Compact Video Code and Its Application to Robust Face Retrieval in TV-Series

Yan Li
yan.li@vipl.ict.ac.cn
Ruiping Wang
wangruiping@ict.ac.cn
Zhen Cui
zhen.cui@vipl.ict.ac.cn
Shiguang Shan
sgshan@ict.ac.cn
Xilin Chen
xlchen@ict.ac.cn

Key Lab of Intelligent Information Processing, Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing, 100190, China

Abstract

We address the problem of video face retrieval in TV-Series which searches video clips based on the presence of specific character, given one video clip of his/hers. This is tremendously challenging because on one hand, faces in TV-Series are captured in largely uncontrolled conditions with complex appearance variations, and on the other hand retrieval task typically needs efficient representation with low time and space complexity. To handle this problem, we propose a compact and discriminative representation for the huge body of video data, named Compact Video Code (CVC). Our method first models the video clip by its sample (i.e., frame) covariance matrix to capture the video data variations in a statistical manner. To incorporate discriminative information and obtain more compact video signature, the high-dimensional covariance matrix is further encoded as a much lower-dimensional binary vector, which finally yields the proposed CVC. Specifically, each bit of the code, i.e., each dimension of the binary vector, is produced via supervised learning in a max margin framework, which aims to make a balance between the discriminability and stability of the code. Face retrieval experiments on two challenging TV-Series video databases demonstrate the competitiveness of the proposed CVC over state-of-the-art retrieval methods. In addition, as a general video matching algorithm, CVC is also evaluated in traditional video face recognition task on a standard Internet database, i.e., *YouTube Celebrities*, showing its quite promising performance by using an extremely compact code with only 128 bits.

1 Introduction

Video face retrieval in general is to retrieve shots containing a particular person given one video clip of his/hers [19]. It is a promising research direction with increasing demands, especially in the era of Internet multimedia involving huge body of video data (e.g., many different types of videos can be found on the video sharing sites like YouTube, such as

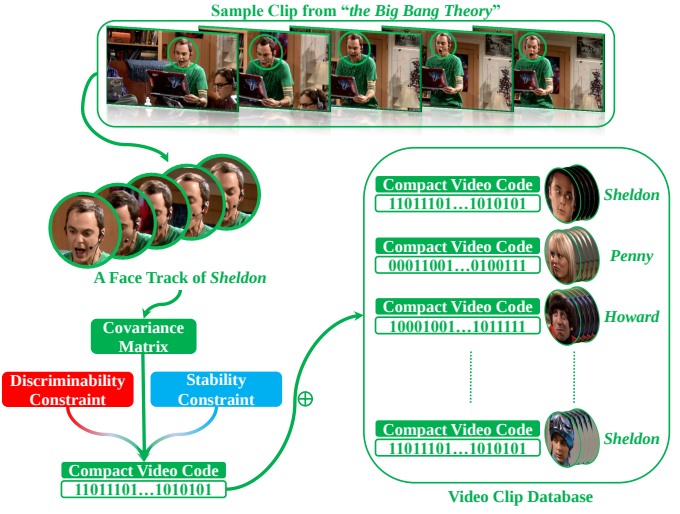

Figure 1: Illustration of the proposed method. Given a video clip of one character as query, we extract the proposed Compact Video Code (CVC) to represent it and use Hamming distance to retrieve video clips containing the specific character in database, which are also encoded in the form of CVC representations.

dramas, news programs and homemade videos, etc). Finding a specific person in videos is crucial to retrieve, analyze and understand videos. There are a wide range of applications relying on it, for example: '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 captured by a digital camera [20]; and rapid locating and tracking of suspects from masses of city surveillance videos. In this paper, we mainly focus on the problem of video face retrieval in TV-Series with character's one video clip as query, as depicted in Figure 1.

