On Using Clustering Algorithms to Produce Video Abstracts for the Web Scenario

Marco Furini
Dip. di Informatica
Via Bellini 25/G
Alessandria, Italy
furini@mfn.unipmn.it

Filippo Geraci
IIT-CNR
Via Moruzzi 1
Pisa, Italy
filippo.geraci@iit.cnr.it

Manuela Montangero
Dip. Ing. Informazione
Via Vignolese 905/b
Modena, Italy
montangero.manuela@unimo.it

Marco Pellegrini
IIT-CNR
Via Moruzzi 1
Pisa, Italy
marco.pellegrini@iit.cnr.it

***Abstract*—A tool to provide an idea of the content of a given video is becoming a need in the current Web scenario, where the presence of videos is increasing day after day. Dynamic summarization techniques can be used to this aim as they set up a *video abstract*, by selecting and sequencing short video clips extracted from the original video. Needless to say, the selection process is critical. In this paper we focus our attention on clustering algorithms to provide such selection and we investigate the effects of their employment in the web scenario. Clustering algorithms are very effecting in producing static video summary, but few works consider them for video abstract production. For this reason, we set up an experimental scenario where we investigate their performance considering different categories of video, different abstract lengths and different low-level video analysis. Results show that clustering techniques can be useful only for some categories of videos and only if the selection process is based on video scene characteristics. Furthermore, the investigation also shows that to provide a customized service (user can freely decide the abstract time length), only fast clustering algorithm should be used.**

I. Introduction

The availability of digital video contents over the Net is increasing due to the advances in networking and multimedia technologies and to the wide use of multimedia applications: videos can be downloaded and played out from almost everywhere using many different devices (e.g., cellphones, palms, laptops) and networking technologies (e.g., EDGE, UMTS, HSDPA, Wi-Fi).

To handle the enormous quantity of video contents, many proposals have been done for indexing, retrieving and categorizing digital video contents. In this paper we focus on *summarization techniques*, which aim at providing a concise representation of a video content. The motivation behind these techniques is to provide a tool able to give an idea of the video content, without watching it entirely, so a user can decide whether to download/watch the entire video or not. In essence, these techniques are well suited for browsing videos.

Two different approaches are usually followed for producing a concise video representation: one is the production of *static video summary*, which is a collection of still video frames extracted from the original video, and the other is the *dynamic video skimming* (or *video abstract*), which is a collection of short video clips. It is worth mentioning that, in both cases, the output is obtained by analyzing some low-level characteristics of the video stream (e.g., colors, brightness, speech, etc.) in order to find out possible aural/visual clues that would allow a high-level semantics video understanding.

In this paper we focus on dynamic summarization techniques that produce *video abstract*. In literature, different techniques have been proposed (see e.g.[1]), but they usually use computationally expensive and very time-consuming algorithms. As a result, this requires video web sites (e.g., The Open Video Project) to pre-compute video abstracts and to present them *as-is*, without offering users customization. In fact, it is unreasonable to think of a user waiting idle for a latency time comparable to the duration of the original video to get an abstract. This is a burden, as customization is becoming more and more important in the current Web scenario, where users have different resources and/or needs.

For these reasons, we focus on clustering techniques. These are commonly used to produce static still image video summary, with considerable benefits in terms of both effectiveness and production time [1], [2], but there are relative few works that address clustering techniques in video skimming.

The contribution of this paper is to investigate the benefits of using clustering techniques to produce video abstracts for the Web scenario. To this aim, we set-up an experimental scenario considering different clustering algorithms (the well known k-means and the fast eFPF), different categories, both in color and in motion terms, of videos (cartoon, tv-show and tv-news), different abstract lengths (2 minutes and 4 minutes), different low-level video analysis (frame-based with HSV color distribution of every frame and scene-based with HSV color distribution of every scene). Note that, in this paper, a video scene is a sequence of consecutive video frames that begins and ends with an abrupt video transition and a silence.

Using a ground truth evaluation, we show that clustering algorithms are useful only for some categories of videos and that to provide a customize service, the only way to do it is to use fast clustering algorithms like eFPF.

The remainder of this paper is organized as follows. In Section II we briefly present related work in the area of dynamic video summarization; The experimental set-up is presented in Section III, whereas performance evaluation is shown in Section IV. Conclusions are drawn in Section V.

