CORE
Metadata, citation and similar papers at core.ac.uk
Provided by University of Liverpool Repository

Stochastic Multi-view Hashing for Large-scale Near-duplicate Video Retrieval

Yanbin Hao, Tingting Mu, *Member, IEEE,* Richang Hong, *Member, IEEE,* Meng Wang, *Member, IEEE,* Ning An, *Senior Member, IEEE,* and John Y. Goulermas, *Senior Member, IEEE,*

***Abstract*—Near-duplicate video retrieval (NDVR) has been a significant research task in multimedia given its high impact in applications, such as video search, recommendation and copyright protection, etc. In addition to accurate retrieval performance, the exponential growth of online videos has imposed heavy demands on the efficiency and scalability of the existing systems. Aiming at improving both the retrieval accuracy and speed, we propose a novel stochastic multi-view hashing algorithm to facilitate the construction of a large-scale NDVR system. Reliable mapping functions, which convert multiple types of keyframe features, enhanced by auxiliary information such as video-keyframe association and ground truth relevance to binary hash code strings, are learned by maximizing a mixture of the generalized retrieval precision and recall scores. A composite Kullback-Leibler (KL) divergence measure is used to approximate the retrieval scores, which aligns stochastically the neighborhood structures between the original feature and the relaxed hash code spaces. The efficiency and effectiveness of the proposed method are examined using two public near-duplicate video collections, and are compared against various classical and state-of-the-art NDVR systems.**

***Index Terms*—Near-duplicate video retrieval, hashing, multi-view learning, semi-supervised learning, divergence.**

I. Introduction

There has been an explosive growth of video-related applications, such as video sharing websites, video broadcasting, recommendation, monitoring and advertising services, etc. This results in a large amount of online video data that keeps growing rapidly. There are increasingly more online users performing video editing, uploading, downloading, searching and viewing activities. The leading internet technology company comScore has reported that the PC users crossed over 300 billion videos in August 2014 alone, with an average of 202 videos and 952 minutes per viewer. Amongst the huge amount of online videos, there exist a substantial portion of near-duplicate videos (NDVs), which possess formatting and/or content differences from the non-duplicate ones [1]–[7]. Various ways of defining NDV can be found in [1], [3], [4], [6]. The existence of NDVs heavily affects video applications such as copyright protection, video monitoring, reranking, recommendation and thread tracking. Accurate and efficient NDV retrieval (NDVR) systems are thus required. For instance, given a search engine, the users are much more interested in diverse videos other than NDVs among the top-ranked retrieval results. Another example is that the copyright video products are exposed to severe risk of being compromised by unauthorized copying, editing and redistribution, and therefore, NDV detection is important for copyright protection.

A general NDVR strategy usually includes three main steps. Videos are first represented by a sequence of keyframes, which are extracted by uniform sampling [8] or shot-based methods [1], [2]. Low-level features are then extracted to characterize each keyframe [1], [2], [8]. Similarities between videos are finally computed based on the extracted keyframes and their low-level features, based on which relevant videos are retrieved [1], [8]. To compute and compare similarity between videos, pairwise keyframe comparison is a classical solution, that is, to exhaustively compare all the available keyframe pairs [1], [9]. Although such exhaustive comparison can offer accurate retrieval results, it is very time-consuming in practice. To improve the efficiency, the sliding window method can be used, examining only keyframes within a certain sliding window [1]. Instead of pairwise comparison over the keyframe level, video signature offers a more efficient way to compute similarity over the video level [6] based on the compact signature generated for each video by processing and combining its low-level keyframe features (e.g., signatures based on the local and global information of keyframes [1], [10] and signatures based on the spatial and temporal information of the videos [8]). There are also works combining the pairwise comparison and video signature [4], [10], which however, can still be time-consuming and not suitable for large-scale applications.

In addition to high accuracy, good scalability has become increasingly important in modern information retrieval to accommodate the big data era. Apart from a few research works that have been developed to address the scalability issue [7], [8], many existing NDVR works [11]–[13] cannot be efficiently applied to process large-scale videos in real time because they use certain photometric or geometric transformations. For instance, the retrieval system in [13] represents each keyframe by more than 400 local descriptors and its keyframe matching is computationally expensive. Another issue is that there does not exist a single feature type, robust enough to capture all the information variations. Different feature types may contain complementary information. For example, the global feature is sensitive to brightness, scale and contrast changes, while the local feature is sensitive to changes in frame rate, video length and captions [7]. The strategy of combining

Y. Hao, R. Hong, M. Wang and N. An are with the School of Computer and Information, Hefei University of Technology, Hefei, 230009, China. Email: haoyanbin@mail.hfut.edu.cn, {hongrc.hfut, eric.mengwang}@gmail.com, ning.an@hfut.edu.cn.

T. Mu is with the School of Computer Science, University of Manchester, Kilburn Building, Manchester, M13 9PL, UK. Email: tingtingmu@me.com.

J.Y. Goulermas is with the Department of Computer Science, University of Liverpool, Ashton Building, Liverpool, L69 3BX, UK. Email: j.y.goulermas@liverpool.ac.uk.

multiple types of features to improve video representation has received growing interest in multimedia [14], [15] and NDV [16], [17] research. This is often called multi-feature fusion [18], or more generally multi-modal [19]–[22] and multi-view learning [14], [15], [23]. Compared to multi-feature fusion, multi-modal and multi-view also employ general data analysis and machine learning tasks that involve joint processing of multiple information resources available to the studied objects.

This work proposes a novel stochastic multi-view hashing (SMVH) algorithm, which contributes to the development of an efficient large-scale NDVR system. The proposed hashing algorithm learns binary strings to characterize data samples by combining multiple feature types and auxiliary information through a stochastic matching procedure of the neighborhood probabilistic models. In the NDVR system, the multiple views include multiple types of video feature information (e.g., the global color histogram and the local texture pattern), the keyframe and video association information, as well as certain amount of ground truth relevance knowledge that is partially available to some videos. They are converted to simple binary strings through a set of mapping functions, so that the similarity comparison can be efficiently implemented by computing the Hamming distance between the strings based on fast XOR operations. The mapping functions are learned stochastically by maximizing a mixture of the generalized retrieval precision and recall scores. The scores are approximated by the composite Kullback-Leibler (KL) divergence computed between two probabilistic models constructed in the original feature space and a relaxed hash code space. Given a query object, its hash code can be rapidly generated using the learned mapping functions to support the subsequent hash code matching. The efficiency and effectiveness of our NDVR system are examined and compared against various state-of-the-art NDVR systems using the two public video collections CC_WEB_VIDEO and UQ_VIDEO.

The remaining of this paper is organized as follows. Section II briefly reviews some related work. Section III outlines the structure of the developed NDRV system, while Section IV explains the proposed hashing algorithm. In Section V, the performance of the proposed system is assessed and compared with several state-of-the-art methods in terms of both retrieval accuracy and efficiency. Section VI concludes the work.

II. Related Work

A. Feature Extraction

Feature extraction utilizes domain knowledge to generate from the raw videos numerical features that make the retrieval algorithms work. Most of the existing NDVR approaches conduct feature extraction based on the video content information [1], [4], [7], [8], [24], [25]. One common strategy is to first select keyframes from videos via uniform sampling, then extract low-level features to characterize each keyframe. It is also possible to directly generate compact signature representations for videos by skipping the keyframe selection procedure. Usually, the global and (or) local information are considered during feature extraction.

The most commonly used global feature is color histogram, e.g. RGB and HSV histogram [1], [7], [8], [26], but it can only be used to retrieve videos that are almost identical to the query video with minor variations. Another type of global features is based on the temporal shape information. e.g., the video signature based on the ordinal measure of the re-sampled video frames [27]. It reflects the relative intensity distribution within a frame, but the curve length of the used ordinal measure increases as the video length increases, consequently can be sensitive to video length.

Compared to global features, local features can be more robust to complex editing, photometric and geometric changes, and they generally provide better performance when processing videos with complex scenes and different lengths. Commonly used local point detectors for extracting local features include the difference of Gaussian (DOG) [28], the scale invariant feature transform (SIFT) [29], a mixture of the principal component analysis (PCA) and SIFT [30] referred to as PCA-SIFT, and the local binary pattern (LBP) [31], etc. To facilitate video retrieval and NDVR, trajectories of the local descriptors along the video sequence have been particularly studied [32]. Although good performance is offered by local features, the computational cost can be high due to the large number of pixels and the exhaustive match of keypoints [13]. Various research works aim at speeding up the computation, by applying dimensionality reduction approaches [30], hashing algorithms [2] and fast indexing structure (FIS) techniques [1], [25], [33]. Although improvement has been achieved, in general, it remains a challenge for the local feature approaches to reach both high accuracy and efficiency, due to for example a substantial amount of pairwise comparisons.

B. Dimensionality Reduction

The use of high-dimensional feature descriptors results in high-dimensional data spaces to process. For example, there exist from hundreds to thousands of local points per keyframe for some videos with complex scenes [10]. It is known that dependencies between different data dimensions often restrict the data points to a manifold with its dimensionality much lower than the dimensionality of the original data space [34]. To reduce the redundancy and discover hidden structure in high-dimensional data, many sophisticated methods have been developed over the last few decades, aiming at discovering and unfolding a lower-dimensional manifold. In addition to the traditional dimensionality reduction techniques such as PCA [35] and multidimensional scaling [36], successful manifold learning approaches are developed, including Isomap [34], locally linear embedding [37], and different versions of stochastic neighbor embedding [38], [39] that offer satisfactory neighbor preservation performance in the reduced space. Most of these methods attempt to bring data points that are similar to (or are neighbors of) each other in the original high-dimensional space closely together in the low-dimensional embedding space. In order to learn a data representation that particularly suits the information retrieval purpose, the neighbor retrieval visualizer (NeRV) [40] is proposed to optimize the low-dimensional space based on a measure of mean smoothed precision and recall that approximates the retrieval performance. As shown later, this measure is adapted in our proposed multi-view

hashing method to seek an optimal embedding space that maintains the inherent relationships between video keyframes to facilitate the NDVR task.

C. Hashing

Hashing is a technique that is able to support large-scale retrieval by enabling a fast pairwise similarity comparison between videos [41]. It takes an arbitrary-length data vector as the input and outputs a fixed-length binary string. In general, longer hash code may result in better performance, but this is associated with a larger memory consumption. Most existing hashing approaches are projection driven. The classical approach of locality sensitive hashing (LSH) maps the video data to binary codes along a set of random projections, so that similar objects are more likely to be mapped into the same buckets [42]. Various extensions of LSH have been developed. For example, [43] employs LSH to index the local descriptors, and [44] projects the extracted features into an auxiliary space using LSH, and represents the projected features as a histogram. The performance of LSH and its variations is however limited, because random projections do not fully exploit the inherent data distributions. Many machine learning algorithms have been proposed to derive the hash code projections or mapping functions through more sophisticated computations rather than random projections. For example, the hashing method proposed in [45] jointly learns the pseudo class labels and the hash code for the given objects based on a discriminant embedding framework driven by linear discriminant analysis. Algorithms such as spectral hashing (SPH) [46], self-taught hashing (STH) [47], semi-supervised hashing (SSH) [48], and supervised hashing with kernels [49] use different distance measures to construct a similarity graph between objects. These graph based methods learn the hash code or hash function by schemes such as the binarization of the eigenvectors of the graph or support vector mapping. Recently, convolutional neural network (CNN) is employed to work with graph based hashing method to achieve deep hashing [50]. More detailed review on state-of-the-art hashing techniques can be found in [51].