The key technique for video face retrieval is face recognition which has long been established as one of the most active research areas in computer vision. However, video face retrieval has its unique characteristics compared with traditional face recognition tasks. Specifically, different from traditional image based face recognition, video provides much more information, which can be exploited to resolve the inherent ambiguities of still image based recognition like sensitivity to variations caused by lighting, pose, different resolution and occlusion [19]. However, how to effectively utilize the rich information of video needs to be considered adequately. In this paper, we utilize the second-order statistic covariance matrix which has been proved natural and efficient for front-end video representation [26]. Another unique characteristic of video face retrieval is the strong demand in more compact and discriminative representation of video for both time fast and space saving search. Although covariance as a general descriptor has a certain discriminability due to its modeling of video data variations, it does not involve any supervised information which is crucial for retrieval. Besides, it is not compact enough due to its high dimension for fast search or retrieval especially when the database is very large. For this problem, we further encode the covariance matrix to a Compact Video Code (CVC) as the final video signature, which comes in the form of a much lower-dimensional binary code, where each bit of the code is learned by explicitly optimizing for discrimination in a max margin framework, inspired by an at-

tribute learning method in [18]. By doing this discriminability and stability are considered jointly.

To verify the effectiveness of our method, we conduct video retrieval experiments on two challenging TV-Series databases (*the Big Bang Theory* and *Buffy the Vampire Slayer* [3]). Experimental results show the competitiveness of the proposed CVC over state-of-the-art retrieval methods. In addition, our method is also evaluated in traditional video face recognition tasks on a standard benchmark, i.e., *YouTube Celebrities* [11]. It is shown that, as a general video matching algorithm, our method achieves quite promising performance by using a rather compact code with only 128 bits.

The rest of this paper is organized as follows: Section 2 discusses the related work of the proposed method. Then Section 3 describes our Compact Video Code designed for video face retrieval. Next, Section 4 exhaustively evaluates our method on different video classification tasks. Finally, we end with a summary of conclusions and future work in Section 5.

2 Related Work

Recent years have witnessed more and more studies on video face retrieval. Arandjelović and Zisserman [1, 2] built an end-to-end system to retrieve film shots, given one or more query face images. To address the uncontrolled appearance variations encountered in films, they proposed to obtain a signature image by a cascade of processing steps for every detected face. Anyway, it is a single image based face matching method which doesn't make full use of the video information. Instead of matching single faces, Sivic *et al*. [20] developed a video shot retrieval system by matching sets of faces (also called face tracks) for each person. The entire face track is represented as a distribution in the form of histogram. While such previous works have been devoted to a complete retrieval system including those preprocessing stages such as shot boundary detection, face detection and tracking, etc, it is generally believed that the key technical components lie in two aspects, i.e., the video data modeling and the final retrieval process, which are the topics of this paper.

As video is comprised of frames, it is often to be treated as image set in practice. Recently, image set based methods have been proved to be efficient representation for face videos after the pioneering work of Yamaguchi *et al*. [23]. Existing methods mainly focus on the key issues of how to model the image sets and how to measure the similarity between them [4, 8, 12, 24, 25, 26, 28]. One class of prevalent methods is to use subspace learning techniques to account for the image set variability globally either by a single linear subspace [12], or by a more sophisticated manifold [24, 25]. However, the linear subspace modeling cannot well accommodate the case in real world when the set has complex data variations. Another class of prevalent methods is based on affine subspace [4, 8]. More recently, [26] viewed image sets as points lying on a Riemannian manifold and developed a kernel learning method on the manifold. While such methods have gained success in image set classification task, high-dimensional representation limits their applicability to the video retrieval scenario which typically requires not only accurate but also compact and efficient representation of video for fast search.

Binary code is a natural solution to the above problem, because it is easy to match, and the space capacity of very short binary codes is so large that all the digital images in this world can be indexed with relatively short binary codes [15]. Binary codes have been widely used as hash keys and important examples include: locality sensitive hashing [6]

and its variants [17], spectral hashing [27], anchor graph hashing [13]. However none of these approaches would necessarily result in discriminative codes. In order to incorporate discriminability, a couple of semi-supervised and supervised binary code learning methods recently form a blowout. Representative methods include: semi-supervised hashing [23], kernel-based supervised hashing [14], supervised iterative quantization [7], and predictable discriminative binary code [18], etc.