II. Related Work

Different approaches have addressed the problem of video skimming [1], [2]. In general one can classify the proposed methods according to several categorical axis: the data domain (generic, news, home videos, etc.), the features used (visual, audio, motion, etc.), the intent (personalization, highlights, information coverage), the duration (defined a priori, a posteriori, or user-defined). Here we focus on techniques for generic videos, using only visual and audio features.

Sampling based methods. Authors in [3] propose a frame sampling technique where the sampling rate is proportional to a local notion of ”visual activity”. Such sampling based methods produce quickly shorter videos but suffer from uneven visual quality, and visual discomfort, and are usually not suitable for dealing with the associated audio trace.

Frame-based methods. In principle any method for selecting a static storyboard can be turned into a dynamic one by selecting and concatenating the shots/scenes containing the key-frames of the storyboard. For example, the method in [4] works at frame level using a partitional clustering method applied to all the video frames. The optimal number of clusters is determined via a cluster-validity analysis and key frames are selected as centroids of the clusters. Video shots, to which key frames belong, are concatenated to form the abstract sequence. In this approach the dynamic efficiency/quality depends directly from those of the static case.

Scene-Based methods. Authors in [5] formulate the problem of producing a video abstract as a graph partitioning problem over a graph where each node is associated to a shot, and an edge is set up if the similarity of two shots is higher than a pre-defined threshold. The corresponding incidence matrix is clustered using an iterative block ordering method. One can notice that setting up the graph is already quadratic in the number of shots, thus this method is likely unsuitable for on-the-fly processing of long videos. The notion of a *scene transition graph* is also used in [6]: a complete link hierarchical agglomerative clustering is used together with a time-weighted distance metric, introducing an overhead that is unsuitable for on-the-fly computations. Authors in [7] use a *scene transition graph* that is clustered via spectral matrix decomposition. In this case, the mechanism needs 23 minutes to analyze a 69 minutes video. Once again, the approach is unsuitable for on-the-fly operations

As mentioned, customization and production of on-the-fly abstracts are important properties. Hence, an analysis of the benefits introduced by clustering techniques is necessary. In fact, many different methods are surveyed in [2], but none claims to have on-the-fly and user-oriented characteristics, needed in web video browsing applications.

III. Our setting

In this section we explain in details what is the setting in which our experiments have been carried out.

The quality of the abstract produced by these experiments and the time needed to produce them will be discussed in the next section.

A. Video Segmentation

A video abstract is composed of a sequence of the most important segments of the original video and hence the abstract quality also depends on the video segmentation process. We observe that it is of crucial importance for the process to consider both audio and video features. In fact, if video is divided according only to visual information (for instance by splitting the video where there is a video cut, which happens when two consecutive video frames have few parts in common), it is likely that a video segment has an incomplete audio.

To avoid this, we consider the approach presented in [8], which takes into account both audio and video features. In such mechanism, when a video cut is detected, audio energy at video transition is checked: if there is silence, the transition is considered to be the end of a segment, otherwise it is assumed that the segment is not over. The result is that, when combining segments obtained in this way, we get a fluid, understandable abstract in which audio is completely intelligible and not interrupted.

B. Clustering

In literature, among the few clustering techniques designed to produce video abstracts, some start by clustering frames and successively recovering scenes from the selected frames; others cluster scenes and then select one scene per cluster. Although the definition of a scene might vary from mechanism to mechanism, the final output is always produced by sequencing the selected video scenes.

In this paper we analyze both approaches (i.e., frame-based and scene-based selection), and the scene is a segment of video that begins and ends with a silence and with a video cut. Note that, when talking about the scene to which the frame belongs to, we mean the only scene in which the frame appears.

To represent frames/scenes we consider the HSV color distribution, and we use the 256 bin colors of the MPEG7 generic color histogram description [9]. The HSV color histogram is stored in a matrix for clustering purpose. In particular, when clustering video frame, we extract and store in a matrix M_{HSV} the 256 bin colors histogram in the HSV color space of all frames. Conversely, according to the approach used in [5], when clustering video scenes, for each scene, we compute a 256-dimension vector, that is the average of the HSV vectors of the frames in the scene. The vector is then stored in a matrix M_{avgHSV} for clustering purpose.

