Video Segmentation via Temporal Pattern Classification

Matthew Cooper, *Member, IEEE,* Ting Liu, and Eleanor Rieffel

Abstract

We present a general approach to temporal media segmentation using supervised classification. Given standard low-level features representing each time sample, we build intermediate features via pairwise similarity. The intermediate features comprehensively characterize local temporal structure, and are input to an efficient supervised classifier to identify shot boundaries. We integrate discriminative feature selection based on mutual information to enhance performance and reduce processing requirements. Experimental results using large-scale test sets provided by the TRECVID evaluations for abrupt and gradual shot boundary detection are presented, demonstrating excellent performance.

I. Introduction

Despite continued progress in video shot boundary detection, it remains a focus of active research in multimedia analysis. Because shots provide the most natural organizational unit for video above the frame, shot segmentation enables hierarchical processing of content in video management systems. The need for effective and efficient multimedia management has been exacerbated by recent trends. The confluence of decreasing storage costs, increasing processing power, and the growing availability of broadband data connections is producing rapid growth in both the size and number of personal and institutional video repositories.

A great deal of current video analysis research focuses on automatically extracting semantics from multimedia within the broader context of multimedia information retrieval. Semantic video annotation and video retrieval are also the focus of the highly successful TRECVID evaluations [1], [2]. Shot boundary detection is a core task at TRECVID, and shots serve as the units for both higher-level semantic annotation and retrieval tasks.

Automatic and semi-automatic approaches to extracting semantics from visual data in the absence of textual descriptors are critically important to content management. Multimedia metadata standards such as MPEG-7 are emerging to support multimedia management [3]. The rate at which content is now being generated precludes metadata creation with substantial manual processing. Many current systems operate at the shot-level following an initial temporal segmentation. This is desirable for both computational efficiency and the extraction of semantics associated with some temporal duration.

M. Cooper and E. Rieffel are with FX Palo Alto Laboratory, Palo Alto, CA. T. Liu is with Google, Inc. Mountain View, CA. For correspondence, email cooper@fxpal.com.

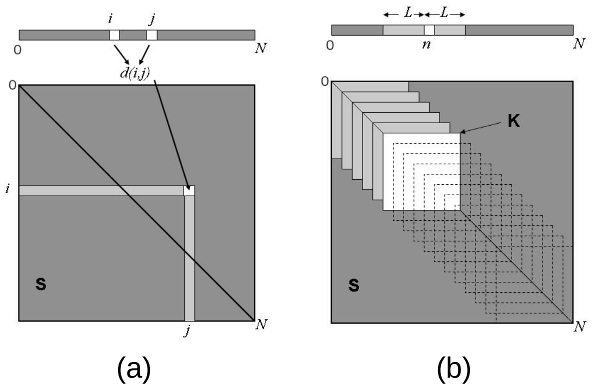


Fig. 1. Panel (a) shows the similarity matrix embedding. Panel (b) depicts the kernel correlation of (1).

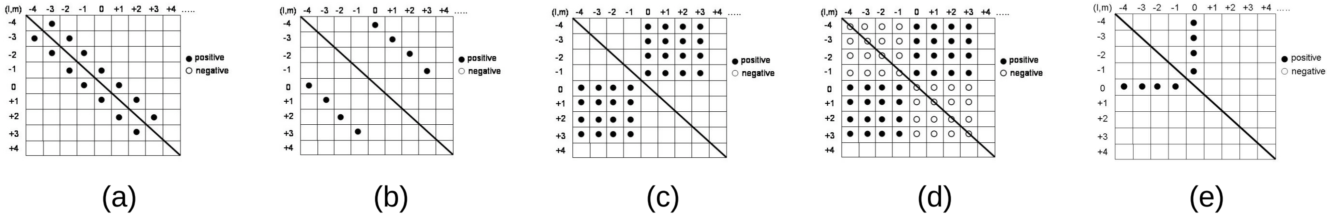


Fig. 2. The figure shows different kernels proposed for shot boundary detection via kernel correlation ($L = 4$). The kernels correspond to (a) scale-space analysis, (b) diagonal cross-similarity, (c) cross-similarity, (d) full similarity, and (e) row.