3 Compact Video Code

Our goal is to learn codes for each video clip such that: (a) They are *efficient* to represent videos, which means it is better to utilize the information of the whole video clip; (b) The codes should be *compact*, preferably binary codes; (c) *Discriminability* should be considered when encoding, that is, all video clips of the same class should be distributed to the similar codes in the Hamming space.

3.1 Video Modeling

The first step we are going to take is to model the video clip. What we need is an efficient representation to characterize a whole video clip and its continuous appearance variations. The first choice coming into our mind is the widely exploited technique - linear subspace [12, 24, 25, 28], however, it cannot well accommodate the case when the video clip is of small size but has complex intra-class variations, see Figure 3. In fact, linear subspace models usually originate from an eigen-decomposition of the covariance matrix while discarding all the eigenvalues and non-leading eigenvectors. That makes the resulting subspace too loose to reflect the underlying data distribution. In contrast, as the raw second-order statistic of the video frames, covariance matrix provides a natural representation for a video clip with any type of features and any number of frames, and characterizes the video clip structure more faithfully [26]. Taking such into consideration, we resort to covariance matrices for representing video clips here.

Let $F = [f_1, f_2, ..., f_n]$ be the data matrix of a video clip with n frames, where $f_i \in \mathbb{R}^d$ denotes the i^{th} frame with d-dimensional feature. We represent the video clip with the $d \times d$ sample covariance matrix:

$$C = \frac{1}{n-1} \sum_{i=1}^{n} (f_i - \bar{f})(f_i - \bar{f})^T, \tag{1}$$

where $\bar{f}$ is the mean of all frames in the video clip. The diagonal entries of the covariance matrix represent the variance of each individual feature, and the off-diagonal entries are their respective correlations. It is well known that the nonsingular covariance matrices do not lie in a Euclidean space but on a Riemannian manifold $\mathcal{M}$ [9] [16] [21]. However, it is not trivial to learn a binary encoder on the manifold since typical code learning methods are devoted to operating in Euclidean space. So here we utilize the Log-Euclidean Distance (LED) to bridge the gap between Riemannian manifold and Euclidean space as in [26]:

$$d_{LED}(C_1, C_2) = \|\log(C_1) - \log(C_2)\|_F, \tag{2}$$

where C_1, C_2 are two nonsingular covariance matrices, log is the ordinary matrix logarithm operator, and $\|\cdot\|_F$ denotes the matrix Frobenius norm. Let $C = U \sum U^T$ be the eigen-

decomposition of C , its log is a symmetric matrix and can be computed easily by

$$\log(C) = U \log(\Sigma) U^T, \tag{3}$$

where $\log(\sum)$ is the diagonal matrix of the eigenvalue logarithms. By doing this, a point C on the Riemannian manifold $\mathcal{M}$ is projected to a Euclidean space via the logarithm map:

$$\Psi_{\log} : \mathcal{M} \mapsto T_I, C \to \log(C). \tag{4}$$

The image $\Psi_{\log}(\mathcal{M})$ is the tangent space T_I of the manifold $\mathcal{M}$ at the point of the identity matrix I, which is a vector space spanned by $d \times d$ symmetric matrices, i.e., $\log(C)$. In the tangent space, the off-diagonal entries of the matrix $\log(C)$ are counted twice during Euclidean vector norm computation. Let we denote the elements of $\log(C)$ by v_{ij} where $i = 1,2,\cdots,d$, $j = 1,2,\cdots,d$. As indicated in [22], we can re-write the matrix $\log(C)$ in a vector form as follows:

$$vec_I(\log(C)) = [v_{1,1}, \sqrt{2}v_{1,2}, \cdots, v_{2,2}, \sqrt{2}v_{2,3}, \cdots, v_{d,d}]. \tag{5}$$

As can be seen, $vec_I(\log(C))$ is a $d(d+1)/2$-dimensional vector and serves as the final representation of the video clip, which will be used as input to the following binary code learning.

