Using Self-similarity Matrices for Structure Mining on News Video

Arne Jacobs

Center for Computing Technologies, University of Bremen
jarne@tzi.de

Abstract. Video broadcast series like news or magazine broadcasts usually expose a strong temporal structure, along with a characteristic audio-visual appearance. This results in frequent patterns occurring in the video signal. We propose an algorithm for the automatic detection of such patterns that exploits the video's self-similarity induced by the patterns. The approach is applied to the problem of anchor shot detection, but can also be used for other related purposes. Tests on real-world video data show that it is possible with our method to detect anchor shots fully automatically with high reliability.

1 Introduction / Related Work

The detection of frequent patterns in video is useful in the case when the given videos expose a strong temporal structure. The results of an anchor shot detection, for example, may be used to help finding story boundaries in news or magazine broadcasts. Other examples are special screens or video sequences used in news videos to announce different topics. Patterns are used in these videos to make it possible for the viewer to easily follow the broadcast and to recognize its structure. They also make it possible to do this so-called "Video Parsing" [1] automatically.

Hauptmann et al. [2] use an SVM on color and face features to identify anchor shots in news videos. In addition to training the SVM, their classifier requires learning of a weighted image grid for the color features to account for changing anchors, clothing, etc. between the different news videos.

In [3], similar features are used, but instead of training, models for each expected presentation type have to be provided manually.

Kobla et al. [4] use the notion of "Video Trails" to identify similar shots in a video without training. However, they directly work on features extracted from the video frames, which makes it difficult to identify clusters because of the vast amount of frames that do not belong to any class and clog the feature space.

In this paper we present an automatic method for identifying frequent patterns, that does not work directly on image features, but on a matrix of similarity between different time points in the video. The method is very general and may be applied to other problems as well, but here we focus on the problem of anchor shot detection.

The next section descibes our algorithm, which is followed by our experimental results. We conclude with Sect. 4.

G. Antoniou et al. (Eds.): SETN 2006, LNAI 3955, pp. 87–94, 2006.

2 Proposed Approach

Our proposed algorithm consists of two steps:

- Computation of a self-similarity matrix
  Based on a similarity measure between two time points in the video, a two-dimensional symmetric matrix is computed, whose height and width is dependent on the length of the video.
- Clustering on rows of the self-similarity matrix
  Here we use a simple K-Means clustering algorithm to find patterns in the matrix, but other clustering methods might also be used.

Both steps can be customized depending on the problem at hand. Here we focus on anchor shot detection. In the next section, the computation of the self-similarity matrix is described in further detail, followed by an overview of the clustering step.

2.1 Self-similarity Matrix

The self-similarity matrix M is a $N \times N$ matrix defined as:

$$M_{ij} = S(t_i, t_j), \tag{1}$$

where $S(t_i, t_j)$ is a similarity measure between two timepoints t_i and t_j in the video. In our case, we choose frame-based timepoints $t_1, \ldots, t_i, \ldots, t_j, \ldots, t_N$ that are multiples of ten, i.e., $N = N'/10$ for a video with N' frames.

The form of the similarity measure $S \in [0, 1]$ (where 1 corresponds to maximal similarity, and 0 corresponds to minimal similarity) depends on the kind of patterns to be detected. Assuming that presentations are always shot in a certain setting and show the same presenter, for our purpose we define S as the product of a color distribution-based and a face-based similarity:

$$S(t_i, t_j) = S_{color}(I(t_i), I(t_j)) \cdot S_{face}(I(t_i), I(t_j)), \tag{2}$$

where $I(t)$ is the video frame at timepoint t. This measure is based on visual information only, i.e., the video frames, but other modes, e.g., audio data, may also be incorporated, if they contain information useful for the solution of the problem.

The color-based similarity is given by:

$$S_{color}(I(t_i), I(t_j)) = s(\|CC(I(t_i)) - CC(I(t_j))\|), \tag{3}$$

where $CC(I(t))$ is the color correlogram according to [5] of frame $I(t)$, and $\|\ldots\|$ denotes the L_2-Norm. Our assumption here is that presentations are always placed in the same or just a limited number of different studio settings. We choose the color correlogram feature as a representation of the visual appearance of the setting.

