Object Mining for Large Video Data

Ronak Shah
ronakshah@ee.iitb.ac.in
Rishabh Iyer
rishabh_nad@iitb.ac.in
Subhasis Chaudhuri
sc@ee.iitb.ac.in

Vision and Image Processing Lab,
Department of Electrical Engineering,
Indian Institute of Technology,Bombay,
India.

Abstract

We propose a method for achieving a novel concise graph-based representation for retrieval of objects from large video data. The emphasis in this paper is towards achieving a compact representation of video data for faster retrieval. Specifically, we use information available from scripts and subtitles in order to group all occurrences of an object in video data, which provides a separate representation for each scene. Further, based on the premise that the number of objects in a shot are typically much less than the number of video frames in that shot, we propose a graph-based representation in which vertices represent objects rather than video frames. Key advantages of the proposed approach include faster retrieval, efficiency in performing tasks such as spatial re-ranking and graph partitioning and a single representation for both retrieval and summarization applications. We demonstrate efficacy of the proposed approach in retrieval and summarization applications over video data consisting of episodes of a popular TV series "Friends".

1 Introduction

The goal of this work is to unsupervisedly mine various objects occurring in video data and represent them in a concise manner. Mining objects has many applications, including a mechanism for accessing large data in an organized way, selective video playback and summarization of video. Most current approaches which aim to achieve a graph-based representation, denote each frame as a vertex in the graph. Hence, operations to perform at query time and other processing requirements increase with an increase in the number of frames. In this work, we utilize scripts and subtitles for achieving a disjoint graph-based representation for video data. Also, in the proposed representation, each vertex serves as a definition for an object rather than a video frame. This provides compactness in the representation as typically only 2-8 objects are present in a shot of about 60-120 frames (from "Friends" video data we analysed).

The advantage of this framework over all existing methods is three-fold: Firstly, vertices grow linearly with the number of objects rather than the number of frames, which yields faster retrieval. Secondly, due to conciseness of representation, tasks such as spatial re-ranking and graph partitioning can be performed efficiently. Finally, video summarization is achieved as a spin-off from the proposed representation.

Several approaches have been suggested for mining video objects. Text-based retrieval by Sivic [19] is the bag-of-words concept for object retrieval. In [20], a fixed neighborhood

BMVC 2011 http://dx.doi.org/10.5244/C.25.2

was defined around center points to capture spatial information. Quack et al. [16] describe video mining for frequently occurring itemsets. Jamieson et al. [6] find frequently occurring objects in a video through graph matching, where each object is represented as a graph. Though these approaches consider collection of visual words for object definitions, they still fail to capture an explicit spatial arrangement of visual words. Also, such representations are not amenable to video summarization. The approach proposed here utilizes object grouping at its core. Grouping of objects occurring over a large number of frames is utilized by Sivic et al. [22]. However, their aim is limited to achieving object associations for varying poses.

Recently, several approaches have been proposed for organizing a very large number of images in an ordered set. Such approaches include min-Hash algorithm [3], mining based on matching graph [14] and retrieval based on fast spatial matching [15]. While these approaches are successful for their intended applications, they are not directly useful for video mining. For example, [3] computes matching between images rapidly through a randomized data mining scheme but suffers from a low recall rate, whereas [14] and [15] consider each image as a vertex in the graph, which provides a very redundant representation for video data. The approach proposed here is similar in the spirit but differs significantly in representation and utility. Specifically, we make use of subtitles and scripts and address the problem of concise representation, which to the best of our knowledge, has not been addressed for this application.

2 Problem Definition

Let, P_T and B_T represent scripts and subtitles for movie or daily soap (serials) video data (D). Let, $\mathbf{C} = [C_1, C_2, \ldots, C_N]$ denote N distinct scenes, places where actions take place, obtained from P_T and B_T. The goal is to achieve a graph based representation $G_i(\mathbb{V}_i, \mathbb{E}_i)$ for each scene, C_i, where $\mathbb{V}_i$ represents a set of vertices consisting of unsupervisedly mined video objects and $\mathbb{E}_i$ represents a set of edges indicating similarities between connected objects.