3.2 Binary Encoding

For each video clip, we can obtain a vector form representation as in Eqn. (5), which comes from the second-order statistic of the video clip, i.e., covariance matrix. The covariance feature can describe textural structures as well as a certain action information of the whole video clip, but meanwhile it magnifies the feature dimension, which naturally leads to high time and space complexity, especially conflicting with the demand of retrieval task. Moreover, the covariance feature does not incorporate supervision information, which will undoubtedly favor the retrieval accuracy. To address this problem, our strategy is to map the high-dimensional covariance features into a much lower-dimensional Hamming space by discriminative learning, i.e., a binary vector for each video clip. By doing this, two advantages can be induced, the concise space demand (only 128 bits can work well in our experiments), and the low time cost (only XOR operation on bits). Inspired by the recent binary code learning work for attribute discovery [18], next we will discuss how to efficiently extract compact binary codes from those high-dimensional features.

First, the **discriminability** of binary code in Hamming space is expected. To this end, we further characterize the discriminability into two parts: intra-class compactness and inter-class separability. That is, the video clips from the same class should have similar codes, better with the same code, and the video clips from different classes should have better separability in Hamming space. Formally, let $b \in \{-1,1\}^{N \times K}$ denote the binary codes of training instances, where N and K denote the total number of training instances and the length of binary code respectively. Here, we use $\{-1,1\}$ instead of $\{0,1\}$ for the sake of formulation concision, and actually it is quite easy to switch between the two forms. $b_m \in \{-1,1\}^{1 \times K}$ denotes the binary code of the m^{th} training instance. Then the measures of

within class (S_W) and between class (S_B) distances can be formulated as,

$$S_W = \sum_{c\in\{1:M\}} \sum_{m,n\in c} dis(b_m, b_n), \tag{6}$$

$$S_B = \sum_{\substack{c_1\in\{1:M\} \\ p\in c_1}} \sum_{\substack{c_2\in\{1:M\} \\ c_1\neq c_2, q\in c_2}} dis(b_p, b_q), \tag{7}$$

where M is the total number of classes, $dis(\cdot)$ is the distance measurement of binary codes in Hamming space. Thus, to implement a strong discrimination, we should minimize the following energy function E_{disc}.

$$E_{disc} = S_W - \lambda_1 S_B, \tag{8}$$

where λ_1 serves as a normalization factor to balance the number of data pairs in S_W and S_B.

Second, the generality should also be considered while the above discriminant model only minimizes the empirical risk on the training instances. Like the classical technique, Support Vector Machine (SVM) [5], the discriminant hyperplane should be largely marginalized, which may make the mapping more stable on unseen test data, referred to **stability** in this work. Moreover, it implies that the similar-looking instances have similar codes, while the dissimilar-looking instances have dissimilar codes. To make better stability, we build the K hyperplanes by using SVM, and each generates one bit of the binary code (please see Figure 2 (a)). Concretely, we denote the k^{th} hyperplane by ω^k ($k = 1, \cdots, K$), and the energy function can be formulated as follow.

$$E_{stab} = \frac{1}{2} \sum_{k\in\{1:K\}} {\omega^k}^T \omega^k + \lambda_2 \sum_{\substack{k\in\{1:K\} \\ i\in\{1:N\}}} \max(1 - b_i^k({\omega^k}^T x_i), 0), \tag{9}$$

where x_i denotes the input feature, b_i^k indicates in which side of the k^{th} hyperplane the i^{th} training instance lies, and λ_2 balances the empirical training error and the hyperplane margin.

