The COST292 Experimental Framework for Rushes Summarization Task in TRECVID 2008

Umut Naci, Uros Damnjanovic, Boris Mansencal, Jenny Benois-Pineau, Christian Kaes, Marzia Corvaglia, Eliana Rossi, Naiara Aginako

The COST292 Experimental Framework for Rushes Summarization Task in TRECVID 2008

S.U. Naci[1], Uros Damnjanovic[2], Boris Mansencal[3], Jenny Benois-Pineau[3], Christian Kaes[3], Marzia Corvaglia[4], Eliana Rossi[4], Naiara Aginako[5]
[1]Delft University of Technology, Delft, The Netherlands
[2]Queen Mary University London, UK
[3]LaBRI, University of Bordeaux 1, France
[4]University of Brescia, Italy
[5]VicomTech, San Sebastian, Spain
[1]s.u.naci@tudelft.nl
[4]marzia.corvaglia@ing.unibs.it

ABSTRACT

In this paper, the method used for Rushes Summarization task by the COST 292 consortium is reported. The approach proposed this year differs significantly from the one proposed in the previous years because of the introduction of new processing steps, like repetition detection in scenes. The method starts with junk frames removal and follows with clustering and scene detection; then for each scene, repetitions are detected in order to extract once the real scene; the following step consists in face detections (faces are considered semantically relevant) and in pan, tilt and zoom detections (other camera motions are usually related to technical operations in the backstage); finally the summary is extracted.

Categories and Subject Descriptors

H.5.1 [**Information Interfaces and Presentation**]: Multimedia information systems:video

General Terms

Algorithms, Experimentation

Keywords

repetition detection, spectral clustering, normalized cuts, mid-level features, face detection, camera motion

1. INTRODUCTION

In this paper we present the activities lead with the purpose of participating in the TRECVID 2008 Rushes Summarization task. The COST 292 consortium has participated to this initiative since 2006 with satisfying and progressively improved results, as well as with a rigorous approach. Concerning the Rushes Summarization task, this year two significant improvements were performed:

- In depth study of development data which allowed the understanding of data structure and thus the consequent understanding of the best strategy for summarization;
- Introduction of repetition detection in scenes in oder to remove the redundant parts and to extract only one scene in a set even if that scene was shot several times.

These two innovative aspects generated a completely revised framework based on five main milestones. Given a video, after shot boundary segmentation, first step consists of filtering that removes the junk frames from the video; junk frames are those frames with few colors, frames which are saturated and color bars. Second, clustering is performed using the spectral video clustering algorithm, specifically the *Normalized Cuts* on frames and then the scenes are extracted considering temporally continuous segments of the same clusters which are close enough. Third, within each scene, repetitions in time are detected using the spectral graph theory. Fourth, the mid–level features are extracted: face detection has been implemented in order to extract segments with additional semantic components; camera motion has be computed with the aim of ignoring segments where camera motion do not belong to the final edited video program or movie. Fifth, the summary of 2% of the initial video is extracted on the basis of the extracted features and the constrains given by NIST.

The organization of this paper is as follows: the framework proposed by COST 292 for Rushes Summarization task is presented in Section 2; results and conclusions are respectively reported in Section 3 and in Section 4.

2. COST 292 FRAMEWORK

The framework for Rushes Summarization task proposed by COST 292 is show in Figure 1. Each box in that diagram is explained in this section.

2.1 Junk frames extraction

According to *instructions for judging video summaries*, to assess how much "junk" a summary contains, a judge has

draft
TVS'08, October 31, 2008, Vancouver, British Columbia, Canada.
draft.

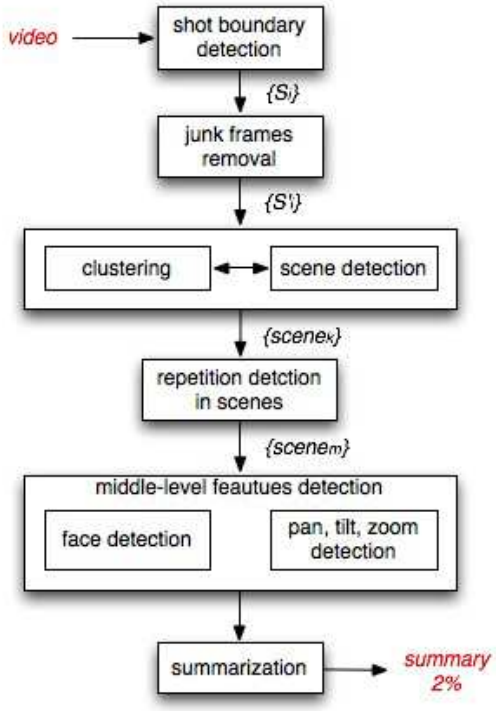

Figure 1: Rushes framework.

