Int J Multimed Info Retr (2013) 2:203–212
DOI 10.1007/s13735-012-0027-z

REGULAR PAPER

Mobile video concept classification

Wei Jiang

Received: 9 October 2012 / Accepted: 13 November 2012 / Published online: 11 December 2012

Abstract Mobile content-based multimedia analysis has attracted much attention with the growing popularity of high-end mobile devices. Most previous systems focus on mobile visual search, i.e., to search images with visually duplicate or near-duplicate objects (e.g., products and landmarks). There remains a strong need for effective mobile video classification solutions, where videos that are not visually duplicate or near-duplicate but are from similar high-level semantic categories can be identified. In this work, we develop a mobile video classification system based on multi-modal analysis. On the mobile side, both visual and audio features are extracted from the input video, and these features are further compressed into compact hash bits for efficient transmission. On the server side, the received hash bits are used to compute the audio and visual Bag-of-Words representations for multi-modal concept classification. We propose a novel method where hash functions are learned based on the multi-modal information from the visual and audio codewords. Compared with traditional ways of computing visual-based and audio-based hash functions based on raw visual and audio local features separately, our method exploits the co-occurrences of audio and visual codewords as augmenting information and significantly improves the classification performance. The cost budget of our system for mobile data storage, computation, and transmission is similar to that in state-of-the-art mobile visual search systems. Extensive experiments over 10,000 YouTube videos show that our system can achieve similar classification accuracy with conventional server-based video classification systems using uncompressed raw descriptors.

W. Jiang (✉)
Kodak Technology Center, Eastman Kodak Company,
Rochester, USA
e-mail: vwjiang@gmail.com

Keywords Mobile video concept classification · Multi-modal hashing

1 Introduction

The growing popularity of high-end mobile devices has triggered many interesting research and applications, such as mobile visual search for objects (e.g., products, buildings, logos, printed media, and artwork). In the context of mobile devices, successful applications need to tackle several challenging conditions. The low computation power, memory and storage space on the mobile side make it necessary to use remote servers for massive computation. The limited bandwidth for wireless transmission makes it important to carefully select the amount of data sent from the mobile device to the servers. Fast, sometimes faster than real-time, computation on the server side is usually required.

A popular workflow of a mobile visual search system is described in Fig. 1. Local features such as SIFT, SURF, or Compressed Histogram of Gradients (CHoG) [3] are extracted at the mobile side, and such features are then compressed, e.g., by coding techniques or compact hashing methods. The compressed features usually have low bit rate and can be efficiently transmitted from the mobile device to remote servers. On the server side, similar images are searched from the database based on distances between the compressed local features. The systems have shown impressive performances for searching images with visually duplicate or near-duplicate objects [7,10,12]. For example, to search for products, logos, landmarks, paintings, book covers, and so on. However, when searching for videos that are not visually duplicate or near duplicate but are from similar high-level semantic categories, e.g., "birthday" videos from different birthday parties, the performance of such systems

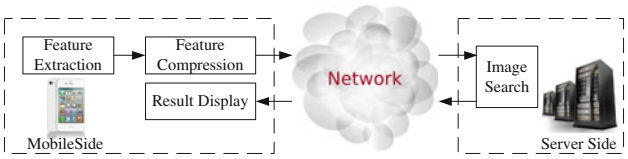


Fig. 1 A popular framework of mobile visual search

often suffers. In the case of video classification and categorization, it has been shown by prior arts that multi-modal (e.g., joint audio-visual) analysis is usually necessary to achieve satisfactory performance [14,29].

As the amount of videos growing increasingly large in recent years, many researchers have started to investigate accelerated video concept classification [13,14,29]. A widely used framework can be described as follows. The input video is downsampled first, to reduce the resolution and the number of visual and audio frames to process, so that the computation cost of feature extraction can be reduced. Then multi-modal features are extracted from the downsampled data. The most commonly used features are the visual local descriptors, e.g., SIFT and SURF, and the audio MFCC descriptor.[1] Next, the raw descriptors are further converted to the Bag-of-Words (BoW) feature vectors. Specifically, the raw visual (audio) descriptors are matched against the visual (audio) codewords that are computed based on the training videos, and the visual (audio) word frequencies of the input video form the visual (audio) BoW representation. Several techniques have been developed for fast BoW computation, such as the vocabulary-tree-based methods [23,25]. For final classification, the audio and visual BoW vectors can be either early fused by concatenation, or individually fed into classifiers to generate classification scores that are then combined as late fusion. The SVM classifier is usually used due to its proven performance for various video concept classification tasks. In [20], an efficient framework has been further developed to speedup the SVM classification using the histogram intersection kernels. It has been reported that the state-of-the-art video concept classification systems can work real-time on server machines [14,29].

In this work, we study real-time video concept classification on mobile devices. Figure 2 gives the overall workflow of our system. Similar to mobile visual search systems, we need to maintain fairly low computation and storage costs at the mobile side, transmit as few data as possible through the wireless network, and ensure fast responses from servers. To achieve these goals, on the mobile side, we first downsample the input video to reduce the resolution and the number of visual and audio frames. Next, audio and visual features are extracted from the downsampled data. These raw features are then compressed to generate compact hash bits for efficient transmission. On the server side, the received hash bits are used to compute the audio and visual BoW representations, which are further used for multi-modal concept classification. To cope with the state-of-the-art bit budget for data transmission (tens of bits per feature and a few hundred features per video) similar to that used in mobile visual search [3,12], and at the same time, to maintain good classification performance using the BoW representations, we propose the following strategy for feature compression. Instead of computing hash functions based on raw visual or audio features, we learn our hash functions based on the visual and audio codebooks. The advantages of our approach are threefold.