3 Approach and Preprocessing Steps

In a typical movie or episode production, scripts (P_T) are written beforehand, whereas, subtitles (B_T), are produced post-production to aid viewers. Scripts contain information such as scenes, speakers and corresponding dialogues, whereas, subtitles contain dialogues and timing information. However, for the task of achieving the proposed concise representation, scene as well as timing information are useful. Hence, we search matches for dialogues of subtitles in corresponding scripts and assign scene information ($C \in mathbf{C}$) and speakers to these dialogues. As subtitles are generally provided by amateurs and sometimes due to changes in screenplay, inconsistencies in scripts and subtitles are commonplace. To cater for inconsistencies, we use Levenshtein distance [7] for carrying out matches between dialogues with dynamic time warping [13]. This approach is similar in spirit to Marszalek et al. [10] and Cour et al. [9]. The procedure is depicted in Figure 1.

For further processing, we extract video segments belonging to the same scene together. Hence, video frames are now indexed by the scene (an element of $\mathbf{C}$) to which they belong to. Subsequently, data for each scene is processed separately to achieve a disjoint graph-based representation.

Further, we perform shot segmentation over video segments of a scene through color histogram [8]. A representative frame for each shot is selected for further processing. For that purpose, similarity measure between two frames, f_i and f_j, is computed in the following

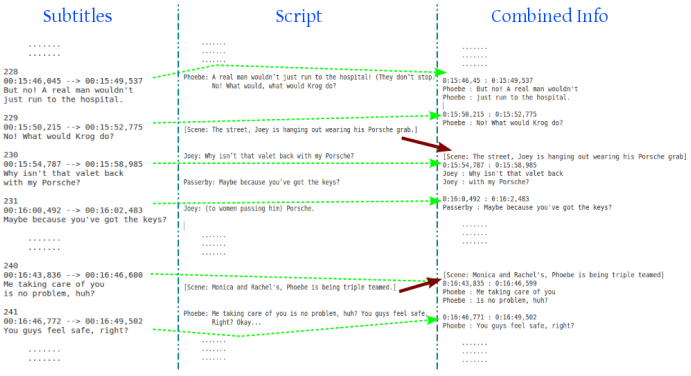


Figure 1: Procedure for combining complementary information from subtitles and scripts. Read text for elaborate procedure.

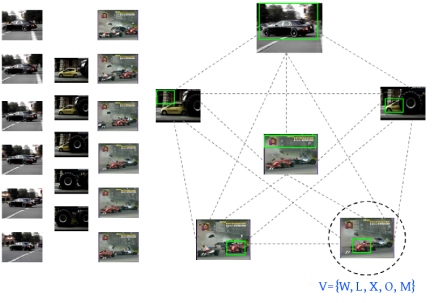


Figure 2: The proposed graph-based representation for example video frames shown on left side. Each vertex in the graph represents an object definition, shown highlighted in figure. Each vertex $V \in \mathbb{V}$ consists of visual word labels $\mathbf{L}$, spatial coordinates $\mathbf{X}$, associated object instances $\mathbf{O}$, their matches with the representative object definition $\mathbf{M}$ and tf-idf descriptor $\mathbf{W}$ [2].

manner:

$$Sim^f(f_i, f_j) = \frac{\mathbf{H}_i \cdot \mathbf{H}_j}{\|\mathbf{H}_i\|_2 \|\mathbf{H}_j\|_2} \tag{1}$$

where $\mathbf{H}_i$ and $\mathbf{H}_j$ are color histograms of f_i and f_j frames and $\|\cdot\|_2$ denotes an L_2 norm. The representative frame (f_R), for the shot (S), is chosen in the following manner:

$$f_R = \underset{\forall k: f_k \in S}{argmax} \sum_{\forall l: f_l \in S \setminus f_k} Sim^f(f_k, f_l) \tag{2}$$

