Video Segmentation Combining Similarity Analysis and Classification

Matthew Cooper
FX Palo Alto Laboratory
3400 Hillview Ave. Bldg. 4
Palo Alto, CA 94304 USA
cooper@fxpal.com

ABSTRACT

In this paper, we compare several recent approaches to video segmentation using pairwise similarity. We first review and contrast the approaches within the common framework of similarity analysis and kernel correlation. We then combine these approaches with non-parametric supervised classification for shot boundary detection. Finally, we discuss comparative experimental results using the 2002 TRECVID shot boundary detection test collection.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*Indexing methods*

General Terms

algorithms, management

Keywords

temporal media indexing and segmentation

1. INTRODUCTION

Numerous video retrieval and management tasks rely on accurate segmentation of scene boundaries. Many existing systems compute frame-indexed scores quantifying local novelty within the media stream. The novelty scores are calculated in two steps. First, an affinity or similarity matrix is generated, as in Figure 1. Next, the frame-indexed score is computed by correlating a small *kernel function* along the main diagonal of the similarity matrix. Typically, detected local maxima in the novelty score are labelled as segment boundaries.

In this paper, we compare several kernels used for media segmentation based on similarity analysis. We first review similarity analysis in Section 2. Section 3 examines each of the kernels used to produce a frame-indexed correlation or novelty score. Section 4 presents comparative experimental results for shot boundary detection using the TRECVID data and evaluation tools [1]. In the first experiment, we use the different kernels and corresponding novelty scores as input to a binary K-nearest-neighbor (KNN) classifier that labels frames as either cut boundaries or non-boundaries. In the second experiment, we directly use the pairwise similarity data as input to train and test the KNN classifier. For this experiment, we vary the specific local set of similarity data again according to the proposed choices of kernels.

MM'04, October 10-16, 2004, New York, New York, USA.

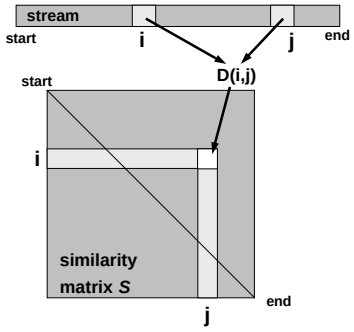

Figure 1: Diagram of the similarity matrix embedding.

2. SIMILARITY ANALYSIS

2.1 Matrix embedding

We detect scene boundaries by quantifying the similarity between pairs of video frames. First, low-level features are computed to represent each frame. Throughout this paper, we extract histograms in the YUV colorspace; these features are a common choice for segmentation, e.g. [2]. Denote the frame-indexed feature data $\mathbf{V} = \{V_n : n, = 1, \cdots, N\}$. A measure D of the similarity between frame parameters V_i and V_j is calculated for every pair of video frames i and j. The *similarity matrix* $\mathbf{S}$ contains the similarity measure calculated for all frame combinations, as depicted in Figure 1. Throughout this paper, we compare feature vectors using the squared Euclidean vector distance:

$$\mathbf{S}(i,j) = D(V_i, V_j) \equiv \|V_i - V_j\|^2 \quad . \tag{1}$$

This choice for D measures *dissimilarity.* Time, or the frame

index, runs along both axes as well as the diagonal. **S** has minimum dissimilarity (zero) along the leading diagonal where each frame is compared to itself. Because D is symmetric, **S** is also symmetric.

2.2 Kernel-based features

Segment boundaries exhibit a distinct pattern in **S**. Specifically, the frames comprising coherent segments exhibit low *within-segment* dissimilarity, creating square regions along the main diagonal of **S** with low values. The boundary between two such segments produces a checkerboard pattern in **S** if the two segments have high *between-segment* dissimilarity, creating rectangular regions off the main diagonal with high values. This suggests that finding the scene boundary transitions is as simple as finding the checkerboards along the main diagonal of **S**. This can be done using a matched filter: correlating **S** with a kernel, **K**, that itself looks like a checkerboard [3]. The correlation produces a frame-indexed novelty score that can be processed to detect segment boundaries. Specifically, define the kernel correlation, or equivalently, novelty score:

$$\nu(n) = \sum_{l=-L}^{L-1} \sum_{m=-L}^{L-1} \mathbf{K}(l,m)\mathbf{S}(n+l, n+m) \quad . \tag{2}$$

By varying the kernel width L, the novelty score can be tuned to detect boundaries between segments of a specific minimum length. The kernel function can be viewed as a generalization of the local linear processing of adjacent frame differences used for segmentation. As discussed in Section 3, several different kernels have been proposed.