Generic video segmentation systems are implemented in three stages. First, a set of time-indexed low-level features are extracted. For simplicity, we assume that the time index corresponds to the frame index, although this is not required. Next, features corresponding to pairs of frames are compared. In the simplest case, the first difference is computed by comparing adjacent frames using an appropriate distance measure. Finally, the differences between frames are analyzed to detect boundaries, usually with thresholds. Often, thresholding methods are customized for specific low-level descriptors or inter-frame similarity measures, thus making it difficult to associate performance variations with the choice of features or similarity measures.

We generalize inter-frame analysis in two steps: similarity comparison and kernel correlation. We represent each frame n with a low-level feature vector V_n. Given a similarity measure d quantifying the similarity between pairs of feature vectors, we embed the similarity between every possible pair of frames in the similarity matrix as in Figure 1(a): $\mathbf{S}(i, j) = d(V_i, V_j)$. Thus, the number of rows and columns of $\mathbf{S}$ is the total number of frames, N, in the source video.

Abrupt shot boundaries exhibit a distinct pattern in the similarity matrix. Frames in visually coherent shots have low within-shot dissimilarity (high similarity). Frames from two such shots that are adjacent in time generally show high between-shot dissimilarity (low similarity). This produces a checkerboard along the main diagonal of $\mathbf{S}$ whose crux is the diagonal element corresponding to the boundary frame. This observation has motivated matched filter approaches to boundary detection, which we refer to as kernel correlation. The matched filter is a square kernel matrix, $\mathbf{K}$, that represents the appearance of an ideal boundary in $\mathbf{S}$ [4]. To produce a quantitative frame-indexed novelty score, we correlate $\mathbf{K}$ along the main diagonal of $\mathbf{S}$:

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

Here, $\mathbf{K}$ is $2L \times 2L$. This process is illustrated in Figure 1(b). By varying the maximal lag L, the novelty score can be tuned to detect boundaries between segments of a specific minimum duration. This approach has also been successfully adapted to audio and music segmentation [5], as well as text segmentation [6].

Maxima in the correlation of (1) correspond to locally novel frames and are good candidate shot boundaries. We use these measures to form an intermediate-level feature set to comprehensively represent local temporal structure, performing boundary detection via supervised non-parametric classification. We thus formulate temporal segmentation as *temporal pattern classification*. We use the k-nearest-neighbor (kNN) classifier, which has been extensively studied and can be related to fundamental algorithm-independent bounds on performance. The application of kNN classification here necessitates an efficient implementation which can handle test and training sets of tens of thousands of frames.

Throughout, we represent local temporal structure using pairwise similarity data. Increasing the extent over which similarities are computed provides a more robust characterization of local temporal structure. Performance in turn improves with the expense of increased computation. We integrate feature selection to reap these performance gains while minimizing additional computation. We employ mutual information as the criterion for feature selection following Vasconcelos *et al.* [7], [8], so that the resulting system makes no parametric or otherwise limiting assumptions about the input content. Section III describes the technical details of our system. We review extensive experimental results using manually labeled test data from the TRECVID evaluations [1], [2] in Section IV. We conclude with a brief summary and describe possible directions for future research.

II. Related Work

There is an abundant literature on shot boundary detection that is impossible to review exhaustively. Comparative reviews include [9] and [10]. State of the art systems exhibit excellent abrupt transition detection performance, while gradual transition detection remains relatively poor. The principal challenge is distinguishing amongst transition effects (e.g. fades), object motion, and camera motion using low-level frame features.