Henceforth, in this paper we use $\mathbf{H}_i$ and $\mathbf{H}_j$ to denote color histograms of representative frames of shots S_i and S_j, respectively, unlike in the above discussion, where they denote color histograms of frames f_i and f_j, respectively. The approach we take for generating object definitions builds on the bag-of-words formulation of Sivic [19]. For each video frame affine-invariant regions and maximally stable regions, suggested by Mikolajczyk [12] and Matas et al. [11], respectively are detected, which are described by invariant descriptors suggested by Lowe [9]. The descriptors are vector quantized through a visual vocabulary generated using the approximate k-means algorithm of Simon et al. [18]. Most frequent 10% of the vocabulary words serve as the stop-list [21]. Further, matching between these salient points of consecutive images is then carried out to attain trajectories across images of a shot. Subsequently, we perform object grouping and achieve object definitions to represent

them in the manner proposed in the next section. The proposed representation, $G(\mathbb{V},\mathbb{E})$ for a very small video consisting of three shots is shown in Figure 2 along with the thumbnail images of the video for example. Note that we achieve one such representation for each scene. Object definitions are described by $\{\mathbf{L},\mathbf{X}\}$, which are visual vocabulary labels and spatial coordinates of salient points, respectively.

4 Object Grouping and Graph Construction for a Scene

The procedure to perform object grouping over a scene, consisting of multiple shots, builds on object grouping over a video shot, which is described next.

4.1 Object grouping over a shot

As trajectories across images of a shot are available, we proceed to group them so that trajectories belonging to the same object can be grouped together. Object grouping is computationally involved and hence we perform it over the data sampled at 1 frame per second (fps). The mentioned approach builds on the approach taken by Yuen [23] in multiple ways, including utilization of appearance information, deciding upon the variance of data and importantly, deciding upon number of groups in which trajectories should be clustered. The distance between two trajectories is computed in the following manner:

$$d(T_i,T_j) = \begin{cases} \frac{1}{|T_i \cap T_j|} \sum_{\forall k: T_i \in f_k, T_j \in f_k} d_S^k(\mathbf{x}_i,\mathbf{x}_j) \times d_A^k(\mathfrak{h}_i,\mathfrak{h}_j); & \text{if } |T_i \cap T_j| > 0 \\ 0; & \text{Otherwise} \end{cases} \tag{3}$$

where $d_S^k(\mathbf{x}_i,\mathbf{x}_j)$ and $d_A^k(\mathfrak{h}_i,\mathfrak{h}_j)$ are distances in spatial domain and appearance in images where trajectories overlap ($T_i \cap T_j \neq \{\phi\}$). $|T_i \cap T_j|$ denotes number of frames in which trajectories overlap, whereas, $\mathbf{x}_i = (x_i, y_i)$ and $\mathbf{x}_j$ are spatial coordinates and $\mathfrak{h}_i$ and $\mathfrak{h}_j$ are color histograms. The spatial distance is defined below:

$$d_S^k(\mathbf{x}_i,\mathbf{x}_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{4}$$

where x_i and y_i denote spatial coordinates of i^{th} trajectory and x_j and y_j denote spatial coordinates of j^{th} trajectory in k^{th} image. Distance in appearance is computed as

$$d_A^k(\mathfrak{h}_i,\mathfrak{h}_j) = \frac{\|\mathfrak{h}_i - \mathfrak{h}_j\|_1}{\|\mathfrak{h}_i + \mathfrak{h}_j\|_1} \tag{5}$$

where $\mathfrak{h}_i$ and $\mathfrak{h}_j$ denote local color histograms of i^{th} and j^{th} trajectories in the k^{th} image, computed as an eight-bin histogram for each color band in a 7×7 neighbourhood and $\|\cdot\|_1$ denotes L_1 norm of an argument.

