Comparison of Content Selection Methods for Skimming Rushes Video

Werner Bailer and Georg Thallinger
JOANNEUM RESEARCH, Institute of Information Systems & Information Management
Steyrergasse 17, 8010 Graz, Austria
{firstName.lastName}@joanneum.at

ABSTRACT

We compare two methods for selecting segments to be included in a video skim, using lists of relevant as well as redundant segments created from different visual features as input. One approach is rule-based, and creates a weighted sum of the input relevances. The other is HMM based, using a model trained on the TRECVID 2007 rushes data. The redundant segments are created from detection of repeated takes and junk content, the selected segments from visual activity and face detection. The results show that the approaches create very short summaries which only contain a part of the relevant information in the video, but reach very high scores in terms of the usability measures non-duplicates, non-junk and pleasant tempo. The HMM based approach contains more information despite shorter duration of the summaries.

Categories and Subject Descriptors

H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems; H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Abstracting methods*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Experimentation

Keywords

Video skim, Rushes, Summarization, Content abstraction, HMM, TRECVID

1. INTRODUCTION

Our work is motivated by the problem of content management in digital cinema and video post-production. After shooting, a large amount of raw material ("rushes") must be organized and viewed in order to identify the small fraction that will be used in the final content. In previous work we have explored methods for reducing redundancy in the content, most importantly by clustering multiple similar takes of one scene [3, 4]. Once redundant segments have been identified a crucial task is to decide which clips from the remaining segments shall be included in the final summary.

The problem can be illustrated by looking at the ground truth provided by NHK [7] for the TRECVID 2007 rushes data set. An average of 38.02% of the video (minimum 11.13% and maximum 87.75% for individual videos) is annotated as relevant content in the ground truth. This comprises all "meaningful" content, including repeated takes of the same scene. If we discard all segments except for takes without repetition and the longest take of each scene (assuming that this is the one containing the complete action) still on average 15.20% of the content duration remain "relevant". This means that if we want to create summaries shorter than that, we have to decide which clips of these segments are sufficiently representative.

In this paper we compare two approaches for segment selection. A rule-based approach, which is an extension of the method described in [2] and a Hidden Markov Model (HMM) based approach. Section 2 gives an overview of the skim creation process and briefly describes all steps except for the segment selection, which is discussed in more detail in Section 3. We then present and discuss in Section 4 the results of the two runs submitted to the TRECVID 2008 evaluation [8] and conclude with Section 5.

2. SKIM CREATION PROCESS

We use the skim creation process proposed in [3], which we briefly summarize here. As shown in Figure 1 this process comprises five major steps: A set of content analysis algorithms is applied to extract visual features, which are used for the subsequent content selection and clustering steps. This includes the removal of junk content (e.g. color bars), detection and clustering of retakes and the selection of representative video clips from the takes. Finally editing of the skim is performed.

The main innovation of this work are the selection methods for the clips to be included in the skim. Thus this step is discussed in more detail in Section 3, while the other steps are briefly described in the following.

2.1 Content Analysis and Removing Junk Content

The content analysis module aims at extracting several low-level features from the uncompressed input video stream.

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

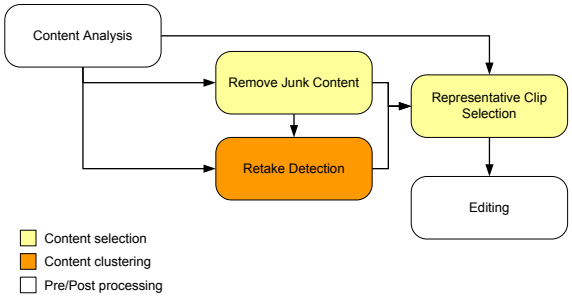

Figure 1: Overview of the skim creation process.

In particular, this analysis includes shot boundary detection, extraction of key frames and their visual features, calculation of visual activity, detection of test patterns such as color bars and monochrome frames, and face detection.