Several design choices can be used to partially categorize existing systems. Systems either operate on uncompressed streams, processing every frame, or compressed streams, using features directly extracted from the MPEG sequence. While performance is degraded in the latter case, time complexity can be reduced by factors of up to 70 [11], [12], [13], [14]. There are also numerous choices of low-level features to represent each time sample. Many methods model global (entire frame) pixel intensities directly, image sub-blocks, or both using statistical measures or histograms. Motion compensated features have been proposed, as well as more specialized features from the computer vision literature including edge or texture features, and estimates of object or camera motion. Finally, specialized features may include detectors of specific objects or phenomena such as faces or camera flashes.

Common measures for comparing frames' histograms include the L-1 and euclidean (L-2) distance. We use the chi-squared similarity [15]:

$$\mathbf{S}(i,j) = d(V_i, V_j) = \frac{1}{2}\sum_k \frac{(V_i(k) - V_j(k))^2}{V_i(k) + V_j(k)} \quad . \tag{2}$$

Here k indexes the feature vector's elements. This choice produces non-negative similarity matrices in which the main diagonal is zero: $\mathbf{S}(n,n) = 0$. Boundary detection is typically achieved using either statistical approaches or threshold-based schemes. The former include Bayesian methods [16], statistical classifiers [17], and hypothesis testing [18]. The latter include simple thresholds for difference scores, multiple thresholds [19], [20] or adaptive data-dependent thresholds [12].

A. Systems using kernel correlation

Several recent algorithms are characterized by a specific kernel used to generate a novelty score per (1). We emphasize the differences between the kernels in terms of their relative weighting of elements of $\mathbf{S}$ as in Figure 2. In each panel, a blank element does not contribute to the corresponding novelty score (i.e. $\mathbf{K}(l,m) = 0$ in (1)). 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 $\mathbf{K}$ align with the main diagonal elements of $\mathbf{S}$ in the correlation, where $\mathbf{S}(n,n) = d(V_n, V_n) = 0$. Below, we assume a kernel of size $2L \times 2L$, so that $-L \leq l, m \leq L-1$.

The results of comparing adjacent video frames appear in the first diagonal above (and below) the main diagonal, i.e. the elements $\mathbf{S}(n, n \pm 1)$. Scale space (SS) analysis [21] is based on applying a kernel of the form shown in Figure 2(a). Scale-space analysis was used in [22] for video segmentation. Pye, *et al.* [23], 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 $\mathbf{S}$ that compare frames from different segments. This kernel is:

$$\mathbf{K}^{(L)}_{DCS}(l,m) = \begin{cases} \frac{1}{2L} & |l-m| = L \\ 0 & \text{otherwise} \end{cases} \quad . \tag{3}$$

We refer to this kernel as the diagonal cross-similarity (DCS) kernel. $\mathbf{K}_{DCS}$ has been used in the segmentation system in [20] whose results appear in the experimental section.

Weighting all the inter-segment elements implies the kernel of Figure 2(c), which we denote the cross-similarity

(CS) kernel. The CS kernel is precisely the matched filter for an ideal abrupt boundary in $\mathbf{S}$. The kernel in Figure 2(d) is the full similarity (FS) kernel used in [4], and it includes both between-segment and within-segment terms. This kernel replaces the zero elements in the CS kernel with negative weights to penalize high within-segment dissimilarity:

$$\mathbf{K}_{FS}(l,m) = \begin{cases} \frac{1}{2L^2} & l \geq 0 \text{ and } m < 0 \\ \frac{1}{2L^2} & m \geq 0 \text{ and } l < 0 \\ -\frac{1}{2L^2} & \text{otherwise} \end{cases} . \quad (4)$$

Figure 2(e) shows the row (ROW) kernel used by [17]. This kernel weights only comparisons between the current frame and previous frames: $\mathbf{S}(n, n-l)$.