In order to perform grouping of similar trajectories, we compute an entry in the similarity matrix $\mathbf{S}$ in the following manner:

$$S(i,j) = e^{\left(-\frac{d(T_i,T_j)}{\sigma^2}\right)};\ \sigma^2 = \frac{1}{\sum_i \sum_j I[|T_i \cap T_j| > 0]} \sum_i \sum_j \sum_{\forall k: T_i, T_j \in f_k} \frac{d_S^k(\mathbf{x}_i,\mathbf{x}_j)}{|T_i \cap T_j|} \tag{6}$$

where I[·] is a function which is unity only when the argument is true. σ is computed in this manner to keep it up to the scale of the data. In order to fix the scale of the data,

we normalize spatial coordinates of salient points in each image such that their centroid is shifted to origin and the mean distance of points in this shifted coordinate system is $\sqrt{2}$ [5]. These normalized coordinates are utilized in Equations (4) and (6). We estimate the number of clusters (c) in which trajectories should be grouped, in the following way:

$$\text{Number of clusters}\,(c) = \sum_{k=1}^{|S|} I\left[\frac{\lambda_{max}}{\lambda_k} > \theta\right] \tag{7}$$

where I[·] is defined as above over eigenvalues (λ) of the similarity matrix $\mathbf{S}$. We empirically choose θ to be 15. Choice of such a function can be understood through the number of modes (eigenvalues) required for explaining variance in the underlying data. Only modes with a significant contribution are considered as potential clusters. Further, we utilize normalized cuts [17] to obtain 'c' number of clusters from trajectories. Each cluster serves as an object definition ($\{\mathbf{L}, \mathbf{X}\}$) taken from the representative frame of the shot. So, rather than representing each frame as a vertex in a graph, we represent object definition as a vertex and obtain a compact representation. However, similar objects might occur in subsequent shots of a scene, which need to be identified for achieving further compactness in a representation.

4.2 Object grouping over a scene

In order to extend the methodology of object grouping of the previous subsection to all shots of that scene, all instances of an object need to be identified across all shots of that scene. Hence, we perform threading across shots to reduce the search space. For creating threads of shots, we compare histograms of representative frames, as shown below, and link two shots if similarity is higher than a pre-determined threshold.

$$Sim^S(S_i, S_j) = \frac{\mathbf{H}_i \cdot \mathbf{H}_j}{\|\mathbf{H}_i\|_2 \|\mathbf{H}_j\|_2} \tag{8}$$

where $\mathbf{H}_i$ and $\mathbf{H}_j$ denote color histograms of representative frames of shots S_i and S_j, respectively. In order to extract instances of an object across a scene, we wish to have consistency in clustering of trajectories. However, due to changes in lighting conditions, partial occlusion, changes in camera angles, etc., clustering of trajectories might be inconsistent. For this purpose, we keep track of the past object definitions in sorted order of visual vocabulary and match them with the visual words of a representative frame of the current shot, which are also sorted for reducing the matching complexity. Similarity between trajectories which are grouped together in already generated object definitions is boosted up by taking square root of the original similarity between them. Typical effect of this step over object grouping is shown in Figure 3.

During grouping over a shot if we find object definition to be similar (based on occurrence of visual words) to the previous object definition then we treat such a group as an instance of the already mined object definition and store all instances as attributes $\{\mathbf{O}, \mathbf{M}\}$ to the corresponding vertex, where $\mathbf{O}$ and $\mathbf{M}$ denote matched object instances in other frames and key-point matches with an object definition from the representative frame. Otherwise, we generate a new vertex for this object definition ($\{\mathbf{L}, \mathbf{X}\}$) and store other instances as attributes to this new vertex. We limit the search to the object definitions created from ten most recent video shots. In most cases, visually similar shots tend to be temporally closer, which allows us to achieve excellent results. The representation at this stage for a typical vertex in the graph is shown in Figure 4(a).

(a) Grouping from the Previous Shot (b) Grouping for the Current Shot (c) Adapted Grouping the for Current Shot

Figure 3: Effect of adaptive grouping. Green labels imposed on image indicate to which group the point belongs. (a) Grouping from the previous shot, (b) Grouping for the current shot without taking information from the previous shot and (c) Grouping for the current shot with information from the previous shot. Regions in red ellipses in image (b) indicate incorrectly clustered points compared to image (a). Corresponding regions in image (c) indicate improvement over the previous grouping.