The shot boundary detector (SBD) and test pattern detector split the entire video into smaller pieces which can be handled more easily. We do not want the selected video segments to include shot boundaries or test patterns, since they are of no value in our context. As the BBC rushes data includes hard cuts only the detection of gradual transitions has been deliberately excluded. The SBD is implemented by calculating the pixel-wise frame differences $d_{i,o}$ between the current frame i and two following frames with offsets $o = 1$ and $o = 2$. In order to find the hard cut, we determine the frame position k where $d_{k,2}$ is significantly greater than $d_{k,1}$. If this criterion is fulfilled, then we assume a hard cut occurs between frame $k + 1$ and $k + 2$. This task is solved using a linear support vector machine (SVM) implemented through LIBSVM [5]. To train the SVM, ground truth from the TRECVID 2006 SBD task [9] has been used.

In order to facilitate sequence matching for retake detection, key frames are extracted by subsampling with a fixed rate. The MPEG-7 descriptors ColorLayout and EdgeHistogram [6] are calculated for each of the key frames. We also skip short shots (duration less than 10 seconds) and discard short content segments at the beginning and end of each shot.

To determine if the current frame comprises only color bars or junk content (i.e. black frames), we first calculate the standard deviation of the RGB-pixel value in each pixel column and channel. When the maximum of the column's standard deviations is lower than a threshold, the frame is labeled as junk. Note that this method utilizes the strong assumption that only standard vertical color bars appear in the content.

The face detector is based on the algorithm developed by Viola and Jones [10] which is implemented in Intel's OpenCV[1] library. The face detection result forms a discrete function. To eliminate short-term false detections (i.e. sudden appearance or disappearance of faces) mathematical morphological closing and opening operators are applied to the sequence of face detection results.

2.2 Retake Detection

The problem of detecting and clustering multiple takes of the same scene shot from the same or similar camera positions is formulated as a problem of matching sequences of visual features (ColorLayout and EdgeHistogram descriptors for every key frame, average visual activity between the key frames) of parts of the input video. The algorithm uses the LCSS (Longest Common Subsequence) model and is briefly summarized in the following. A detailed description can be found in [4].

An algorithm based on a variant of the LCSS algorithm for matching object trajectories in 2D space has been proposed in [11]. In order to adapt the LCSS algorithm to matching takes of a video the following modifications are made. As each element of the sequence is a multidimensional feature vector, a vector $\theta_{sim} = (\epsilon_1, \ldots, \epsilon_K)$ is defined, that contains the matching thresholds for all K features. Similarly a vector $W = (w_1, \ldots, w_K)$ representing the relative weights ($\sum_k w_k = 1$) of the features is introduced. The offset in the sequence δ introduced in [11] is not relevant for this problem, as the matching subsequences can be anywhere in the parts. Instead the maximum gap size γ of the subsequence is introduced as constraint. Note that the longest common subsequence does not necessarily fulfill the condition of having only gaps $\leq \gamma$, so instead of just finding the LCSS all sufficiently long subsequences (length $> \theta_{len}$) with gaps no longer than γ need to be found. Matching sequences are considered as take candidates and are clustered by similarity in order to group them by scene.

The matching thresholds θ_{sim} for the feature vectors can be replaced by a sigmoid weighting function with a gain parameter tuned by the standard deviation of the input feature sequences. It has been shown in [1] that the algorithm in this case is more sensitive to errors in the segmentation. As we cannot expect the hard cut detection method to yield correct take boundaries, we have used the LCSS method with matching thresholds.

2.3 Editing Videos

The final step of the process is to produce the skim. The selection step outputs a list of video segments to be included in the summary. This list is represented as an MPEG-7 document. An XSL stylesheet is used to transform the MPEG-7 document into a Linux shell script that calls the command line tools performing the edit operations. The edit script contains the following steps: (a) extraction of clips from the source content, (b) demultiplexing, (c) audio processing, (d) multiplexing and (e) concatenating the segments. The *ffmpeg* and *mencoder*[2] tools are used for the video related tasks, SoX[3] is used for audio processing. Audio processing consists of adding a short fade in and fade out effect to each of the clips in order to avoid artifacts at the clip boundaries. For clips shorter than 3.5× the fade duration the audio signal is completely suppressed. As the clips in our runs are very short, audio remained only in a tiny fraction of the clips included in the skims. Thus we decided to entirely discard audio in our skims for the submitted runs.

