SCENE SEGMENTATION USING TEMPORAL CLUSTERING FOR ACCESSING AND RE-USING BROADCAST VIDEO

Lorenzo Baraldi, Costantino Grana, Rita Cucchiara

Dipartimento di Ingegneria "Enzo Ferrari"
Università degli Studi di Modena e Reggio Emilia
Via Vivarelli 10, Modena MO 41225, Italy
name.surname@unimore.it

ABSTRACT

Scene detection is a fundamental tool for allowing effective video browsing and re-using. In this paper we present a model that automatically divides videos into coherent scenes, which is based on a novel combination of local image descriptors and temporal clustering techniques. Experiments are performed to demonstrate the effectiveness of our approach, by comparing our algorithm against two recent proposals for automatic scene segmentation. We also propose improved performance measures that aim to reduce the gap between numerical evaluation and expected results.

Index Terms— Scene detection, performance measures, temporal clustering.

1. INTRODUCTION

In recent years, the large availability of videos has led to great interest in fields different from simple entertainment or news broadcasting, such as education. Many modern approaches to education, like Massive Open Online Courses (MOOC), make use of videos as a tool for transmitting knowledge and educate a large number of potential students. This has also led to a strong interest in the re-use of video content coming from major broadcasting networks, which have been producing high quality edited videos for popular science purposes, such as documentaries and similar programs.

Unfortunately, re-using videos in ones own presentations or video aided lectures is not an easy task, and requires video editing skills and tools, on top of the difficulty of finding the parts of videos which effectively contain the specific content the instructor is interested in. Of course the basic unit for this task cannot be the single frame, and higher level groupings are needed, such as DVD chapters. The problem is that most of the on-line reusable content is not provided with editor defined video sub units. Scene detection has been recognized as a tool which effectively may help in this situation, going beyond frames and even beyond simple editing units, such as shots. The task is to identify coherent sequences (scenes) in videos, without any help from the editor or publisher.

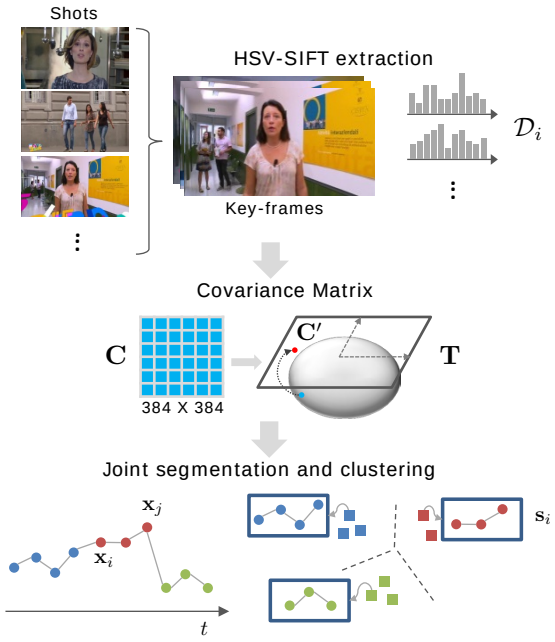

Fig. 1. Summary of our approach.

We present a novel proposal for scene segmentation, based on temporal clustering, that shows competitive results when compared to state-of-the-art methods. We also try to tackle the problem of evaluating scene detection techniques, by proposing an improved definition of the coverage/overflow measures [1], which solves frequently observed cases in which the numeric interpretation would be quite different from the expected results.

The rest of this paper is organized as follows: Section 2 presents a summary of the existing approaches to scene detection and temporal clutering. In Section 3 we describe our algorithm; in Section 4 we evaluate it, propose appropriate improvements to evaluation measures, and show a sample use case. We make our code, data and annotations publicly available.

2. RELATED WORK

Scene detection. Video decomposition techniques aim at partitioning a video into sequences, like shots or scenes, according to semantic or structural criteria. Shots are elementary structural segments that are defined as sequences of frames taken without interruption by a single camera, and shot detection is nowadays regarded as an essentially solved task [2].

Scenes, on the contrary, are often defined as series of temporally contiguous shots characterized by overlapping links that connect shots with similar content [3]. Therefore, the fundamental goal of scene detection algorithms is to identify semantically coherent shots that are temporally close to each other. Most of the existing works can be roughly categorized into two categories: *rule-based methods* and *graph-based methods*. They can rely on visual, audio, and textual features.