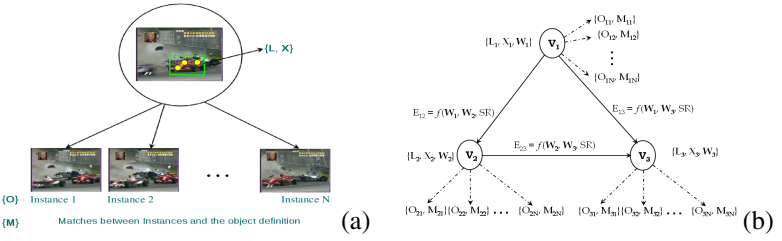


Figure 4: (a) A typial vertex representation. A vertex V is represented as $\{\mathbf{L},\mathbf{X},\mathbf{O},\mathbf{M}\}$, which are vocabulary labels, spatial coordinates, related object instances and their matches, respectively and (b) Illustration of a final representation. Each vertex has an added attribute $\mathbf{W}$, which is a tf-idf descriptor, whereas other attributes are carried over from Figure 4(a). Edge weights are determined as functions of tf-idf descriptors and spatial re-ranking (SR).

4.3 Graph Construction for entire Video

In order to connect vertices of a graph through edges carrying similarities between them, we compute term frequency-inverse document frequency (tf-idf) descriptor [2], shown below, for each object definition.

$$\mathbf{W}_t = (W_1, W_2, \cdots, W_K)^T \tag{9}$$

where $\mathbf{W}_t$ represents a tf-idf descriptor for a vertex t and each entry in this vector, for a total of size of vocabulary, K entries, is computed in the following manner:

$$W_k = \frac{n_{kt}}{n_t} log \frac{N}{N_k}; \forall k \in K \tag{10}$$

where W_k, n_{kt}, n_t, N and N_k denote k^{th} entry in vector, number of occurrences of k^{th} word in vertex t, total number of words in vertex t, total number of vertices in the graph for that scene and number of vertices containing k^{th} word in the entire graph of that scene, respectively. Note that we keep separate statistics for each scene. Similarity between tf-idf descriptors is computed in the following manner:

$$Sim^t(\mathbf{W}_i, \mathbf{W}_j) = \frac{\mathbf{W}_i \cdot \mathbf{W}_j}{\|\mathbf{W}_i\|_2 \|\mathbf{W}_j\|_2} \tag{11}$$

where $\mathbf{W}_i$ and $\mathbf{W}_j$ denote tf-idf descriptors for i^{th} and j^{th} vertices, respectively. Further, we perform spatial re-ranking between vertices of the obtained graph. Object instances associated with a representative definition do not participate in spatial re-ranking as they are similar in form to the representative object definition. This saves a significant amount of time while mining. We perform spatial re-ranking by fitting homography between representative

object definitions. Homography with a larger number of points in consensus weighs the edge connecting two vertices higher than its counterpart. We re-weigh 15 edges connecting best matched vertices for each vertex, which typically re-ranks at least $80-100$ object instances. This clearly is the most time consuming part of the object mining algorithms and we save a significant amount of time and computation through the proposed representation.

Example representation for a very small graph with 3 vertices is shown in Figure 4(b). Note that we achieve one such representation for each scene. In order to make retrieval faster, we implement inverted file index (for each graph corresponding to a scene). At the time of retrieval, scene information about the frame carrying query object is known and hence only vertices of the graph under consideration need to be ranked. Instances related to these vertices are taken as retrieved frames carrying the query object. Localization in such instances at the time of retrieval is perfromed using the attributes $\{\mathbf{M}, \mathbf{X}\}$. The proposed algorithm for achieving the proposed representation is summarized below.

Objective: For a given collection of videos, D, achieve a disjoint graph-based representation $G(\mathbb{V},\mathbb{E})$.
Algorithm:

I. **Dynamic time warping** *(Section 3)*: Use dynamic time warping to obtain scenes $\mathbf{C} = [C_1, \cdots, C_N]$ from P_T and B_T and segment D in N parts.

for $\forall C_i \in \mathbf{C}$ **do**

II. **Shot segmentation and representative frames** *(Section 3)*: Perform shot segmentation such that $C_i = S_1 \cup S_2 \cup \cdots \cup S_M$ and find representative frames (f_R) for each shot (S_j).