to tell if there are "many color bars, clap boards, all black or all white frames". Two kinds of color bars are presented on figure 2 a and b. We think that junk frames should also include frames with few colors, such as in figure 2 c, and saturated frames, such as in figure 2 d. All these kinds of frames would not be included in the final edited movie and thus should not be included in the summary.

Figure 2: Junk frames: a) sharp color bars; b) diffuse color bars; c) grey/black frame; d) saturated frame

To detect frames with few colors, thus in particular sharp color bars and uniform color frames, we use a thresholded histogram on each channel of a frame in RGB format. We then check that we have a reduced number of colors. Furthermore, to detect diffuse color bars (such as in Figure 2 b), we apply the same algorithm to picture downscaled to 8x8. These images can be considered as having undergone low-pass filtering.

For performance reasons, we apply this detection at I-frame temporal resolution. We filter the result with a median filter (of width 5). We then interpolate to P-frames resolution. As frames with few colors, and in particular color bars, often last several seconds, this combined filtering and interpolation method seems to work pretty well.

Our method may falsely detect parts of scripted scenes with very few colors, such as very dark scenes. But we observed that events in such scenes are not very understandable and so do not need to be in the summary.

For the detection and extraction of saturated frames in rushes videos Accumulated Histogram Difference (AHD) [8] technique is used. Each frame is converted into HSV color space in order to process each channel independently. A frame is classified as saturated if it has a low value in the S-channel and a high value in the V-channel.

Firstly, a 256 bins histogram is created for both S- and V-channels and the AHD (see Equation 1) is computed in V-channel for the last 20 bins. For S- and V-channels histogram values are summed for the first and last 35 bins, respectively. These three results are normalized and thresholded in order to classify the frame.

$$AHD_n(x) = \sum_{l=x_{min}}^{x_{max}} \Delta H_n(l) = \sum_{l=x_{min}}^{x_{max}} H_{n+1}(l) - \sum_{l=x_{min}}^{x_{max}} H_n(l) \quad (1)$$

$x_{min} \leqslant x \leqslant x_{max}$; where H_n is the histogram of frame n and x is the number of bin.

For the frames that come after a frame that is classified as saturated, only histogram values are used. In this case, the number of bins of the histogram is reduced to 25 and the threshold used is more restrictive. Therefore, only saturated frames are considered. As a consequence, meaningless saturated frames are completely extracted from rushes videos.

Our summaries have been judged as not containing too much junk frames: indeed, it seems we are 7th/44 on this criteria.

We still lack a tool to detect clap boards. This could help us to improve our results on junk frames removal. But it could also help us to better separate unscripted and scripted parts of video and thus restrain event detection to scripted parts.

2.2 Clustering and Scene detection

Clustering of frames is used as a first step of the scene detection process. Scenes are treated as segments that hold same semantic information, or which are part of the same semantic story not necessarily visually similar. Segmenting the video into scenes is more natural way of representing videos compared to the simple clustering based summarisation. In the scene detection task, spectral clustering approach by normalised cuts [9] is used to cluster the frames. In normalised cuts each frame of a video is treated as a node of the graph, and algebraic graph partitioning techniques are used to find clusters in the dataset. For each frame i we extract MPEG7 Colour layout descriptor [3] and store it in a feature vector f_i of the frame i. Similarity between two data points describes the relation between two frames. If Euclidean distance $\|f_i - f_j\|_2$ between two feature vectors is high, the frames similarity will be close to zero. On the other hand if the distance is small, the similarity will be close to one. The Similarity w_{ij} between two frames i and j withe feature vectors f_i and f_j is calculated using a gaussian function:

$$w_{ij} = e^{-\frac{\|f_i - f_j\|_2}{\sigma^2}} \quad (2)$$

Parameter σ serves as a scaling parameter which determine how fast will similarity measure decrease with increas-