After the above analysis, we can reach the final objective function by combining Eqn. (8) and Eqn. (9) to simultaneously consider the discriminability and stability of the target binary code:

$$\min_{b,\omega} \quad E_{disc} + E_{stab}. \tag{10}$$

Since the objective function is non-convex, it is infeasible to find a global analytical solution. In practice, we independently optimize each individual component to iteratively update b and ω, similar to the optimization process in [18]. **First**, b is initialized by quantize the first K components of principal component analysis (PCA) [10]; **Second**, using b^k ($k = 1, \cdots, K$) as training labels to train the k^{th} SVM hyperplane ω^k; **Third**, update b to reflect the binary codes that these SVM's actually predict with $b_i^k = sgn({\omega^k}^T x_i)$; **Fourth**, optimize b with Equ. (8) via an efficient subgradient descend method proposed in [18] with computational complexity $O(NK)$. The whole optimization is looped from the second to fourth step by iteratively update b and ω, and in practice we find that usually two or three times iterations can make the objective function convergence. To further illustrate the optimization process, we provide a conceptual illustration as shown in Figure 2 (b), which only considers a one-hyperplane two-class toy case.

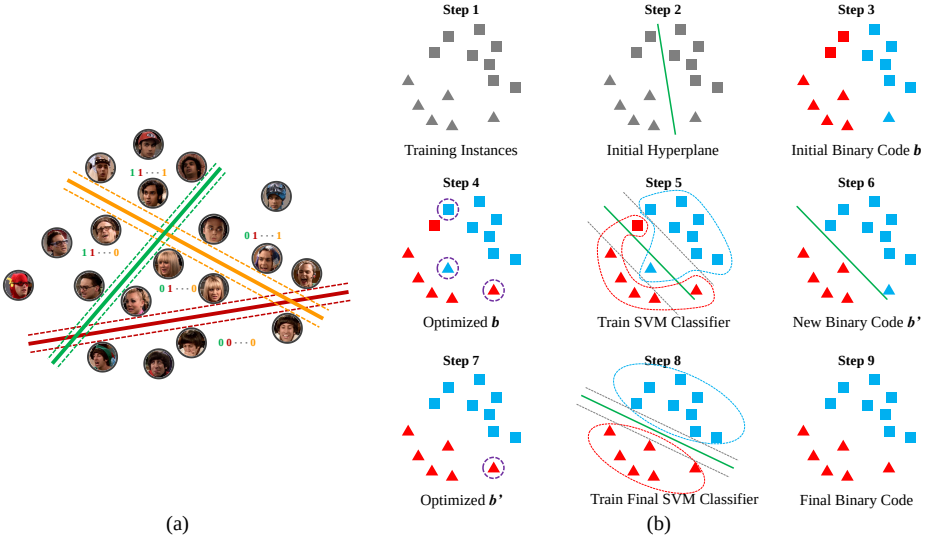

Figure 2: Illustration of Hamming space and optimization process. (a) Each bit of the binary code can be visualized as a hyperplane in the Hamming space. Each bit of the binary code for an instance is generated by checking the instance lies in which side of the corresponding hyperplane. (b) Illustration of the iterative optimization process of a one-hyperplane two-class case. The square and triangular shapes represent two classes. Blue and red colors represent bit values of binary code respectively. The purple circles indicate the updated instances after optimization in Eqn. (8). The green solid line indicates the learned hyperplane.

4 Experiments

To verify the effectiveness of the proposed method, in this section we conduct retrieval experiments on two challenging TV-Series databases. Before that, as a general video matching algorithm, we first evaluate our method in traditional video face recognition task.

4.1 Evaluation on Traditional Video Face Recognition

Although our method is designed for video face retrieval, it is also a general video matching algorithm, but with more concise features. In this part, we focus on the traditional video face recognition task on *YouTube Celebrities*.

YouTube Celebrities (YTC) is a widely studied and challenging benchmark [11] and contains of 1,910 video clips involving 47 celebrities collected from YouTube. Each video clip contains hundreds of frames, which are mostly highly compressed, with low resolution and covering large intra-class variations. We use the data features and 10-fold cross validation splits exactly the same as [26]. Specifically, each face is resized into 20×20 and only histogram equalization is used for pre-processing. In each folder, one subject has 3 randomly chosen video clips for the gallery and 6 for the probe.