1. The number of raw audio and visual features (e.g., visual SURF and audio MFCC) from training videos can be very large (billions or even trillions). To generate well-performing short codes, data-dependent hashing techniques such as spectral hashing [31] should be used. Significant downsampling of the raw descriptors is therefore necessary to be able to effectively use the data-dependent hashing techniques. Compared with random downsampling, the audio and visual codebooks capture the informative centroids to represent the raw audio and visual features. Hash functions computed based on codebooks should outperform those computed based on randomly downsampled raw features.
2. The hash functions computed based on codebooks are targeted at mapping raw descriptors to similar corresponding codewords. This is naturally aligned with the final goal of using the compact hash bits, i.e., to compute the BoW representations for classification.
3. There are rarely exact or near duplicate audio and visual content from different videos. The hash functions found based on the relatively small number of codewords usually result in higher recall, i.e., higher probability of finding similar matching codewords, for the raw descriptors, and therefore, result in higher hit rate of the true codewords in the final BoW representation.

Feature compression reduces the amount of data for effective transmission. As a trade-off, it usually results in performance drop. In this paper, to alleviate such performance degradation for final classification, a multi-modal hashing method is further developed. Based on the observation that the low-dimension audio feature results in large performance degradation, we use the visual information to help compress the audio descriptors. Specifically, the co-occurrences of audio and visual codewords are used to construct an augmenting multi-modal feature space, and we learn the audio hash functions by preserving the local data neighborhoods in both the original audio feature space and this

[1] Information such as text or meta data is not generally used because its existence is not guaranteed.

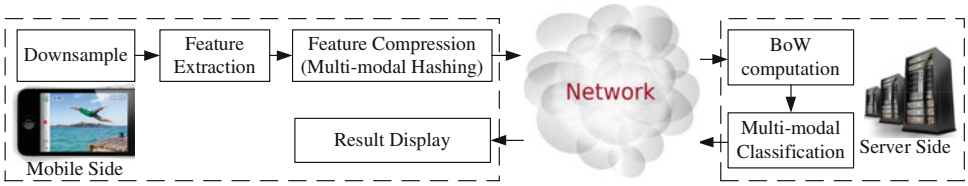


Fig. 2 The workflow of our system

augmenting multi-modal feature space. Extensive experiments over 10,000 YouTube videos show the effectiveness of our system. We can achieve similar classification accuracy with the conventional server-based video classification systems using uncompressed raw descriptors.

2 Related works

2.1 Fast video concept classification

As video sets grow increasingly large, the efficiency of computation for video content analysis has become a critical issue. Most of the state-of-the-art video concept classification systems use the BoW framework that keeps showing optimal performances over a series of benchmarks, such as the TRECVID high-level feature extraction and multimedia event detection [24], the Columbia Consumer Video (CCV) classification [15], and the Hollywood human action classification [21]. To reduce the computational cost of the BoW systems, several different approaches have been developed. To reduce the time cost of quantization, the vocabulary-tree-based methods have been developed [23,25]. To achieve real-time classification, Uijlings et al. use a series of actions in [29], including extracting the densely sampled DURF and DIFT, using linear approximation of the histogram intersection kernel, and so on. The work of [13] significantly downsamples image and audio frames to reduce the cost of feature extraction, and uses an audio-visual grouplet representation to reduce the final BoW dimension for classification. The work of [14] gives an extensive evaluation of various audio and visual descriptors and downsampling rates, and presents a general framework for fast internet video classification.

2.2 Mobile visual search

The increasing popularity of high-end mobile devices enables many interesting applications such as mobile visual search. Using the mobile camera to initiate a query, users can search for visually similar objects in the remote database. One major effort is related to identifying duplicate or near duplicate objects, such as products, logos, buildings and real estates, printed media and artwork. Many commercial systems have been developed, including Google Goggles [10], Amazon Snaptell [2], etc. Active research has also been conducted in recent years to further improve the performance of mobile visual search systems. For example, the low-bit-rate CHoG descriptor has been proposed [3], which reduces costs of storage and transmission while maintaining good precision. The Bag-of-Hash-Bits (BoHB) framework [12] incorporates boundary-feature-based reranking to improve the accuracy of mobile product search. The Active Query Sensing (AQS) method [7] incorporates the active viewing-angle recommendation mechanism to improve the performance of mobile location search.

2.3 Hashing

Many hash coding-based algorithms have been proposed to effectively approximate nearest neighbor search with high-dimensional data. Let $\mathbf{X} \in \mathbb{R}^{d\times n}$ denote n data points, each column $\mathbf{x}_i$ being a datum in a d-dim feature space. Let $\mathbf{Y} \in \mathbb{B}^{r\times n}$ denote the binary codes of $\mathbf{X}$, where each column $\mathbf{y}(\mathbf{x}_i)$ is an r-bit binary code for $\mathbf{x}_i$. The most widely used hash functions use linear projections with the following form:

$$h_k(\mathbf{x}) = sign(\mathbf{w}_k^T\mathbf{x} + b_k), \tag{1}$$

where $\mathbf{w}_k$ and b_k are the projection vector and threshold, respectively, for generating the kth bit. The corresponding hash bit is given by $y_k(\mathbf{x}) = (1+h_k(\mathbf{x}))/2$.