3. SEGMENT SELECTION METHODS

The final segment selection step merges the different lists of selected and redundant segments in order to produce an output selection list of segments which shall be included in the final summary. The input to the final segment selection are lists of selected and redundant segments as created

[1] `http://sourceforge.net/projects/opencvlibrary`

[2] `http://ffmpeg.mplayerhq.hu`

[3] `http://sox.sourceforge.net`

by the different features. Selected segments are segments of the input video that are found to be relevant w.r.t. a certain feature and each segment has a relevance value. Redundancy information consists of *absolutely* redundant segments, which shall not be included in the skim (e.g. color bars) and *relatively* redundant segments, which are redundant w.r.t. to other segments in a group and from which at most one shall be included in the summary (e.g. clusters of takes of the same scene). In the latter case there is a similarity value for each pair of segments in a group.

In our experiments we produce the following selected and redundant segment lists:

- retakes: relative redundancy information with similarity values
- junk content: absolute redundancy information
- motion activity: selected segments with relevance
- presence of faces: selected segments with relevance

We compare two methods for the selection of segments to be included in the summary, a rule-based and a Hidden Markov Model (HMM) based method. As we cannot guarantee that the output of HMM decoding fulfills the maximum length constraint, the final steps of the rule-based method are applied to the selected segments list created by the HMM method to ensure that the length constraint is met. Figure 2 shows an overview of the selection step and the two methods are described in more detail in the following.

3.1 Rule-based selection method

The rule-based selection method basically calculates a weighted sum of the input relevance and redundancy lists. The selection algorithm (i) transforms relative into absolute redundant segment lists, (ii) combines the relevant and redundant segment lists, (iii) selects an appropriate threshold and adapts the segment selection to the length constraints of the summary. Relative redundancy information such as that from take clustering cannot be used directly, as not *all* but only *all but one* segments of a cluster are redundant. The reason for deferring this decision into the final selection is that results from other analysis methods (e.g. detection of junk content) is also available. We have implemented two approaches: The first option is to use the longest of the alternative takes. This ensures that most of the content of the take is included, even if it is unique to this take (e.g. this could be the only complete take, while the others in the cluster are only partial takes). The disadvantage is that parts of this take may need to be discarded later to fulfill length constraints. The second option is to use a clip that is most representative for the cluster, i.e. is shared by most of the alternative takes and has a high relevance. Applying one of these strategies allows treating the selected take or clip as relevant and the others as redundant. The lists of relevant and redundant segment are interpreted as relevance functions $\mathrm{rel}(t)$ and $\mathrm{red}(t)$ over time t. The selection function is then given as

$$\mathrm{select}(t) = \begin{cases} 1, \text{if } \frac{w_{rel}}{n_{rel}} \sum_{k=1}^{n_{rel}} \mathrm{rel}(t) + \frac{w_{red}}{n_{red}} \sum_{k=1}^{n_{red}} \mathrm{red}(t) \geq \theta \\ 0, \text{otherwise,} \end{cases}$$

where n_{rel} and n_{red} are the number of input relevant and redundant segment lists, w_{rel} and w_{red} are the relative weights of relevance and redundancy information and θ is a threshold.

