Unsupervised Summarization of Rushes Videos

Yang Liu[1,2], Feng Zhou[2], Wei Liu[2], Fernando De la Torre[2], Yan Liu[1]
[1]Department of Computing, The Hong Kong Polytechnic University, Hong Kong, P. R. China
[2]Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA
csygliu@comp.polyu.edu.hk, zhfe99@gmail.com, lwbiosoft@gmail.com
ftorre@cs.cmu.edu, csyliu@comp.polyu.edu.hk

ABSTRACT

This paper proposes a new framework to formulate summarization of rushes video as an unsupervised learning problem. We pose the problem of video summarization as one of time-series clustering, and proposed Constrained Aligned Cluster Analysis (CACA). CACA combines kernel k-means, Dynamic Time Alignment Kernel (DTAK), and unlike previous work, CACA jointly optimizes video segmentation and shot clustering. CACA is efficiently solved via dynamic programming. Experimental results on the TRECVID 2007 and 2008 BBC rushes video summarization databases validate the accuracy and effectiveness of CACA.

Categories and Subject Descriptors

I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding—*Video analysis*; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Abstracting methods*; H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems—*Video*

General Terms

Algorithms, Experimentation

Keywords

Constrained aligned cluster analysis, dynamic programming, dynamic time alignment kernel, unsupervised learning, rushes video summarization, TRECVID

1. INTRODUCTION

The aim of rushes video summarization [5, 6] is to summarize a film as a professional film cutter would do using the raw footage. Compared with traditional method for video summarization [2, 4], rushes video summarization [5, 6, 7] has some important difference due to the structural characteristics of rushes videos. Rushes are the unorganized raw footages with considerable junk clips (such as color bars and monochrome frames) and redundant information (such as retakes of the same shot). To produce a compact and informative summary of the rushes video, the junk clips and redundant information should be detected and discarded.

MM'10, October 25–29, 2010, Firenze, Italy.

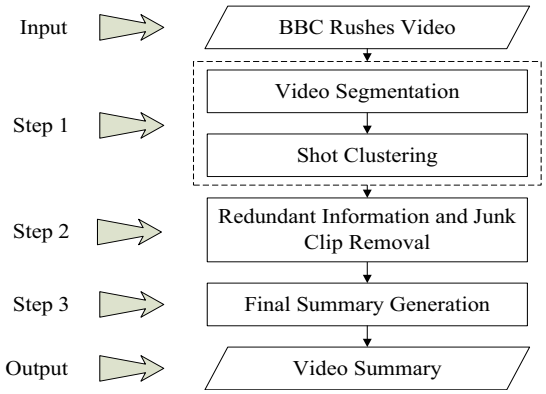


Figure 1: Flowchart of the proposed framework.

Generally, the procedure of rushes video summarization can be divided into four steps: video segmentation, shot clustering, redundant information and junk clip removal, and final summary generation (shown as four solid line boxes in Figure 1) [5, 6]. Among these four steps, video segmentation and shot clustering are of great importance since accurate segmentation and clustering can reflect the semantic meaning of video sequences and thus essentially improve the performance of further processing. However, previous work treat these two steps independently, which might result in unsatisfactory results due to the hard decision of the segmentation process. In this paper, we present an unsupervised framework for joint video segmentation and shot clustering (see step 1, i.e., the dotted line box in Figure 1), called Constrained Aligned Clustering Analysis (CACA). CACA finds the optimal segmentation and shot clustering using dynamic programming in polynomial time. After video segmentation and shot clustering, we detect and remove the redundant information as well as junk clips (step 2). Finally, we generate the summary (step 3). Figure 1 describes the flowchart of proposed framework.

2. PROPOSED FRAMEWORK

This section proposes the new framework for rushes video summarization. Section 2.1 introduces the unified procedure for video segmentation and shot clustering. Section 2.2 de-

scribes the detection and removal of redundant information as well as junk clips. Section 2.3 presents the generation of the final summary.

2.1 Video Segmentation and Shot Clustering

Given a video sequence, we first extract the local color histogram as the feature for each frame. Each video frame is divided into 16 ($= 4 \times 4$) subimages with the same size. For each sub-image, 16 bins' color histogram on HSV color space is extracted according to MPEG-7 [3]. Therefore, each frame is represented as a 256-dimensional feature vector.