A large variety of $\mathbf{w}_k$ have been adopted. The approaches based on locality-sensitive hashing (LSH) [6,8] use random projections following a p-stable distribution for a general l_p norm. These methods have the theoretical advantage that the input distance can be asymptotically preserved as the number of hash bits increases. However, long codes are usually required to achieve good precision, and to alleviate the recall degradation resulting from long codes, multiple hash tables (often several hundred) have to be maintained for high-dimensional datasets. Instead of using random projections, the semi-supervised hashing (SSH) method [30] learns the projection vector by minimizing the empirical error on the labeled data while maximizing the overall entropy. The iterative quantization (ITQ) method [9] first performs orthogonal transformation over the data and then refines the initial transformation by reducing the quantization error. To better cope with linearly inseparable data or high-dim kernelized data with implicitly known embedding, the kernelized methods have been further developed, such as the Kernelized LSH

[17], the Binary Recostructive Embedding (BRE) [16], and the kernel-based supervised hashing (KSH) [19].

The optimal Hamming embedding of the dataset can also be directly learned. Let $\mathbf{A}$ denote the affinity matrix between pair-wise data points in the original feature space, i.e., each A_{ij} measures the similarity between $\mathbf{x}_i$ and $\mathbf{x}_j$. As discussed in [31], the desired binary codes $\mathbf{Y}$ can be obtained by solving the following optimization problem:

$$\min_{\mathbf{Y}} \operatorname{tr}(\mathbf{YLY}^T) \quad \text{s.t.}\ \mathbf{Y} \in \{1,-1\}^{r\times n}, \quad \mathbf{1}^T\mathbf{Y}=0, \quad \mathbf{YY}^T=\mathbf{I}. \tag{2}$$

$\mathbf{L}$ is the graph Laplacian defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$, where $\mathbf{D}$ is a diagonal matrix and $D_{ii} = \sum_j A_{ij}$. The constraint $\mathbf{1}^T\mathbf{Y} = 0$ requires each bit to fire 50 % of the time, and the constraint $\mathbf{YY}^T = \mathbf{I}$ requires mutually uncorrelated bits to reduce redundancy. However, the original problem of Eq. (2) is NP-hard, and the spectral relaxation is usually applied to remove the constraint: $\mathbf{Y} \in \{1,-1\}^{r\times n}$. With the relaxation, the optimal binary codes can be obtained through two steps: (1) computing $\mathbf{Y}$ as the r eigenvectors corresponding to the r smallest non-zero eigenvalues of the graph Laplacian $\mathbf{L}$, and (2) getting binary codes as $sign(\mathbf{Y})$.

When extending the eigenvectors to unseen data, to reduce computation, the spectral hashing method [31] assumes a separable uniform data distribution and obtains a 1-D Laplacian eigenfunction solution, and the AGH method [18] computes an anchor graph Laplacian to preserve the approximate data neighborhood.

3 Mobile video concept classification

3.1 Mobile side: feature extraction

Due to the limited computation and storage power, as well as the limited bandwidth for transmitting data, the original input video needs to be significantly downsampled in both visual and audio channels. We conduct this downsampling from several aspects.

One image frame is uniformly sampled from every 10 s, and correspondingly, around the sampled image frame, a 6-s audio signal is extracted. If the original sampled image frame has high resolution, it is further downsized to 320 × 240. Then, over the image frame, two types of visual features are computed. The first is the SURF descriptor (128-dim), which has proven performance in both accuracy and speed in many previous systems [14,22]. The other is the 225-dim grid-based color moment (GCM), which is very fast to compute and has shown robust performance for general video concept classification [1]. Over the downsampled audio signal, the 13-dim MFCC coefficients as well as their deltas are computed, composing a 26-dim audio feature for each audio frame. Such an audio feature has been proved effective for general audio and video concept classification in previous literature [4,26]. A fairly large audio frame, i.e., 250 ms window with 100 ms hop, is taken here, due to two reasons. First, as shown in [26], the 250 ms audio frame setting can be more appropriate for classifying audio concepts in generic audio signals in consumer-quality videos. Second, the large audio frame leads to a smaller number of audio features and can save computation in later parts of the system.

Considering the constraint of transmission bandwidth, we need to further reduce the amount of raw features, especially for long videos. In our implementation, we uniformly sample at most 1,000 SURF features for each video, and keep at most 1,000 audio features per video. Only one global GCM feature is used per video, which is the feature closest to the mean of all GCM features in this video. Therefore, we have $1,000 \times (128+26)+225$ feature values (about 617 KB data) for each input video. In many situations, these raw features are still too many to be transmitted directly. Therefore, we need to further compress such raw features to reduce the amount of data for fast transmission.

3.2 Mobile side: feature compression

Recently, low-bit-rate descriptors have gained great attention in the research community. In [28], each SURF descriptor is quantized and entropy coded to 36.8 bytes. In [3], an effective CHoG descriptor has been developed, where each local descriptor has only 53 bits. In [11], a bag-of-hash-bits approach has been proposed where each local SURF descriptor is compressed to 80 hash bits.

In this work, we aim to compress the visual SURF descriptors and audio MFCC with delta descriptors into compact bits, with similar bit budget to that of the states-of-the-art [3,11]. We take an approach similar to [11], where we compress the visual and audio features by hashing. Different from [11], we do not compose hash functions based on raw visual or audio descriptors. We learn hash functions based on the visual and audio codebooks, and then use the hash functions to compress the visual and audio descriptors of the input video. On the server side, after receiving the hash bits, the audio and visual BoW features can be computed based on the hash bits. The advantages of our approach are: with a relatively small number of codewords, we can learn hash functions using data-dependent hashing techniques such as spectral hashing [31]; since there are rarely exact or near duplicate descriptors from different videos, hash functions found based on the relatively small number of codewords result in higher recall (probability of finding similar matching codewords) for the raw descriptors, and therefore, result in higher hit rate of the true codewords in the final BoW representation; and finally, hash functions computed based on codebooks aim to map raw descriptors to similar

corresponding codewords, which is naturally aligned with the goal of generating the hash bits, i.e., to compute BoW representations for final classification.