III. **Generating trajectories** *(Section 3)*: Perform key-point detection and description and generate trajectories (T) across all frames of each shot $S_j \in C_i$.

IV. **Object grouping** *(Section 4)*: Estimate number of clusters (c) and use normalized-cuts to obtain clusters. Each object definition forms a vertex $V = \{\mathbf{L}, \mathbf{X}, \mathbf{O}, \mathbf{M}\}$.

V. **Description and similarity between object definitions** *(Section 4)*: Compute tf-idf descriptors ($\mathbf{W}$) such that $V = \{\mathbf{L}, \mathbf{X}, \mathbf{O}, \mathbf{M}, \mathbf{W}\}$ and perform spatial re-ranking so that $\forall i, j : e_{ij} \in \mathbb{E}$, weight of e_{ij} is $f(W_i, W_j, SR)$, where SR denotes spatial re-ranking.

end for

VI. **Final graph-based representation** *(Section 4)* :
Final representation, $G(\mathbb{V},\mathbb{E}) = \bigcup_{i=1}^{N} G_i(\mathbb{V}_i,\mathbb{E}_i)$

Algorithm 1: Outline of the proposed method

5 Experimental results

We evaluate perfromance of the proposed method for applications to object retrieval and video summarization over two different types of datasets. The task of manually generating ground-truth for large videos is cumbersome. Hence, we evaluate results of object retrieval with respect to the ground truth over a relatively smaller duration dataset. We also provide video summarization results over a smaller subset of this dataset due to space constraints. We evaluate compactness of the achieved representation and provide timing analysis along with the retrieval results over a much larger dataset taken from a popular sitcom "Friends".

5.1 Evaluation of object retrieval and video summarization

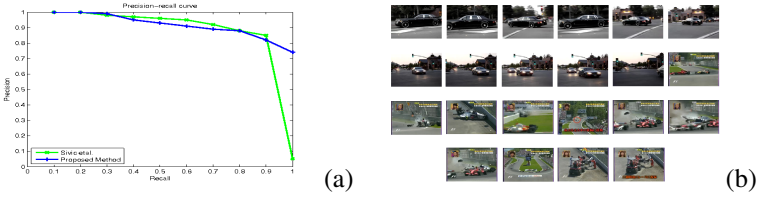
Summary of the dataset over which we evaluate object retrieval results with respect to the ground truth is given in Table 1.

Video	# Frames	# Shots	# Desc	K	# Objects
1	2134	219	$\sim 1.5M$	3.2k	589
2	2560	256	$\sim 1.8M$	3.5k	1096

Table 1: Summary of the representation generated from the proposed method. Table denotes number of frames, number of shots, total number of descriptors in millions ($\#Desc$), size of the codebook (K) and number of vertices ($\#Objects$) generated in final representation.

The first dataset of 2134 frames is an ensemble of many small videos taken from youtube.com. The other dataset is also an ensemble of a number of small TV human interactions taken from [1]. For both these videos, subtitles and scripts were not available and hence all the frames were considered to be emanating from a single scene. Availabilty of scripts and subtitles only account for disjoint representation, which is accounted in next subsection. Note that the proposed method generated 589 and 1096 vertices, rather than 2134 and 2560 vertices, in the final representation for videos 1 and 2, respectively. More vertices in the representation of a second video is due to the larger number of background objects present.

We evaluate the object retrieval results with respect to the manually generated ground-truth for above mentioned videos. Precision and recall achieved by the proposed scheme is compared to the approach of Sivic [19], where, precision-recall was evaluated only on 164 frames of 48 various shots of the movie "Run Lola Run". Results of the proposed scheme are evaluated on a relatively larger dataset of Table 1. For this purpose, we query for 10 randomly selected objects and repeat this process for 10 times. Comparison is shown in Figure 5(a). Clearly, precision we obtain for the proposed scheme is slightly lower compared to [19]. This is due to the object grouping step, which requires matching to be carried out across images of a shot. Due to which, points which are not consistently detected across images are not taken into account. However, the recall that we achieve from the proposed scheme is higher than [19]. This is also expected as we group all instances of an object together under a single vertex. Hence, when a vertex is found as a match for the query object, all instances are retrieved simultaneously. Note that object grouping step decreases precision marginally but provides compactness in the representation.