ing distance between feature vectors. Similarity matrix $W_{n\times n} = [w_{ij}]$, where n is the number of frames, is created by calculating pairwise similarities between all frames. The matrix W created in such a way hold all information necessary to perform spectral clustering. Clusters are found using eigenvectors and eigenvalues of the similarity matrix. Clustering using eigenvectors have its origin in the problem of graph partitioning, where the goal is to find a cut in the graph that satisfies some predefined criteria. In the literature there exists a number of criteria that can be satisfied using eigenvectors of the similarity matrix [5, 6]. It can be shown that most of this criteria are similar or equivalent to each other. Let V be set of all frames of the original video, and let A and B be two clusters satisfying following conditions: $A \cap B = 0$ and $A \cup B = V$. Cut between two disjoint subsets A and B,$cut(A, B)$ of the set V is defined as $cut(A, B) = \sum_{i\in A, j\in B} w_{ij}$ and association $assoc(A, V)$of the subset A is defined as: $assoc(A, V) = \sum_{i\in A, j\in V} w_{ij}$. The clustering criterion $Ncut(A, B)$ that is optimised using eigenvectors of the similarity matrix W is defined as follows:

$$Ncut(A, B) = \frac{cut(A, B)}{assoc(A, V)} + \frac{cut(A, B)}{assoc(B, V)} \quad (3)$$

The clustering problem formulated in terms of cut and association can be seen as identifying groups that have a strong connection between members of the same cluster and weak connections between members of different clusters. Clustering that satisfies these conditions gives minimal value of $Ncut(A, B)$ over a set of all possible clustering results. Searching for the specific clustering that minimise $Ncut(A, B)$ is shown to be NP-hard problem in discrete domain[9]. Algebraic theory of graph spectra properties shows that minimising $Ncut(A, B)$ can be done in continuous domain using eigenvectors and eigenvalues of the similarity matrix W. Letting clustering indicators take continuous values instead of discrete, minimal value of $Ncut(A, B)$can be found as the second smallest eigenvalue of the similarity matrix W. Every entry $x^{(2)}(i)$ of the second eigenvector $x^{(2)}$ corresponds to one point i in the dataset, and its sign is used as a clustering indicator:

$$\text{if } x^{(2)}(i) > 0 \; i \in A \quad (4)$$

$$\text{if } x^{(2)}(i) < 0 \; i \in B \quad (5)$$

Starting from the set of all frames V, in first step $k = 1$, two clusters are found $V(1)^+$ and $V(1)^-$, $V(1)^+$ correspond to the positive eigenvector entries and $V(1)^-$ to the negative ones. In each of the clustering steps k, $Ncut(k)$ value is calculated using formula (3). If $Ncut(k)$ value is bigger then some predefined threshold $NCUT$, elements of $V(k)^+$ and $V(k)^-$ belong to a single cluster, and are left out from further clustering. Choice of $NCUT$ is done experimentally on a manually annotated training dataset. High $Ncut$ value indicates that the similarity between frames of different clusters is high, so it is most likely that they should stay in the same cluster. On the other hand, lower $Ncut$ values indicate that two clusters are separated more. Clustering process can be generalised as follows, for every step k set of frames that is clustered will be denoted by $V(k)$,cluster corresponding to positive eigenvector entries $V(k)^+$ and cluster corresponding to negative eigenvector entries $V(k)^-$. Clusters $V(k)^+$ and

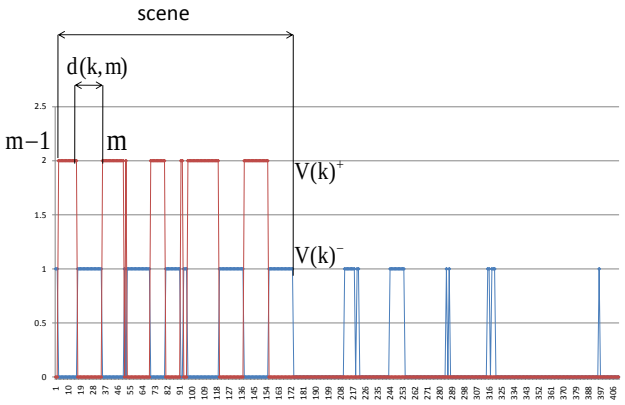

Figure 3: Scene detection example. Cluster indicators values are plotted over the set of frames. If two consecutive continuous segments m and $m-1$ of the same cluster are close enough they belong to the same scene. Distance between consecutive segments is analysed over time, until all scenes of both positive and negative clusters are found.