Finally, given a video and a desired abstract lenght T, we produce two abstracts, one obtained by clustering vectors in M_{HSV} (corresponding to frames), and one by clustering vectors in M_{avgHSV} (corresponding to scenes).

We test two clustering algorithms with different characteristics: one is the well-known k-means [10], widely used and considered in literature, the other is an enhanced version of the Furtherest-Point-First algorithm (eFPF) [11], which has been considered for its speed-up processing. Both algorithms require to know the number of clusters k to make in advance, and output clusters each provided with a representative element. To measure frame (scene) similarity, we consider the Generalized Jacard Distance [12] , as this metric has shown to perform well for HSV vectors [13].

We do not take into consideration other well known clustering algorithm (e.g., Hierarchical clustering) because these are computationally slower than k-means and do not apply to our scenario of on-the-fly customized video abstract production.

We produce abstracts in the following way:
Abstract by frames: given T in seconds and fps, the frame per seconds of the original video, we compute the number of frames that should be in the abstract as $\#SF = T \cdot fps$. We estimate the number of scenes $\#SS$ in the abstract with a value such that the ratio between the number of frames and the number of scenes in the original video and in the abstract is the same. Chosen an arbitrary small integer constant c, we cluster vectors in M_{HSV} in $k = \#SS \cdot c$ clusters, obtaining k representative frames. For each frame we determine the scene to which the frame belongs to and we increment by one a counter associated to the scene (initially all counters are set to zero). Starting from the scene with higher counter, and considering scenes in counter decreasing order, we select the scenes to be in the abstract until the total length of the selected scenes reaches time T. Observe that c is used to produce a number of clusters higher than the number of scenes that will compose the abstract, generating a significant ranking of the scenes by means of the counters.

Abstract by scenes: we cluster vectors of the matrix M_{avgHSV}. The process depends on the clustering algorithm. The eFPF algorithm generates a new permanent center of a new cluster at each iteration, giving a way to rank centers, *i.e.*, a selection order. Note that items that are clustered represents scenes and that the order in which they are considered for the clustering process is completely independent from the order in which the scenes appear in the original video. Hence, at the time in which centers are created, the corresponding scenes are selected and inserted in the abstract. The process continues until the the total abstract length reaches time T.

For the k-means algorithm we proceed with a brute force approach to determine the k clusters necessary to produce an abstract of length T (note that in this paper we are not discussing the best way to choose k).

In both cases, the selected scenes are ordered according to the time in which they appear in the original video and the sequence that has been obtained is presented as the abstract.

C. Random

To evaluate if clustering might help in producing video abstracts, we also compute abstracts by choosing frames and scenes at random. If there is no significant difference between these abstracts and those produced using clustering, the only natural conclusion is that there is no point in spending time and resources with clustering. To produce the randomized abstracts we proceed as follows:
Abstract by frames: choose frames at random and select their corresponding scenes until the total length of the selected scenes reaches T.
Abstract by scenes: randomly chose scenes until the total length of the selected scenes reaches T.

In both cases, we reorder the scenes according to the time in which they appear in the original video and we output the resulting abstract.

IV. Experimental Assessment

To evaluate the benefits of using clustering algorithms to produce video abstract in the web scenario, we set up an experimental scenario investigating the performance of clustering algorithms against a random approach. In order to have a wide test bed, we consider three different categories of videos: cartoons, TV-Shows and TV-News. Movies have not been considered since video abstracts reveal too much contents (e.g, the end of the movie), and hence ad-hoc techniques to produce *highlights* are more suited for this category. In particular, we considered four different 40 minutes long TV-Shows (*Charmed, Roswell*, *Dark Angel* and *Lost*); two different 20 minutes long cartoons (*The Simpsons* and *Futurama*, two episodes each) and two different sources of 15 minutes long TV-News (*Sky TV* and *BBC* sources, two videos each).

First, each video is virtually divided into video scenes using the approach in [8]. The average number of video scenes for each categories of video is: 67 for TV-News, 534 for long TV-Show and 322 for cartoons.

Afterward, a HSV color analysis produces, for each video, two different tables (representing M_{HSV} and M_{avgHSV}), which are the input of the clustering algorithms.