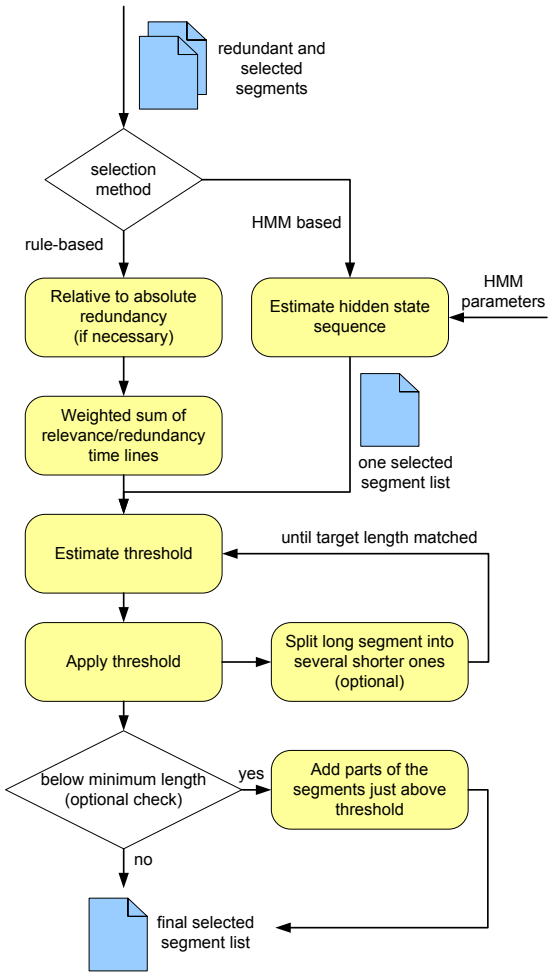

Figure 2: Overview of the selection step with rule-based and HMM based selection methods.

The next step is to determine θ, so that $\sum_t \mathrm{select}(t)$ is maximum while $\sum_t \mathrm{select}(t) \leq T$, with T being the target length of the summary. The optimization problem is solved using binary search. Very short segments (below a user defined threshold) in the result are discarded, as they are hard to perceive and rather disturbing in the summary.

Optionally longer segments are split into several shorter ones in order to increase the number of different clips in the selected segments. This produces results with several shorter segments instead of fewer longer ones. The step is applied before the final shortening of segments.

Especially in cases where few input lists are available, a number of segments with the same total relevance value may exist, so that – depending on the choice of the threshold – all of them are kept or discarded. As a result summaries that are much shorter than the maximum length are created. To avoid this we apply the following strategy. If the length of the summary would be below a fraction of the maximum length, θ is set to the relevance of the segments in question, i.e. so that they are included. If several segments from one shot of the common segmentation are selected, only the longest is kept. Of the remaining segments we iteratively select the longest segment and crop it by a fraction of its length at beginning and end to match the length constraint.

The rule-based method has the following parameters: the minimum and maximum duration of a selected segment, the minimum temporal distance between two selected segments (all specified in seconds), the maximum total duration of selected segments (specified as fraction of the input), the minimum total duration of selected segments (specified as fraction of the maximum total duration), the weights w_{rel} and w_{red} for combining summed relevances with summed redundancies and a flag that determines whether long segments are split into several short ones.

3.2 HMM based selection method

The vector of relevance/redundant scores for each sample of the video is the observation sequence of our HMM. The sequence of hidden states represents the selection state of a sample. As we use a Discrete HMM (DHMM) we convert the input vectors to discrete scalars by assigning a fixed number of bits to each element of the input vector. The same content analysis algorithms are applied to both the training and test set to yield the feature sequences for each video of the sets. For the training set, the sequence of hidden states is created from a ground truth annotation. We use the annotation for the TRECVID 2007 rushes data set provided by NHK [7] which contains lists of relevant segments and an identification to which scene each segment belongs. Without loss of generality we can treat a training set consisting of several videos as one long sequence. The HMM has the following 6 states:

non-relevant ($\mathbf{N_{pre}}$, state 1) A sample of the video that does not lie within any of the segments marked as relevant in the ground truth and is before the selected segment from this scene.

relevant ($\mathbf{R_{pre}}$, state 2) A sample of the video that lies within one of the segments marked as relevant in the ground truth, but shall not be selected for inclusion in the summary, and is before the selected segment from this scene.

selected (S, state 3) A sample of the video that lies within one of the segments marked as relevant in the ground truth and shall be selected for inclusion in the summary.

scene boundary (B, state 4) A sample corresponding to a scene boundary (i.e. the first sample of a scene).

non-relevant ($\mathbf{N_{post}}$, state 5) A sample of the video that does not lie within any of the segments marked as relevant in the ground truth and is after the selected segment.

relevant ($\mathbf{R_{post}}$, state 6) A sample of the video that lies within one of the segments marked as relevant in the ground truth, but shall not be selected for inclusion in the summary, and is after the selected segment.