(a) (b)

Figure 5: (a) Mean precision-recall curves of the proposed scheme and [19] and (b) frames selected for video summarization for a 110 frame video (cc: supplementary material) of 22 different shots.

As object definitions are selected from representative frames and all instances of a similar object are clubbed together, distinct video frames which provide object definitions fulfill likelihood and orthogonality criteria for summarization of Simon et al. [18]. Hence, we achieve video summarization as a by-product of the representation for object retrieval. Summarization result over a very small video consisting of 110 frames of 22 diverse shots, consisting of shots taken from video 1 of Table 1, is shown in Figure 5(b).

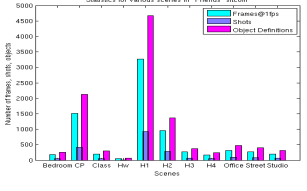
5.2 Compactness and timing analysis

The larger dataset over which we analyse timing and compactness attributes is taken from a sitcom "Friends", which is described in Table 2. Specifically, we provide results over 6 episodes (2 to 7) of season 6. For the task of generating visual vocabulary, we randomly selected 526 frames from the dataset. For the purpose of retrieval, we utilize data achieved at 10 fps (73813 frames), whereas grouping is performed over frames achieved at 1 fps (7381 frames). Note that this dataset is much larger than a baseline dataset used by Sivic [21], which contains 5640 frames achieved at 1 fps from the movie "Groundhog Day".

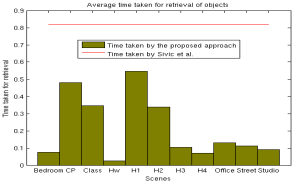
# Episodes	# Frames	Duration	# DS	# Scenes	# Shots	K	# Objects
6	221454	123 mins	11	84	2111	10000	10580

Table 2: Summary of the dataset taken from "Friends" sitcom. Table denotes number of episodes, number of frames in video at 30 fps, duration of the video in minutes, number of distict scenes in video ($\#DS$), total number of scenes, number of shots in entire video, size of the codebook used (K) and number of object definitions generated ($\#Objects$) in the final disjoint representation.

Note that for a total of 73813 frames, only 10580 object definitions were required. Statistics for each individual scene containing number of frames at 1 fps(for displaying all on the same scale), number of shots and number of object definitions is shown in Figure 6(a).



(a)



(b)

Figure 6: (a) Statistics for individual scenes in terms of number of video frames at 1 fps, number of shots and number of object definitions and (b) Timing analysis for retrieval of objects from video data of Table 2. Scenes shown on X-axis are Bedroom, Central Perk (CP), Classroom, Hallway (HW), House 1 (H1), House 2 (H2), House 3 (H3), House 4 (H4), Office, Street and Studio, respectively.

Due to disjoint representation and search over object definitions (10580) rather than over frames (73813) through the inverted file index, retrieval through the proposed scheme is faster. A rough comparison to a baseline of Sivic et al. [21] is provided in Figure 6(b), which shows that the proposed technique is much more efficient in retrieval. The proposed scheme is implemented on Matlab over a 2 GHz processor, which is similar to the configuration reported by Sivic et al. with 0.82 second of average retrieval time.

Note that operations needed to be performed at retrieval time vary slightly in both approaches. In [21] frequency ranking and spatial consistency re-ranking are performed at retrieval time, whereas, in the proposed approach frequency ranking is performed in conjunction with the matches for other instances of the object. Timing analysis for the proposed scheme was done by averaging over retrieval times for 6 randomly selected objects for each scene. As shown in figure, retrieval for scenes with less number of objects is much faster compared to the scenes with relatively more number of frames. Some example object retrievals are shown in Figure 7 over the dataset of Table 1.