Rule-based approaches consider the way a scene is structured in professional movie production. Liu *et al.* [4], for example, propose a visual based probabilistic framework that imitates the authoring process and detects scenes by incorporating contextual dynamics and learning a scene model. In [5], shots are firstly clustered into symbolic groups using spectral clustering. Then, scene boundaries are detected by comparing successive non-overlapping windows of shot labels using a modified version of the Needleman-Wunsh algorithm, that considers the visual similarity of shot clusters and the frequency of sequential labels in the video. Of course, the drawback of this kind of methods is that they tend to fail when directors intentionally break film-editing rules, or when two adjacent scenes are similar and follow the same rules.

In graph-based methods, instead, shots are arranged in a graph representation and then clustered by partitioning the graph. The Shot Transition Graph (STG), proposed in [6], is one of the most used models in this category: here each node represents a shot and the edges between the shots are weighted by shot similarity. In [7], color and motion features are used to represent shot similarity, and the STG is then split into subgraphs by applying the normalized cuts for graph partitioning.

More recently, Sidiropoulos *et al.* [8] extended the Shot Transition Graph using multimodal low-level and high-level features. To this aim, multiple STGs are constructed, one for each kind of feature, and then a probabilistic merging process is used to combine their results. The features used include visual features, such as HSV histograms, outputs of visual concept detectors trained using the Bag of Words approach, and audio features, like background conditions classification results, speaker histogram, and model vectors constructed from the responses of a number of audio event detectors.

Temporal Clustering. Three of the most popular approaches to the clustering of temporal data are change-point detection, switching linear dynamical systems and evolutionary clustering. The goal of change-point detection [9] is to identify changes at unknown times and to estimate the location of changes in stochastic processes. Hierarchical Cluster Analysis, proposed in [10] for human motion segmentation, looks at the change points that minimize the error across several segments that belong to one of k clusters. This is an unsupervised hierarchical bottom-up framework that finds a partition of a given multidimensional time series into disjoint segments, and combines kernel k-means with the generalized dynamic time alignment kernel to cluster time series data.

Switching linear dynamical systems (SLDS) [11, 12], on the other hand, describe the dynamics of time series by switching several dynamical systems over time. The switching states then implicitly provide the segmentation of an input sequence. Evolutionary Clustering, proposed in [13], processes input data in an online fashion to produce a sequence of clusterings, by simultaneously optimizing two criteria: the clustering at any point in time should remain faithful to the current data, and clustering should not shift dramatically from one timestep to the next.

3. SCENE DETECTION AS A CLUSTERING TASK

To generate sets of semantically coherent scenes, we extract key-frames from each shot of the input video and describe them using HSV-SIFT descriptors, which are then summarized using their covariance matrix. Unlike previous scene detection algorithms that incorporated prior knowledge, such as production rules, to infer temporal dynamics, we jointly create a decomposition of the input video into contiguos segments, and assign each of them to one of k clusters, in a fully unsupervised way. A summary of our approach is presented in Figure 1.

3.1. Representing shots

The first step of our algorithm is shot description. Instead of describing each frame of a given shot, we choose to rely on keyframes. This step allows to reduce the computational requirements of the feature extraction phase, while still allowing an effective description of the shot.

To perform key-frame extraction, we firstly represent all the frames of a given shot with a 16-bin HSV normalized histogram. Then, we cluster them using the spectral clustering algorithm [14] with the Normalized Laplacian matrix. We employ the Euclidean distance as similarity measure and the maximum eigen-gap criterion to select the number of clusters, that therefore is equal to $\arg\max |\lambda_i - \lambda_{i-1}|$, where λ_i is the i-th eigenvalue of the Normalized Laplacian. The medoid of each group, defined as the frame of a cluster whose average similarity to all other frames of his group is maximal, is marked as a key-frame [5].

Having detected several key-frames for each shot, our goal is to encode the visual similarity of two shots. Therefore, we extract HSV-SIFT descriptors [15] using the Harris-

Laplace detector. These are obtained by computing SIFT descriptors over all three channels of the HSV representation of each key-frame, and since the SIFT descriptor is 128-dimensional, this gives 384 dimensions per descriptor, for a variable number of key-points.