D. Learning From Multiple Types of Information

As mentioned in Section II-A, the global and local features are two main types of features for representing videos. Often, they offer complementary information that can enhance each other. This naturally leads to the feature fusion strategy to combine them, such as the early fusion (EF) [52] and the late fusion (LF) [53], and is widely applied to the representation of multimedia data [54]–[58]. The EF strategy performs the fusion of multiple features at the input stage. For example, [54] designs a hierarchical regression model to exploit the information derived from each feature type and then, collaboratively fuses these features to be fed into a multimedia semantic concept classifier. However, it is often difficult to construct a perfect unified space to project the multiple types of features, which makes it challenging for EF to well preserve the individual structural information of each feature type. Differently, LF attempts to combine results obtained individually by each feature type at the output stage, which however, considers less the correlation information between the feature types.

Apart from combining different types of feature characterizations to improve the video representation, researchers have also investigated more generally how to utilize multiple information resources to improve the searching performance. For instance, search over different media types of queries and results, such as text documents, images, audio and video [20], [22]. In NDVR, the context information associated with the web videos (e.g., thumbnail images, time durations, and number of views, etc.) is combined with the video content to boost the retrieval performance [10]. Specifically, the time duration of the videos is used to rapidly, but coarsely identify the preliminary groups of the NDVs. Then, a seed video is selected from each group based on the color histograms of the thumbnail images and their view counts. The final step of the NDV detection is reduced to compare the thumbnail images of the candidate videos with the selected seed videos. This approach can reach around an 164 fold speedup, with a slight loss of the retrieval accuracy. But it can only be used to retrieve web videos, due to the use of the web context information, which is unfortunately not always available in other video corpora.

To improve the retrieval performance, while maintaining good searching scalability, a substantial amount of research works have been developed to learn high-quality hash codes from multiple types of information. The current state-of-the-art multi-view hashing algorithms are mostly algebraic approaches based on trace (or norm) minimization, matrix factorization or their mixtures [7], [14], [15], [59], or approaches based on trace/norm induced objective functions alternatively optimized over variables stored in multiple matrices [60], [61]. There are few works studying hash code generation based on a stochastic strategy. One relevant example is the linear cross-modal hashing, which stochastically preserves the neighborhood relationships under each view (or modal) via neighborhood components analysis [62]. Given the recent success of stochastic neighborhood preservation [39], [40] in embedding generation and data visualization, we aim at constructing appropriate probabilisitic models for multi-view hash code generation to further improve the retrieval performance over the commonly used algebraic models.

III. The NDVR System

An information retrieval task is defined as a search task that outputs a ranked list of objects that are relevant to a specified query provided by the user. To search among NDVs, we construct the following retrieval system step by step.

1) **Keyframe Extraction**: Given a collection of V videos, multiple representative keyframes are extracted for each video by using the shot-based sampling method. Assuming n keyframes are extracted from V videos, the later processing steps are focused on information provided by the n extracted keyframes.
2) **Feature Extraction**: The global HSV (hue, saturation, value) features and the local binary pattern (LBP) features [31] are extracted, characterizing the global colour

histogram and the local texture feature for each keyframe. These result in two separate feature representations for each of the n keyframes, stored in the two feature matrices of $\mathbf{X}^{(1)} = \left[x_{ij}^{(1)}\right]$ and $\mathbf{X}^{(2)} = \left[x_{ij}^{(2)}\right]$ with the sizes of $n \times d_1$ and $n \times d_2$, respectively. The two column vectors $\boldsymbol{x}_i^{(1)} = \left[x_{i1}^{(1)}, x_{i2}^{(1)}, \ldots, x_{id_1}^{(1)}\right]^T$ and $\boldsymbol{x}_i^{(2)} = \left[x_{i1}^{(2)}, x_{i2}^{(2)}, \ldots, x_{id_2}^{(2)}\right]^T$ are used to denote the HSV and LBP feature vectors, respectively, for the ith keyframe.

3) **Hash Code Learning**: In this step, a set of s hash functions $\{\mathfrak{h}_i(\cdot)\}_{i=1}^s$ is learned. Each function takes the extracted features of a keyframe as the input and returns a binary number. The s hash functions lead to a binary string of length s for each keyframe. The derived strings of the n keyframes are stored in the rows of the $n \times s$ binary hash code matrix $\mathbf{H} = [h_{ij}]$.
4) **Video Similarity Computation**: Finally, a unique hash code string is generated for each video from the relaxed hash codes of its representation keyframes by Eq. (4). The Hamming distance between the generated strings is used to assess the similarity between videos. A list of videos that possess the highest similarities to the query video is returned.

In this system, the key video information supporting the retrieval task is characterized by the HSV and LBP features of the representative keyframes, stored in $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$. The success of this system relies on the hash code learning stage. The quality of the learned hash code decides how well the information contained in $\mathbf{X}^{(1)}$ and $\mathbf{X}^{(2)}$ can be transferred to the binary hash code matrix $\mathbf{H}$. It drives the quality of the retrieval system through hash code matching.

Apart from HSV and LBP, there exist many other feature extraction methods [8], [28]–[30] that can be used to characterize the keyframes. To allow researchers in the field to explore and combine various feature extraction methods of their choices, we define the targeted research problem in a general manner. Given a set of n objects characterized by multiple feature matrices $\{\mathbf{X}^{(g)}\}_{g=1}^m$ each with the size of $n \times d_g$, the goal is to derive an optimal $n \times s$ binary matrix $\mathbf{H}$ by simultaneously preserving the useful information provided by the m feature matrices, and auxiliary adjacency information if available. An explicit mapping function between the binary feature matrix $\mathbf{H}$ and the input features $\{\mathbf{X}^{(g)}\}_{g=1}^m$ is constructed.

IV. Proposed Stochastic Multi-view Hashing

A. Hash Function Mapping

Given multiple feature matrices $\{\mathbf{X}^{(g)}\}_{g=1}^m$ to characterize a collection of n objects, which are referred as different feature views, the column vector $\boldsymbol{x}_i^{(g)} = \left[x_{i1}^{(g)}, x_{i2}^{(g)}, \ldots, x_{id_g}^{(g)}\right]^T$ stores the features of the ith object under the gth view. In the NDVR system, the n objects correspond to the n keyframes. For each object, to build a reliable connection between its features $\{\boldsymbol{x}_i^{(g)}\}_{g=1}^m$ and its length s binary hash code string $\boldsymbol{h}_i = [h_{i1}, h_{i2}, \ldots, h_{is}]$ where $h_{il} \in \{0, 1\}$, we construct s hash functions $\{\mathfrak{h}_l\}_{l=1}^s$ so that $h_{il} = \mathfrak{h}_l\left(\{\boldsymbol{x}_i^{(g)}\}_{g=1}^m\right)$. These functions are formulated as

$$\mathfrak{h}_l\left(\left\{\boldsymbol{x}_i^{(g)}\right\}_{g=1}^m\right) = T\left(z_{il}\right), \quad (1)$$

$$z_{il} = \text{sigmoid}\left(\tilde{z}_{il}\right), \quad (2)$$

$$\tilde{z}_{il} = \sum_{g=1}^{m}\sum_{j=1}^{d_g} x_{ij}^{(g)} w_{lj}^{(g)} + b_l. \quad (3)$$

In the above equations, a one-dimensional embedding is first computed, by assuming it is a linear combination of all the observed features, where $w_{lj}^{(g)} \in R$ are the combination coefficients and $b_l \in R$ is a bias parameter. Then, the sigmoid function is used to convert the positive and negative embedding values to numbers close to one and zero. In the end, a thresholding function, given as $T(x) = 1$ if $x > 0.5$, and $T(x) = 0$ otherwise, is applied to convert a real-valued input to a binary number. The embedding vector $\boldsymbol{z}_i = [z_{i1}, z_{i1}, \ldots, z_{is}]^T$ obtained without applying thresholding, referred to as the relaxed hash code, for which the n embedding vectors of the n keyframes constitute an $n \times s$ relaxed hash code matrix $\mathbf{Z} = [z_{il}]$. In NDVR, one classical way to generate hash code for a video is to process the relaxed hash codes of its representative keyframes by first performing averaging and then the thresholding operations [7]. Letting $h_{il}^{(v)}$ denote the lth digit of the ith video's hash code, Ind_i the set of keyframe indices of this video and $|\text{Ind}_i|$ its cardinality, the video hash code can be expressed by

$$h_{il}^{(v)} = T\left(\frac{1}{|\text{Ind}_i|}\sum_{j \in \text{Ind}_i} z_{jl}\right). \quad (4)$$

These codes constitute the $V \times s$ video hash code matrix $\mathbf{H}^{(v)} = \left[h_{il}^{(v)}\right]$.

The composition of the linear combination function and the sigmoid function as in Eqs. (2) and (3) constructs a smooth mapping to transform $\{\mathbf{X}^{(g)}\}_{g=1}^m$ into a relaxed hash code (embedding) matrix $\mathbf{Z}$ with each element $z_{il} \in [0, 1]$. This operation is equivalent to taking all the observed features as the input of a single layer perceptron. An alternative setting is to map $\{\mathbf{X}^{(g)}\}_{g=1}^m$ to $\mathbf{Z}$ by employing a neural network with multiple layers to realize a nonlinear combination of the features. As shown in the experimental section, we have obtained satisfactory results using Eq. (3), which could be an alternative. The use of the thresholding function enables the approximation of the Hamming distance between two binary strings $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$ with the Euclidean distance between their corresponding embeddings $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$. The benefit of such an approximation is to avoid the more time consuming discrete optimization of the hash code, by constructing a differentiable cost function to directly optimize the embeddings, so that the hash code strings are indirectly optimized.

B. Retrieval Score via Space Matching

In a retrieval system, the returned result is a set of objects that are close to a query object. Thus, an accurate similarity representation between objects plays a significant role in the

success of a retrieval system. As a result of hashing, the similarity between the ith and jth objects is evaluated by the Hamming distance between the two binary strings $\boldsymbol{h}_i$ and $\boldsymbol{h}_j$. As explained earlier, due to Eqs. (1) and (2), the Hamming distance can be closely approximated by the Euclidean distance between the two embedding vectors $\boldsymbol{z}_i$ and $\boldsymbol{z}_j$. Thus, the research focus becomes how to compute the optimal embeddings $\{\boldsymbol{z}_i\}_{i=1}^n$ from the multiple views $\{\mathbf{X}^{(g)}\}_{g=1}^m$ so that correct similarity information between objects can be reflected by the Euclidean distances between their corresponding embeddings. This results in two tasks: (1) how to encode the similarity structure based on $\{\mathbf{X}^{(g)}\}_{g=1}^m$ and (2) how to preserve such a structure in the embedded space through Euclidean distance.