Calculating **S** requires $O(N^2)$ computations, where N is the number of frames. In practice, there is no reason to calculate similarity matrix values beyond the extent of the kernel, i.e. elements $\mathbf{S}(i,j)$ where $|i-j| > L$. Additionally, because both **S** and **K** are typically symmetric, many computations are redundant. For this reason, we compute only a small portion of **S** near the main diagonal, and the algorithmic complexity is $O(N)$.

3. RELATED WORK

There is a vast literature on video segmentation, including comparative reviews such as [4]. Here, we review only the algorithms used in the experiments of Section 4. Each algorithm is characterized by a specific kernel used to generate a novelty score per (2). For comparison, we emphasize the differences between the kernels in terms of their relative weighting of the elements of **S** to form the novelty scores. Figure 2 graphically depicts the kernels considered here. In each panel, a blank element does not contribute to the corresponding novelty score (i.e. $\mathbf{K}(l,m) = 0$ in (2)). The elements containing solid circles contribute positively to the novelty score ($\mathbf{K}(l,m) > 0$). The elements containing unfilled circles contribute negatively to the novelty score ($\mathbf{K}(l,m) < 0$). Notice that the elements along the main diagonal of **K** align with the main diagonal elements of **S** in the correlation, where $\mathbf{S}(n,n) = D(V_n, V_n) = 0$.

The results of comparing adjacent video frames appear in the first diagonal above (and below) the main diagonal, i.e. the elements $\mathbf{S}(n, n+1)$. Scale space analysis [5] is based on applying a kernel of the form shown in Figure 2(a). The full analysis uses a family of Gaussian kernels of varying standard deviation to calculate a corresponding family of

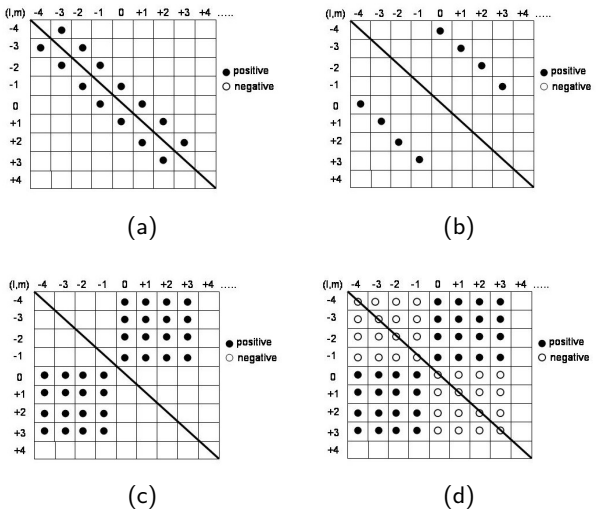

Figure 2: The figure shows different kernels proposed for segment boundary detection via kernel correlation ($L = 4$). The kernels correspond to scale-space analysis (a), diagonal cross-similarity (b), cross-similarity (c), and full similarity (d).

novelty scores. Define the $2L \times 2L$ scale space (SS) kernel as:

$$\mathbf{K}_{SS}^{(\sigma)}(l,m) = \begin{cases} \frac{1}{Z(\sigma)} \exp\left(-\frac{l^2}{2\sigma^2}\right) & |l-m| = 1 \\ 0 & \text{otherwise} \end{cases} \quad . \tag{3}$$

where $Z(\sigma)$ is a normalizing factor[1]. Scale-space analysis was used in [6] for video segmentation.

Pye, *et al.* [7], presented an alternative approach using kernels of the form of Figure 2(b). When centered on a segment boundary, this kernel weights only elements of **S** that compare frames from different segments. This kernel is defined:

$$\mathbf{K}_{DCS}(l,m) = \begin{cases} \frac{1}{2L} & |l-m| = L \\ 0 & \text{otherwise} \end{cases} \quad . \tag{4}$$

We refer to this kernel as the diagonal cross-similarity (DCS) kernel, as the elements of **S** for which $\mathbf{K}_{DCS} > 0$ lie on the L^{th} diagonal above (and below) the main diagonal of **S**. $\mathbf{K}_{DCS}$ has been used in the segmentation systems by Pickering *et al.* [8].