To obtain a representation of a shot $\mathbf{x}_i$ with fixed size, we summarize the HSV-SIFT descriptors of its key-frames by computing their covariance matrix:

$$\mathbf{C} = \frac{1}{\#\mathcal{D}_i - 1} \sum_{\mathbf{d}_j \in \mathcal{D}_i} (\mathbf{d}_j - \bar{\mathbf{d}})(\mathbf{d}_j - \bar{\mathbf{d}})^T \tag{1}$$

where $\mathcal{D}_i$ is the set of HSV-SIFT of shot $\mathbf{x}_i$, and $\bar{\mathbf{d}}$ is their mean.

Since covariances belong to the Riemannian manifold of symmetric positive semi-definite matrices, Euclidean operations cannot be computed among them, and it would be difficult to define a similarity measure between two shots. Therefore, as in [16], we exploit a projection from the Riemannian manifold to an Euclidean tangent space, that is given by:

$$\mathbf{C}' = \log_{\mathbf{T}}(\mathbf{C}) = \mathbf{T}^{\frac{1}{2}} \log\left(\mathbf{T}^{-\frac{1}{2}} \mathbf{C} \mathbf{T}^{-\frac{1}{2}}\right) \mathbf{T}^{\frac{1}{2}} \tag{2}$$

where $\log(\cdot)$ is the matrix logarithm and the tangency point is denoted as matrix $\mathbf{T}$. The choice of the tangency point is arbitrary and, even if it could influence the distortion of the projection, from a computational point of view the best choice is the identity matrix, further simplifying the projection step which reduces to a simple matrix logarithm. Since $\mathbf{C}'$ is also symmetric, the final descriptor for a shot $\mathbf{x}_i$, $\psi(\mathbf{x}_i)$, is the upper triangular part of matrix $\mathbf{C}'$. Distances between shots can then be computed as simple Euclidean norms.

Having defined a suitable distance measure between shots, we can now cluster them to obtain the final scene boundaries, that must follow the temporal coherence and shot similarity principles.

3.2. Scene detection via temporal clustering

The task of clustering is to partition n datapoints into k disjoined clusters. Popular clustering algorithms used in scene detection, like k-means, kernel k-means and spectral clustering, minimize a variant of the following energy function:

$$J(\mathbf{G}) = \sum_{c=1}^{k} \sum_{i=1}^{n} g_{ci} \|\phi(\mathbf{x}_i) - \mathbf{m}_c\|^2 \tag{3}$$

where $\mathbf{G}$ is a binary indicator matrix such that $g_{ci} = 1$ if $\mathbf{x}_i$ belongs to cluster c, and zero otherwise, $\mathbf{m}_c$ is the center of class c, and $\phi(\cdot)$ is a mapping to a higher dimensional feature space. In traditional k-means, $\phi(\mathbf{x}_i) = \mathbf{x}_i$, and in spectral clustering $\mathbf{G}$ is relaxed to be continuous. Of course, the major limitation of these algorithms is that the temporal ordering of data is not taken into account.

Given a sequence of shots $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]$, we want to decompose it into a set $\mathbf{s}$ of m scenes, where each scene $\mathbf{s}_i$ is a set of contiguous shots. To this aim, we exploit a variant of Eq. 3 that considers sequences instead of points [10]:

$$J(\mathbf{G}, \mathbf{s}) = \sum_{c=1}^{k} \sum_{i=1}^{m} g_{ci} \|\phi(\mathbf{s}_i) - \mathbf{m}_c\|^2 \tag{4}$$

where $\mathbf{s}_i$ is a scene that begins at shot $\mathbf{x}_{s_i}$ and ends at shot $\mathbf{x}_{s_{i+1}}$, and $\|\phi(\mathbf{s}_i) - \mathbf{m}_c\|^2$ is the squared distance between the i-th scene and the center of class c.

Minimizing $J(\mathbf{G}, \mathbf{s})$ means finding a decomposition of $\mathbf{X}$ into m scenes (where m is not known a priori), and to assign each of them to one of k clusters, with the objective of maximizing intra-class similarities. The key insight of our method, therefore, is that a video can be divided into scenes by finding the partition that fits this clustering problem in the best way. This is consistent with the way a video is structured, where contiguous scenes often have different visual content, but similar visual appearances can be found in distant scenes.

