE Trans. Pattern Anal. Mach.Intell. 35 (1) (2013) 221–231.
[75] D. Yi, Z. Lei, S. Liao, S.Z. Li, Learning face representation from scratch, arXiv preprint arXiv:1411.7923 (2014).
[76] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proc. AISTATS, 2010, pp. 249–256.
[77] M. Kim, S. Kumar, V. Pavlovic, H.A. Rowley, Face tracking and recognition with visual constraints in real-world videos, in: Proc. CVPR, 2008.
[78] A. Bansal, A. Nanduri, C.D. Castillo, R. Ranjan, R. Chellappa, UMDFaces: An annotated face dataset for training deep networks, in: IEEE, IJCB, 2017, pp. 464–473.
[79] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, M.J. Black, Towards understanding action recognition, in: IEEE, ICCV, 2013, pp. 3192–3199.

Shishi Qiao received the B.S. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2014. He is currently pursuing the Ph.D. degree with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China. His research interests mainly include computer vision, pattern recognition, machine learning and, in particular, video face recognition, face retrieval, object and scene understanding with deep generative models.

Ruiping Wang (S'08 M'11) received the B.S. degree in applied mathematics from Beijing Jiaotong University, Beijing, China, in 2003, and the Ph.D. degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, in 2010. He was a Post Doctoral Researcher with the Department of Automation, Tsinghua University, Beijing, from 2010 to 2012. He also spent one year as a Research Associate with the Computer Vision Laboratory, Institute for Advanced Computer Studies, University of Maryland at College Park, College Park, from 2010 to 2011. In 2012, he joined the Faculty of the Institute of Computing Technology, Chinese Academy of Sciences, where he has been a Professor since 2017. His re search interests include computer vision, pattern recognition, and machine learning.

Shiguang Shan (M'04-SM'15) received the M.S. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 1999, and the Ph.D. degree in computer science from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), Beijing, China, in 2004. In 2002, he joined ICT, CAS, where he has been a Professor since 2010. He is currently the Deputy Director of the Key Laboratory of Intelligent Information Processing, CAS. He has authored over 200 papers in refereed journals and proceedings in computer vision and pattern recognition. His research interests include computer vision, pattern recognition, and machine learning. He especially focuses on face recognition related research topics. He was a recipient of the Chinas State Natural Science Award in 2015 and the Chinas State S&T Progress Award in 2005 for his research work. He is an Associate Editor of several journals, including the IEEE TRANSACTIONS ON IMAGE PROCESSING, the Computer Vision and Image Understanding, the Neurocomputing, and the Pattern Recognition Letters. He has served as the Area Chair for some international conferences, including ICCV11, ICPR12/14/20, ACCV12/16/18, FG13/18/20, ICASSP14, BTAS18, and CVPR19/20.

Xilin Chen (M'00-SM'09-FF'16) is a professor with the Institute of Computing Technology, Chinese Academy of Sciences (CAS). He has authored one book and more than 300 papers in refereed journals and proceedings in the areas of computer vision, pattern recognition, image processing, and multimodal interfaces. He is currently an associate editor of the IEEE Transactions on Multimedia, and a Senior Editor of the Journal of Visual Communication and Image Representation, a leading editor of the Journal of Computer Science and Technology, and an associate editor-in-chief of the Chinese Journal of Computers, and Chinese Journal of Pattern Recognition and Artificial Intelligence. He served as an Organizing Committee member for many conferences, including general co-chair of FG13/FG18, program co-chair of ICMI 2010. He is/was an area chair of CVPR 2017/2019/2020, and ICCV 2019. He is a fellow of the IEEE, IAPR, and CCF.