The face-based similarity is given by:

$$S_{face}(I(t_i), I(t_j)) = \begin{cases} 0 & \text{if } \neg face\_in(I(t_i)) \vee \neg face\_in(I(t_j)) \\ s(\|face(I(t_i)) - face(I(t_j))\|) & \text{otherwise} \end{cases} \quad (4)$$

where $face\_in(I(t)) \in \{\text{true}, \text{false}\}$ is true iff a face was detected in $I(t)$, and $face(I(t)) \in \mathcal{N}^4$ are the coordinates of the rectangle enclosing the dominant detected face in $I(t)$, if any. For face detection we use the algorithm by Lienhardt et al. [6].

Note that the form of the face-based measure means that a frame is dissimilar to each other frame (including itself) if it contains no face. However, this does not affect the algorithm in general. It just reflects our assumption that the anchor shots we want to find always show a face, and that this face is always shown at the same or a limited number of different positions in the video.

Finally, $s(x)$ as referenced in Equations 3 and 4 is the *GemanMcLure* probability distribution, scaled to $[0, 1]$:

$$s(x_i) = \frac{e^{\frac{-x_i^2}{\sigma^2 + x_i^2}} - e^{-1}}{1 - e^{-1}} \quad (5)$$

where the scale σ is proportional to the median of the distance values:

$$\sigma = \frac{\text{median}(x_1, \ldots, x_i, \ldots, x_{N^2})}{3} \quad (6)$$

We use $s(x)$ to convert distance values (low values correspond to high similarity) to similarity values between 0 and 1 (where high values denote high similarity).

Figure 1 shows an example similarity matrx. Dark areas denote high similarity. The main anchor sequences in the beginning and the financial news presentations at the beginning of the second half can be easily identified by the human eye.

2.2 Clustering

The similarity matrix computed in the previous section reveals temporal patterns of reoccurring frames, which in our case correspond to frames showing a face in a similar size and position, and a similar color distribution. Each row of the matrix contains a frame's similarity to each other considered frame. Frames belonging to a type of presentation will be more similar to other frames of the same type, and those in turn will also be similar to all the other frames of that type. If we see a row in the similarity matrix as a multidimensional feature vector, we find that rows corresponding to a frame of the same type of presentation will form a cluster in this new feature space.

Note that by clustering using the similarity matrix instead of using image features directly, we just need a similarity or distance measure between two frames (or time points in a video, respectively). We thus omit the need for features that lie in a vector space.