Many data-dependent hashing methods have been developed to overcome the limitations of the data-independent LSH that long codes and a large number of multiple hash tables are usually necessary to achieve reasonable precision and recall. As discussed in Sect. 2.3, these methods learn compact binary codes based on properties of the underlying dataset. Representative approaches are the unsupervised spectral hashing [31], BRE [16], AGH [18], the semi-supervised SSH [30], and the supervised semantic hashing [27] and KSH [19]. Since we do not have supervised training labels over the audio or visual codewords, we need to use unsupervised methods. Due to the limited memory and storage on mobile devices, methods that require storing part of the original training data in memory are too expensive to use for large-scale problems. For example, the kernelized LSH [17] and BRE [16] require to store the original training data and compute kernel values between the test data and the training data at run time. The AGH method [18], although significantly reducing the storage and computation costs, still requires, at run time, storing the original anchors in memory and computing distances between the raw descriptors to the anchor points. As a result, in our baseline approach, we use the spectral hashing algorithm [31] to compress the audio and visual descriptors.

3.2.1 The baseline approach

In the training process, a visual codebook is generated by clustering all the SURF descriptors from all the training videos into different codewords. The K-means algorithm is used for this purpose, and the entire codebook has 10,000 codewords. By empirical study, such a codebook size is an appropriate trade-off between the classification accuracy and the computational cost. Similarly, a 10,000-codeword audio codebook is generated by clustering all the MFCC plus delta audio descriptors from all the training videos.

Then we generate hash functions based on the visual codebook and the audio codebook, respectively. In the visual aspect, we hash each of the original 128-dim SURF feature to 40 bits. Then on the server side, after receiving the hash bits, 5 hash tables are used to increase recall when computing the visual BoW representation. The 5 hash tables are composed by reusing the 40 bits, i.e., 24 bits are randomly sampled from the 40 bits to generate each table. As suggested by [11], such a method improves the recall without adding burden to transmission. Also, the spatial location of each SURF descriptor generates a 2-bit code (recording the location in the 2×2 spatial layout). So finally, each SURF descriptor has 42 bits. In the audio aspect, each of the original 26-dim MFCC with delta descriptor is hashed to 24 bits, and only one hash table is used. The 225-dim GCM feature can be either directly transmitted, or quantized then entropy coded (similar to [3]) for transmission. In combination, for a 1-min video, we transmit about 9 KB feature data from the mobile device to the server. This bit budget is similar to that of the state-of-the-art image search systems [3,11].

As can be seen from Fig. 3 in Sect. 4, compared with the original descriptors, the compressed feature will sacrifice the final classification performance. The performance degradation is especially severe for the audio feature (more than 50 % MAP drop). It is highly desirable that a more effective hashing method can be used to alleviate the performance drop, especially in the audio aspect.

3.2.2 Multi-modal hashing

In this subsection, we develop a multi-modal hashing algorithm for compressing the audio descriptors. The basic idea is to improve the hashed audio feature with the help of visual information. One possible reason for the large performance drop in the audio aspect observed in the baseline approach is the relatively low dimensionality (26 dim) of the original raw audio descriptors. That is, it is especially hardly true that the PCA-aligned descriptors can be modeled by a multivariate uniform distribution, and the selected 24 hash functions (eigenfunctions) to generating the 24 audio hash bits tend to completely ignore the underlying data structure. Therefore, we propose to learn audio hash functions by preserving the local data neighborhoods. To maintain similar storage and computation budget with the baseline spectral hashing method, we follow the idea of learning a projection vector $\mathbf{w}_k$ that is used to compose the linear-projection-based hash function described in Eq. (1). Let $\mathbf{W}$ denote the $r \times d$ projection matrix whose rows are projection vectors $\mathbf{w}_k^T$. We learn the optimal $\mathbf{W}$ to preserve the local data neighborhood that is defined based on information from both audio and visual channels. At run time, we only need to store the projection matrix $\mathbf{W}$ in memory, and only need to perform one matrix multiplication to compress raw descriptors.

Without loss of generality, we assume that the dataset $\mathbf{X}$ is zero-mean, i.e., $\mathbf{X}$ has been normalized. According to Eq. (1), the kth bit $y_k(\mathbf{x})$ of a datum $\mathbf{x}$ is given by:

$$y_k(\mathbf{x}) = (1 + h_k(\mathbf{x}))/2 = (1 + sign(\mathbf{w}_k^T \mathbf{x}))/2. \quad (3)$$

Following Eq. (2), we compute the optimal $\mathbf{W}$ by solving the following problem:

$$\min_{\mathbf{W}} \text{tr}(sign(\mathbf{W}^T \mathbf{X})\, \mathbf{L}\, sign(\mathbf{X}^T \mathbf{W})) \quad \text{s.t. } \mathbf{W}^T \mathbf{W} = \mathbf{I}.$$

The constraint $\mathbf{W}^T \mathbf{W} = \mathbf{I}$ requires orthogonal projections, which avoids redundancy in hash bits. As shown in [30], the requirement of using exact balanced bits makes the above

objective function intractable. Therefore, similar to [30], we take the relaxation by replacing the sign of the projection with its signed magnitude. That is, similar points should not only have the same signs but also have large projection magnitudes, while dissimilar points should not only have different signs but also have far apart projections. With such a relaxation, the original cost function turns to:

$$\min_{\mathbf{W}} \operatorname{tr}(\mathbf{W}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{W}) \quad \text{s.t. } \mathbf{W}^T\mathbf{W} = \mathbf{I}. \tag{4}$$