We produce two sets of video abstract: one contains 2 minutes long abstracts and the other contains 4 minutes long abstracts. The length has been chosen as thought of reasonable for a video abstract. For each set, we compute two different video abstracts for each video: one is frame-based and the other is scene-based. This means that, considering also randomly produced abstracts, for each given video we have 6 different video abstracts.

The goal of this experimental scenario is to investigate the quality of the produced video abstracts and also the production time in order to potentially offer a customized service.

A. Quality evaluation using a Ground-truth

The evaluation of a video abstract is a difficult task to set up: objective metrics like PSNR cannot be applied to videos of different length, hence, user evaluation has to be considered [2]. However, since the presence of long videos may discourage a truthful evaluation, a more effective method is to compute a *ground-truth* abstract of each video (a manual built abstract containing the most important video scenes), and compare the produced abstracts with it. We proceed as follows: (a) Given the original video, we manually divide it into *Super-Scenes* (s-scene) each having a self contained meaning (*e.g.*, dialog between two characters in the kitchen; trip from here to there by car and so on). A s-scene might contain more than one scene (as defined in this paper) or can be a fraction of a scene (e.g., two different actions taking place during one

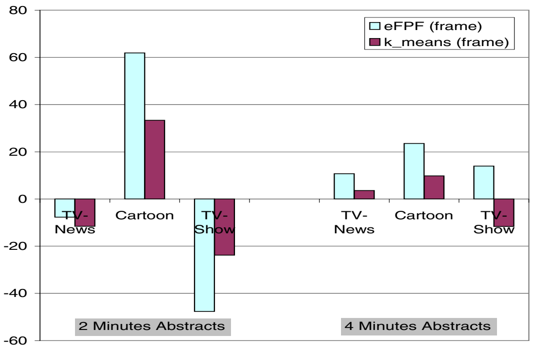

Fig. 1. Ground Truth Evaluation: Comparison of abstracts produced using a frame analysis. Results are normalized to the random abstract scores (positive values mean better results than random, negative worst).

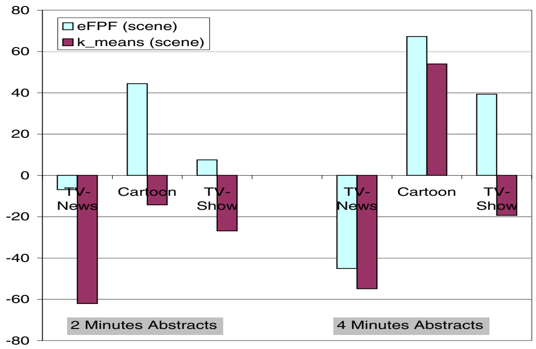

Fig. 2. Ground Truth Evaluation: Comparison of abstracts produced using a scene analysis. Results are normalized to the random abstract scores (positive values mean better results than random, negative worst).

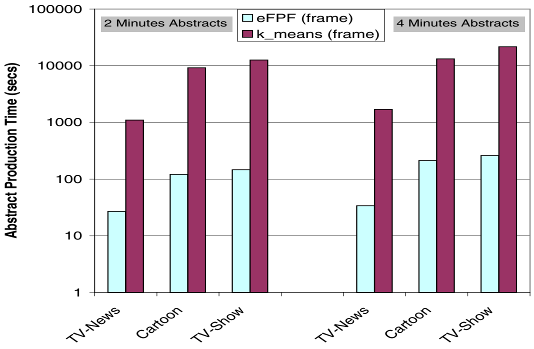

Fig. 3. Production Time: Comparison of abstracts produced using a frame analysis. Results are presented on a logarithmic scale.

single background piece of music). (b) We ask a set of 10 users (undergraduate and Ph.D. students, young researchers and non-academic) to score each s-scene with a value from zero to five (0 = un-influent, 5 = fundamental). Then, to each s-scene we associate a score that is computed as the average of the scores given by the users. (c) Given an abstract, each scene is scored with the score given to the s-scene it belongs to or with the sum of the scores given to the scenes it is composed of. The abstract receives a score that is equal to the sum of the scores of the scenes it is composed of.

It might be observed that also this evaluation approach needs the intervention of several users, as it was in the user study approach. On the other hand, in the ground-truth approach, once the process of scoring s-scene is done, experiments can be carried out and automatically evaluated.