The distance $\|\phi(\mathbf{s}_i) - \mathbf{m}_c\|^2$ can be implicitly computed without knowing $\mathbf{m}_c$ [17]:

$$\|\phi(\mathbf{s}_i) - \mathbf{m}_c\|^2 = \phi(\mathbf{s}_i)^T \phi(\mathbf{s}_i) - \frac{2}{n_c} \sum_{j=1}^{m} g_{cj} \phi(\mathbf{s}_i)^T \phi(\mathbf{s}_j) + \frac{1}{n_c^2} \sum_{j_1, j_2 = 1}^{m} g_{cj_1} g_{cj_2} \phi(\mathbf{s}_{j_1})^T \phi(\mathbf{s}_{j_2}) \tag{5}$$

where n_c is the number of segments that belong to class c.

The dot product of the mapping $\phi(\cdot)$, on the other hand, determines the similarity between two scenes, possibly of different lengths. We define it as the average similarity between the shots belonging to the two scenes:

$$\phi(\mathbf{s}_i)^T \phi(\mathbf{s}_j) = \frac{\sum_{h=s_i}^{s_{i+1}} \sum_{k=s_j}^{s_{j+1}} \kappa_{hk}}{(s_{i+1} - s_i + 1)(s_{j+1} - s_j + 1)} \tag{6}$$

where κ_{ij} is the similarity of shots $\mathbf{x}_i$ and $\mathbf{x}_j$, that we compute applying a Gaussian kernel to feature vectors $\psi(\mathbf{x}_i)$ and $\psi(\mathbf{x}_j)$:

$$\kappa_{ij} = \exp\left(-\frac{\|\psi(\mathbf{x}_i) - \psi(\mathbf{x}_j)\|^2}{2\sigma^2}\right). \tag{7}$$

The number of clusters, k, is selected using again the maximum eigen-gap criterion on the Normalized Laplacian of matrix κ_{ij}.

Summarizing, we turn distances between feature vectors into similarities by means of a Gaussian kernel. Then, shots similarities are averaged to compute scene similarities, and shots boundaries are obtained minimizing Eq. 4. Since optimizing over $\mathbf{G}$ and $\mathbf{s}$ is NP-hard, we employ the coordinate-descent scheme used in [10], that alternates between computing $\mathbf{s}$ using dynamic programming and $\mathbf{G}$ with a winner-take-all strategy.

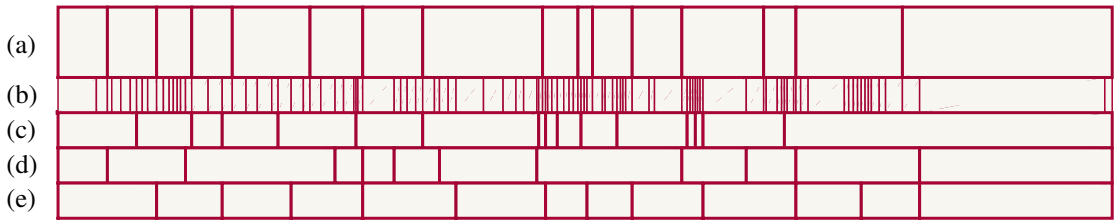

Fig. 2. Scene segmentation results on a frame sequence from our dataset. Row (a) shows the ground truth segmentation, (b) the individual shots boundaries, row (c) shows the results of [5], (d) those of [8] and (e) the results of our method.

4. EXPERIMENTS

We firstly describe the measures used to evaluate scene segmentation techniques, then assess the effectiveness of our approach by comparing it against two recent methods. We also address two drawbacks of the existing measures.

4.1. Performance measures

We adopt the Coverage, Overflow and F-Score measures, proposed in [1], to evaluate our results. Coverage $\mathcal{C}$ measures the quantity of shots belonging to the same scene correctly grouped together, while Overflow $\mathcal{O}$ evaluates to what extent shots not belonging to the same scene are erroneously grouped together. Formally, given the set of automatically detected scenes $\mathbf{s} = [\mathbf{s}_1, \mathbf{s}_2, ..., \mathbf{s}_m]$, and the ground truth $\tilde{\mathbf{s}} = [\tilde{\mathbf{s}}_1, \tilde{\mathbf{s}}_2, ..., \tilde{\mathbf{s}}_n]$, where each element of $\mathbf{s}$ and $\tilde{\mathbf{s}}$ is a set of shot indexes, the coverage $\mathcal{C}_t$ of scene $\tilde{\mathbf{s}}_t$ is proportional to the longest overlap between $\mathbf{s}_i$ and $\tilde{\mathbf{s}}_t$:

$$\mathcal{C}_t = \frac{\max_{i=1...,m} \#(\mathbf{s}_i \cap \tilde{\mathbf{s}}_t)}{\#(\tilde{\mathbf{s}}_t)} \tag{8}$$