2.1.1 Similarity Measure of Video Sequences

In order to segment and cluster video sequences, we need to define a distance metric to measure the similarity between video sequences with different lengths. To align time series, a frequent approach is Dynamic Time Warping (DTW). A known drawback of using DTW as a distance metric is that it fails to satisfy the triangle inequality. To address this issue, Shimodaira et al. [8] proposed Dynamic Time Alignment Kernel (DTAK).

Given two video sequences $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_{n_x}] \in \mathbb{R}^{d \times n_x}$ and $\mathbf{Y} = [\mathbf{y}_1, ..., \mathbf{y}_{n_y}] \in \mathbb{R}^{d \times n_y}$ (d is the feature dimension of the video frame, n_x and n_y are the frame numbers of video sequences $\mathbf{X}$ and $\mathbf{Y}$, respectively), DTAK is defined as:

$$\tau(\mathbf{X}, \mathbf{Y}) = \frac{u_{n_x n_y}}{n_x + n_y} \tag{1}$$

where $u_{ij} = \max\{u_{i-1,j} + \kappa_{ij}, u_{i-1,j-1} + 2\kappa_{ij}, u_{i,j-1} + \kappa_{ij}\}$, and $\kappa_{ij} = \phi(\mathbf{x}_i)^T \phi(\mathbf{y}_j)$ is the kernel similarity between two frames $\mathbf{x}_i$ and $\mathbf{y}_j$, which composes the similarity matrix $\mathbf{K} \in \mathbb{R}^{n_x \times n_y}$. The DTAK is constructed in a recursive manner. The cumulative similarity matrix $\mathbf{U} \in \mathbb{R}^{n_x \times n_y}$ is initialized at the upper-left, i.e., $u_{11} = 2\kappa_{11}$.

Figure 2 illustrates the procedure of constructing $\mathbf{U}$ for two video sequences based on the similarity matrix $\mathbf{K}$. Figure 2(a) shows two video sequences and the alignment result on two-dimensional PCA subspace. In this example, $\mathbf{x}_1$ and $\mathbf{y}_1$ are aligned; $\mathbf{x}_2$ and $\mathbf{y}_2$ are aligned; $\mathbf{x}_3$ is aligned with both $\mathbf{y}_3$ and $\mathbf{y}_4$; finally, $\mathbf{x}_4$ is aligned with $\mathbf{y}_5$. Figure 2(b) is the similarity matrix $\mathbf{K}$ of these two video sequences, in which the frame similarity is calculated by the RBF kernel. Figure 2(c) is the cumulative similarity matrix. The final value of DTAK, $\tau(\mathbf{X}, \mathbf{Y}) = \frac{6.5}{9}$, is computed by normalizing the bottom-right of cumulative similarity matrix with the sum of sequence lengths.

2.1.2 Video Segmentation and Shot Clustering

In this section, we propose Constrained Aligned Cluster Analysis (CACA) an extension of Aligned Cluster Analysis (ACA) [9], for video segmentation and shot clustering. Given a video sequence $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n] \in \mathbb{R}^{d \times n}$ with n frames, we aim to decompose $\mathbf{X}$ into m disjointed segments, each of which belongs to one of k clusters. The i^{th} segment $\mathbf{Y}_i = \mathbf{X}_{[s_i, s_{i+1})} = [\mathbf{x}_{s_i}, ..., \mathbf{x}_{s_{i+1}-1}]$. The length of $\mathbf{Y}_i$, $n_i = s_{i+1} - s_i$, is constraint as $n_i \in [1, n_{\max}]$. An indicator matrix $\mathbf{G} \in \{0, 1\}^{k \times m}$ is used to assign each segment to a cluster: $g_{ci} = 1$ if $\mathbf{Y}_i$ belongs to cluster c, otherwise $g_{ci} = 0$. The objective function of CACA can be written as:

$$J_{caca}(\mathbf{G}, \mathbf{s}) = (1-\lambda) \sum_{c=1}^{k} \sum_{i=1}^{m} g_{ci} dist_{\psi}^2(\mathbf{Y}_i, \mathbf{z}_c) + \lambda \sum_{i=2}^{m} b_{s_i} \tag{2}$$

where $\mathbf{s} \in \mathbb{R}^{(m+1) \times 1}$ is the vector that contains the start and end of each segment. $dist_{\psi}^2(\mathbf{Y}_i, \mathbf{z}_c)$ is the squared distance