B. Systems using statistical boundary detection

A variety of statistical techniques have been used in shot boundary detection. Zhang, *et al.* [19] used a Gaussian model to determine a threshold for boundary detection. Vasconcelos and Lippman [16] presented a Bayesian formulation of abrupt shot boundary detection, using Weibull and Erlang priors for the shot duration and activity histograms to characterize shot boundaries. Conditional models for the activity histograms for boundary and non-boundary frames are estimated using Gaussian mixture models. Hanjalic [18] used statistical detection theory to adaptively determine a threshold to optimize performance, integrating motion compensated features and extending to gradual transitions. This work used a Poisson prior for shot duration, and parametric templates for the class-conditional models for discontinuity (dissimilarity) features. It also included a “detector cascade” approach, similar to that employed in our system, to handle detection of multiple types of shot transitions. Qi, *et al.* [17] applied supervised classification to shot boundary detection. Their intermediate features correspond to the ROW kernel of Figure 2(e). They used the same efficient kNN classifier and also a cascade of binary classification steps to detect both abrupt and gradual transitions. This work provides the core of our system. We build on this foundation with enhanced intermediate similarity features for improved classification as demonstrated in Section IV. Our intermediate representation makes specialized motion analysis as in [17] unnecessary. We extend our previous work in [24] to perform gradual boundary detection. We also incorporate discriminative feature selection to enhance performance.

C. Top performing systems at TRECVID

Several interesting high performance systems were presented in the 2004 TRECVID evaluation [2], including a version of the system presented in this paper [6]. Hoashi *et al.* [14] use a system processing MPEG features directly with motion compensation to improve the temporal resolution. They use pixel and edge-based low-level features and detectors for specific gradual transition types such as wipes and dissolves, as well as a (camera) flash detector. Volkmer *et al.* [25] used frame-indexed block HSV color histogram features and a local similarity ranking-based method, which ranks frames in terms of similarity to the current frame. For improved detection of gradual transitions, they use a larger set of inter-frame similarities and adaptively threshold the ratio between the average similarity between pairs of previous frames, and pairs of future frames. Yuan *et al.* [26] used several separate components: a fade detector, a cut detector, and a gradual transition detector based on a finite state automaton. These processed several low-level feature modalities and were integrated using a cascade. Petersohn [27] combined adaptive thresholding with specialized flash detection, wipe detection and motion analysis.

We formulate shot boundary detection as classification of frame-indexed features describing local temporal structure. Unlike most related work based on statistical detection, the classifier makes no parametric assumptions about the content. In contrast to other top performing systems from TRECVID, we forego the use of both multiple specialized detectors for effects such as flashes and complex low-level features such as estimated motion. Additionally, we avert the need for thresholds. Rather, we use a supervised classifier with a single parameter to trade off false positive and false negative classification error.

III. System Description

In this Section, we present a system for temporal multimedia segmentation with minimal specialized processing. We develop generic intermediate features to represent local temporal structure using standard frame histograms and pairwise inter-frame similarity. These intermediate features are classified to detect shot boundaries. This system represents a basic framework in which any combination of low-level features and similarity measures can be used. Additionally, the approach can be adapted to any other modality or time-ordered data collection. Here, we present and validate the system architecture for temporal video segmentation.

A. Building intermediate features

We detect scene boundaries by quantifying the similarity between pairs of video frames. First, low-level features are computed from each frame. We use YUV color histograms, which are a simple and common feature parametrization [28]. We compute 32-bin global frame histograms, and 8-bin block histograms using a 4×4 uniform spatial grid for each channel[1]. Denote the frame-indexed histogram feature data by $\mathbf{V} = \{V_n : n = 1, \cdots, N\}$ for N frames. The chi-squared similarity between frames is embedded in $\mathbf{S}$ as in Figure 1(a).