where $\#(\mathbf{s}_i)$ is the number of shots in scene $\mathbf{s}_i$. The overflow of a scene $\tilde{\mathbf{s}}_t$, $\mathcal{O}_t$, is the amount of overlap of every $\mathbf{s}_i$ corresponding to $\tilde{\mathbf{s}}_t$ with the two surrounding scenes $\tilde{\mathbf{s}}_{t-1}$ and $\tilde{\mathbf{s}}_{t+1}$:

$$\mathcal{O}_t = \frac{\sum_{i=1}^{m} \#(\mathbf{s}_i \setminus \tilde{\mathbf{s}}_t) \cdot \min(1, \#(\mathbf{s}_i \cap \tilde{\mathbf{s}}_t))}{\#(\tilde{\mathbf{s}}_{t-1}) + \#(\tilde{\mathbf{s}}_{t+1})} \tag{9}$$

The computed per-scene measures can then be aggregated into values for an entire video as follows:

$$\mathcal{C} = \sum_{t=1}^{n} \mathcal{C}_t \cdot \frac{\#(\tilde{\mathbf{s}}_t)}{\sum \#(\tilde{\mathbf{s}}_i)}, \quad \mathcal{O} = \sum_{t=1}^{n} \mathcal{O}_t \cdot \frac{\#(\tilde{\mathbf{s}}_t)}{\sum \#(\tilde{\mathbf{s}}_i)} \tag{10}$$

finally, an F-Score metric can be defined to combine Coverage and Overflow in a single measure, by taking the harmonic mean of $\mathcal{C}$ and $1 - \mathcal{O}$.

We identify two inconveniences of these measures, hence we propose an improved definition. The first one is that, being computed at the shot level, an error on a short shot is given the same importance of an error on a very long shot. On the other hand, we propose to normalize $\mathcal{O}_t$ with respect to the length of $\tilde{\mathbf{s}}_t$ instead of that of $\tilde{\mathbf{s}}_{t-1}$ and $\tilde{\mathbf{s}}_{t+1}$, since we believe that the amount of error due to overflowing should be related to the current scene length, instead of its two neighbors. As an example, consider a ground truth segmentation where a long scene is surrounded by two short scenes: if the detected scene is the union of all three, the actual amount of overflow for the middle scene is quite small, while the usage of the original measures would result in a 100% overflow.

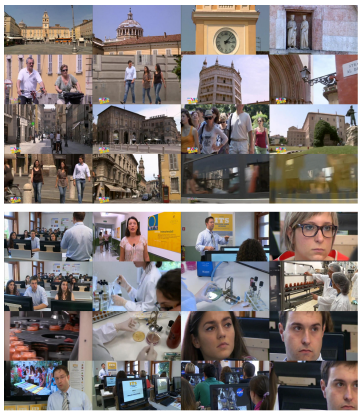

Fig. 4. Two consecutive scenes from the RAI dataset.

Therefore, we propose the Coverage* and Overflow* measures, where the cardinality operator $\#$ is replaced with the number of frames of a scene, $l(\mathbf{s}_i)$, and overflow is redefined as follows:

$$\mathcal{O}^*_t = \min\left(1, \frac{\sum_{i=1}^{m} l(\mathbf{s}_i \setminus \tilde{\mathbf{s}}_t) \cdot \min(1, l(\mathbf{s}_i \cap \tilde{\mathbf{s}}_t))}{l(\tilde{\mathbf{s}}_t)}\right) \tag{11}$$

Note that we limit the amount of overflow to one. The corresponding $\mathcal{C}^*$ and $\mathcal{O}^*$ for an entire video can be obtained in the same way of Eq. 10, using the newly defined cardinality operator.

4.2. Evaluation

We evaluate our approach on a collection of ten randomly selected broadcasting videos from the Rai Scuola video archive[1], mainly documentaries and talk shows (see Figure

[1] http://www.scuola.rai.it

Fig. 3. Effective video browsing using our algorithm. Users can visualize a summary of the content by means of the extracted scenes.

4). Shots have been obtained running a state-of-the-art shot detector [18] and manually grouped into scenes by a set of human experts to define the ground truth. Our dataset and the corresponding annotations are available for download at `http://imagelab.ing.unimore.it`.