Before presenting details of the ground-truth evaluation, it is worth pointing out that the data produced by the set of users, presented a large statistical difference in the scores related to TV-News, whereas more homogeneous scores have been given for cartoon and TV-Show videos. This shows the importance of a storyline in the video: TV-news has multiple storyline, each one presented in a different video clip and some users might prefer some storylines to others (e.g., the same soccer video may be evaluated as very important by a soccer fan, whereas it can be meaningless for his wife). Conversely, when there is a single (or few) storyline, as in cartoon/TV-shows, evaluation of the video clips tend to be more oriented to the video storyline and less to the users interests.

Figure 1 reports results of the ground-truth evaluation of abstracts produced with frame-based analysis. Results are normalized to the quality achieved by the random approach (i.e., positive values mean better results than random, negative worst). Clustering techniques are worth using only for cartoon videos; for TV-News there is no significant difference with the random approach; for TV-Show videos clustering are not worth using for 2 minutes abstracts, whereas some benefits are present for 4 minutes abstracts. The TV-News behavior is not surprising considering the large statistical difference of the ground-truth evaluation. It is a little bit surprising the bad results of 2 minutes long TV-Show abstracts. An explanation might be that the abstracts are too short compared to the original videos (2 minutes abstract against a 40 minutes video) and hence it difficult to pick up interesting scenes to fill in such a limited time abstract (for instance, *Roswell* abstracts behave very similar to the one randomly produced, whereas *Charmed* abstracts behave much worse than the random ones). In fact, looking at the longer 4 minutes abstracts, we can see better results for clustering.

Figure 2 reports results of the ground-truth evaluation of abstracts produced with scene-based analysis. Also in this case, results are normalized to the achieved random quality. Clustering techniques are not worth using for TV-News videos, whereas there are benefits for cartoon videos. For TV-Show videos there are no significant benefits for 2 minutes abstracts, whereas clear benefits are present for 4 minutes long abstracts produced with the eFPF technique.

B. *Time production evaluation*

In this section we analyze the abstract production time, which is very important for video abstract lengths customization, as abstracts have to be produced on-the-fly to meet the user request. The following results are obtained using a simple Pentium D 3.4 GHz with 3GB RAM. Although more powerful hardware can be employed to lower the production time, the ratio is likely to be the same.

Figure 3 reports results related to the abstract production time (in seconds) with frame analysis, given on a logarithmic scale. Production time of random abstract is not reported as it is less than one second, regardless of the type of video. Needless to say, the lower the production time, the better for a customized service. Observe that k-means is out of the game (note that we don't consider the time spent looking for a good value of k), as it takes too much time to produce a video abstract. eFPF has reasonable performances only for TV-News (27 seconds to produce an abstract of a 15 minutes video).

Figure 4 reports results related to the abstract production time (in seconds) with scene analysis, given on a logarithmic scale. Again, production time of random abstract is not reported as it is around 0.1 seconds, regardless of the type of video. eFPF has always good performances (19 seconds to produce a 4 minutes abstract of a 40 minutes video and less than one second for TV-News videos), whereas, k-means has reasonable production time only for TV-News videos.

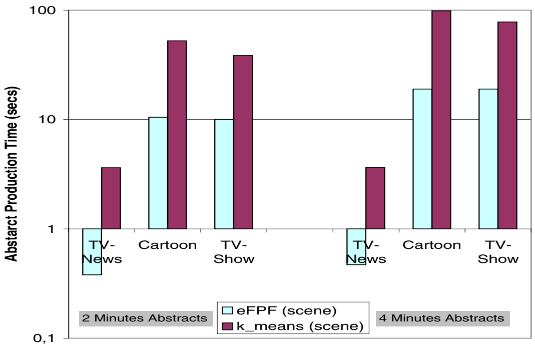

Fig. 4. Production Time: Comparison of abstracts produced using a scene analysis. Results are presented on a logarithmic scale.