Figure 3 shows a graphical representation of this model. A global parameter λ is introduced in the transition matrix to control the number and length of the selected segments. The distinction of before/after states is introduced in order to ensure that zero or one segments are selected from each scene. There are four groups of states: states that can be taken before a selected segment in the scene has been encountered (N_{pre} and R_{pre}, changing between them is possible), the selected state (S), states that can be taken after a selected segment in the scene has been encountered (N_{post} and R_{post}, changing between them is possible) and the scene boundary (B). These groups of states can only be traversed in this order and after the scene boundary only the pre-selection states N_{pre} and R_{pre} can follow. The only exception is the direct path from N_{pre} to N_{post} which allows for zero selected segments, but it also prevents the state to be relevant, i.e. it aims to avoid selection from junk segments.

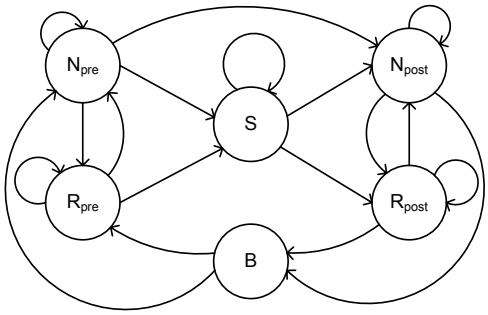

Figure 3: 6 state HMM for segment selection.

4. RESULTS

We have submitted two runs based on the same features, i.e. the same selected/redundant segments lists have been used, and only the clip selection method has been varied. Run JRS1 uses the rule-based selection method, JRS2 uses the HMM based method. Table 1 shows the parameters that have been used for the submitted runs and the results achieved for the evaluation criteria.

The results show poor scores for the inclusion rate but very good results for the non-duplicate, non-junk and tempo scores. Both runs reach the maximum value of the non-duplicate score. This shows that the retake detection algorithm works satisfactorily on this material. The submitted runs are among the shortest summaries, well below the maximum length of 2%: the rule based run is 58.00% of the maximum length, the HMM run is 49.65% of maximum length (median values over all videos), that corresponds to 1.20% and 0.99% of the total content lengths respectively.

If we look at the comparison of the two runs we see that the HMM based selected method yields a 6% higher inclusion rate (increase of 27%), although the duration is 24% shorter. This comes at the expense of less pleasant tempo and slightly higher time needed for judging the summary. The lower non-junk score is caused by the difficulty to constrain the HMM: while in the rule-based approach segments classified as redundant by the junk detector get a lower value in the selection function, the most likely state sequence of the HMM may in some cases include junk segments.

A comparison of the results for individual videos shows that both durations (correlation $r = 0.37$) and inclusion rates ($r = 0.11$) are quite different for the two approaches. This supports the assumption that the use of the final steps of the rule-based approach for post processing the HMM based result with similar parameters does not bias the HMM based result towards that of the rule-based method.

The summaries have been created on a Pentium D 3.0 GHz with 2 GB of RAM, using Windows XP for content analysis, retake detection and segment selection and Fedora Core 5 in VMware machine (on the same computer) for running the editing scripts. The total runtime for creating the

	JRS1		JRS2	
Parameters				
min. segment length	0.5		2.0	
max. segment length	3.0		3.0	
min. segment distance	5.8		5.8	
w_{rel}=w_{red}	0.5		n/a	
max. total length	0.02		0.02	
min. total length	0.30		0.30	
rel. to abs. redundancy	longest		n/a	
split long segments	true		true	
Results				
DU (median)	18.50	(0.15)	14.00	(0.03)
XD (median)	13.38	(0.24)	14.20	(0.22)
TT (median)	25.33	(0.09)	26.67	(0.13)
VT (median)	20.00	(0.05)	18.33	(0.00)
IN (median)	0.22	(0.19)	0.28	(0.27)
IN (min)	0.00		0.08	
IN (max)	0.53		0.67	
JU (median)	3.67	(1.00)	3.00	(0.50)
RE (median)	4.00	(1.00)	4.00	(1.00)
TE (median)	3.33	(1.00)	2.33	(0.50)