1) Probabilistic Encoding of Similarity Structure: The goal is to construct an accurate representation to reflect the reliable similarity structure between the objects based on their multi-view feature representations $\{\mathbf{X}^{(g)}\}_{g=1}^m$. We start from the computation based on one single view by following the probabilistic relevance model for information retrieval in [40]. Given the ith object as a query, a conditional probability $p_{j|i}^{(g)}$ of returning the jth ($i \neq j$) object as its related object can be formulated by

$$p_{j|i}^{(g)} = \frac{\exp\left(-\frac{\left\|\boldsymbol{x}_i^{(g)} - \boldsymbol{x}_j^{(g)}\right\|_2^2}{2\sigma_{ig}^2}\right)}{\sum_{l\neq i}\exp\left(-\frac{\left\|\boldsymbol{x}_i^{(g)} - \boldsymbol{x}_l^{(g)}\right\|_2^2}{2\sigma_{ig}^2}\right)}, \tag{5}$$

where $\|\cdot\|_2$ denotes the l_2-norm, and the Gaussian parameter $\sigma_{ig} > 0$ controls how fast the probability $p_{j|i}^{(g)}$ vanishes over the Euclidean distance between two objects. The value of $p_{j|i}^{(g)}$ reflects the similarity information between the ith and jth objects under the gth view. The parameter value of σ_{ig} is selected by examining the Shannon entropy of $\mathbb{H}\left(\mathbf{P}_{\cdot|i}^{(g)}\right) = -\sum_{j\neq i} p_{j|i}^{(g)} \log_2 p_{j|i}^{(g)}$. When the entropy is equal to $\log_2 K$, where the integer K controls the upper bound of the number of the relevant objects of a given query and is set by the user, the value of σ_{ig} is shown to be a good choice [38], [40]. For a retrieval task, it is not of any interest for a query object to return itself as a relevant one, thus, it is assumed that $p_{i|i}^{(g)} = 0$ for all cases. All these conditional probabilities computed under different views constitute a set of $n \times n$ probability matrices $\left\{\mathbf{P}^{(g)} = \left[p_{j|i}^{(g)}\right]\right\}_{g=1}^m$ for the n objects, each indicating a relevance structure computed based on a feature representation $\mathbf{X}^{(g)}$.

For the specific application of NDVR, the n objects correspond to different representative keyframes extracted from different videos. Apart from the feature information provided by $\{\mathbf{X}^{(g)}\}_{g=1}^m$, whether the keyframes are extracted from the same video naturally contribute to the knowledge regarding the keyframes. Therefore, this information can be used to control the neighbor structure between the keyframes in the embedded space. By assuming a rewarding score of 1 to pick up the keyframes that are extracted from the same video as the query keyframe, we construct the matrix $\mathbf{P}^{(\mathrm{W})} = \left[p_{ij}^{(\mathrm{W})}\right]$ as

$$p_{ij}^{(\mathrm{W})} = \begin{cases} 1, & \text{if the keyframes } \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j (i \neq j) \text{ are} \\ & \text{extracted from the same video,} \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

It drives those keyframes from the same video to be related to each other.

When there is ground truth information available regarding to the relevance between objects, it is helpful to construct a proximity matrix between the objects by rewarding the truly related objects with a score of 1, while the non-related or unknown ones with a score of 0. A supervised proximity matrix $\mathbf{P}^{(\mathrm{S})} = \left[p_{ij}^{(\mathrm{S})}\right]$ can be constructed to facilitate the NDVR task, given as

$$p_{ij}^{(\mathrm{S})} = \begin{cases} 1, & \text{if the keyframes } \boldsymbol{x}_i \text{ and } \boldsymbol{x}_j (i \neq j) \text{ are} \\ & \text{extracted from near-duplicated videos,} \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

It lets the keyframes from the truly related videos to be related to each other.

All these matrices of $\left\{\mathbf{P}^{(g)}\right\}_{g=1}^m$, $\mathbf{P}^{(\mathrm{W})}$ and $\mathbf{P}^{(\mathrm{S})}$ can be treated as relevance matrices with each element representing a relevance score between 0 and 1. A soft voting scheme can be implemented by constructing an overall relevance matrix $\mathbf{P} = [p_{ij}]$ representing an accumulation of the relevance scores offered by different views, such that

$$\mathbf{P} = N\left(\sum_{g=1}^m \alpha_g \mathbf{P}^{(g)} + \alpha_{m+1}\mathbf{P}^{(\mathrm{W})} + \alpha_{m+2}\mathbf{P}^{(\mathrm{S})}\right), \tag{8}$$

where the summation weights $\{\alpha_g\}_{g=1}^{m+2}$ are all positive and satisfy $\sum_{g=1}^{m+2}\alpha_g = 1$. Given an input matrix $\mathbf{A} = [a_{ij}]$, the function $N(\cdot)$ normalizes each of its rows $\sum_j a_{ij} = 1$, so that the resulting matrix $\mathbf{P}$ can be viewed as a conditional probability matrix with the ijth element $p_{j|i}$ representing the probability of returning the jth object as the related one to the query object i.

2) Structure Matching in the Embedded Space: The constructed matrix $\mathbf{P}$ as in Eq. (8) retains the structural information between objects offered by all the views. The embeddings $\{\boldsymbol{z}_i\}_{i=1}^n$ to be learned should be able to preserve optimally the relevance structure contained by $\mathbf{P}$. Given the fact that each element of $\mathbf{P}$ represents the probability of returning the jth object given the query object i, a natural way to learn the embeddings is to re-compute such probabilities in the embedded space and minimize the difference between the two sets of probabilities. Specifically, we use the following probability formulation in the embedded space

$$q_{j|i} = \frac{\exp\left(-\left\|\boldsymbol{z}_i - \boldsymbol{z}_j\right\|_2^2\right)}{\sum_{l\neq i}\exp\left(-\left\|\boldsymbol{z}_i - \boldsymbol{z}_l\right\|_2^2\right)}. \tag{9}$$

Slightly different from Eq. (5), a fixed scaling is adopted in the embedded space without introducing the Gaussian parameter σ_i. It is not necessary to scale the distances in both spaces, since similar effects can be achieved by scaling one and

fixing the other. These newly computed quantities constitute another probability matrix $\mathbf{Q} = [q_{j|i}]$. In order to preserve the information contained by $\mathbf{P}$, the quality of the embeddings can be assessed by examining how well the structures of $\mathbf{P}$ and $\mathbf{Q}$ match.

Following the structure matching scores as used in [38]–[40], we employ a composite KL divergence score to assess the difference between the two conditional probability matrices of $\mathbf{P}$ and $\mathbf{Q}$, given as

$$\begin{aligned} S_{\mathrm{KL}} &= \lambda \sum_{i=1}^{n} \mathrm{KL}\left(\boldsymbol{p}_{\cdot|i} \parallel \boldsymbol{q}_{\cdot|i}\right) + (1-\lambda) \sum_{i=1}^{n} \mathrm{KL}\left(\boldsymbol{q}_{\cdot|i} \parallel \boldsymbol{p}_{\cdot|i}\right) \\ &= \lambda \sum_{i=1}^{n} \sum_{j \neq i} p_{j|i} \log \frac{p_{j|i}}{q_{j|i}} + (1-\lambda) \sum_{i=1}^{n} \sum_{j \neq i} q_{j|i} \log \frac{q_{j|i}}{p_{j|i}}, \end{aligned} \tag{10}$$

where $0 < \lambda < 1$. Under the binary neighborhood assumption, minimizations of the the terms $\left\{\mathrm{KL}\left(\boldsymbol{p}_{\cdot|i} \parallel \boldsymbol{q}_{\cdot|i}\right)\right\}_{i=1}^{n}$ and $\left\{\mathrm{KL}\left(\boldsymbol{q}_{\cdot|i} \parallel \boldsymbol{p}_{\cdot|i}\right)\right\}_{i=1}^{n}$ are equivalent to the maximizations of the generalizations of recall and precision, respectively, for a retrieval task [40]. The overall score S_{KL} contains the smoothed recall and precision averaged over all the n observed objects. In most real-world retrieval applications, it is difficult for a system to achieve the maximization of both the precision and recall simultaneously, thus, the parameter λ is used to control the system preference between its accuracy and completeness in search.

C. Model Optimization

According to the mapping functions in Eqs. (2) and (3), the computation of the embeddings is driven by the weight parameters $\{w_{lj}^{(g)}\}_{l,j,g}$ and the bias parameters $\{b_l\}_l$. Therefore, the discovery of the optimal embeddings can be converted to the minimization problem of the composite KL divergence score with respect to the weight and bias parameters. Incorporating Eqs. (2) and (3) into Eq. (10) and introducing a regularization term, we construct the following optimization problem

$$\min_{w_{lj}^{(g)}, b_l} O = S_{\mathrm{KL}}\left(w_{lj}^{(g)}, b_l\right) + \frac{\mu}{2} \sum_{g=1}^{m} \sum_{l=1}^{s} \sum_{j=1}^{d_g} \left(w_{lj}^{(g)}\right)^2, \tag{11}$$

where $\mu > 0$ is the user-set regularization parameter. The objective function O is smooth and differentiable. A gradient descent algorithm can be employed to find a good solution.

To compute the gradient, we first decompose the objective functions into multiple components as

$$O = \lambda \mathrm{KL}_1 + (1-\lambda)\mathrm{KL}_2 + \mu O_r, \tag{12}$$

$$\mathrm{KL}_1 = \sum_{i=1}^{n} \sum_{t \neq i} p_{t|i} \log \frac{p_{t|i}}{q_{t|i}}, \tag{13}$$

$$\mathrm{KL}_2 = \sum_{i=1}^{n} \sum_{t \neq i} q_{t|i} \log \frac{q_{t|i}}{p_{t|i}}, \tag{14}$$

$$O_r = \frac{1}{2} \sum_{g=1}^{m} \sum_{l=1}^{s} \sum_{j=1}^{d_g} \left(w_{lj}^{(g)}\right)^2. \tag{15}$$

Then, following the compound function derivation law, we have

$$\frac{\partial O}{\partial w_{lj}^{(g)}} = \left[\lambda \frac{\partial \mathrm{KL}_1}{\partial z_{il}} + (1-\lambda) \frac{\partial \mathrm{KL}_2}{\partial z_{il}}\right] \frac{\partial z_{il}}{\partial w_{lj}^{(g)}} + \mu w_{lj}^{(g)}, \tag{16}$$

$$\frac{\partial O}{\partial b_l} = \left[\lambda \frac{\partial \mathrm{KL}_1}{\partial z_{il}} + (1-\lambda) \frac{\partial \mathrm{KL}_2}{\partial z_{il}}\right] \frac{\partial z_{il}}{\partial b_l}. \tag{17}$$

It can be seen that the targeted gradients depend on different components of $\frac{\partial \mathrm{KL}_1}{\partial z_{il}}$, $\frac{\partial \mathrm{KL}_2}{\partial z_{il}}$, $\frac{\partial z_{il}}{\partial w_{lj}^{(g)}}$ and $\frac{\partial z_{il}}{\partial b_l}$.