In these tasks, we mainly compare the proposed method with the prevalent image set classification methods, including linear subspace based methods, Mutual Subspace Method (MSM) [28], Discriminative Canonical Correlations (DCC) [12]; nonlinear manifold based methods, Manifold-Manifold Distance (MMD) [25], Manifold Discriminant Analysis (M-

Table 1: Comparison with prevalent image set classification methods on *YouTube Celebrities*.

Methods	Ref.	Feature Length	Recognition Rate
MSM	[28]	Hundreds of Float	0.611
DCC	[12]	Hundreds of Float	0.648
MMD	[25]	Hundreds of Float	0.629
MDA	[24]	Hundreds of Float	0.653
AHISD	[4]	Hundreds of Float	0.637
CHISD	[4]	Hundreds of Float	0.663
SANP	[8]	Hundreds of Float	0.684
CDL	[26]	Hundreds of Float	0.701
CVC	**Ours**	**128 Bits**	**0.702**

DA) [24]; affine subspace based methods, Affine Hull based Image Set Distance (AHISD) [4], Convex Hull based Image Set Distance (CHISD) [4], Sparse Approximated Nearest Points (SANP) [8] and a covariance based method Covariance Discriminative Learning (CDL) [26]. For fair comparison, the important parameters of each method are empirically tuned according to the recommendations in the original literatures as well as the source codes provided by the authors. For all the experiments, we set the parameters as follows. In DCC, the eigen subspace dimension of each vector set is set to 10. In MDA, the number of between-class NN local models and the subspace dimension are specified as [24]. For both AHISD and CHISD, we use their linear version. The error penalty in CHISD is set to C = 100 as [4]. For SANP, we adopted the same weight parameters as [8] for the convex optimization. For CDL, we use partial least squares for identification task, and linear discriminant analysis for the later retrieval task.

For our method, we didn't tune the parameters λ_1 and λ_2, but just simply set λ_1 to normalize for the size of classes and λ_2 to 1. The feature used for covariance calculation is gray image intensity. Moreover, we typically fix the code length to 128, which is a trade-off between training cost and accuracy. Table 1 shows the performance evaluation. From this table, we can reach two consistent observations: (1) Compared with those subspace-based methods, the covariance-based model is more robust to characterize large appearance variations of video clips. The reason is that it is a natural statistical model and can faithfully capture the original information, containing appearance textures and dynamic actions; (2) CVC achieves a comparable performance compared with these state-of-the-art methods, which are elaborately designed for video face recognition. More importantly, our method has an extremely compact feature, i.e., only 128 bits, which naturally leads to higher efficient matching process than those traditional methods especially in large scale database.

4.2 Evaluation on Video Face Retrieval

In this part, we evaluate our method in video face retrieval task on two challenging TV-Series databases [3]. The first one consists of the first 6 episodes from season 1 of ***the Big Bang Theory*** (BBT), and the second one consists of the first 6 episodes from season 5 of ***Buffy the Vampire Slayer*** (BVS). These two TV-Series are quite different in their filming style, and therefore pose different challenges. BBT is a sitcom mostly taking place indoors with a main cast of 5∼8 characters. On the other hand, BVS has a main cast size around 12 sometimes up to 18 in specific episodes. Many shots are set at night and outdoors, resulting in a large

Figure 3: Examples of the two TV-Series. The first row are some frames from *the Big Bang Theory*, and the second row for *Buffy the Vampire Slayer*. Large variations caused by expression, illumination, head pose can be found here.

range of different illumination. Some examples are shown in Figure 3. We use the extracted face tracks represented by block discrete cosine transformation feature as used in [3] for all the competitive methods (including ours). The distribution of face tracks per character can be found in supplementary material.