$V(k)^-$ will be further clustered in the steps 2^k and $2^k + 1$ respectively if $Ncut(k) < NCUT$. Every clustering step k for which $Ncut(k) < NCUT$ gives useful information for the problem of scene detection. In every step k each frame $i \in V$ can have one of three possible labels $l(i)$:

$$l(i) = 0 \text{ if } i \notin V(k) \quad (6)$$

$$l(i) = 1 \text{ if } i \in V(k)^+ \quad (7)$$

$$l(i) = 2 \text{ if } i \in V(k)^- \quad (8)$$

If we assume that we are in the clustering step k, with parent cluster $V(k)$, and child clusters $V(k)^+$ and $V(k)^-$. When clustering video segments, parent cluster $V(k)$ is not necessarily continuous in time, resulting in $V(k)^+$ and $V(k)^-$ being scattered over the time axis, see 3. Every cluster is then composed of a number of continuous segments. We will denote $m-th$ continuous segment of the cluster $V(k)^+$ $V(k, m)^+$ and $n-th$ continuous segment of the cluster $V(k)^-$ $V(k, n)^-$. Let $d(k, m)^+$ be the distance in time between $m-th$ and$(m-1)-th$ segment of the positive cluster, and $d(k, n)^-$ be the distance between $n-th$ and $(n-1)-th$ segment of the negative cluster. Scene boundary detection starts from the first frame of $V(k)$ on the time axis. We define scene as a time segment which contain segments with $d(k, m)^+ < Dseg$ where $Dseg$ is temporal threshold. It means that scene will be formed of temporally continuous segments with distance between every consecutive segment being smaller than $Dseg$. $Dseg$ is not set to fix value, but is dynamically determined on the run. Two consecutive segments $V(k, m)$ and $V(k, m-1)$ will be put in the same scene if distance between them $d(k, m)$ is smaller than a weigthed sum of lengths of the two segments:

$$d(k, m) < T * (length(V(k, m) + length(k, m-1))) \quad (9)$$

here the weight T is an experimentally determined constant.

2.3 Repetition detection

Repetition detection is based on the spectral graph theory, saying that eigenvector entries that belong to the same

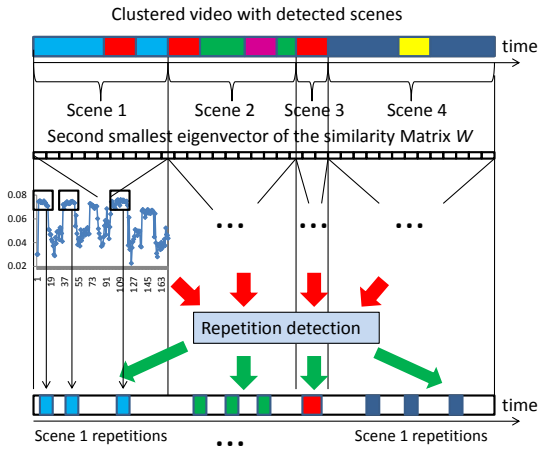

Figure 4: Overview of the repetition detection process. Different colours represent different clusters, with detected scenes. Entries of the second smallest eigenvector of the similarity matrix W , corresponding to the specific scene are used in the repetition detection process. Detecting similar patterns of the eigenvector entries is done for every scene until all repetitions are found.

cluster, in ideal case will be pair wise constant [5]. This practically means that all elements of the same cluster, in ideal case, will have same eigenvector entries which are different from entries belonging to other clusters in the dataset. It is also worth noting that stronger the similarity between two clusters is, stronger the similarity between eigenvector entries corresponding to these clusters will be. On the other hand difference between entries of dissimilar clusters will be high. We use these facts to analyse scenes by analysing the structure of its eigenvector entries. Let S be the scene with $n(S)$ frames, in which we are looking for the repetitions, with $i_{1(S)}$ being the first frame of the scene, and $i_{n(S)}$ last frame. We use second smallest eigenvector of the first clustering step, which is done on the whole video. The second smallest eigenvector of the first step is $x_i^{(2)}(1), i \in 1..n$ where n is a number of frames in the video. For each scene S analysis is done on a subset of $x_t^{(2)}(1)$ corresponding to the scene S:

$$i_{1(S)} \geq t \geq i_{n(S)} \tag{10}$$

Repetitions are found by searching for segments in the $x_t^{(2)}(1)$ that have similar value of eigenvector entries, that are separated in time, and have similar temporal structure. Condition that repetitive segments have to have similar eigenvector entries comes from the fact that these segments have similar if not the same visual layout. Separation is logical consequence of the repetition definition, as repetitive taking of the same scene mixed with recordings of technical preparations is found to be main feature of the BBC Rushes videos. Since interesting segments are taken according to a script which is usually fixed, similar duration of the repetitive segments showed to be useful condition for detecting these scenes. It means that the repetitive segments will have similar distribution of eigenvector entries in time. Assuming we are in the scene S, we analyse second smallest eigenvector entries corresponding to the frames belonging to the scene S, see (4). Let $x_{max}^{(2)}(S)$ be the maximal eigenvector value within the scene S and $x_{min}^{(2)}(S)$ be the minimal value. We define span of the scene S as a segment of eigenvector values between minimal and maximal value:

$$span(S) = x_{max}^{(2)}(S) - x_{min}^{(2)}(S) \tag{11}$$

We divide the span of the eigenvector entries $span(S)$ into r partitions P_i, $i = 1..r$,of length $span(S)/r$. Every partition P_i will have some number of points $N(P_i)$, whose eigenvector entries belong to P_i. By defining partitions P_i we captured information about eigenvector entries distribution over the $span(S)$. Looking into distribution of $N(P_i)$ over the set of partitions P_i we can detect such partitions P_i^m that have maximal values of $N(P_i)$ on the local level. These partitions correspond to peaks in the histogram of $N(P_i)$ over P_i and satisfy following conditions:

$$P_i^m > P_{i-1} \text{ and } P_i^m > P_{i+1} \tag{12}$$

Now, we treat each local peak as a centre of a possible repetition e in the space of eigenvector values. Possible repetition e is defined as a set of frames whose eigenvector entries fall into area around the local peaks P_e. P_i^m denote detected local peak in the histogram of $N(P_i)$, and $N(P_i^m)$ is a number of frames whose eigenvector entries belong to the partition P_i^m. Area $span(P_i^m)$ around the local peak P_i^m which defines the possible repetition e is defined as a set of partitions with number of frames being bigger then half the number of the frames $N(P_i^m)$ in the partition P_i^m. By the nature of repetitions, it is assumed that they are all of similar duration. Every possible repetition e_i of the same event E which have significantly different duration $dur(e_i)$ compared to other possible repetition of the same event is discarded. First we calculate mean duration of the repetition segments $dur(R_i \in E)_{mean}$ of the event E, and then we test if the current segment satisfies following conditions:

$$R_{max} * dur(R_i \in E)_{mean} \geq dur(R_i) \tag{13}$$

$$dur(R_i) \geq R_{min} * dur(R_i \in E)_{mean} \tag{14}$$

R_{max} and R_min are thresholds used to keep duration values have small standard deviations. This is done for all scenes and all events within the scene until all scenes are analysed and all repetitions are found.

2.4 Detection of Mid-level features

In order to create a summary, due to the nature of the rushes content, the detection of events constitutes an interesting tool. Our approach is based on the usage of some meaningful mid-level features that, according to our experience, are relevant for event detection. Specifically we have implemented: face detection and camera motion description which is a significant mid-level feature because directors usually use this characteristic to highlight some relevant events in the film and explicitly camera events make a part of semantic Ground Truth annotation. The developed algorithm tags frames for the different camera motion types.

We believe a human is one of the most recognizable object in a video, especially for a human summarizer. Moreover, most of the events in ground truth produced for evaluation contain a reference to a human posture or action. Our approach is a combination of two detectors: one of Viola and Jones, extended by Lienhart, from OpenCV [1], using Haar-like features, thus working on textural features, the other

one uses skin color appearance model trained on the faces detected by OpenCV. The first implementation of this approach was described in [2]. Compared to pure OpenCV, this method allows to increase the recall, without degrading precision.

For significant camera motion detection, we use the algorithm described in Kraemer et al [4]. First, we estimate the global camera motion, extracting only motion vectors from P-frames of MPEG compressed stream. Then we use a likelihood significance test of the camera parameters to classify specific camera motions. The algorithm of [4] allows for classification of camera motion as pure physical motions, such as "pan/travelling", "tilt", "zoom", "rotation" or complex motions.