Letting $\frac{\partial \mathrm{KL}_1}{\partial \boldsymbol{z}_i} = \left[\frac{\partial \mathrm{KL}_1}{\partial z_{i1}}, \frac{\partial \mathrm{KL}_1}{\partial z_{i2}}, \ldots, \frac{\partial \mathrm{KL}_1}{\partial z_{il}}, \ldots, \frac{\partial \mathrm{KL}_1}{\partial z_{is}}\right]^T$ and $\frac{\partial \mathrm{KL}_2}{\partial \boldsymbol{z}_i} = \left[\frac{\partial \mathrm{KL}_2}{\partial z_{i1}}, \frac{\partial \mathrm{KL}_2}{\partial z_{i2}}, \ldots, \frac{\partial \mathrm{KL}_2}{\partial z_{il}}, \ldots, \frac{\partial \mathrm{KL}_2}{\partial z_{is}}\right]^T$, we conduct the computation of $\frac{\partial \mathrm{KL}_1}{\partial z_{il}}$ and $\frac{\partial \mathrm{KL}_2}{\partial z_{il}}$ by operating on the vector level. Two auxiliary variables d_{it} and U_j are introduced, given as

$$d_{it} = \left\|\boldsymbol{z}_i - \boldsymbol{z}_t\right\|_2, \tag{18}$$

$$U_j = \sum_{k \neq j} \exp\left(-\left\|\boldsymbol{z}_j - \boldsymbol{z}_k\right\|_2^2\right) = \sum_{k \neq j} \exp\left(-d_{jk}^2\right), \tag{19}$$

which simplify the following quantities as

$$q_{t|i} = \frac{\exp\left(-d_{it}^2\right)}{U_i}, \tag{20}$$

$$\log q_{t|i} = -d_{it}^2 - \log U_i. \tag{21}$$

Although d_{it} and d_{ti} possess exactly the same formulation, they are treated as two independent terms in the cost function, both of which involve $\boldsymbol{z}_i$. The gradient computation further proceed as

$$\frac{\partial \mathrm{KL}_1}{\partial \boldsymbol{z}_i} = \frac{\partial \mathrm{KL}_1}{\partial d_{it}} \frac{\partial d_{it}}{\partial \boldsymbol{z}_i} + \frac{\partial \mathrm{KL}_1}{\partial d_{ti}} \frac{\partial d_{ti}}{\partial \boldsymbol{z}_i}, \tag{22}$$

$$\frac{\partial \mathrm{KL}_2}{\partial \boldsymbol{z}_i} = \frac{\partial \mathrm{KL}_2}{\partial d_{it}} \frac{\partial d_{it}}{\partial \boldsymbol{z}_i} + \frac{\partial \mathrm{KL}_2}{\partial d_{ti}} \frac{\partial d_{ti}}{\partial \boldsymbol{z}_i}. \tag{23}$$

We first focus on computing $\frac{\partial \mathrm{KL}_1}{\partial d_{it}}$ and $\frac{\partial \mathrm{KL}_2}{\partial d_{it}}$. Ignoring the constant terms in KL_1 with respect to d_{it}, it has

$$\frac{\partial \mathrm{KL}_1}{\partial d_{it}} = \sum_{k \neq i} -p_{k|i} \frac{\partial\left(\log q_{k|i}\right)}{\partial d_{it}}. \tag{24}$$

Following a similar routine of ignoring the constant terms, we have

$$\begin{aligned} \frac{\partial \mathrm{KL}_2}{\partial d_{it}} &= \sum_{k \neq i} \left[\frac{\partial\left(q_{k|i} \log q_{k|i}\right)}{\partial d_{it}} - \frac{\partial\left(q_{k|i} \log p_{k|i}\right)}{\partial d_{it}}\right] \\ &= \sum_{k \neq i} \log q_{k|i} \frac{\partial q_{k|i}}{\partial d_{it}} + q_{k|i} \frac{\partial\left(\log q_{k|i}\right)}{\partial d_{it}} - \log p_{k|i} \frac{\partial q_{k|i}}{\partial d_{it}} \\ &= \sum_{k \neq i} \left(\log q_{k|i} + 1 - \log p_{k|i}\right) \times q_{k|i} \times \frac{1}{q_{k|i}} \times \frac{\partial q_{k|i}}{\partial d_{it}} \\ &= \sum_{k \neq i} \left(q_{k|i} \log \frac{q_{k|i}}{p_{k|i}} + q_{k|i}\right) \frac{\partial\left(\log q_{k|i}\right)}{\partial d_{it}} \end{aligned} \tag{25}$$

It can be seen that both derivatives in Eqs. (24) and (25) are depending on $\frac{\partial\left(\log q_{k|i}\right)}{\partial d_{it}}$, which can be calculated as follows after incorporating Eqs. (20) and (21):

$$\begin{aligned}\frac{\partial\left(\log q_{k|i}\right)}{\partial d_{it}} &= \frac{\partial\left(-d_{ik}^2\right)}{\partial d_{it}} + \frac{\partial\left(-\log U_i\right)}{\partial d_{it}} \\ &= \begin{cases} -2d_{it} - \frac{1}{U_i}\frac{\partial U_i}{\partial d_{it}}, & \text{if } k = t, \\ -\frac{1}{U_i}\frac{\partial U_i}{\partial d_{it}}, & \text{if } k \neq t, \end{cases} \\ &= \begin{cases} -2d_{it} - \frac{1}{U_i}\exp\left(-d_{it}^2\right)\left(-2d_{it}\right), & \text{if } k = t, \\ -\frac{1}{U_i}\exp\left(-d_{it}^2\right)\left(-2d_{it}\right), & \text{if } k \neq t, \end{cases} \\ &= \begin{cases} -2d_{it} + 2q_{t|i}d_{it}, & \text{if } k = t, \\ 2q_{t|i}d_{it}, & \text{if } k \neq t. \end{cases} \end{aligned} \tag{26}$$

By incorporating Eq. (26) into Eqs. (24) and (25), and utilizing $\sum_{k\neq i} p_{k|i} = 1$ and $\sum_{k\neq i} q_{k|i} = 1$, we obtain

$$\frac{\partial \mathrm{KL}_1}{\partial d_{it}} = 2p_{t|i}d_{it} - 2q_{t|i}d_{it}\sum_{k\neq i} p_{k|i} = 2(p_{t|i} - q_{t|i})d_{it}, \tag{27}$$

and

$$\begin{aligned}\frac{\partial \mathrm{KL}_2}{\partial d_{it}} =& -2d_{it}\left(q_{t|i}\log\frac{q_{t|i}}{p_{t|i}} + q_{t|i}\right) \\ &+ 2q_{t|i}d_{it}\sum_{k\neq i}\left(q_{k|i}\log\frac{q_{k|i}}{p_{k|i}} + q_{k|i}\right) \\ =& 2\left(\sum_{k\neq i} q_{k|i}\log\frac{q_{k|i}}{p_{k|i}} - \log\frac{q_{t|i}}{p_{t|i}}\right)q_{t|i}d_{it}. \end{aligned} \tag{28}$$

As shown above, $\frac{\partial \mathrm{KL}_1}{\partial d_{ti}}$ and $\frac{\partial \mathrm{KL}_2}{\partial d_{ti}}$ can be computed from Eqs. (27) and (28), and it is easy to calculate

$$\frac{\partial d_{it}}{\partial \boldsymbol{z}_i} = \frac{\partial d_{ti}}{\partial \boldsymbol{z}_i} = \frac{\boldsymbol{z}_i - \boldsymbol{z}_t}{d_{it}}. \tag{29}$$

Substituting these into Eqs. (22) and (23), it results in

$$\frac{\partial \mathrm{KL}_1}{\partial \boldsymbol{z}_i} = 2(p_{t|i} + p_{i|t} - q_{t|i} - q_{i|t})(\boldsymbol{z}_i - \boldsymbol{z}_t), \tag{30}$$

and

$$\begin{aligned}\frac{\partial \mathrm{KL}_2}{\partial \boldsymbol{z}_i} =& 2\left(q_{t|i}\sum_{k\neq i} q_{k|i}\log\frac{q_{k|i}}{p_{k|i}} + q_{i|t}\sum_{k\neq t} q_{k|t}\log\frac{q_{k|t}}{p_{k|t}}\right. \\ &\left. - \log\frac{q_{t|i}}{p_{t|i}} - \log\frac{q_{i|t}}{p_{i|t}}\right)(\boldsymbol{z}_i - \boldsymbol{z}_t). \end{aligned} \tag{31}$$

Now, the computation remains $\frac{\partial z_{il}}{\partial w_{lj}^{(g)}}$ and $\frac{\partial z_{il}}{\partial b_l}$. Based on Eqs. (2) and (3), it can be easily obtained that

$$\frac{\partial z_{il}}{\partial w_{lj}^{(g)}} = \mathrm{sigmoid}\left(\tilde{z}_{il}\right)\left[1 - \mathrm{sigmoid}\left(\tilde{z}_{il}\right)\right]x_{ij}^{(g)}, \tag{32}$$

$$\frac{\partial z_{il}}{\partial b_l} = \mathrm{sigmoid}\left(\tilde{z}_{il}\right)\left[1 - \mathrm{sigmoid}\left(\tilde{z}_{il}\right)\right]. \tag{33}$$

By substituting Eqs. (30), (31), (32) and (33) into Eqs. (16) and (17), the complete formulations of $\frac{\partial O}{\partial w_{lj}^{(g)}}$ and $\frac{\partial O}{\partial b_l}$ can be derived. With gradient descent optimization, the embeddings $\{\boldsymbol{z}_i\}_{i=1}^n$ can be computed, then the video hash code can be generated base on Eq. (4). We provide the pseudocode for the proposed multi-view hashing method in Algorithm 1.

Algorithm 1 Stochastic Multi-view Hashing (SMVH)

Input: n keyframes $\left\{\boldsymbol{x}_i^{(g)}\right\}_{i=1}^n$ extracted from V videos represented by m types of d_g-dimensional features ($g = 1, 2, \ldots m$).

Output: The $V \times s$ video hash code matrix $\mathbf{H}^{(v)}$.

Algorithm parameters: Hash code length s, neighbor bound K, balancing parameter λ, regularization parameter μ, multi-view weights $\{\alpha_i\}_{i=1}^{m+2}$.

Optimization parameters: Iteration number T, learning rate η and momentum $\zeta(t)$.