We randomly select 147 and 374 face tracks for training algorithms on BBT and BVS respectively, and leave 3,194 and 4,405 face tracks for evaluating the retrieval performance with standard precision/recall curve. Specifically, for each character the precision/recall curve is computed by averaging the randomly selected 20 queries of his/hers. In this retrieval task, we compare not only with the image set classification methods as Section 4.1 did but also several binary code methods, including Locality-Sensitive Hashing (LSH) [6], Shift-Invariant Kernels based Locality-Sensitive Hashing (SIKLSH) [17], Spectral Hashing (SH) [27], Random Rotation (RR) [7], Semi-Supervised Hashing (SSH) [23], Kernel-Based Supervised Hashing (KSH) [14], Iterative Quantization (ITQ) [7] and Supervised Iterative Quantization (SITQ) [7]. For fair comparison, we fix the bit number to 128 (please find more experimental results in supplementary material) and use covariance feature in Section 3.1 for all the competitive methods. Figure 4 shows the precision/recall curves of two groups of comparison. Due to the limited space, we only show several main characters for each TV-Series, please refer to supplementary material for more characters.

Although CDL and DCC achieve near comparable performance with CVC, the storage space they consumed for representing a face track far exceeds ours (128 bits) by several orders of magnitude. We can also obviously find that supervised hashing methods achieve higher retrieval accuracy than those unsupervised and semi-supervised methods, which may attribute to the full use of supervised information. Compared with those state-of-the-art supervised hashing methods, CVC achieves more excellent performance. A possible reason is CVC also incorporates the stability while considering the discriminability, which makes the learnt model better generality on the unseen test set. In another word, sometimes we would rather sacrifice some discriminability on training data to ensure the stability, or vice versa.

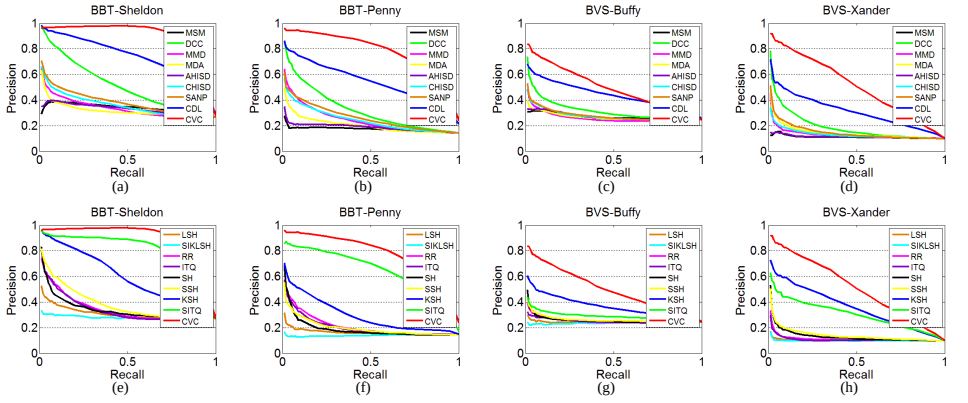

Figure 4: Comparisons with image set classification methods ((a)-(d)) and binary code learning methods ((e)-(h)) in video face retrieval.

5 Conclusion

In this paper, we address the problem of video face retrieval in TV-Series. To solve this problem, we have proposed a Compact Video Code (CVC) which at first models video clip by its sample covariance matrix, then forms the binary representation by jointly optimizing for discriminability and stability which are particularly crucial for retrieval. The learned CVC has been successfully applied to different video classification tasks, including identification and retrieval with rather concise code with only 128 bits. As CVC is a binary code, from another perspective, each bit of the CVC can be seen as an attribute classifier which shows the presence or absence of specific attribute of video clips. Although these attributes have been proven discriminative but they cannot be described by human beings, that is to say, there is nothing of explicit semantic information. In the future, we would explore the connection between CVC and semantic attributes for more convenient and practical retrieval applications, not limited to faces, but extending to behavior retrieval from massive surveillance data.

6 Acknowledgments

This work is partially supported by Natural Science Foundation of China under contracts Nos. 61025010, 61222211, 61379083, and 61390511.

References

[1] Ognjen Arandjelović and Andrew Zisserman. Automatic face recognition for film character retrieval in feature-length films. In *CVPR*, volume 1, pages 860–867. IEEE, 2005.