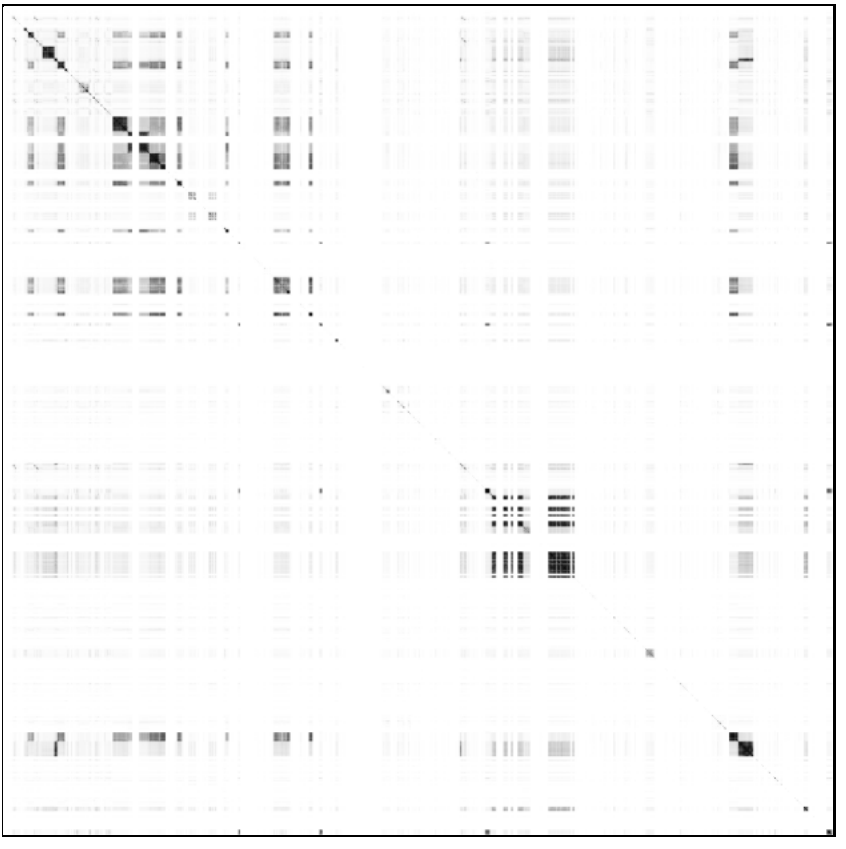

Fig. 1. Example of a similarity matrix computed on a video of "CNN Headline News", using color and face similarity

By applying a clustering algorithm to the matrix rows we can find the different types of presentations and we will also have a mapping from frame to presentation type. We use an adapted simple K-Means clustering algorithm. The number of classes to use as input for the algorithm depends on how many frequently occurring different presentation types we expect in a video. One additional class is added for all frames not belonging to any presentation. Our changes to the original K-Means algorithm ensure that these frames are all put into the same class.

We adapt the K-Means algorithm by weighting rows according to the sum of all entries, and by using a slightly different distance measure and a slightly different mean. The reasons for these changes are given below. The distance $d(i, j)$ between two matrix rows i and j used for clustering is defined as:

$$d(i,j) = \sqrt{\sum_{k=1}^{N} d(i,j,k)^2} \tag{7}$$

$$d(i,j,k) = (M_{ik} - M_{jk}) \cdot w_d(i,j) \tag{8}$$

$$w_d(i,j) = \begin{cases} 0 & \text{if } |i-j| < 50, \\ 1 & \text{otherwise} \end{cases} \tag{9}$$

By using this adapted distance measure we give more weight to similar frames that are not temporally adjacent, supporting frequently occurring patterns. The adapted mean $c(C,k)$ for column k of all rows belonging to the class C is defined by:

$$c(C,k) = \frac{\sum_{i \in C} M_{ik} \cdot w_d(i,k) \cdot w_r(i)}{\sum_{i \in C} w_d(i,k) \cdot w_r(i)} \tag{10}$$

where

$$w_r(i) = \sum_{k=1}^{N} M_{ik} \tag{11}$$

is the weight of row i. The adapted mean makes sure that high values in the diagonal of the similarity matrix do not distort clustering and that all frames not belonging to any frequently occurring pattern are clustered in the same class.

3 Experimental Results

We test our algorithm on several videos from the news broadcast series "CNN Headline News" from 1998 and the german magazine broadcast "ZDF Auslandsjournal" from 2002. In "CNN Headline News" we expect three frequently used types of presentations (see Fig. 2 for examples): Main anchor centered (a), main anchor with image on the right (b), and financial news presenter (c). We therefore cluster using four classes, three for the presentation types, and one for all other frames. For "ZDF Auslandsjournal" we use two classes, one for the main presentation (see Fig. 2 (d) as an example), and one for all other frames.

Figure 3 shows the results of our algorithm compared to the manually annotated ground truth. Table 1 lists the error rate. The second column shows the total amount of false classified frames. The third column shows the amount of false classified frames in relation to the total number of frames that belong to one presentation type. The latter measure is shown because the number of frames not belonging to any presentation type is comparatively large. Note that there is only one presentation (at the end of the third video) that was not detected at all, and just one presentation (in the beginning of the first video) that was mistaken for another presentation type. The other errors are mainly due to single missed frames, probably caused by a wrong face detection result.

Figure 4 shows example frames from the detected presentations, one for each type and video. Note that a model for each presentation type would be difficult to create because of varying setting, clothing, and camera position between the different videos. It would probably at least require some supervision, which means a considerable effort for a human annotator.