Initialization: Assign random values to the weight $\left\{w_{l,j}^{(g,0)}\right\}$ and bias variables $\left\{b_l^{(0)}\right\}$.

for $t = 1$ to T **do**

Compute gradient $\frac{\partial O}{\partial w_{lj}^{(g,t)}}$.

Compute gradient $\frac{\partial O}{\partial b_l^{(t)}}$.

Set the updates:

$w_{lj}^{(g,t+1)} = w_{lj}^{(g,t)} + \eta\frac{\partial O}{\partial w_{lj}^{(g,t)}} + \zeta(t)\left(w_{lj}^{(g,t)} - w_{lj}^{(g,t-1)}\right)$.

$b_l^{(t+1)} = b_l^{(t)} + \eta\frac{\partial O}{\partial b_l^{(t)}} + \zeta(t)\left(b_l^{(t)} - b_l^{(t-1)}\right)$.

end for

Hash code computation: Obtain the video hash code by Eqs. (2), (3) and (4).

After characterizing each video with a unique hash code string, only the efficient XOR and bit count operations are needed for the video retrieval task. This can avoid the costly pairwise keyframe comparison and effectively improve the retrieval efficiency. Overall, the training phase of the developed NDVR system is conducted off-line, so that it does not affect the online retrieval speed. In the training phase, the computational complexity of obtaining the composite probability matrix $\mathbf{P}$ is approximately $O(mn^2+2n^2)$ and only needs to be computed once before the optimization starts. The complexity of the weight and bias parameter updating procedure is approximately $O(n^3)$ in each iteration with standard gradient descent approach. It can be seen that the training cost is dominated by the number of the used representative keyframes from the training videos. The training cost can be further reduced when stochastic gradient descent is performed, where the gradient is estimated from a mini batch of the keyframes instead of all and this leads to a reduced cost of $O(n_s^3)$ ($n_s \ll n$) in each iteration with n_s denoting the batch size. The trained NDVR system is able to convert any new video to a hash code string very rapidly. This is because (1) the proposed system only employs very simple HSV and LBP features that can be computed much faster than the more expensive features such as DOG, SIFT, PCA_SIFT and some mixed features; and (2) the hash code is computed based on very simple operations such as linear combination, sigmoid and thresholding with a low cost of around $O(1)$. This, being combined with the fast XOR and bit count operations for hamming distance calculation, leads to a very fast online NDVR system. Its efficiency will be demonstrated in the results section.

V. Experiments and Results

A. Datasets and Experimental Setup

We examine the performance of the NDVR system constructed based on the proposed hashing algorithm SMVH using two publicly available web video datasets. The CC_WEB_VIDEO dataset [1] consists of 12,790 video clips downloaded from the video sharing websites such as YouTube, Google and Yahoo! through keyword search, and is organized into 24 sets. Within each set, the most popular video is used as the query, and the remaining videos were manually labeled by two non-expert assessors to create ground truth. Shot boundaries of each video are detected and each shot is represented by a keyframe[1] , which results in a total of 398,015 keyframes. There are 162 HSV and 256 LBP features extracted for each keyframe. The UQ_VIDEO dataset [7] expands the CC_WEB_VIDEO dataset with YouTube videos dowloaded by searching against 400 most popular queries selected from the Google Zeitgeist Archives from 2004 to 2009. This results in a total of 169,952 videos, for which 2,570,554 keyframes are extracted, and 162 HSV and 256 LBP features are used to characterize each keyframe.

Comparative analysis is conducted against a set of existing NDVR systems constructed using either the classic feature extraction methods or the state-of-the-art hashing methods:

- **Global Feature** (GF): This GF-based NDVR system [1] is used as a baseline for performance comparison. It employs the most typical global feature of color histogram to characterize the keyframes. A 24-dimensional vector based on a normalized color histogram over all the keyframes in each video is used as the video signature.
- **Hierarchical Fusing** (HF): This system [1] combines the global and local features, by firstly using the color histogram signature to detect the NDVs with high confidence and filtering out the very novel ones, and then performing a pairwise comparison based on the local features to further determine the uncertain videos. When this system is implemented using 36-dimensional rotation invariant LBP features [1], it is referred to as HF-36.
- **Spectral Hashing** (SPH): This system relies on the hashing method SPH [46], which is based on spectral relaxation and rectangular approximation of the eigenfunction of the weighted Laplacian operator. It includes the HSV and LBP features within a vector to be used as the input of SPH.
- **Self-taught Hashing** (STH): This system relies on the hashing method STH [47], which shares similar hash code training procedure to SPH, but achieves out-of-sample extension through a different scheme based on linear SVM. Similar to above, the HSV and LBP features are included within a vector as the input of STH.
- **Multiple Feature Hashing** (MFH): This system is based on a sophisticated multi-view extension of SPH, referred to as MFH [2], [7]. It encodes the information provided by the HSV and LBP features as a neighbor graph and seeks a hash function to preserve the desired neighbor structure. A semi-supervised extension of MFH is also implemented by utilizing the ground truth information to improve the neighbor adjacency graph, referred as SMFH.
- **Proposed SMVH**: This system is based on the proposed hashing method SMVH. An unsupervised version of SMVH is also implemented by setting $\alpha_{m+2} = 0$, referred as USMVH. The same HSV and LBP features as used by the existing methods are employed as the input of SMVH.

[1]The shot-based extraction and uniform sampling are the two most commonly used keyframe extraction methods for videos in NDVR [2], [7]. Apart from detecting the shot boundaries, we also experiment with uniform sampling, of which the extracted keyframes are used to perform the same retrieval task as the shot-based ones. We compare these two keyframe extraction methods for both the proposed hashing method and the state-of the-art method multiple feature hashing [2], [7] in Table I. It is observed that shot-based extraction outperforms uniform sampling with subtle difference, and we use shot-based keyframe extraction to conduct all the remaining experiments in this work.

TABLE I: Comparison of keyframe extraction methods of shot based and uniform sampling using randomly selected query sets of CC_WEB_VIDEO data.

Methods	Q1	Q5	Q9	Q14	Q20	Mean
shot (proposed)	0.999	0.950	1.000	0.974	0.962	0.977
uniform (proposed)	0.974	0.968	0.984	0.974	0.937	0.967
shot (MFH)	0.991	0.944	0.999	0.977	0.903	0.963
uniform (MFH)	0.994	0.953	0.988	0.960	0.903	0.960

TABLE II: Parameter setting for the proposed algorithm.

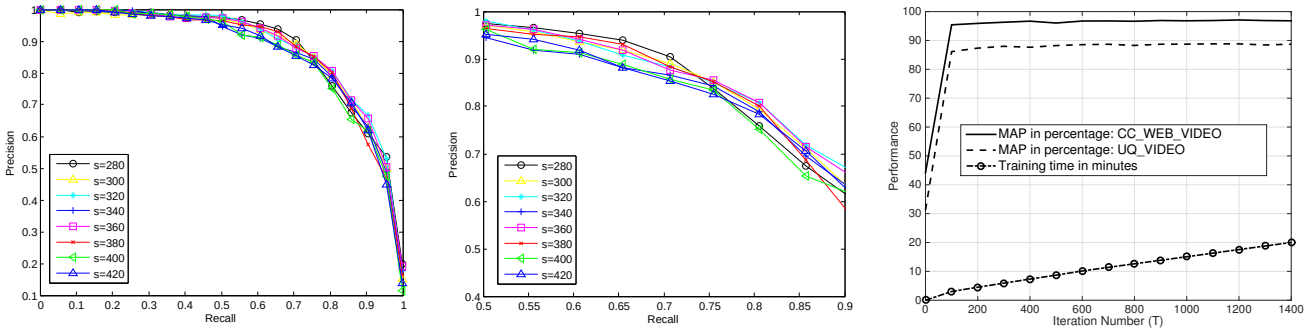
Optimization param.	Value	Algorithm param.	Value
T	1200	λ	0.9
η	0.05	μ	0.01
$\zeta\,(t)\,(t < 250)$	0.5	s	320
$\zeta\,(t)\,(t \geq 250)$	0.75	K	20

TABLE III: SMVH performance change given varying length of the hash code evaluated using the UQ_VIDEO data.

Hash Code Length (s)	MAP	Time(s)
s=280	0.8782	0.0531
s=300	0.8803	0.0568
s=320	0.8882	0.0592
s=340	0.8738	0.0623
s=360	0.8865	0.0645
s=380	0.8777	0.0679
s=400	0.8661	0.0703
s=420	0.8698	0.0737

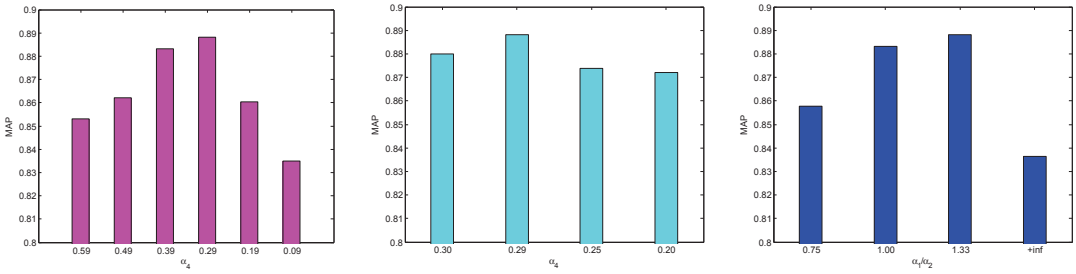
Although MFH, SMFH, USMVH and SMVH are designed to process multi-view information, it is also important to observe and compare how they respond to one single feature type. This indicates the information preservation power of the learned hash code. The single-view implementation takes only the HSV features as the algorithm input, referred as MFH-HSV, SMFH-HSV, USMVH-HSV and SMVH-HSV. To evaluate the retrieval performance, the classic metric of the mean average precision (MAP) commonly used in the NDVR community is employed [1], [7]. The precision-recall curve is used to provide a more thorough view of the retrieval performance.

For SMVH and its different versions, the optimization parameter setting as listed in the left column of Table II is adopted, which are set by following the empirical recommendations for gradient descent optimization [39]. The parameter η is initially set as 0.05 and then updated in each iteration following the adaptive learning rate scheme in [63]. The algo-



(a) precision-recall curve (b) enlarged portion (c) effect of iteration number T

Fig. 1: (a) compares the precision-recall curve with varying values of the hash code length s for the SMVH-based retrieval system examined using the UQ_VIDEO data. (b) displays an enlarged portion of (a). (3) demonstrates the effect of iteration number T in terms of the MAP performance in percentage for both video collections and the training time in minutes.



(a) $\alpha_1 = \alpha_2$, $\alpha_3 = 0.01$. (b) $\alpha_1 = \alpha_2 = 0.35$. (c) $\alpha_3 = 0.01$, $\alpha_4 = 0.29$.