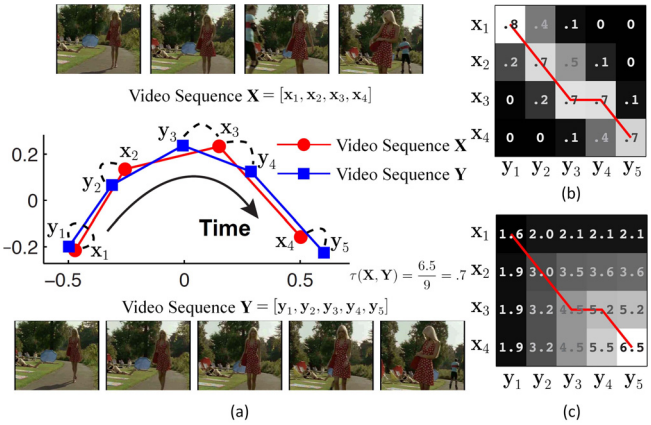


Figure 2: Dynamic time alignment kernel (DTAK) as similarity measure for two video sequences. (a) Projection of the video into the the first two principal components. Dotted lines denote the final alignment of frames. (b) Similarity matrix K. (c) Cumulative similarity matrix U.

between the i^{th} segment and the center of cluster c in the feature space defined by the implicit mapping $\psi(\cdot)$, i.e.,

$$dist_{\psi}^2(\mathbf{Y}_i, \mathbf{z}_c) = \tau_{ii} - \frac{2}{m_c} \sum_{j=1}^{m} g_{cj} \tau_{ij} + \frac{1}{m_c^2} \sum_{j_1, j_2 = 1}^{m} g_{cj_1} g_{cj_2} \tau_{j_1 j_2} \tag{3}$$

where $m_c = \sum_{j=1}^{m} g_{cj}$ is the number of segments that belong to cluster c. The dynamic kernel function τ is defined as $\tau_{ij} = \psi(\mathbf{Y}_i)^T \psi(\mathbf{Y}_j)$. CACA extends ACA [9] by adding a regularization term $\mathbf{b} = [b_1, ..., b_n]^T$ to further improve the segmentation performance, where b_{s_i} penalizes the segmentation between $(s_i - 1)^{th}$ and s_i^{th} frames. $\lambda \in [0, 1]$ is a trade-off parameter to balance the within-cluster error $dist_{\psi}^2(\mathbf{Y}_i, \mathbf{z}_c)$ and the boundary error b_{s_i}.

2.1.3 Dynamic Programming for CACA

As in the case of ACA [9], we use a coordinate-descent algorithm to minimize CACA:

$$\mathbf{G}, \mathbf{s} = \underset{\mathbf{G}, \mathbf{s}}{\operatorname{argmin}} (1-\lambda) \sum_{c=1}^{k} \sum_{i=1}^{m} g_{ci} dist_{\psi}^2(\mathbf{Y}_i, \dot{\mathbf{z}}_c) + \lambda \sum_{i=1}^{m} b_{s_i} \tag{4}$$

where $\dot{\mathbf{z}}_c$ is the cluster center computed from the segmentation $(\dot{\mathbf{G}}, \dot{\mathbf{s}})$ obtained in the previous step. Given a video sequence $\mathbf{X}$ of length n, however, the number of all possible segmentations is $O(2^n)$, which makes a brute-force search infeasible. In this subsection, we present a dynamic programming (DP) based algorithm to find the optimal solution in polynomial time. First we rewrite (2) as follows:

$$J_{caca}(\mathbf{G}, \mathbf{s}) = (1 - \lambda) \sum_{i=1}^{m} dist_{\psi}^2(\mathbf{Y}_i, \mathbf{z}_{c_i^*}) + \lambda \sum_{i=1}^{m} b_{s_i} \tag{5}$$

where c_i^* denotes the label of the closest cluster for segment $\mathbf{Y}_i$, i.e., $g_{c_i^* i} = 1$. Observe that $\mathbf{G}$ is determined once $\mathbf{s}$ is known. To further leverage the relationship between $\mathbf{G}$ and $\mathbf{s}$, we introduce an auxiliary function $J : [1, n] \to \mathbb{R}$,

$$J(v) = \min_{\mathbf{G}, \mathbf{s}} J_{caca}(\mathbf{G}, \mathbf{s})|_{\mathbf{X}_{[1,v]}} \tag{6}$$