Fig. 2. Examples of frequent presentations in "CNN Headline News" (a), (b), (c) and "ZDF Auslandsjournal" (d)

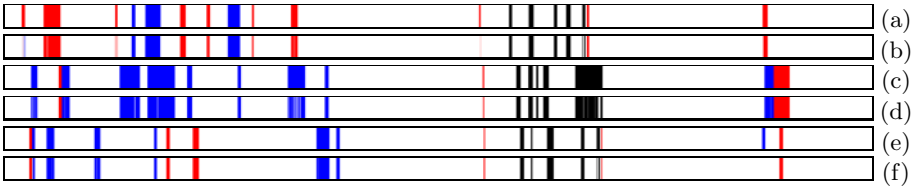

Fig. 3. Ground truth (a, c, e) and results of our algorithm (b, d, f) applied to three instances of the news broadcast series "CNN Headline News". The three identified types of presentations are shown in red (main anchor centered), blue (main anchor with image on the right), and black (financial news presenter), respectively.

Table 1. Error rate for three test videos from "CNN Headline News"

Video	Total error	Error regarding presentations only
1	1.37%	10.85%
2	1.18%	6.22%
3	0.73%	8.20%

The magazine "ZDF Auslandsjournal" just contains one presentation type that occurs often enough to be identified as a class. All three presentations of that type were correctly detected, Fig. 5 shows example frames from these presentations.

Fig. 4. Examples for each presentation type found by the algorithm. Each row corresponds to one video of "CNN Headline News". Each column corresponds to one identified presentation type.

Fig. 5. Examples for each presentation found in the video from "ZDF Auslandsjournal"

4 Conclusions and Outlook

We have presented a method for detection of frequently occurring patterns in videos and successfully applied it to the problem of anchor shot detection. The method works fully automatically and just needs as input the number of expected presentation types. No examples are required.

A suggestion to further reduce the error rate is to incorporate some kind of temporal constraint to account for the single misclassified frames inside a presentation. It may also be possible to find missed presentations by looking for frames that are similar to examples from the automatically identified classes.

Another extension of our method could be to create an inter-video similarity matrix to find sequences occurring in several videos but not frequently enough in one video to be clustered into one class. This includes, e.g., introduction sequences, weather maps, etc.

Acknowledgement

This work was partly supported by the research project AVAnTA [7] which is funded by the German Research Foundation (DFG) within the strategic research initiative No. 1041 "3D2 - Distributed Processing and Delivery of Digital Document".

References

1. Swanberg, D., Shu, C.F., Jain, R.: Knowledge guided parsing in video databases. In: Proceedings of the IS-T/SPIE Conference on Storage and Retrieval for Image and Video Databases. Volume 1908., San Jose, CA, USA (1993) 13–24
2. Hauptmann, A., Baron, R., Chen, M.Y., Christel, M., Duygulu, P., Huang, C., Jin, R., W.-H.Lin, Ng, T., Moraveji, N., Papernick, N., Snoek, C., Tzanetakis, G., Yang, J., Yang, R., Wactlar, H.: Informedia at TRECVID 2003: Analyzing and searching broadcast news video. In: Proceedings of the 12th Text Retrieval Conference (TREC). (2003)
3. Wang, W., Gao, W.: A fast anchor shot detection algorithm on compressed video. In: IEEE Pacific Rim Conference on Multimedia. (2001) 873–878
4. Kobla, V., Doermann, D.S., Faloutsos, C.: Videotrails : Representing and visualizing structure in video sequences. In: ACM Multimedia. (1997) 335–346
5. Huang, J., Zabih, R.: Combining color and spatial information for content-based image retrieval. In: Proceedings of ECDL 1998. (1998)
6. Lienhart, R., Maydt, J.: An extended set of haar-like features for rapid object detection. In: Proceedings of ICIP 2002. Volume 1. (2002) 900–903
7. Miene, A., Herzog, O.: AVAnTA - Automatische Video Analyse und textuelle Annotation. it + ti Informationstechnik und Technische Informatik **42** (2000) 24 – 27