Fig. 2: Comparison of MAP performance against different settings of the weight parameters for the SMVH-based retrieval system using the UQ_VIDEO data.

rithm parameter setting listed in the right column of Table II is used, where the settings of the balancing parameter λ and the neighbor bound K follow the empirical recommendations for stochastic neighbor preservation [40]. The regularization parameter μ does not affect the performance much when it is within a reasonable range. The hash code length s is tuned from 100 and 420 with a step size of 20. To demonstrate the performance sensitivity against the hash code length s of the SMVH-based retrieval system, we show in Table III and Fig. 1(a) the changes of the MAP performance and the precision-recall curves for different values of s using the UQ_VIDEO data. An enlarged portion of Fig. 1(a) is displayed in Fig. 1(b). It can be seen that within a certain range, e.g., $280 \leq s \leq 380$, different lengths actually provide quite similar retrieval performance. The retrieval time though can differ according to the used hash code length and the employed binarization scheme of the relaxed hash code (embedding). It can be seen from Fig. 1(a) that a longer hash code reduces the retrieval speed. We report in Fig. 1(c) how the setting of the iteration number T affects the retrieval performance and the training speed. It can be seen that the MAP performance for both video collections stabilizes after around 150 iterations, and of course larger iteration number requires higher training time. Although we employ the suggested iteration number setting $T = 1200$ by [39], it is actually sufficient to use a smaller number of T to accelerate the training without sacrificing much the retrieval performance.

For the multi-view weights $\{\alpha_g\}_{g=1}^{m+2}$ ($m = 2$), instead of exhaustively searching a good setting within the range of $[0, 1]^4$, we first determine a preliminary weight arrangement by applying a traditional pairwise comparison [1] in each feature space among a small collection of videos. Specifically, the duplicate keyframes are first identified under each feature type g for this small collection, and then the number of shared duplicate keyframes is computed by

$$s_{ij}^{(g)} = \frac{1}{2}\left(\frac{\left|I_i^{(g)} \cap I_j^{(g)}\right|}{\left|I_i^{(g)}\right|} + \frac{\left|I_i^{(g)} \cap I_j^{(g)}\right|}{\left|I_j^{(g)}\right|}\right), \tag{34}$$

where $I_i^{(g)}$ denotes the duplicate keyframe set contained by the ith video. Given a targeted similarity s_{ij} between two videos (e.g., the ground truth relevance information corresponding to a binary value representing whether the jth video is retrieved given the ith video as a query), a linear relationship between s_{ij} and $\{s_{ij}^{(g)}\}_{g=1}^{m}$ can be used to approximate the contributions of the different feature types to achieve a good approximation of s_{ij}. By restricting the linear combination coefficients as nonnegative values, the contribution estimation can be realized by solving the following nonnegative least squares problem

$$\min_{\{w_g \geq 0\}_{g=1}^{m}} \sum_{i \neq j} \left(s_{ij} - \sum_{g=1}^{m} w_g s_{ij}^{(g)}\right)^2. \tag{35}$$

TABLE IV: Performance comparison in terms of the MAP performance and the retrieval speed. The best performance and speed are highlighted in bold and the second best underlined. HF and HF-36 are only examined using the comparatively small data CC_WEB_VIDEO due to their low speed.

			CC_WEB_VIDEO		UQ_VIDEO	
Methods	Feature	Type	MAP	Time (10^{-4} s)	MAP	Time (s)
GF [1]	HSV	unsupervised	0.892	25.0	0.640	0.2190
MFH-HSV [7]	HSV	unsupervised	0.918	<u>4.81</u>	0.715	0.0362
SMFH-HSV	HSV	semi-supervised	0.920	4.81	0.718	0.0363
USMVH-HSV	HSV	unsupervised	0.934	**4.61**	0.787	<u>0.0306</u>
SMVH-HSV	HSV	semi-supervised	0.951	**4.61**	0.836	**0.0305**
HF [1]	HSV, PCA-SIFT	unsupervised	0.952	>8000.0	—	—
HF-36 [1]	HSV, LBP	unsupervised	0.936	>8000.0	—	—
SPH [46]	HSV, LBP	unsupervised	0.864	6.14	0.546	0.0634
STH [47]	HSV, LBP	unsupervised	0.932	6.50	0.775	0.0642
MFH [7]	HSV, LBP	unsupervised	0.928	6.38	0.757	0.0679
SMFH	HSV, LBP	semi-supervised	0.936	6.39	0.766	0.0679
USMVH	HSV, LBP	unsupervised	<u>0.955</u>	6.06	<u>0.851</u>	0.0593
SMVH	HSV, LBP	semi-supervised	**0.971**	6.07	**0.888**	0.0592

Instead of supervised learning of $\{w_g\}_{g=1}^m$ based on ground truth relevance, it is also possible to conduct an unsupervised learning by employing a voted version of s_{ij} based on the multi-view features [64]. For instance, $s_{ij} = 1$ when more than 50% of the views agree that the jth video is retrieved given the ith video as a query, and $s_{ij} = 0$ otherwise. After obtaining $\{w_g\}_{g=1}^m$ by Eq. (35), the selection of the $m+2$ weight parameters $\{s_{ij}^{(g)}\}_{g=1}^{m+2}$ can be reduced to the selection of the two parameters of $0 < \alpha_{m+1} < 1$ and $1 - \alpha_{m+1} < \alpha_{m+2} < 1$, with which the remaining weights are computed by $\alpha_g = (1 - \alpha_{m+1} - \alpha_{m+2})\frac{w_g}{\sum_{g=1}^m w_g}$ for $g = 1, 2, \ldots m$. Solution of Eq. (35) indicates $w_1 \approx w_2$ for the used datasets, representing almost equal contributions of the HSV and LBP features. After letting $\frac{\alpha_1}{\alpha_2} = 1$, a rough search of α_3 and α_4 suggests $0.01 \leq \alpha_3 \leq 0.1$ and $0.19 \leq \alpha_4 \leq 0.39$ offer good performance. Searching within these two suggested ranges and allowing $\frac{\alpha_1}{\alpha_2}$ to vary slightly around one, the fine tuning results in the final setting of $\alpha_1 = 0.4$, $\alpha_2 = 0.3$, $\alpha_3 = 0.01$ and $\alpha_4 = 0.29$ for SMVH. Similar parameter selection strategy leads to the setting of $\alpha_2 = 0$, $\alpha_1 = 0.7$, $\alpha_3 = 0.01$ and $\alpha_4 = 0.29$ for SMVH-HSV, and $\alpha_4 = 0$, $\alpha_1 = 0.55$, $\alpha_2 = 0.4$ and $\alpha_3 = 0.05$ for USMVH.

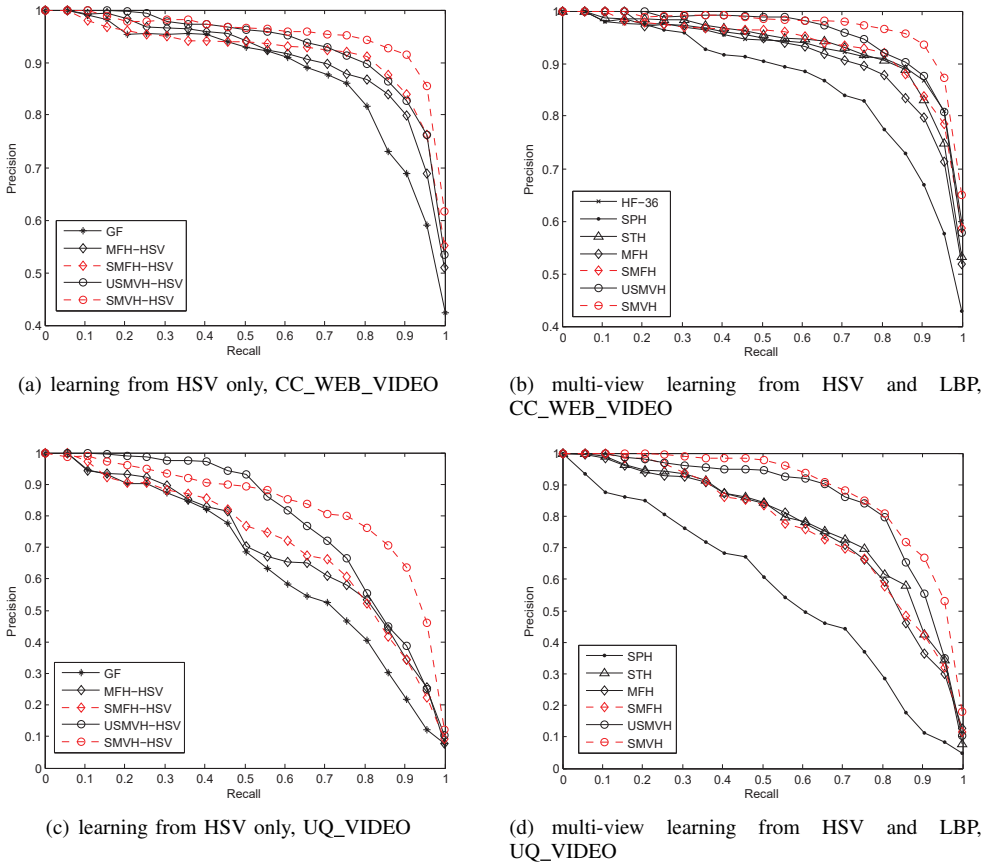
To demonstrate the algorithm sensitivity of SMVH, we display its performance change against different values of the weight parameters in Fig. 2 using the UQ_VIDEO data. In general, as expected, the ground truth information $\mathbf{P}^{(\mathrm{S})}$ is more important than the association information $\mathbf{P}^{(\mathrm{W})}$ between keyframes and videos. This is evidenced by the observation that $\alpha_4 > \alpha_3$ usually leads to better performance, as seen in Figs. 2(a) and 2(b). Both HSV and LBP features contribute significantly and similarly to the retrieval task. This is evidenced by the observation that, by setting $\frac{\alpha_1}{\alpha_2}$ around one, good performance can be achieved, as seen in Fig 2(c). It is worth to mention that, although the contribution weight of $\mathbf{P}^{(\mathrm{W})}$ is low, its participation is necessary, as we have observed that a small value of α_3 provides better performance than a zero α_3.

For the competing methods, we either conduct the implementation using the same setting as reported in their published works or employ the existing code provided by the authors. In all experiments, the semi-supervised training setup is implemented by randomly selecting 360 videos for training, where 240 videos (10 from each of the 24 video clip sets) are provided with the ground truth information and the remaining videos not. The unsupervised setup is implemented by randomly selecting 600 videos for training, where none of them is provided with the ground truth information. The hash code length is fixed as $s = 320$ for all the multi-view methods and $s = 100$ for ones using only HSV features in all the experiments. The online retrieval speed is computed using MATLAB R2013a running on a server with Intel Xeon E5-2630 2 CPUs, 32 GB RAM and 64-bit Windows Server 2012 operating system.