to relate the minimum energy directly with the tail position v of the subsequence $[\mathbf{x}_1, ..., \mathbf{x}_v]$. Actually, J satisfies the

principle of optimality [1], i.e.,

$$J(v) = \min_{1<i\leqslant v} \Big(J(i-1) + \min_{\mathbf{G},\mathbf{s}} J_{caca}(\mathbf{G},\mathbf{s})|\mathbf{x}_{[i,v]} \Big) \qquad (7)$$

which implies that the optimal decomposition of the subsequence $\mathbf{X}_{[1,v]}$ is achieved only when the segmentations on both $\mathbf{X}_{[1,i-1]}$ and $\mathbf{X}_{[i,v]}$ are optimal and their sum is minimal. Although the number of possible ways to decompose sequence $\mathbf{X}$ is exponential in n, DP [1] offers an efficient way to minimize J by using Bellman's equation:

$$J(v) = \min_{v-n_{\max}<i\leqslant v} \Big(J(i-1) + \lambda b_i + (1-\lambda) \min_{\mathbf{g}} \sum_{c=1}^{k} g_c dist_{\psi}^2(\mathbf{X}_{[i,v]}, \dot{\mathbf{z}}_c) \Big) \qquad (8)$$

where $dist_{\psi}^2(\mathbf{X}_{[i,v]}, \dot{\mathbf{z}}_c)$ is the squared distance between segment $\mathbf{X}_{[i,v]}$ and the center of cluster c. When $v = n$, $J(n)$ is the optimal cost of the segmentation that we seek. The inner values, $i_v^*, \mathbf{g}_v^* = \text{argmin}_{i,\mathbf{g}} J(v)$, are the head position and label for the last segmentation respectively that lead to the minima. We repeat the following forward-backward steps alternately until $J(n)$ converges:

Forward step: Scan from the beginning ($v = 1$) of the sequence to its end ($v = n$). For each v, $J(v)$ is computed according to (8), as well as i_v^* and $\mathbf{g}_v^*$.

Backward step: Trace back from the end of sequence ($v = n$). Cut off the segment whose head $s = i_v^*$. The indicator vector $\mathbf{g} = \mathbf{g}_v^*$ can be indexed from the stored positions. Repeat this operation on the left part of the sequence ($v = i_v^* - 1$).

The computational cost of DP based search is $O(n^2 n_{\max} t)$, where t is the number of iterations.

2.2 Redundant Information and Junk Clip Removal

To exclude the redundant information, we select only the longest shot from each cluster. To remove junk clips, we use the mean value of gradients in the vertical direction (on gray scale) as the feature to distinguish junk frames from normal frames:

$$F_{vg} = \frac{1}{W \times (H-1)} \sum_{i=1}^{W} \sum_{j=1}^{H-1} |I(i,j+1) - I(i,j)| \qquad (9)$$

where $I(i,j)$ is the gray level pixel value at location (i,j). W and H are the width and height of the frame, respectively.

We define a threshold F_{th}. If $F_{vg} > F_{th}$, the corresponding frame is regarded as a normal frame; otherwise it is regarded as a junk frame. For each selected shot, we uniformly sample n_s frames (if the total frame number of the shot is less than n_s, we regard this shot as a junk clip since it is too short to include any useful information). If $\frac{n_{ju}}{n_s} > \varepsilon$, i.e., the ratio of the number of junk frames to the number of sample frames is larger than a predefined value ε, the shot is regarded as a junk clip and then removed. In our experiment, we set $F_{th} = 1$, $n_s = 10$, and $\varepsilon = 0.5$.

2.3 Final Summary Generation

After redundant information and junk clip removal, the remaining shots are representative and informative. We generate the final summary based on the extension of keyframes of these remaining shots. To extract keyframes, we need to calculate how many keyframes we can have in final summary.

According to the requirement of rushes summarization task in TRECVID 2008, the total length of each summary should not be longer than 2% of the original full video. Note that the requirement of rushes summarization task in TRECVID 2007 is slightly different: the total length of each summary should not be longer than 4% of the original full video. In our framework, we follow the stricter requirement in TRECVID 2008. Furthermore, in our framework, we extend each keyframe to a one-second clip (25 frames). Therefore, the maximum number of keyframes that we can have in the summary is: $n_{kf} = [\frac{n_{total} \times 2\%}{25}]$, where n_{total} is the total frame number of original rushes video and $[\cdot]$ denotes the rounding operation.