The graph laplacian $\mathbf{L}$ is computed based on an affinity matrix $\mathbf{A}$, and in our approach, this affinity matrix incorporates information from both audio and visual channels. Specifically, based on the set of training videos $\mathcal{V}$, an $n^v \times n^a$ dimensional co-occurrence matrix $\mathbf{M}$ can be computed, where each item M_{ij} measures how likely the ith visual codeword and the jth audio codeword co-occur. Let v_k be a training video in $\mathcal{V}$, $\mathbf{f}^v_{BoW}(v_k) = [f^v_{BoW}(v_k, 1), \ldots, f^v_{BoW}(v_k, n^v)]$ be the BoW feature of video v_k computed against the visual codebook, and $\mathbf{f}^a_{BoW}(v_k) = [f^a_{BoW}(v_k, 1), \ldots, f^a_{BoW}(v_k, n^a)]$ be the BoW feature of video v_k computed against the audio codebook. M_{ij} can be computed as:

$$M_{ij} = \frac{1}{|\mathcal{V}|} \sum_{v_k \in \mathcal{V}} f^v_{BoW}(v_k, i) + f^a_{BoW}(v_k, j).$$

Using the original 26-dim audio MFCC with delta descriptors, we can compute an affinity matrix $\mathbf{A}^a$ measuring the pair-wise similarity between audio codewords, where each item A^a_{ij} is given by:

$$A^a_{ij} = \begin{cases} \exp\left(-||\mathbf{x}_i - \mathbf{x}_j||^2/(\sigma^a)^2\right), & \forall j \in \langle i \rangle \\ 0, & \text{otherwise} \end{cases}$$

$\langle i \rangle \subset [1:m]$ denote the indices of m nearest neighbors of $\mathbf{x}_i$ (i.e., m defines the size of local neighborhoods that we want to preserve). In practice, we set $m = 10$. The parameter σ^a is simply set as the mean value of all pair-wise distances (measured in the original MFCC plus delta feature space) between the audio codewords.

In addition, each column $\mathbf{m}_j$ of the co-occurrence matrix $\mathbf{M}$ can be treated as an n^v-dim feature representation of the jth audio codeword. Using this feature, another affinity matrix $\mathbf{A}^{a-v}$ can be computed with items A^{a-v}_{ij} as:

$$A^{a-v}_{ij} = \begin{cases} \exp\left(-||\mathbf{m}_i - \mathbf{m}_j||^2/(\sigma^{a-v})^2\right), & \forall j \in \langle i \rangle \\ 0, & \text{otherwise} \end{cases}$$

where $\langle i \rangle \subset [1:m]$ denote the indices of m nearest neighbors of $\mathbf{m}_i$. We also set $m = 10$ in practice. The parameter σ^{a-v}, again, is simply set as the mean value of all pair-wise distances (measured based on co-occurrence-based features) between audio codewords.

Therefore, the final affinity matrix $\mathbf{A}$ can be computed as a weighted combination of $\mathbf{A}^a$ and $\mathbf{A}^{a-v}$:

$$\mathbf{A} = \mathbf{A}^a + \eta\mathbf{A}^{a-v}. \tag{5}$$

Using the graph Laplacian $\mathbf{L}$ that is computed based on the affinity matrix $\mathbf{A}$ defined in Eq. 5, we can solve the problem of Eq. 4 to obtain the optimal projection matrix $\mathbf{W}$ as the r eigenvectors corresponding to the r smallest non-zero eigenvalues of the graph Laplacian $\mathbf{L}$. At run time, to compress a raw audio descriptor $\mathbf{x}$, we only need to store the $r \times d$-dim projection matrix $\mathbf{W}$ in memory, and only need to compute the matrix multiplication $\mathbf{W}\mathbf{x}$. Both storage and computation costs are similar to (actually a little less than) those in the baseline spectral hashing approach.

Similar to spectral hashing, we have the limitation that the length of the binary codes cannot exceed the length of the original feature dimension. In practice, we choose r as the number of non-zero eigenvalues of $\mathbf{L}$ ($r < 26$). Similar to the baseline approach, only one hash table is used to compute the audio BoW feature at the server side.

3.3 Server side

After obtaining the hashed visual and audio features, 42 bits for each visual SURF descriptor (2 bits recording the spatial information), and r bits for each audio descriptor, the visual and audio BoW representations can be effectively computed. We use Hamming ball of radius 1. Each visual BoW feature has $5n^v$ dimensions ($n^v = 10,000$ in practice), which is the concatenation of two BoW features corresponding to the 1×1 and 2×2 spatial layouts, respectively. The audio BoW feature has n^a dimensions ($n^a = 10,000$ in practice). Assume that we have C concepts. To classify each semantic concept, the SVM classifiers using χ^2-RBF kernels corresponding to that concept are applied to the audio and visual BoW features, respectively, to obtain the audio and visual-based classification scores. At the same time, the SVM classifier using the Gaussian RBF kernel corresponding to that concept is applied to the GCM feature to obtain the visual global-based classification score. All these three scores are first normalized by using a sigmoid function and then averagely combined to obtain the final classification score. It is worth mentioning that when the number of concepts C increases, the computational time does not increase as fast, since the most time consuming part in the whole process is to compute the BoW features, and the same BoW features, once computed, are used by all concept classifiers. In addition, the final SVM classification can be parallelled for further speedup.