Naively calculating $\mathbf{S}$ requires $O(N^2)$ computations. We use $\mathbf{S}$ to construct a frame-indexed set of intermediate features based on the kernels of Section II. The lag parameter L determines the size of the kernel, and in practice $L \ll N$. Thus, we do not calculate elements of the similarity matrix values beyond the extent of the kernel, i.e. elements $\mathbf{S}(i,j)$ where $|i-j| > L$. Additionally, because both $\mathbf{S}$ and $\mathbf{K}$ are typically symmetric, many computations

[1] Thus, the global histogram data has dimensionality 96 and the block histogram data has dimensionality 384 for each frame.

are redundant. We compute only a small portion of $\mathbf{S}$ near the main diagonal, reducing the complexity to $O(N)$. We generate similarity matrices $\mathbf{S}^{(G)}$ and $\mathbf{S}^{(B)}$ corresponding to the global and block color histogram features, respectively.

Next, we construct intermediate features $\mathbf{X} = \{X_n : n = 1, \cdots, N\}$ based on $\mathbf{S}$, so that X_n represents the local temporal structure around frame n. For any kernel in Section II, we construct feature vectors concatenating those elements of $\mathbf{S}$ that contribute to the corresponding correlation. That is, X_n contains the elements of $\mathbf{S}$ that are multiplied by non-zero weights $\mathbf{K}(l, m)$ in (1). For example, for the SS features (Figure 2(a)) and $L = 5$, frame n is represented by the column vector:

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

We eliminate elements from the main diagonal (always zero) and remove duplicates due to the symmetry of the similarity matrices.

B. Efficient kNN Classification

The intermediate features $\mathbf{X}$ are classified to detect shot boundaries using a binary kNN classifier. The kNN classifier has two appealing properties. First, it is non-parametric, making no limiting assumptions about the statistics of the transition classes expressed in our intermediate features. Secondly, the asymptotic error rate of the 1-nearest-neighbor classifier does not exceed twice the Bayes (minimum) rate [29]. This algorithm-independent bound gives us a basis for comparatively assessing different choices for intermediate features for shot boundary detection.

The principal disadvantage of the kNN is its computational complexity. To reduce complexity, a number of efficient implementations have been devised. The accelerated version we employ uses metric tree data structures to achieve efficient spatial search [30]. Furthermore, we use an implementation designed for unbalanced problems in which one class is far more frequent than the rest. In our case, the number of frames that are not part of a transition is substantially greater than the number of transition frames.

In experimental testing using video data, speedups between a factor of 20 and 30 in run time over the naive implementation of kNN have been recorded [30]. This acceleration is crucial in the present context. We perform classification in two sequential binary steps, as depicted in Figure 3. In each step, we produce training sets from approximately six hours of video labeled according to manual ground truth. Typically, we randomly discard 90% of the non-transition frames, resulting in a labeled training set of about 60,000 examples. Our test set is comprised of twelve thirty minute videos, each containing approximately 54,000 frames with corresponding intermediate features.

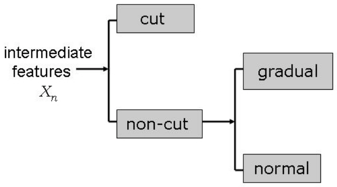


Fig. 3. The figure depicts the classification process. In the first step, cut boundaries are detected. In the second step, the non-cut frames are classified as either gradual transition frames or non-transition (normal) frames.

C. Information-theoretic feature selection

We consider detection of two types of shot boundaries: abrupt (cut) and gradual. Intuitively, we expect that specific inter-frame comparisons will be of varying relevance to detecting these two classes. Numerous existing systems using solely adjacent frame comparisons of the form $\mathbf{S}(n, n \pm 1)$ can detect abrupt transitions reasonably well. Robust gradual transition detection requires analysis of frames over a greater temporal neighborhood. We greedily select elements of the intermediate feature vectors $\mathbf{X}$ that best discriminate among the transition classes. To determine feature subsets, we calculate mutual information measures between the elements of the intermediate feature vectors and the corresponding class labels from our training sets.