Table 1: Parameters and results for the two JRS runs (JRS1: rule-based, JRS2: HMM). The values specify the median (in case of IN also minimum and maximum) for each of the runs over all videos. The values m'_k in parentheses are calculated by mapping the value range of the median values m_i of all submitted runs to the unity interval, i.e. $m'_k = \frac{m_k - \min_i(m_i)}{\max_i(m_i) - \min_i(m_i)}$. For the time measures (DU, XD, TT, VT), a value of 0.0 corresponds to the shortest, a value 1.0 to the longest duration of all participants (median over all videos). For inclusion (IN) and the subjective measures (JU, RE, TE) a value of 0.0 corresponds to the worst score, a value 1.0 to the best score of all participants (median over all videos).

runs is 0.84× real-time for the rule-based method and 0.89× real-time for the HMM based method. Content analysis is the major factor, taking 67% (rule-based) and 63% (HMM based) of the total processing time. It has to be noted that editing is quite slow as it was running in a virtual machine, taking 18% (rule-based) and 27% (HMM based) of the total time. The longer runtime of the HMM based method is caused by the fact that this result contains more and shorter segments, which increases the time for editing by 47%. The training time for the HMM is negligable, as this is only done once and estimating the most likely state sequence takes on average 4.75 seconds per video.

5. CONCLUSION

Selection of relevant segments to be included in summaries is a crucial step in video skimming. In this paper we compare two approaches: a rule-based approach and a HMM based approach. Both are applied to relevant and redundant segment lists created from detection and clustering of multiple takes of one scene, junk content detection, motion activity and face detection.

The evaluation shows very good results in terms of the scores for non-duplicates, non-junk and pleasant tempo. The two methods produce short summaries well below the maximum length, at the cost of a low inclusion rate. Despite shorter duration of the summaries the HMM based method yields a 6% higher inclusion rate. Due to this denser packing of information the tempo is perceived less pleasant.

The results show that the HMM based summaries contain more junk. This is due to the fact that junk content cannot be deterministically excluded as in the rule-based case. Also the results of the HMM based method need post-processsing in order to constrain the results to the maximum length and exclude extremely short clips. HMMs are not powerful enough to model these constraints so the use of more complex models will be investigated.

6. ACKNOWLEDGMENTS

The authors would like to thank Felix Lee, Harald Stiegler, Herwig Rehatschek and Werner Haas for their support. The research leading to this paper has been partially supported by the European Commission under the contracts "SEMEDIA" (FP6-045032) and "K-Space" (FP6-027026).

7. REFERENCES

[1] W. Bailer. A comparison of distance measures for clustering video sequences. In *Proceedings of 1st Workshop on Automated Information Extraction in Media Production*, Turin, IT, Sept. 2008.

[2] W. Bailer, E. Dumont, S. Essid, and B. Mérialdo. A collaborative approach to automatic rushes video summarization. In *Proceedings of IEEE International Conference on Image Processing*, San Diego, CA, USA, Sept. 2008.

[3] W. Bailer, F. Lee, and G. Thallinger. Skimming rushes video using retake detection. In *Proceedings of the TRECVID Workshop on Video Summarization (TVS'07)*, pages 60–64, September 2007.

[4] W. Bailer, F. Lee, and G. Thallinger. Detecting and clustering multiple takes of one scene. In *MMM*, pages 80–89, Kyoto, JP, Jan. 2008.

[5] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001.

[6] MPEG-7. Multimedia Content Description Interface—Part 3: Visual. ISO/IEC 15938-3, 2001.

[7] NHK STRL. Test modules for TRECVID activity. Use case scenario. Ver.1.2.0E, Apr. 2008.

[8] 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, 2008.

[9] A. F. Smeaton and P. Over. TRECVID 2006: Shot boundary detection task overview. In *Proceedings of the TRECVID Workshop*, November 2006.

[10] P. Viola and M. Jones. Fast and robust classification using asymmetric adaboost and a detector cascade. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.

[11] M. Vlachos, G. Kollios, and D. Gunopoulos. Discovering similar multidimensional trajectories. In *ICDE '02: Proceedings of the 18th International Conference on Data Engineering*, pages 673–684, San Jose, CA, USA, 2002. IEEE Computer Society.