Building on these intuitions, we present two additional kernels for comparison. Including *all* the inter-segment elements implies the kernel of Figure 2(c). This kernel "includes" the DCS kernel, and adds the remaining between-segment (cross-similarity) terms within the kernel's temporal extent. The cross-similarity (CS) kernel is defined:

$$\mathbf{K}_{CS}(l,m) = \begin{cases} \frac{1}{2L^2} & l \geq 0 \text{ and } m < 0 \\ \frac{1}{2L^2} & m \geq 0 \text{ and } l < 0 \\ 0 & \text{otherwise} \end{cases} \quad . \tag{5}$$

For the Euclidean distance of (1), this kernel is precisely the matched filter for an ideal cut boundary in **S**. The inter-

[1]The SS and DCS kernels are easily defined using a single variable, i.e. l, but we use the two variables (l,m) for consistency.

segment (cross-similarity) terms will be maximally dissimilar, while the intra-segment terms will exhibit zero dissimilarity.

The final kernel Figure 2(d) is the full similarity (FS) kernel used in [3], and it includes both between-segment and within-segment terms. This kernel replaces the zero elements in $\mathbf{K}_{CS}$ with negative weights. The negative weights penalize high within-segment dissimilarity:

$$\mathbf{K}_{FS}(l,m) = \begin{cases} \frac{1}{2L^2} & l \geq 0 \text{ and } m < 0 \\ \frac{1}{2L^2} & m \geq 0 \text{ and } l < 0 \\ -\frac{1}{2L^2} & \text{otherwise} \end{cases} . \quad (6)$$

4. EXPERIMENTAL RESULTS

In this section, we examine cut boundary detection using pairwise similarity features. For each frame, we extract a global YUV histogram, and block YUV histograms using a uniform 4×4 grid. We then compute *separate* similarity matrices for the global histogram data, $\mathbf{S}^{(G)}$ and for the block histogram data, $\mathbf{S}^{(B)}$. For the experiments, we follow the approach of [9] and employ supervised binary classification for boundary detection. We use an efficient implementation of KNN classification [10] to label each frame as either a boundary or non-boundary. This algorithm offers potential speedups of a factor of 20 over the naive implementation of KNN, as tested on video data. This allows a consistent boundary detection scheme for comparing the various kernels in the testing below. We concatenate frame-indexed data computed from $\mathbf{S}^{(G)}$ and $\mathbf{S}^{(B)}$ to train and test the KNN classifier to detect cut (abrupt) segment boundaries.

For testing, we use the TRECVID 2002 test data and evaluation software for the shot boundary detection task [1]. This data was viewed as poorer quality than the 2001 data used in [9]. From TRECVID 2002, the average recall and precision for cut detection was 0.86 and 0.84, respectively [8]. The test set consists of almost 6 hours of video containing 1466 cut transitions, per the manual ground truth. For the KNN training, we use cross-validation and train separate classifiers for each video using the remaining videos in the test set. The results are combined for the entire test set. Throughout, $K = 11$.

4.1 Kernel-based features

We present results for two sets of experiments. In the first, we produce novelty features for shot boundary detection corresponding to kernels of extent $L = 2, 3, 4, 5$. For each L, we compute a frame-indexed kernel correlation separately using $\mathbf{S}^{(G)}$ and $\mathbf{S}^{(B)}$ following (2). We concatenate these novelty scores across scale, so that we have four scores for each frame for both the global and the block histogram features. We finally combine this data into a single 8×1 vector to represent each frame n:

$$X_n = \Big[\nu_2^{(G)}(n) \;\; \nu_3^{(G)}(n) \;\; \nu_4^{(G)}(n) \;\; \nu_5^{(G)}(n) \;\; \nu_2^{(B)}(n) \;\; \nu_3^{(B)}(n) \;\; \nu_4^{(B)}(n) \;\; \nu_5^{(B)}(n) \Big]^T .$$

where $\nu_L^{(G)}$ denotes the novelty score computed using $\mathbf{S}^{(G)}$ with kernel width L, and $\nu_L^{(B)}$ denotes the novelty score computed using $\mathbf{S}^{(B)}$. We use the input data $\mathbf{X} = \{X_n : n = 1, \cdots, N\}$ with the manual ground truth to train and test the KNN classifier.