[2] Ognjen Arandjelović and Andrew Zisserman. On film character retrieval in feature-length films. In *Interactive Video*, pages 89–105. Springer, 2006.

[3] Martin Bäuml, Makarand Tapaswi, and Rainer Stiefelhagen. Semi-supervised learning with constraints for person identification in multimedia data. In *CVPR*. IEEE, 2013.

[4] Hakan Cevikalp and Bill Triggs. Face recognition based on image sets. In *CVPR*, pages 2567–2573. IEEE, 2010.

[5] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20 (3):273–297, 1995.

[6] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *VLDB*, volume 99, pages 518–529, 1999.

[7] Yunchao Gong and Svetlana Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR*, pages 817–824. IEEE, 2011.

[8] Yiqun Hu, Ajmal S Mian, and Robyn Owens. Sparse approximated nearest points for image set classification. In *CVPR*, pages 121–128. IEEE, 2011.

[9] Sadeep Jayasumana, Richard Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Harandi. Kernel methods on the riemannian manifold of symmetric positive definite matrices. In *CVPR*, pages 73–80. IEEE, 2013.

[10] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2005.

[11] Minyoung Kim, Sanjiv Kumar, Vladimir Pavlovic, and Henry Rowley. Face tracking and recognition with visual constraints in real-world videos. In *CVPR*, pages 1–8. IEEE, 2008.

[12] Tae-Kyun Kim, Josef Kittler, and Roberto Cipolla. Discriminative learning and recognition of image set classes using canonical correlations. *PAMI*, 29(6):1005–1018, 2007.

[13] Wei Liu, Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Hashing with graphs. In *ICML*, pages 1–8, 2011.

[14] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, and Shih-Fu Chang. Supervised hashing with kernels. In *CVPR*, pages 2074–2081. IEEE, 2012.

[15] Peter Lyman and Hal Varian. How much information? 2003. 2004.

[16] Xavier Pennec, Pierre Fillard, and Nicholas Ayache. A riemannian framework for tensor computing. *IJCV*, 66(1):41–66, 2006.

[17] Maxim Raginsky and Svetlana Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In *NIPS*, pages 1509–1517, 2009.

[18] Mohammad Rastegari, Ali Farhadi, and David Forsyth. Attribute discovery via predictable discriminative binary codes. In *ECCV*, pages 876–889. Springer, 2012.

[19] Caifeng Shan. Face recognition and retrieval in video. In *Video Search and Mining*, pages 235–260. Springer, 2010.

[20] Josef Sivic, Mark Everingham, and Andrew Zisserman. Person spotting: video shot retrieval for face sets. In *Image and Video Retrieval*, pages 226–236. Springer, 2005.

[21] Suvrit Sra. A new metric on the manifold of kernel matrices with application to matrix geometric means. In *NIPS*, pages 144–152, 2012.

[22] Oncel Tuzel, Fatih Porikli, and Peter Meer. Pedestrian detection via classification on riemannian manifolds. *PAMI*, 30(10):1713–1727, 2008.

[23] Jun Wang, Sanjiv Kumar, and Shih-Fu Chang. Semi-supervised hashing for scalable image retrieval. In *CVPR*, pages 3424–3431. IEEE, 2010.

[24] Ruiping Wang and Xilin Chen. Manifold discriminant analysis. In *CVPR*, pages 429–436. IEEE, 2009.

[25] Ruiping Wang, Shiguang Shan, Xilin Chen, and Wen Gao. Manifold-manifold distance with application to face recognition based on image set. In *CVPR*, pages 1–8. IEEE, 2008.

[26] Ruiping Wang, Huimin Guo, Larry S. Davis, and Qionghai Dai. Covariance discriminative learning: a natural and efficient approach to image set classification. In *CVPR*, pages 2496–2503. IEEE, 2012.

[27] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *NIPS*, pages 1753–1760, 2008.

[28] Osamu Yamaguchi, Kazuhiro Fukui, and Ken-ichi Maeda. Face recognition using temporal image sequence. In *FG*, pages 318–323. IEEE, 1998.