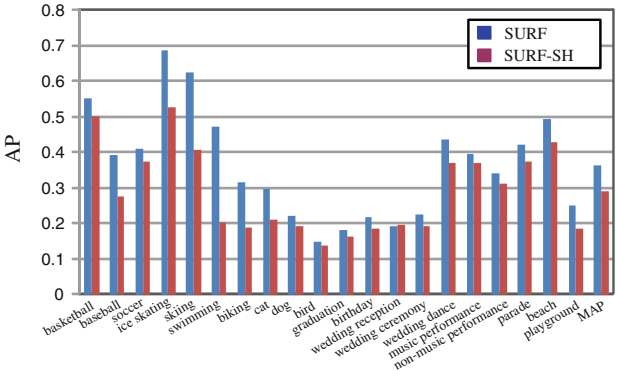
4 Experiments

We evaluate our system based on the CCV dataset [15], which is one of the largest video set of open-domain consumer videos. The dataset contains 9,317 videos downloaded from YouTube, 4,659 videos for training and 4,658 videos for testing. The videos are manually annotated to 20 consumer concepts focusing on human activities and events, objects, and scenes. The average video duration is about 80 s. On average, each concept has about 400 positive samples, which are evenly distributed in the training and test sets. The performance is measured by Average Precision (AP, the area under uninterpolated PR curve) and Mean AP (MAP, averaged AP across concepts).

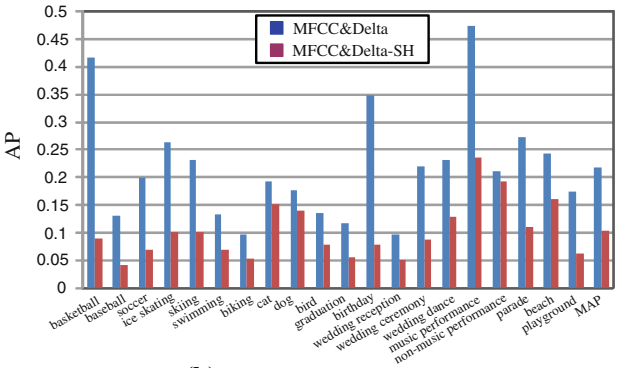
4.1 The baseline performance

Figure 3 (a–c) shows the performance of the baseline approach where spectral hashing (SH) is used for feature compression. We have the following observations.

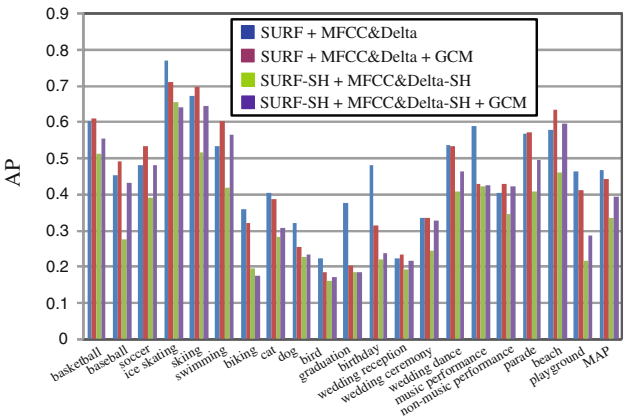
1. The original SURF feature with two spatial layouts gets roughly 0.36 MAP. This result is consistent with numbers reported in [14] over the same dataset, since we use at most 1,000 SURF points per video, which is equivalent to using only 2–3 frames per video in [14]. The original MFCC with delta feature gets roughly 0.22 MAP, which is slightly better than the corresponding downsampled MFCC results reported in [14]. After feature compression, both MFCC and SURF have large performance drop. The MAP of "SURF-SH" drops 20 % from the original "SURF," and the MAP of "MFCC-SH" drops even more (53 %).
2. In Fig. 3c, we combine classification scores from various features by late fusion, and we can see that although the largely downsampled MFCC with delta descriptor performs not satisfactorily, the combination of audio and visual features can still get big performance improvements (30 % MAP gain compared with SURF alone). Similarly, for the compressed feature, the combined audio and visual descriptor can improve the MAP by 16 % compared with "SURF-SH," despite the bad performance of "MFCC-SH" alone.
3. It is also interesting to see that by adding the global GCM feature, for classification using original audio and visual descriptors, no benefit is obtained overall. Instead, the MAP drops by 5 %. This confirms the importance of using local visual descriptors for video event and object classification, as observed by others [14,24]. However, the global GCM feature contributes a lot to the final classification performance using hashed audio and visual descriptors. The "SURF-SH+MFCC&Delta-SH+GCM" outperforms the corresponding "SURF-SH+MFCC&Delta-SH" by 17 % in terms of MAP, and the improvement is consistent over most of the concepts. This phenomenon confirms the importance of including the global GCM feature in our mobile video classification system, since the global GCM feature is fairly low-cost to compute and transmit and has the ability to boost the final classification performance. Finally, compared with the best performing original features (i.e., the "SURF+MFCC&Delta" method), the



(a) the visual SURF descriptor



(b) the audio MFCC and delta descriptor



(c) late fusion of audio and visual descriptors

Fig. 3 Baseline performance: feature compression by spectral hashing

best performing compressed features (i.e., the "SURF-SH+MFCC&Delta-SH+GCM" method) still have 17 % MAP drop.

To further demonstrate the advantage of the proposed codeword-based hashing framework, we compare our baseline approach with the intuitive alternative method that computes hash functions based on downsampled raw visual and audio descriptors. Specifically, 1 million SURF descriptors and 1 million MFCC plus delta descriptors are randomly sampled from the training videos, and spectral hashing is applied to the visual and audio raw descriptors, respectively, to generate the visual and audio hash functions. For fair comparison, we use the same bit budget here, i.e., 40 bits and 5 hash tables (24 bits for each table) for the visual part, 24 bits for the audio part. Figure 4 shows the performance comparison. From the figure, our baseline approach using codeword-based hash functions significantly and consistently outperforms the corresponding hash functions computed over raw descriptors. That is, "SURF-SH" outperforms "SURF-SH (Raw)" (by 32 % MAP improvement), "MFCC&Delta-SH" outperforms "MFCC&Delta-SH (Raw)" (by 50 % MAP improvement), and "SURF-SH+MFCC&Delta-SH" outperforms the corresponding "SURF-SH+MFCC&Delta-SH (Raw)" (by 33 % MAP improvement).