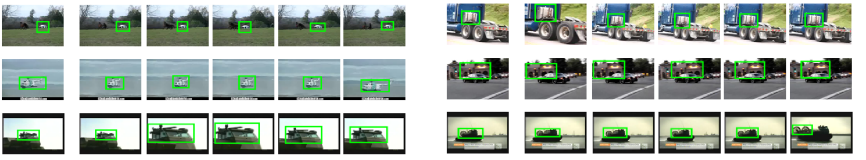


Figure 7: Example retrieval results over the dataset described in Table 1. Frames in the first columns of left and right sides contain query objects, whereas frames in the subsequent columns on both sides represent corresponding frames retrieved. Objects are shown in green rectangles.

6 Conclusions

An efficient method for automatic video object mining for retrieval and summarization has been presented with emphasis on compact representation and faster retrieval. The conciseness of representation is possible primarily due to redundancy in video data, which is exploited and coded as associations with selected object definitions. This reduces time required for retrieval. Also, use of textual information allows the representation to be disjoint and hence reduces the search space while retrieving objects. This representation opens up questions like what is the most compact possible representation which can provide high precision and recall and trade-off between storage space requirements and efficiency in retrieval.

References

[1] P. Alonso, M. Marszalek, A. Zisserman, and I. Reid. High five: Recognizing human interactions in tv shows. In Proc. BMVC, 2010.

[2] R. Baeza-Yates and B. Ribeiro-Neto. Modern Information Retrieval. Addison-Wesley Longman Publishing Co., ISBN: 020139829X, 1999.

[3] O. Chum and J. Matas. Large scale discovery of spatially related images. IEEE Transactions on PAMI, 32(2):371–377, 2008.

[4] T. Cour, C. Jordan, E. Miltsakaki, and B. Taskar. Movie/script: Alignment and parsing of video and text transcription. In Proc. ECCV, pages 158–171, 2008.

[5] R. I. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, ISBN: 0521623049, 2000.

[6] M. Jamieson, A. Fazly, S. Stevenson, S. Dickinson, and S. Wachsmuth. Using language to learn structured appearance models for image annotation. IEEE Transactions on PAMI, 32(1):148–164, 2010.

[7] V. Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. In Soviet Physics Doklady, 10:707–710, 1966.

[8] R. Lienhart. Reliable transition detection in videos: A survey and practitioner's guide. International Journal of Image and Graphics, 1(3):469–486, 2001.

[9] D. Lowe. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision, 60(2):91–100, 2004.

[10] M. Marszalek, I. Laptev, and C. Schmid. Actions in context. In Proc. CVPR, pages 2929–2936, 2009.

[11] J. Matas, O.Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. Image and Vision Computing, 22(10):761–767, 2004.

[12] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In Proc. ECCV, 2350:128–142, 2002.

[13] C. Myers and L. Rabiner. A comparatuve study of several dynamic time warping algorithms for connected word recognition. The Bell system technical journal, 60(7): 1389–1409, 1981.

[14] J. Philbin and A. Zisserman. Object mining using a matching graph on large image collections. In Indian conf. on computer vision, graphics and image processing, pages 738–745, 2008.

[15] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabulary and fast spatial matching. In Proc. CVPR, pages 1–8, 2007.

[16] T. Quack, V. Ferrari, and Luc Van Gool. Video mining with frequent itemset configurations. In Proc. CIVR, Springer, pages 360–369, 2006.

[17] J. Shi and J. Malik. Normalized cuts and image segmentation. IEEE transactions on PAMI, 22(8):888–905, 2000.

[18] I. Simon, N. Snavely, and S. Seitz. Scene summarization for online image collections. In Proc. ICCV, pages 1–8, 2007.

[19] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In Proc. ICCV, 2:1470–1477, 2003.

[20] J. Sivic and A. Zisserman. Video data mining using configuration of viewpoint changes. In Proc. CVPR, 1:488–495, 2004.

[21] J. Sivic and A. Zisserman. Efficient visual search of videos cast as text retrieval. IEEE transactions on PAMI, 31(4):591–606, 2009.

[22] J. Sivic, F. Schaffalitzky, and A. Zisserman. Object level grouping for video shots. International Journal on Computer Vision, 67(2):189–210, 2006.

[23] J. Yuen and A. Torralba. A data-driven approach for event prediction. In Proc. ECCV, pages 707–720, 2010.