B. Comparative Analysis

Table IV summarizes the MAP performance of all the methods and records their averaged online retrieval speed in seconds for the two datasets. Because HF and HF-36 are time consuming in searching, we only examine their performance with the smaller dataset CC_WEB_VIDEO. The averaged precision-recall curves are displayed in Fig. 3, where the multi-view systems and the single-view systems are compared separately. A bar graph comparing the average precision (AP) over each query is provided in Fig. 4 for the UQ_VIDEO data.

It can be seen from Table IV and Fig. 3 that the proposed SMVH outperforms all the competing methods under all the tested learning environments, that is, single-view, multi-view, semi-supervised and unsupervised. For example, when learning from one feature type HSV in an unsupervised manner, USMVH-HSV provides better performance than MFH-HSV and much better performance than GF. When unsupervised multi-view learning is performed using both the HSV and LBP features, USMVH performs better than HF, STH, MFH, and much better than SPH. The poor performance of SPH is possibly caused by the non-uniform distribution of the video data within a hyper-rectangle in the feature space which is against the assumption required by SPH. When partially available ground truth information is incorporated to enhance the learning quality, the sem-supervised extensions of MFH/MFH-HSV



(a) learning from HSV only, CC_WEB_VIDEO

(b) multi-view learning from HSV and LBP, CC_WEB_VIDEO

(c) learning from HSV only, UQ_VIDEO

(d) multi-view learning from HSV and LBP, UQ_VIDEO

Fig. 3: Comparison of the averaged precision-recall curves for different methods and datasets.

and USMVH/USMVH-HSV in general perform better than the unsupervised ones, and overall, the proposed SMVH achieves the best performance. It can be seen from Fig. 3(d) that, for the large UQ_VIDEO data, SMVH is able to achieve over 90% precision given the recall values up to 70%. As shown in Table IV, the MAP performance achieved by the proposed system is 0.971 for CC_WEB_VIDEO and 0.888 for UQ_VIDEO, which is significantly better than the best performance of 0.952 for CC_WEB_VIDEO and 0.775 for UQ_VIDEO achieved by the competing methods. This is also better than the current state-of-the-art MAP performance reported by [7], which is 0.958 for CC_WEB_VIDEO and 0.883 for UQ_VIDEO. The system in [7] learns from 1000 unlabeled training videos, while ours learns from 240 labeled and 120 unlabeled training videos, all randomly chosen. Because both systems work on the scale of $O(n^3)$, heavily relying on the used keyframe number n during training, the much smaller training data size required by our system greatly improves the training speed. In terms of unsupervised learning, we are able to achieve comparable performance of 0.955 for CC_WEB_VIDEO and 0.851 for UQ_VIDEO by learning from only a small training set of 600 videos.

Table IV also compares the retrieval speed. All the hashing-based systems, such as SPH, STH, as well as MFH and SMVH and their corresponding variations, are able to achieve real-time retrieval, even when using MATLAB (less than 7×10^{-4} seconds in CC_WEB_VDIEO dataset). Both GF and HF rely on time consuming distance computation and searching strategies, and are thus not suitable for large-scale applications.

Finally, in Fig. 4 we summarize the AP performance of different methods over each of the 24 queries using the UQ_VIDEO data. For most queries, the multi-view methods such as STH, MFH, SMFH and SMVH perform better than the single-view methods such as GF. By incorporating partially available ground truth information, both the single-view and multi-view performance can be improved, which are exemplified by most queries (e.g., SMFH, SMFH-HSV, SMVH-HSV and SMVH). There exist a few individual cases such as Q5 and Q13 for which the multi-view performance is not better than the single-view one. Also Q11 and Q13, for which the semi-supervised learning does not improve the unsupervised one. However, this does not change the overall conclusion that, taking into account all the queries, the global and local features are in general complementary to each other in video representation, and that it is effective to combine both views, and moreover that it is helpful to perform semi-supervised learning to improve performance.

VI. Conclusion

We have proposed a novel stochastic multi-view hashing method (SMVH), to facilitate the construction of a large-scale NDVR system. The proposed method addresses the accuracy, efficiency and scalability issues that are very important in recent NDVR applications and contributes to the performance

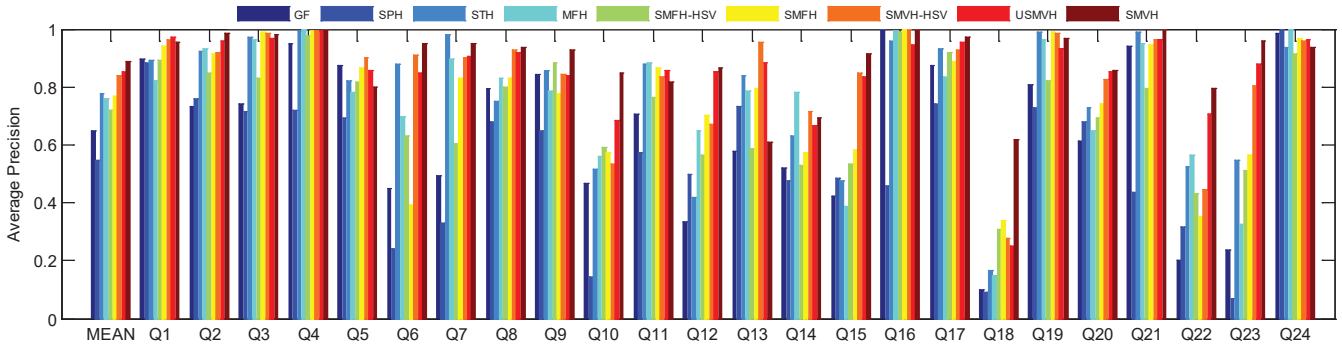


Fig. 4: Comparison of the AP performance of different methods over each query (Q1-Q24) evaluated using the UQ_VIDEO dataset. The first bar "MEAN" represents the averaged MAP performance over all the 24 queries.

improvement. A reliable mapping function is learned to convert the video content to a unique hash code signature so that a fast matching can be achieved by inexpensive XOR operations. In order to accurately retrieve the NDVs, multi-type feature information of the video keyframes is considered, and is further enhanced by auxiliary information such as association between videos and keyframes and partially available ground truth to further boost the system performance. Given that both the feature and auxiliary information can be extracted from the videos in efficient and straightforward ways, the main challenge remains how to accurately blend, refine and preserve such information using hash codes, in order to realize accurate and fast retrieval. To tackle the problem, a multi-view probabilistic relevance model is constructed, which accurately represents the stochastic neighbor structure between video keyframes in a target space. Moreover, a composite KL divergence score is used to align the constructed neighborhood structures in the original and the relaxed hash code spaces, which approximates the retrieval precision and recall scores to suit the particular nature of the retrieval task. This work, for the first time, contributes a multi-view and hashing extension of the stochastic neighbor embedding strategy that is particularly tailored for a large-scale retrieval task.

Performance of the developed NDVR system using the proposed hashing method is examined using public datasets based on various performance metrics, such as MAP and AP, as well as the precision-recall curves, tested under different learning environments with single-view, multi-view, unsupervised and semi-supervised setups. Extensive experiments verify the superior performance of the proposed system in terms of both effectiveness and efficiency. Compared with a series of classical and state-of-the-art NDVR systems, the proposed one provides the best MAP performance of 0.971 and 0.888 for the CC_WEB_VIDEO and UQ_VIDEO datasets, respectively, and achieves an on-line retrieval speed of no greater than 6×10^{-4} seconds, which is fast enough even under the MATLAB implementation. For the particularly large video collection UQ_VIDEO, over 90% precision has been achieved with a recall performance of up to 70%.

ACKNOWLEDGEMENTS

This work was supported by "The International Research Base for Developing Innovative Gerontechnology" sponsored by the national "111" project (No. B14025) of China. The authors would like to thank the three anonymous reviewers for their constructive comments and suggestions.

REFERENCES

[1] X. Wu, A. G. Hauptmann, and C.-W. Ngo, "Practical elimination of near-duplicates from web video search," pp. 218–227, 2007.
[2] J. Song, Y. Yang, Z. Huang, H. T. Shen, and R. Hong, "Multiple feature hashing for real-time large scale near-duplicate video retrieval," pp. 423–432, 2011.
[3] M. Cherubini, R. De Oliveira, and N. Oliver, "Understanding near-duplicate videos: a user-centric approach," pp. 35–44, 2009.
[4] H. T. Shen, X. Zhou, Z. Huang, J. Shao, and X. Zhou, "Uqlips: a real-time near-duplicate video clip detection system," pp. 1374–1377, 2007.
[5] H.-K. Tan, C.-W. Ngo, R. Hong, and T.-S. Chua, "Scalable detection of partial near-duplicate videos by visual-temporal consistency," pp. 145–154, 2009.
[6] J. Liu, Z. Huang, H. Cai, H. T. Shen, C. W. Ngo, and W. Wang, "Near-duplicate video retrieval: Current research and future trends," *ACM Computing Surveys (CSUR)*, vol. 45, no. 4, p. 44, 2013.
[7] J. Song, Y. Yang, Z. Huang, H. T. Shen, and J. Luo, "Effective multiple feature hashing for large-scale near-duplicate video retrieval," *IEEE Trans. on Multimedia*, vol. 15, no. 8, pp. 1997–2008, 2013.
[8] L. Shang, L. Yang, F. Wang, K.-P. Chan, and X.-S. Hua, "Real-time large scale near-duplicate web video retrieval," pp. 531–540, 2010.
[9] C.-W. Ngo, W.-L. Zhao, and Y.-G. Jiang, "Fast tracking of near-duplicate keyframes in broadcast domain with transitivity propagation," pp. 845–854, 2006.
[10] X. Wu, C.-W. Ngo, A. G. Hauptmann, and H.-K. Tan, "Real-time near-duplicate elimination for web video search with content and context," *IEEE Trans. on Multimedia*, vol. 11, no. 2, pp. 196–207, 2009.
[11] A. Hampapur, K. Hyun, and R. M. Bolle, "Comparison of sequence matching techniques for video copy detection," pp. 194–201, 2001.
[12] J. Law-To, L. Chen, A. Joly, I. Laptev, O. Buisson, V. Gouet-Brunet, N. Boujemaa, and F. Stentiford, "Video copy detection: a comparative study," pp. 371–378, 2007.
[13] M. Douze, H. Jégou, C. Schmid, and P. Pérez, "Compact video description for copy detection with precise temporal alignment," in *Computer Vision–ECCV 2010*, 2010, pp. 522–535.
[14] X. Liu, L. Huang, C. Deng, J. Lu, and B. Lang, "Multi-view complementary hash tables for nearest neighbor search," pp. 1107–1115, 2015.
[15] X. Shen, F. Shen, Q. Sun, and Y. Yuan, "Multi-view latent hashing for efficient multimedia search," pp. 831–834, 2015.
[16] L. Cao, X. Liu, W. Liu, R. Jia, and T. Huang, "Localizing web videos using social images," *Information Sciences*, vol. 302, pp. 122–131, 2015.
[17] X. Nie, Y. Yin, and J. Sun, "Comprehensive feature-based robust video fingerprinting using tensor model," *CoRR*, vol. abs/1601.07270, 2016.
[18] J. Chen, Y. He, and J. Wang, "Multi-feature fusion based fast video flame detection," *Building and Environment*, vol. 45, no. 5, pp. 1113–1122, 2010.
[19] J. Xu, V. Jagadeesh, and B. Manjunath, "Multi-label learning with fused multimodal bi-relational graph," *IEEE Trans. on Multimedia*, vol. 16, no. 2, pp. 403–412, 2014.