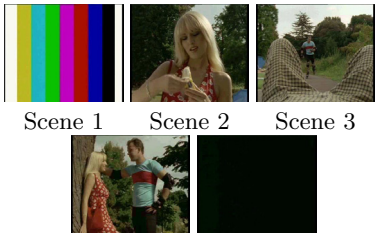
In this paper, we use two methods to extract keyframes. The first method is uniform sampling, i.e., n_{kf} keyframes are uniformly extracted from the remaining shots to generate the final summary. The second method is k-means clustering. We cluster all frames of the remaining shots into n_{kf} groups, and then select the frame closest to each group center as the keyframe. When we get these keyframes, we can generate the final summary by extending each keyframe to a one-second clip and concatenating them orderly.

3. EXPERIMENTS

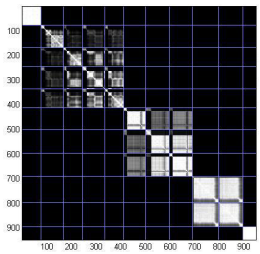
This section evaluates the performance of proposed framework using the test datasets from TRECVID 2007 and 2008 BBC rushes video summarization tasks [5, 6]. These two datasets include about 40 hours' rushes videos provided by BBC Archive. The videos are in MPEG-1 format with 352×288 resolution and the frame rate is 25 fps. First we test the segmentation and clustering performance of proposed framework on a short video and give detailed analysis. Then we report the statistical results of rushes summarization on TRECVID 2007 and 2008 test datasets. In our experiments, we use the RBF kernel in DTAK, and σ is set as the average Euclidean distance between all data points and their 10% closest neighbors (Here each point is a 256-dimensional feature vector introduced in Section 2.1).

3.1 Segmentation and Clustering

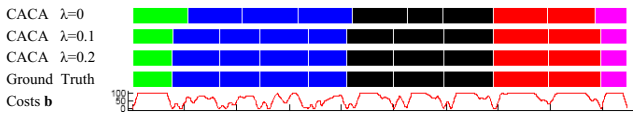
In the first experiment, we tested CACA's performance to segment and cluster a short video from the BBC rushes video MRS148090 of TRECVID 2008. This video lasts three minutes and nine seconds (4732 frames). It includes 5 different scenes (as shown in Figure 3(a)) and 11 shots. The first shot belongs to scene 1; the second to fifth shots belong to scene 2; the sixth to eighth shots belong to scene 3; the ninth and tenth shots belong to scene 4; and the last shot belongs to scene 5. For this video, we uniformly sampled the frames at a rate of $\frac{1}{5}$, and set $k = 5$ and $n_{max} = 105$. Figure 3(b) shows the similarity matrix as well as the segmentation positions obtained by CACA with $\lambda = 0.2$. Different scenes and shots are clearly represented by big and small squares, respectively (pure white point denotes the value 1, which means that the two frames are totally the same; while pure black point denotes the value 0, which means the two frames are totally different). Figure 3(c) shows the segmentation and clustering results of proposed framework with different values of λ, as well as the ground truth provided by human (same color denotes same cluster). If we set $\lambda = 0$, i.e., ignore the regularization term $\sum_{i=2}^{m} b_{s_i}$, the clustering result is approximately correct but boundaries are not very precise. When λ is increased to balance both terms in (2), the



(a) Five different scenes



(b) Similarity matrix



(c) Segmentation and clustering results

Figure 3: Segmentation and clustering results of proposed framework on a 4732-frame video from TRECVID 2008 BBC rushes.

results become more accurate. Specifically, if we set $\lambda = 0.1$ or $\lambda = 0.2$, the segmentation and clustering results of CACA are the same as ground truth provided by human.

In our case, we computed the boundary costs $\mathbf{b} \in \mathbb{R}^{n\times 1}$ as $b_i = \sum_{i_1=i-n_b}^{i-1} \sum_{i_2=i}^{i+n_b-1} \kappa_{i_1 i_2}$, which measures the similarity between the sequences $[\mathbf{x}_{i-n_b}, ..., \mathbf{x}_{i-1}]$ and $[\mathbf{x}_i, ..., \mathbf{x}_{i+n_b-1}]$. In this paper, we set $n_b = 10$. By comparing the ground truth and the values of costs $\mathbf{b}$ in Figure 3(c), we can observe that by setting to a small value $\mathbf{b}$ the correct segmentation is found, outperforming ACA.