4.2 Performance of multi-modal hashing

Figure 5a, b shows the performance of our method where multi-modal hashing is used to improve the performance of audio feature compression.

From Fig. 5a, using the co-occurrence of audio and visual codewords as additional information, the compressed MFCC with Delta feature with multi-modal hashing ("MFCC-&Delta-MMH" with $\eta > 0$) consistently outperforms the

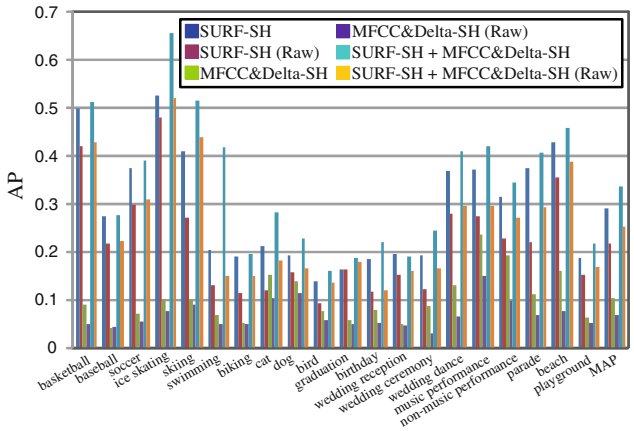
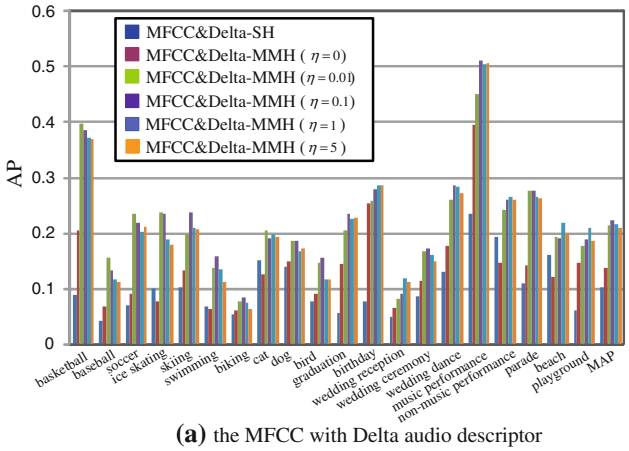
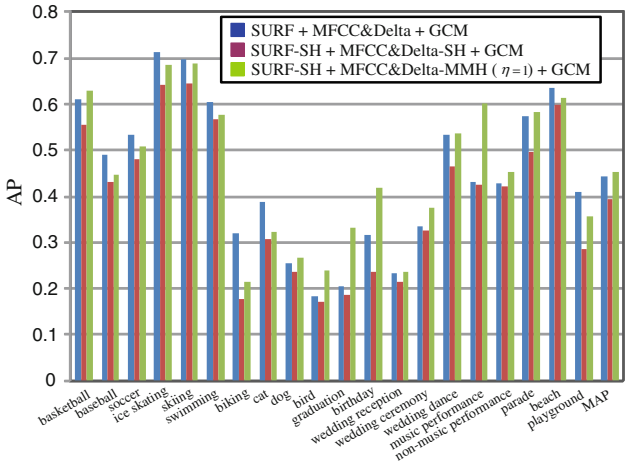


Fig. 4 Performance comparison with hash functions computed using raw descriptors



(a) the MFCC with Delta audio descriptor



(b) late fusion of multiple descriptors

Fig. 5 Performance of our approach

compressed feature with spectral hashing ("MFCC&Delta-SH"), over every concept. When $\eta = 0$, the problem of Eq. (4) reduces to solving for a projection matrix $\mathbf{W}$ based on the pair-wise affinity matrix defined in the audio feature space alone, without using the audio-visual co-occurrence information. As shown in Fig. 5a, when $\eta > 0$, the audio-visual co-occurrence information can help to obtain a better compressed audio feature. In addition, the final performance is not very sensitive to parameter η. As η varies between 0.01 and 5, the overall MAP varies only a little bit. Therefore, in practice, we simply set $\eta = 1$. In such a case, the "MFCC&Delta-MMH" significantly improves the performance of compressed audio feature by pulling up the MAP back to roughly 0.22, which is the same as the original uncompressed "MFCC&Delta" in Fig. 3b.

When we combine multi-modal classification scores through late fusion, as shown in Fig. 5b, we can bring the overall MAP of using compressed feature back to about 0.46, which is roughly the same with (actually slightly better than) the MAP of using original uncompressed descriptors. Over some concepts where audio feature plays

critical roles, e.g., "bird", "birthday", "graduation" ,"wedding ceremony", "wedding dance", and "music performance", the performance improvement compared with spectral hashing is quite significant. This is because over these concepts, our multi-modal hashing can largely improve the quality of compressed audio feature, resulting in the final AP boost for combined classification. Such experimental results are quite encouraging, which show that with much less costs in computation, storage, and transmission, the compressed features can actually achieve the same classification performance as the uncompressed features.

5 Conclusion

We develop a mobile video concept classification system in this paper. Our approach uses a new method to learn hash functions based on the multi-modal information from the visual and audio codewords, and the generated compact hash bits significantly outperform the traditional alternatives where hash functions are generated based on raw visual and audio local features. Extensive experiments over the large-scale YouTube video set show that our system can conduct real-time video concept classification on mobile devices with similar classification accuracy to the conventional server-based video classification systems using corresponding uncompressed raw descriptors.