C. Summary of Results

Experimental results lead to the following conclusions:

- Clustering techniques seems not to be useful for multiple storyline videos like TV-News. If videos are based on a storyline, as cartoons and TV-Shows, the benefits of clustering are significant, especially for 4 minutes abstracts based on video scene analysis.
- Random selection has to be preferred to clustering for 2 minutes long abstracts. In such videos, the limited number of scenes that can be selected to compose the abstract, compared to the total number of video scenes, does not leave much space for interesting choices.
- Production of abstracts by frame analysis takes much longer than those by scene (e.g., the eFPF scene-based is one order of magnitude more efficient than the frame-based) and hence abstracting by frame-analysis is not a winning strategy.
- k-means is too time consuming to think of it as a mechanism to produce on-the-fly abstracts.
- If user customization is enabled only a very fast clustering algorithm as eFPF can be used.

Another important issue is the estimation of the number of clusters k to get an abstract of the desired length. Since, it is not easy to associate k with scene lengths, clustering algorithms should be not obliged to start over again if the choice of k results incorrect.

V. Conclusions

In this paper we analyzed the benefits of using clustering techniques to produce video abstracts. We investigated two well known clustering algorithms, one known as very fast, the other known as very accurate. Both methods have been evaluated with several different videos and using two different approaches to find similarity: one based on low-level frame HSV color histogram and one based on low-level scene HSV color histogram. Results showed that clustering algorithms are not worth using for TV-News videos. In this case, additional information, like the subjects relevant to a user, should be taken into account. Conversely, for storyline videos like cartoons and TV-Shows, clustering techniques provide benefits. We also show that the only way to use clustering while offering a customized service, is to use very fast algorithm as eFPF.

VI. Acknowledgments

The work of Filippo Geraci and Marco Pellegrini has been partially supported by the Italian Registry of ccTLD "it". The work of Manuela Montangero has been partially supported by the M.I.U.R. under the PRIN 2006018748_004 initiative.

References

[1] J. Oh, Q. Wen, J. Lee, and S. Hwang, *Video Abstraction*. Idea Group Inc.and IRM Press, 2004, ch. XIV, pp. 321–346.

[2] B. T. Truong and S. Venkatesh, "Video abstraction: A systematic review and classification," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 3, no. 1, pp. 1–37, 2007.

[3] J. Nam and A. Tewfik, "Video abstract of video," in *IEEE 3rd Workshop on Multimedia Signal Processing*, 1999, pp. 117–122.

[4] A. Hanjalic and H. Zhang, "An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis," *IEEE Trans. Cir. and Sys. for Video Tech.*, vol. 9, no. 8, pp. 1280–1289, 1999.

[5] Y.-P. Tan and H. Lu, "Video scene clustering by graph partitioning," *Electronics Letters*, vol. 39, no. 11, pp. 841–842, 2003.

[6] M. M. Yeung and B.-L. Yeo, "Time-constrained clustering for segmentation of video into story unites," in *ICPR '96: Proceedings of the International Conference on Pattern Recognition (ICPR '96)*. Washington, DC, USA: IEEE Computer Society, 1996, pp. 375–380.

[7] C.-W. Ngo, Y.-F. Ma, and H. Zhang, "Video summarization and scene detection by graph modeling." *IEEE Trans. Circuits Syst. Video Techn.*, vol. 15, no. 2, pp. 296–305, 2005.

[8] M. Furini, "On ameliorating the perceived playout quality in chunk-driven p2p media streaming systems," in *ICC '07: Proceedings of the IEEE International Conference on Communications*, 2007.

[9] B. S. Manjunath, J. R. Ohm, V. V. Vasudevan, and A. Yamada, "Color and texture descriptors," *IEEE Transactions on Circuits and Systems For Video Technology*, vol. 11, pp. 703–715, 2001.

[10] S. J. Phillips, "Acceleration of k-means and related clustering algorithms," in *Proceedings of ALENEX-02, 4th International Workshop on Algorithm Engineering and Experiments*, San Francisco, US, 2002, pp. 166–177.

[11] T. F. Gonzalez, "Clustering to minimize the maximum intercluster distance," *Theoretical Computer Science*, vol. 38, no. 2/3, pp. 293–306, 1985.

[12] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proceedings of STOC-02, 34th Annual ACM Symposium on the Theory of Computing*, Montreal, CA, 2002, pp. 380–388.

[13] M. Furini, F. Geraci, M. Montangero, and M. Pellegrini, "VISTO: VIsual Storyboard for Web Video Browsing," in *CIVR '07: Proceedings of the ACM International Conference on Image and Video Retrieval*, July 2007.