3.2 Rushes Video Summarization

In the second experiment, we reported CACA's qualitative results on the TRECVID 2007 and 2008 BBC rushes datasets. There were a total of eight evaluation criteria for the summarization tasks [5, 6]. In this paper, we use the following four to evaluate performance: *DU* - duration of the summary (secs.); *IN* - fraction of inclusions found in the summary (0 - 1); *JU* - Summary contained lots of junk: 1 strongly agree - 5 (best) strongly disagree; *RE* - Summary contained lots of duplicate video: 1 strongly agree - 5 (best) strongly disagree. For *DU*, the lower the score is, the better the performance is. For *IN*, *JU*, and *RE*, the higher the scores are, the better the performance is. In this experiment, we set $k = 30$, $n_{max} = 50$, and $\lambda = 0.2$.

Table 1 and 2 list CACA's results as well as the mean and median results of all participants' submissions in TRECVID 2007 and 2008, respectively. Even though we use very simple methods to remove useless information and generate final summary, due to the effectiveness of the proposed framework in jointly optimizing video segmentation and shot clustering, the results are promising.

Note that the best summarization results reported in [5, 6] are slightly better than the results of proposed algorithm. However, it is worth pointing out that our framework uses very simple algorithms for redundant information removal and final summarization. In the future, we plan to consider more complicated steps to further improve the performance.

Table 1: Summarization performance on TRECVID 2007 rushes summarization task.

Criteria	*DU*	*IN*	*JU*	*RE*
Mean of 22 research groups	50.54	0.49	-	3.65
Median of 22 research groups	50.64	0.51	-	**3.69**
Proposed - uniform sampling	**25.60**	0.59	3.55	3.53
Proposed - k-means	26.17	**0.63**	**3.58**	3.57

Table 2: Summarization performance on TRECVID 2008 rushes summarization task.

Criteria	*DU*	*IN*	*JU*	*RE*
Mean of 31 research groups	27.10	0.44	3.16	3.27
Median of 31 research groups	28.11	0.45	3.11	3.37
Proposed - uniform sampling	**26.15**	0.62	**3.56**	3.53
Proposed - k-means	26.41	**0.67**	3.54	**3.59**

4. CONCLUSION AND FUTURE WORKS

In this paper, we present a new framework for rushes video summarization. We propose CACA, an extension of ACA that allows joint segmentation and shot clustering. The unification of video segmentation and shot clustering not only reduces cumulative errors of these two tasks, but also takes into account the close relationship between them, which further improves the summarization performance. CACA is solved in polynomial time with Dynamic programming. Although we have illustrated the benefits of CACA in the summarization of rushes video, our approach is more general and it can be viewed as a general framework useful for other unsupervised clustering and segmentation video segmentation problems.

5. REFERENCES

[1] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.

[2] A. Ekin, A. Tekalp, and R. Mehrotra. Automatic soccer video analysis and summarization. *IEEE Trans. Image Process.*, 12(7):796–807, 2003.

[3] B. S. Manjunath, J. R. Ohm, V. V. Vasudevan, and A. Yamada. Color and texture descriptors. *IEEE Trans. Circuits Syst. Video Technol.*, 11(6):703–715, 2001.

[4] C.-W. Ngo, Y.-F. Ma, and H.-J. Zhang. Video summarization and scene detection by graph modeling. *IEEE Trans. Circuits Syst. Video Technol.*, 15(2):296–305, 2005.

[5] P. Over, A. F. Smeaton, and G. Awad. The TRECVID 2008 bbc rushes summarization evaluation. In *TVS'08*, pages 1–20. ACM, 2008.

[6] P. Over, A. F. Smeaton, and P. Kelly. The TRECVID 2007 bbc rushes summarization evaluation pilot. In *TVS'07*, pages 1–15. ACM, 2007.

[7] J. Ren and J. Jiang. Hierarchical modeling and adaptive clustering for real-time summarization of rush videos. *IEEE Trans. Multimedia*, 11(5):906–917, 2009.

[8] H. Shimodaira, K.-I. Noma, M. Nakai, and S. Sagayama. Dynamic time-alignment kernel in support vector machine. In *NIPS 14*, pages 921–928, 2001.

[9] F. Zhou, F. de la Torre, and J. K. Hodgins. Aligned cluster analysis for temporal segmentation of human motion. In *IEEE FG*, 2008.