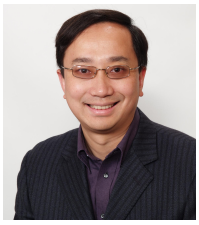
[20] J. Song, Y. Yang, Y. Yang, Z. Huang, and H. T. Shen, "Inter-media hashing for large-scale retrieval from heterogeneous data sources," pp. 785–796, 2013.
[21] M. M. Bronstein, A. M. Bronstein, F. Michel, and N. Paragios, "Data fusion through cross-modality metric learning using similarity-sensitive hashing." 2010.
[22] V. Ranjan, N. Rasiwasia, and C. V. Jawahar, "Multi-label cross-modal retrieval," pp. 4094–4102, 2015.
[23] S. Sun, "A survey of multi-view machine learning," *Neural Computing and Applications*, vol. 23, no. 7-8, pp. 2031–2038, 2013.
[24] X. Zhou, X. Zhou, L. Chen, A. Bouguettaya, N. Xiao, J. Taylor *et al.*, "An efficient near-duplicate video shot detection method using shot-based interest points," *IEEE Trans. on Multimedia*, vol. 11, no. 5, pp. 879–891, 2009.
[25] C.-L. Chou, H.-T. Chen, and S.-Y. Lee, "Pattern-based near-duplicate video retrieval and localization on web-scale videos," *IEEE Trans. on Multimedia*, vol. 17, no. 3, pp. 382–395, 2015.
[26] J. Yuan, L.-Y. Duan, Q. Tian, S. Ranganath, and C. Xu, "Fast and robust short video clip search for copy detection," in *Advances in Multimedia Information Processing-PCM 2004*, 2005, pp. 479–488.
[27] X.-S. Hua, X. Chen, and H.-J. Zhang, "Robust video signature based on ordinal measure," pp. 685–688, 2004.
[28] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.
[29] D.-G. Lowe, "Object recognition from local scale-invariant features," pp. 1150–1157, 1999.
[30] Y. Ke and R. Sukthankar, "Pca-sift: A more distinctive representation for local image descriptors," pp. 506–513, 2004.
[31] G. Zhao and M. Pietikainen, "Dynamic texture recognition using local binary patterns with an application to facial expressions," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 915–928, 2007.
[32] J. Law-To, V. Gouet-Branet, O. Buisson, and N. Boujemaa, "Local behaviours labelling for content based video copy detection," pp. 232–235, 2006.
[33] C.-W. Ngo, W.-L. Zhao, and Y.-G. Jiang, "Fast tracking of near-duplicate keyframes in broadcast domain with transitivity propagation," pp. 845–854, 2006.
[34] J. B. Tenenbaum, V. De Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.
[35] H. Hotelling, "Analysis of a complex of statistical variables into principal components." *Journal of educational psychology*, vol. 24, no. 6, p. 417, 1933.
[36] W. S. Torgerson, "Multidimensional scaling: I. theory and method," *Psychometrika*, vol. 17, no. 4, pp. 401–419, 1952.
[37] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.
[38] G. E. Hinton and S. T. Roweis, "Stochastic neighbor embedding," pp. 833–840, 2002.
[39] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, no. 2579-2605, p. 85, 2008.
[40] J. Venna, J. Peltonen, K. Nybo, H. Aidos, and S. Kaski, "Information retrieval perspective to nonlinear dimensionality reduction for data visualization," *The Journal of Machine Learning Research*, vol. 11, pp. 451–490, 2010.
[41] J. Wang, H. T. Shen, J. Song, and J. Ji, "Hashing for similarity search: A survey," *arXiv preprint arXiv:1408.2927*, 2014.
[42] A. Gionis, P. Indyk, R. Motwani *et al.*, "Similarity search in high dimensions via hashing," pp. 518–529, 1999.
[43] Y. Ke, R. Sukthankar, L. Huston, Y. Ke, and R. Sukthankar, "Efficient near-duplicate detection and sub-image retrieval," p. 5, 2004.
[44] W. Dong, Z. Wang, M. Charikar, and K. Li, "Efficiently matching sets of features with random histograms," pp. 179–188, 2008.
[45] J. Song, L. Gao, Y. Yan, D. Zhang, and N. Sebe, "Supervised hashing with pseudo labels for scalable multimedia retrieval," pp. 827–830, 2015.
[46] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," pp. 1753–1760, 2009.
[47] D. Zhang, J. Wang, D. Cai, and J. Lu, "Self-taught hashing for fast similarity search," pp. 18–25, 2010.
[48] R. Salakhutdinov and G. E. Hinton, "Learning a nonlinear embedding by preserving class neighbourhood structure," pp. 412–419, 2007.
[49] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," pp. 2074–2081, 2012.
[50] L. Gao, J. Song, F. Zou, D. Zhang, and J. Shao, "Scalable multimedia retrieval by deep learning hashing with relative similarity learning," pp. 903–906, 2015.
[51] J. Wang, T. Zhang, J. Song, N. Sebe, and H. T. Shen, "A survey on learning to hash," *arXiv preprint arXiv:1606.00185*, 2016.
[52] C. G. Snoek, M. Worring, and A. W. Smeulders, "Early versus late fusion in semantic video analysis," pp. 399–402, 2005.
[53] M. Wang, X.-S. Hua, R. Hong, J. Tang, G.-J. Qi, and Y. Song, "Unified video annotation via multigraph learning," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 19, no. 5, pp. 733–746, 2009.
[54] Y. Yang, J. Song, Z. Huang, Z. Ma, N. Sebe, and A. G. Hauptmann, "Multi-feature fusion via hierarchical regression for multimedia analysis," *IEEE Trans. on Multimedia*, vol. 15, no. 3, pp. 572–581, 2013.
[55] J. Yang, J.-y. Yang, D. Zhang, and J.-f. Lu, "Feature fusion: parallel strategy vs. serial strategy," *Pattern Recognition*, vol. 36, no. 6, pp. 1369–1381, 2003.
[56] Y. Fu, L. Cao, G. Guo, and T. S. Huang, "Multiple feature fusion by subspace learning," pp. 127–134, 2008.
[57] B. Cui, A. K. Tung, C. Zhang, and Z. Zhao, "Multiple feature fusion for social media applications," pp. 435–446, 2010.
[58] X. Yang, T. Zhang, and C. Xu, "Cross-domain feature learning in multimedia," *IEEE Trans. on Multimedia*, vol. 17, no. 1, pp. 64–78, 2015.
[59] Y. Hu, Z. Jin, H. Ren, D. Cai, and X. He, "Iterative multi-view hashing for cross media indexing," pp. 527–536, 2014.
[60] F. Zou, Y. Chen, J. Song, K. Zhou, Y. Yang, and N. Sebe, "Compact image fingerprint via multiple kernel hashing," *IEEE Transactions on Multimedia*, vol. 17, no. 7, pp. 1006–1018, 2015.
[61] L. Gao, J. Song, F. Nie, Y. Yan, N. Sebe, and H. Tao Shen, "Optimal graph learning with partial tags and multiple features for image and video annotation," pp. 4371–4379, 2015.
[62] X. Zhu, Z. Huang, H. T. Shen, and X. Zhao, "Linear cross-modal hashing for efficient multimedia search," pp. 143–152, 2013.
[63] R. A. Jacobs, "Increased rates of convergence through learning rate adaptation," *Neural networks*, vol. 1, no. 4, pp. 295–307, 1988.
[64] X. Gao, T. Mu, and M. Wang, "Local voting based multiview embedding," *Neurocomputing*, vol. 171, pp. 901–909, 2016.

Y. Hao is a Ph.D. student in School of Computer and Information, Hefei University of Technology. His research interests mainly include machine learning and multimedia data analysis, such as large-scale multimedia indexing and retrieval, and multimedia data embedding.

T. Mu received the B.Eng. degree in electronic engineering and information science from the University of Science and Technology of China, Hefei, China, in 2004, and the Ph.D. degree in electrical engineering and electronics from the University of Liverpool in 2008. She is currently a Lecturer in the Department of Computer Science at the University of Manchester. Her current research interests include machine learning, data visualization and mathematical modeling, with applications to information retrieval, text mining, and bioinformatics.

R. Hong received the Ph.D. degree from the University of Science and Technology of China, Hefei, China, in 2008. He was a Research Fellow with the School of Computing, National University of Singapore, Singapore, from September 2008 to December 2010. He is currently a Professor with the Hefei University of Technology. His current research interests include multimedia question answering, video content analysis, and pattern recognition. He has co-authored more than 50 publications in his areas of expertise. Dr. Hong is a member of the Association for Computing Machinery (ACM). He was the recipient of the Best Paper Award in ACM Multimedia 2010.

M. Wang is a professor at the Hefei University of Technology, China. He received his B.E. and Ph.D. degrees in the Special Class for the Gifted Young and the Department of Electronic Engineering and Information Science from the University of Science and Technology of China (USTC), Hefei, China, respectively. His current research interests include multimedia content analysis, search, mining, recommendation, and large-scale computing. He received the best paper awards successively from the 17th and 18th ACM International Conference on Multimedia, the best paper award from the 16th International Multimedia Modeling Conference, the best paper award from the 4th International Conference on Internet Multimedia Computing and Service, and the best demo award from the 20th ACM International Conference on Multimedia.



N. An has a PhD in Computer Science and Engineering from the Pennsylvania State University and is an IEEE Senior Member and ACM Life Member. After working for Oracle USA, Inc. as technical staff for ten years, Dr. An joined Hefei University of Technology in China as a 'Yellow Mountain' Professor and founded the Gerontechnology Lab which is the first of its kind in mainland China. He is the PI for the International Research Base for Developing Innovative Gerontechnology sponsored by Chinese Ministry of Education and State Administration of Foreign Experts Affairs, and serves as the PI for several research projects including one sponsored by International S&T Cooperation Program of China. His current research interests include data sciences & engineering, gerontechnology, mobile health technologies, health communication, and digital learning.

J. Y. Goulermas (M'98, S'10) obtained the B.Sc.(1st class) degree in computation from the University of Manchester (UMIST), in 1994, and the M.Sc. and Ph.D. degrees from the Control Systems Center, UMIST, in 1996 and 2000, respectively. He is currently a Reader in the Department of Computer Science at the University of Liverpool. His current research interests include machine learning, combinatorial data analysis, data visualization as well as mathematical modeling. He has worked with various application areas including image/video analysis, biomedical engineering and biomechanics, industrial monitoring and control, and security. He is a senior member of the IEEE and an Associate Editor of the IEEE Transactions on Neural Networks and Learning Systems.