In rushes content, during scene setup, there are often short, noisy camera motions. However, such unwanted motions are often complex and thus can be discarded by algorithm [4].

2.5 Merging and summary production

The merging system is developed to utilize the units mentioned above to produce summaries of the rushes videos which are *plausible to watch, informative, clear from redundancy and with the maximum coverage of the events*. The system functions in four steps to produce the final summary whose length is limited to maximum 2% of the input video:

1. Detection of redundant parts in the video.
2. Detection of separate scenes and repetitions in the scenes.
3. Aligning the repetitions and calculation of importance within the repetitions.
4. Creating the summary selecting relevant parts from each repetition.

Proposed system puts emphasis on the "watchability" issue. In that sense the system attempts to produce outputs that follow the story line of the content, that contain the important events and actions in the video and also, in the automatic editing process, it takes the measures to prevent any difficulty or annoyance for the viewer. Firstly, the system never displays segments shorter than 2 seconds. In addition to this, we do not use techniques like fast forwarding and frame-in-frame which limit the access of viewer to audio modality and boost artificially the content of events.

3. EXPERIMENTAL RESULTS

As can be seen in the detailed results in [7], the proposed method has performed quite well in many criteria. Especially taking into account the criteria *repeated segments* (3rd best result among all), *pleasant tempo and rhythm* (2nd best result among all) and *junk frame removal* (7th best result among all), the system can be referred to as capable of producing pleasant and informative summaries of the videos which are cleared from the redundancy. On the other hand, in the results we observe that the *inclusion of the events* (36th best result among all) is below the average and we investigated the reasons behind this. Firstly it is important to consider this important measure together with the other performance measures. Otherwise we observe that the best performing system in this ranking is the CMU's base system which mainly plays the videos in a fast-forward manner. This system, which includes almost all the events (as expected) shows poor performance in other criteria, i.e. it produces a "summary" which shows a piece of every event existing in the video in an unpleasant and difficult to understand manner. In general when we look at all the scores in different criteria and from different groups, we observe an inverse relation between the *inclusion of the events* and other three criteria mentioned above.

4. CONCLUSIONS

In this paper we introduced a system for creating summaries from unedited videos. The system is designed to create an output that is free from redundancy and repetition, and which highlights the most important events in the video, and does all of this in a generic manner. For the system the "watchability" was another issue that we put emphasis on: the created summaries should be composed of continuous segments which are long enough to let the users to understand the content in audio and visual modalities. This suggests that the possible techniques that might be employed for using the 2% time limit more efficiently (i.e. fast forwarding, frame-in-frame or multiple very short segments) are avoided in the summaries. In such a system where the number of video segments that can be added to the summary is limited, the success in chosing the right segments becomes critical. Our efforts in this direction has resulted in a better inclusion rate compared to the last years system, although the summary length has been halved. We plan to continue working on this issue by using the mid-level features in a more efficient way and proposing improvements on the scene clustering algoritms.

5. REFERENCES

[1] Opencv. http://opencvlibrary.sourceforge.net, 2007.

[2] A. Don, L. Carminati, and J. Benois-Pineau. Detection of visual dialog scenes in video content based on structural and semantic features. In *Proc. CBMI'05*, Létonie, 2005.

[3] E. Kasutani and A. Yamada. The mpeg-7 color layout descriptor: a compact image feature description of high-speed image/video segment retrieval. In *ICIP 2001*, Greece, 2001.

[4] P. Kraemer, J. Benois-Pineau, and M. Gràcia Pla. Indexing camera motion integrating knowledge of quality of the encoded video. In *Proc. SAMT'06*, 2006.

[5] M. Meila and J. Shi. Learning segmentation by random walks. In *NIPS*, 2000.

[6] M. Meila and J. Shi. A random walks view of spectral segmentation, 2001.

[7] P. Over, A. F. Smeaton, and G. Awad. The TRECVid 2008 BBC rushes summarization evaluation. In *TVS '08: Proceedings of the International Workshop on TRECVID Video Summarization*, pages 1–20, New York, NY, USA, 2008. ACM.

[8] X. Qian, G. Liu, and R. Su. Effective fades and flashlight detection based on accumulating histogram difference. *IEEE Transactions On Circuits And Systems For Video Technology*, 16(10), 2001.

[9] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), 2000.