We compare our method against the multimodal approach presented in [8] and that of [5]. We use the executable of [8] provided by the authors[2] and reimplement the method in [5]. Parameters of [5] were selected to maximize the performance on our dataset.

Figure 2 shows the segmentations of the compared methods on a frame sequence from our dataset. To visualize the effect of our definitions of coverage, consider, for example, the last scene of the ground truth (represented in the first row), and the segmentation of [8] (depicted in the fourth row). According to the standard definition of Coverage, this scene has 0.67 coverage. With our measures, the computed coverage is 0.92, a much more realistic numerical result.

The overall results on the dataset are shown in Table 1, using Vendrig's measures (Coverage, Overflow and F-Score), and in Table 2, using our improved definitions (Score*, Overflow* and F-Score*). As it can be seen, our method achieves competitive results, using both measures, when compared to recent and state-of-the-art methods like [8], and features a considerably reduced overflow. When shot duration is taken into account, using our measures, the improvement of our method over the others is even more manifest.

4.3. Use cases

Detected scenes, finally, can be used as an input for video browsing or re-using software. As an example, we built a web-based browsing interface for broadcasting videos (see Figure 3) where users can visualize a summary of the content by means of the extracted scenes. Scenes are represented with key-frames in a time-line fashion, and when a particular scene is selected, all its shots are unfolded. To ease the browsing even more, most frequent words, obtained from the transcript of the audio, are reported under each scene. Users can jump from one part of the video to another by clicking on the corresponding scene or shot.

[2] `http://mklab.iti.gr/project/video-shot-segm`

5. CONCLUSIONS

We introduced a model to perform scene detection of broadcasting videos. Our approach relies on the description of key-frames with HSV-SIFT descriptors, summarized using the projection of their covariance matrix in a Euclidean space, and on a temporal clustering approach that jointly segments and clusters the input video. Experimental results showed that our approach outperforms the existing state of the art. Finally, we addressed the problem of evaluating scene detection results, and proposed ways to improve the existing performance measures.

6. REFERENCES

[1] J. Vendrig and M. Worring, "Systematic evaluation of logical story unit segmentation," *IEEE Trans. Multimedia*, vol. 4, no. 4, pp. 492–499, 2002.

[2] B. T. Truong and S. Venkatesh, "Video abstraction: A systematic review and classification," *ACM Trans. Multimedia Comput. Commun. and Appl.*, vol. 3, no. 1, pp. 3, 2007.

[3] A. Hanjalic, R. L. Lagendijk, and J. Biemond, "Automated high-level movie segmentation for advanced video-retrieval systems," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 4, pp. 580–588, 1999.

[4] C. Liu, D. Wang, J. Zhu, and B. Zhang, "Learning a Contextual Multi-Thread Model for Movie/TV Scene Segmentation," *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 884–897, 2013.

[5] V. T. Chasanis, C. Likas, and N. P. Galatsanos, "Scene detection in videos using shot clustering and sequence alignment," *IEEE Trans. Multimedia*, vol. 11, no. 1, pp. 89–100, 2009.

[6] M. M. Yeung, B.-L. Yeo, W. H. Wolf, and B. Liu, "Video browsing using clustering and scene transitions on compressed

Table 1. Performance comparison on the RAI dataset using the Coverage, Overflow and F-Score measures.

Video	Chasanis *et al.* [5]			Sidiropoulos *et al.* [8]			Our method		
	F-Score	$\mathcal{C}$	$\mathcal{O}$	F-Score	$\mathcal{C}$	$\mathcal{O}$	F-Score	$\mathcal{C}$	$\mathcal{O}$
V_1	0.70	0.64	0.24	**0.72**	0.84	0.37	0.63	0.56	0.29
V_2	0.36	0.80	0.77	**0.59**	0.85	0.55	0.50	0.76	0.62
V_3	0.58	0.73	0.52	0.58	0.90	0.57	**0.66**	0.82	0.45
V_4	**0.50**	0.65	0.60	0.33	0.94	0.80	0.48	0.71	0.63
V_5	0.25	0.93	0.86	**0.66**	0.76	0.41	0.62	0.72	0.45
V_6	0.18	0.89	0.90	**0.71**	0.77	0.34	0.47	0.35	0.30
V_7	0.37	0.70	0.75	0.51	0.78	0.62	**0.59**	0.54	0.35
V_8	0.62	0.57	0.32	0.45	0.88	0.70	**0.63**	0.57	0.30
V_9	0.27	0.87	0.84	0.43	0.92	0.72	**0.63**	0.67	0.40
V_{10}	0.54	0.91	0.62	0.44	0.94	0.71	**0.58**	0.66	0.48
Average	0.44	0.77	0.64	0.54	0.86	0.58	**0.58**	0.63	0.43

Table 2. Performance comparison on the RAI dataset using the Coverage*, Overflow* and F-Score* measures.