The multi-modal analysis provides the opportunity of learning multi-modal compact bits both for effectively representing the video content and for efficient transmission. The work described in this paper gives an example of using the audio-visual co-occurrences to facilitate hashing audio features for video concept classification. In the future, more research will be conducted along this direction. Other types of multi-modal information will be studied to help generate effective compact bits, in both visual and audio aspects. Other applications, besides video concept classification, will be investigated, such as video-based search and control.

Acknowledgments Thanks to Dr. Alexander Loui from Eastman Kodak Company for the valuable discussions.

References

1. Yanagawa WHA, Chang S (2006) Brief descriptions of visual features for baseline trecvid concept detectors. Columbia University ADVENT Technical, Report 219-2006-5
2. Amazon. Snaptell. http://a9.amazon.com/-/company/snaptell.jsp
3. Chandrasekhar V, Takacs G, Chen D, Tsai S, Reznik Y, Grzeszczuk R, Girod B (2012) Compressed histogram of gradients: a low-bitrate descriptor. Int J Comput Vis 96(3):384–399
4. Chang S, Ellis D, Jiang W, Lee K, Yanagawa A, Loui A, Luo J (2007) Large-scale multimodal semantic concept detection for consumer video. ACM Multimedia, Information Retrieval, pp 255–264
5. Cotton C, Ellis D, Loui A (2011) Soundtrack classification by transient events. In: IEEE International Conference on Acoustics, Speech, and, Signal Processing, pp 473–476
6. Datar M, Immorlica N, Indyk P, Mirrokni VS (2004) Locality-sensitive hashing scheme based on p-stable distributions. Annual Symposium on Computational Geometry, pp 253–262
7. Rongrong J, Yu FX, Chang S (2011) Active query sensing for mobile location search. In: ACM Multimedia, pp 3–12
8. Gionis A, Indyk P, Motwani R (1999) Similarity search in high dimensions via hashing. In: International Conference on Very Large Data, Bases, pp 518–529
9. Gong Y, Lazebnik S (2011) Iterative quantization: a procrustean approach to learning binary codes. In: IEEE International Conference on Computer Vision and Pattern Recognition
10. Google. Google Goggles. http://www.google.com/mobile/goggles/
11. He J, Feng J, Liu X, Cheng T, Lin TH, chung H, Chang SF (2011) Mobile product search with bag of hash bits. In: ACM Multimedia, pp 839–840
12. He J, Feng J, Liu X, Cheng T, Lin TH, Chung H, Chang SF (2012) Mobile product search with bag of hash bits and boundary reranking. In: IEEE International Conference on Computer Vision and, Pattern Recognition, pp 3005–3012
13. Jiang W, Loui A, Lei P (2012) A consumer video search system by audio-visual concept classification. IEEE Computer Vision and Pattern Recognition Workshops, Providence
14. Jiang Y (2012) Super: towards real-time event recognition in internet videos. In: ACM International Conference on Multimedia Retrieval
15. Jiang Y et al. (2011) Towards optimal bag-of-features for object categorization and semantic video retrieval. In: ACM International Conference on Multimedia Retrieval
16. Kulis B, Darrell T (2009) Learning to hash with binary reconstructive embeddings. NIPS
17. Kulis B, Grauman K (2012) Kernelized locality-sensitive hashing. IEEE Transact Pattern Anal Mach Intell 34(6):1092–1104
18. Liu W, Wang J, Kumar S, Chang SF (2011) Hashing with graphs. International Conference on Machine Learning, Bellevue
19. Liu W, Wang J, Ji R, Jiang Y, Chang SF (2012) Supervised hashing with kernels. In: IEEE International Conference on Computer Vision and Pattern Recognition, Providence
20. Maji S, Berg A, Malik J (2008) Classification using intersection kenrel support vector machines is efficient. IEEE International Conference on Computer Vision and, Pattern Recognition
21. Marszałek M, Laptev I, Schmid C (2009) Actions in context. IEEE International Conference on Computer Vision and, Pattern Recognition
22. Mikolajczyk K, Schmid C (1995) A performance evaluation of local descriptors. IEEE Transact Pattern Anal Mach Intell 27(10):1615–1630
23. Moosmann F, Triggs B, Jurie F (2006) Fast discriminative visual codebooks using randomized clustering forests. NIPS, pp 985–992
24. NIST. TRECVID. http://trecvid.nist.gov/
25. Nister D, Stewenius H (2006) Scalable recognition with a vocabulary tree. IEEE International Conference on Computer Vision and Pattern Recognition, pp 2161–2168
26. Parker C (2010) An exploration of semantic audio classification. In: Technical Report 345596K, Eastman Kodak Company
27. Salakhutdinov R, Hinton G (2009) Semantic hashing. Int J Approx Reason 50(7):969–978
28. Takacs G, Chandrasekhar V, Gelfand N, Xiong Y, Chen WC, Bismpigiannis T, Grzeszczuk R, Pulli K, Girod B (2008) Outdoors augmented reality on mobile phone using loxel-based visual fea-

ture organization. In: ACM international conference on Multimedia, information retrieval, pp 427–434
29. Uijings J, Smeulders A, Scha R (2010) Real-time visual concept classification. IEEE Transact Multimed 12(7):665–681
30. Wang J, Kumar S, Chang S (2012) Semi-supervised hashing for large scale search, IEEE Transact Pattern Anal Mach Intell
31. Weiss Y, Torralba A, Fergus R (2008) Spectral hashing, NIPS