We control the sensitivity of the KNN classification using an integer parameter $\kappa : 1 \leq \kappa \leq K$. If at least κ out of the K nearest neighbors of the vector X_n in the training data are from the "cut" class, we label frame n as a cut and otherwise label it as a non-cut. κ is varied to produce the recall-precision curves of Figure 3 for the FS kernel (circle), the CS kernel ("x"), the SS kernel (square), and the DCS kernel ("+"). The best performance is achieved by the CS and the DCS kernels. As noted above, the CS kernel is the matched filter for the expected pattern produced by segment boundaries in $\mathbf{S}$. Both the CS and DCS kernels emphasize dissimilarity between the segments evident at multiple time scales. The FS kernel performs worst, we believe due to the choice of the Euclidean dissimilarity measure. The FS kernel may be better suited to dissimilarity measures that take positive and negative values such as the cosine similarity measure.

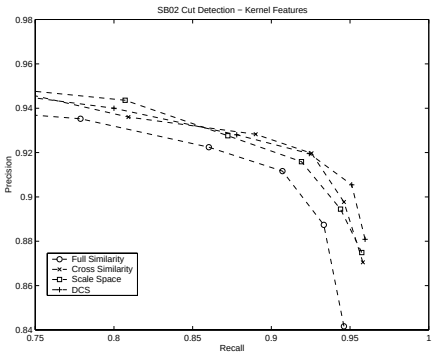

Figure 3: Experimental results for cut detection using the kernel-based features.

4.2 Pairwise similarity features

In the second experiment, we examine performance using the raw pairwise similarity data (without kernel correlation) as input to the KNN classifier. This approach does incur a computational penalty by increasing the dimensionality of the input data $\mathbf{X}$ for classification. Again, we use the separate similarity matrices $\mathbf{S}^{(G)}$ and $\mathbf{S}^{(B)}$. For each kernel, we construct the input feature vectors from those elements of $\mathbf{S}^{(G)}$ and $\mathbf{S}^{(B)}$ that contribute to the corresponding novelty score for $L = 5$. For example, for the SS features (Figure 2(a)) frame n is represented by the column vector:

$$X_n = \Big[\mathbf{S}^{(G)}(n-5,n-4) \;\; \mathbf{S}^{(G)}(n-4,n-3) \;\; \cdots \;\; \mathbf{S}^{(G)}(n+4,n+5) \;\; \mathbf{S}^{(B)}(n-5,n-4) \;\; \mathbf{S}^{(B)}(n-4,n-3) \;\; \cdots \;\; \mathbf{S}^{(B)}(n+4,n+5) \Big]^T$$

and for the DCS features (Figure 2(b)):

$$X_n = \Big[\mathbf{S}^{(G)}(n-5,n) \;\; \mathbf{S}^{(G)}(n-4,n+1) \;\; \cdots \;\; \mathbf{S}^{(G)}(n-1,n+4) \;\; \mathbf{S}^{(B)}(n-5,n) \;\; \mathbf{S}^{(B)}(n-4,n+1) \;\; \cdots \;\; \mathbf{S}^{(B)}(n-1,n+4) \Big]^T$$

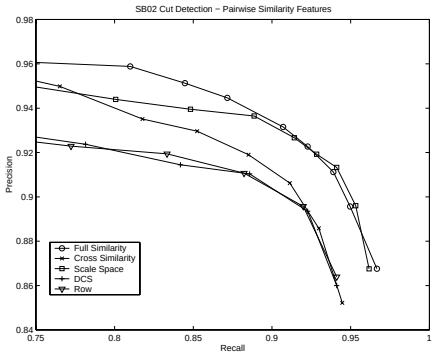

Figure 4: Experimental results for cut detection using the pairwise similarity features.

We only include elements above the main diagonal of $\mathbf{S}$ since $\mathbf{S}$ and $\mathbf{K}$ are symmetric.