Video	Chasanis *et al.* [5]			Sidiropoulos *et al.* [8]			Our method		
	F-Score*	$\mathcal{C}^*$	$\mathcal{O}^*$	F-Score*	$\mathcal{C}^*$	$\mathcal{O}^*$	F-Score*	$\mathcal{C}^*$	$\mathcal{O}^*$
V_1	**0.70**	0.65	0.24	**0.70**	0.63	0.20	0.63	0.56	0.29
V_2	0.60	0.91	0.55	**0.61**	0.73	0.47	0.50	0.76	0.62
V_3	0.51	0.87	0.64	0.51	0.89	0.64	**0.66**	0.82	0.45
V_4	**0.54**	0.70	0.56	0.22	0.95	0.88	0.48	0.71	0.63
V_5	0.34	0.92	0.79	0.57	0.66	0.50	**0.62**	0.72	0.45
V_6	0.20	0.89	0.88	**0.74**	0.72	0.24	0.47	0.35	0.30
V_7	0.37	0.75	0.76	0.56	0.69	0.53	**0.59**	0.54	0.35
V_8	0.59	0.65	0.47	0.15	0.89	0.92	**0.63**	0.57	0.30
V_9	0.07	0.83	0.96	0.15	0.94	0.92	**0.63**	0.67	0.40
V_{10}	0.50	0.93	0.66	0.11	0.93	0.94	**0.58**	0.66	0.48
Average	0.44	0.81	0.65	0.43	0.80	0.63	**0.58**	0.63	0.43

sequences," in *IS&T/SPIE Symp. Electron. Imaging*, 1995, pp. 399–413.

[7] Z. Rasheed and M. Shah, "Detection and representation of scenes in videos," *IEEE Trans. Multimedia*, vol. 7, no. 6, pp. 1097–1105, 2005.

[8] P. Sidiropoulos, V. Mezaris, I. Kompatsiaris, H. Meinedo, M. Bugalho, and I. Trancoso, "Temporal video segmentation to scenes using high-level audiovisual features," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 8, pp. 1163–1177, 2011.

[9] Z. Harchaoui, E. Moulines, and F. R. Bach, "Kernel change-point analysis," in *Adv. Neural Inf. Process. Syst.*, 2009, pp. 609–616.

[10] F. Zhou, F. De la Torre, and J. K. Hodgins, "Hierarchical aligned cluster analysis for temporal clustering of human motion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 3, pp. 582–596, 2013.

[11] J. M. Wang, D. J. Fleet, and A. Hertzmann, "Gaussian process dynamical models for human motion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 2, pp. 283–298, 2008.

[12] A. S. Willsky, E. B. Sudderth, M. I. Jordan, and E. B. Fox, "Nonparametric Bayesian learning of switching linear dynamical systems," in *Adv. Neural Inf. Process. Syst.*, 2009, pp. 457–464.

[13] D. Chakrabarti, R. Kumar, and A. Tomkins, "Evolutionary clustering," in *ACM SIGKDD Int. Conf. Knowl. Discov. and Data Min.*, 2006, pp. 554–560.

[14] A. Y. Ng, M. I. Jordan, Y. Weiss, et al., "On spectral clustering: Analysis and an algorithm," *Advances in neural information processing systems*, vol. 2, pp. 849–856, 2002.

[15] A. Bosch, A. Zisserman, and X. Muoz, "Scene classification using a hybrid generative/discriminative approach," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 4, pp. 712–727, 2008.

[16] C. Grana, G. Serra, M. Manfredi, and R. Cucchiara, "Image Classification with Multivariate Gaussian Descriptors," in *LNCS*, Sept. 2013, vol. 8157, pp. 111–120.

[17] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *ACM SIGKDD Int. Conf. Knowl. Discov. and Data Min.*, 2004, pp. 551–556.

[18] E. Apostolidis and V. Mezaris, "Fast Shot Segmentation Combining Global and Local Visual Descriptors," in *IEEE Int. Conf. Acoustics, Speech and Signal Process.*, 2014, pp. 6583–6587.