The results appear in Figure 4. In this case, the additional similarity information included in the FS data improves performance. The scale-space approach, however outperforms the CS features. This is not surprising since cut detection performance relies largely on first order (adjacent frame) similarity, which is not emphasized by either the CS or DCS features. We also include performance for "row" features (triangle) following [9] where each frame n is represented by the $2L \times 1$ vector:

$$X_n = \Big[\mathbf{S}^{(G)}(n,n-1)\ \ \mathbf{S}^{(G)}(n,n-2)\ \cdots\ \mathbf{S}^{(G)}(n,n-L)$$
$$\mathbf{S}^{(B)}(n,n-1)\ \ \mathbf{S}^{(B)}(n,n-2)\ \cdots\ \mathbf{S}^{(B)}(n,n-L)\Big]^T .$$

Figure 5 shows the curves for all the approaches tested on a single plot. The dashed curves show performance using the kernel-based features, and the solid curves show the performance using the pairwise similarity features. The use of the pairwise similarity data corresponding to the FS kernel shows the best overall performance. All the approaches perform at a high level as input to the KNN classification.

5. CONCLUSION

In this short paper, we have presented preliminary results of an empirical comparison of similarity-based approaches to shot boundary detection. We presented several techniques in a common framework based on kernel correlation of a similarity matrix, and compared them experimentally via testing in combination with supervised classification. Generally, the pairwise similarity features demonstrate superior performance over local correlation-based features. Additionally, we expect that using the pairwise similarity data will benefit the classification of boundaries into subclasses, such as abrupt (cut) versus gradual transitions. There is a complexity tradeoff here, because gradual boundaries require larger values for L which can result in a quadratic increase in the dimensionality of the representation X_n for each frame n. Exploring this tradeoff and integrating gradual transition detection is a key aim of current research. We also intend to continue this study by integrating additional features and broadening the test collection.

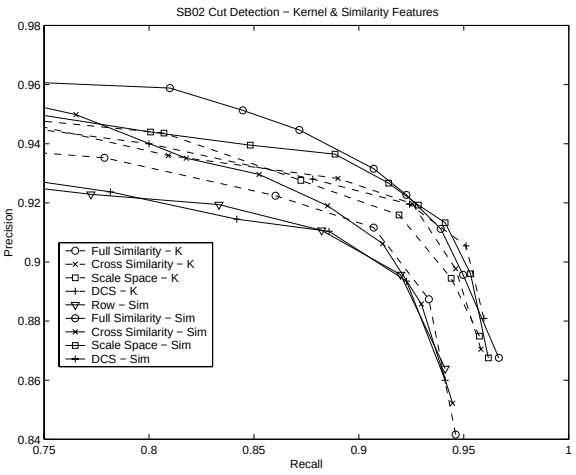

Figure 5: Combined experimental results from Figures 3 and 4.

6. ACKNOWLEDGMENTS

We thank Ting Liu and Andrew Moore for making their KNN software available for these experiments.

7. REFERENCES

[1] A Smeaton and P. Over. The TREC 2002 Video Track Report. *Proc. TREC Video Track*, 2002.

[2] B. Gunsel, M. Ferman, and A. M. Tekalp, Temporal video segmentation using unsupervised clustering and semantic object tracking, *J. Electronic Imaging* **7**(3):592-604, July 1998.

[3] M. Cooper and J. Foote. Scene Boundary Detection Via Video Self-Similarity Analysis. *Proc. IEEE Intl. Conf. on Image Processing*, 2001.

[4] J. Boreczky and L. Rowe. Comparison of video shot boundary detection techniques. *Proc. SPIE Storage and Retrieval for Image and Video Databases*, 1996.

[5] A. Witkin. Scale-space Filtering: A New Approach to Multi-scale Description. *Proc. IEEE ICASSP*, 1984.

[6] M. Slaney, D. Ponceleon, and J. Kaufman. Multimedia edges: finding hierarchy in all dimensions. *Proc. ACM Multimedia*, pp. 29–40, 2001.

[7] D. Pye, N. Hollinghurst, T. Mills, and K. Wood. Audio-visual Segmentation for Content-Based Retrieval. *Proc. Intl. Conf on Spoken Language Processing*, 1998.

[8] M. Pickering, D. Heesch, *et al.*. Video Retrieval using Global Features in Keyframes. *Proc. TREC Video Track*, 2002.

[9] Y. Qi, A. Hauptman, and T. Liu. Supervised Classification for Video Shot Segmentation. *Proc. IEEE Intl. Conf. on Multimedia & Expo*, 2003.

[10] T. Liu, A. Moore, and A. Gray. Efficient Exact k-NN and Nonparametric Classification in High Dimensions. *Proc. Neural Information Processing Systems*, 2003.