The mutual information between two random variables quantifies the information that they share about one another. We focus on the mutual information $I(X(i); Y)$ where $X(i)$ is the i^{th} element of the intermediate feature vectors $\mathbf{X}$, and Y denotes the class label[2]. This measure is referred to as marginal diversity and used for greedy feature selection in [7]. The greedy approach ignores inter-feature redundancies, and the resulting feature subset is not generally maximally informative. To account for inter-feature dependencies, more complicated measures must be calculated. Although mutual information naturally extends to this case, its direct application becomes computationally intractable. Denote the currently selected features by the set $\mathbf{X}^C = \{X^{(1)}, \cdots, X^{(C)}\}$. Then, the next feature selected is

$$X^{(a)} = \underset{X(i) \notin \mathbf{X}^C}{\text{ArgMax}}\, I(X(i); Y | \mathbf{X}^C) \quad .$$

These forms of the mutual information become increasingly difficult to calculate as $\mathbf{X}^C$ grows. The main difficulty is estimation of quantities such as $P(X(i)|X^{(1)}, \cdots, X^{(C)})$, which quickly becomes computationally prohibitive and unreliable without massive amounts of training data. As a result, we seek approximations, and focus on second order terms of the form $I(X(i); Y|X^{(c)})$.

The approximation problem is studied in [8]. We approximate the relevant forms of the mutual information by neglecting higher order terms. For simplicity, we assume *ℓ-decomposability* [8] for $\ell = 1$. The assumption is:

[2] To make the notation explicit, $X_m(i)$ is the i^{th} element of the feature vector associated with the m^{th} labeled frame in the training set. If $\mathbf{X}$ is a $M \times P$ matrix, then $1 \leq i \leq P$ and $X(i)$ appears in the i^{th} column of the matrix $\mathbf{X}$ of training data. We can also think of each observation X_m as an $1 \times P$ vector.

$$I(X(i);Y|\mathbf{X}^C) = I(X(i);Y) + \sum_{X^{(c)} \in \mathbf{X}^C} \Big[I(X(i);X^{(c)}|Y) - I(X(i);X^{(c)}) \Big] \quad . \tag{6}$$

Therefore, we repeatedly select the feature $X^{(a)}$ to add to a previously selected set $\mathbf{X}^C$ such that,

$$X^{(a)} = \underset{X(i) \notin \mathbf{X}^C}{\mathrm{ArgMax}} \Bigg(I(X(i);Y) + \sum_{X^{(c)} \in \mathbf{X}^C} \Big[I(X(i);X^{(c)}|Y) - I(X(i);X^{(c)}) \Big] \Bigg) \quad . \tag{7}$$

The approximation implies that the only critical feature interdependencies are pairwise. While this is not wholly accurate, it significantly improves our resulting feature subset, as demonstrated experimentally below. The framework of ℓ-decomposability can be used to systematically study the tradeoff between the computational complexity of assembling $\mathbf{X}^C$ and the corresponding performance gains in boundary detection. We focus here on the approximation of (7) which underlies a greedy procedure for feature selection outlined in detail in [8].

We apply the kNN in two steps per Figure 3. In each step, we use different training sets. Thus, we perform feature selection separately using the two training sets, producing a feature subset optimized for each classification step. The training and test data sets are then projected to the appropriate lower-dimensional subspace prior to classification.

IV. Experimental Results

A. Data description

In this section we present experimental results to validate the general approach and compare a number of specific system configurations. For testing, we use the data and evaluation protocol of TRECVID 2004 shot boundary determination task [2]. The test data is approximately six hours of broadcast news data produced by CNN and ABC from 1998. A manual ground truth segmentation is also provided in which shot boundaries are labeled as either abrupt (cut) or gradual transitions. The test data contains 618,409 total video frames with 2,774 cut transitions and 2,031 gradual transitions of various types and durations. The available training data for the evaluation is the 2003 TRECVID test set [1] and ground truth segmentation. From this data, we randomly discard 90% of the non-transition frames.The resulting training set contains 63,822 labeled samples. Of the training data, 2,489 frames are labeled cut transitions, 22,074 frames are labeled gradual transitions, and the rest are non-transition frames. The training data is used to build two separate training sets corresponding to our two step classification process. In the

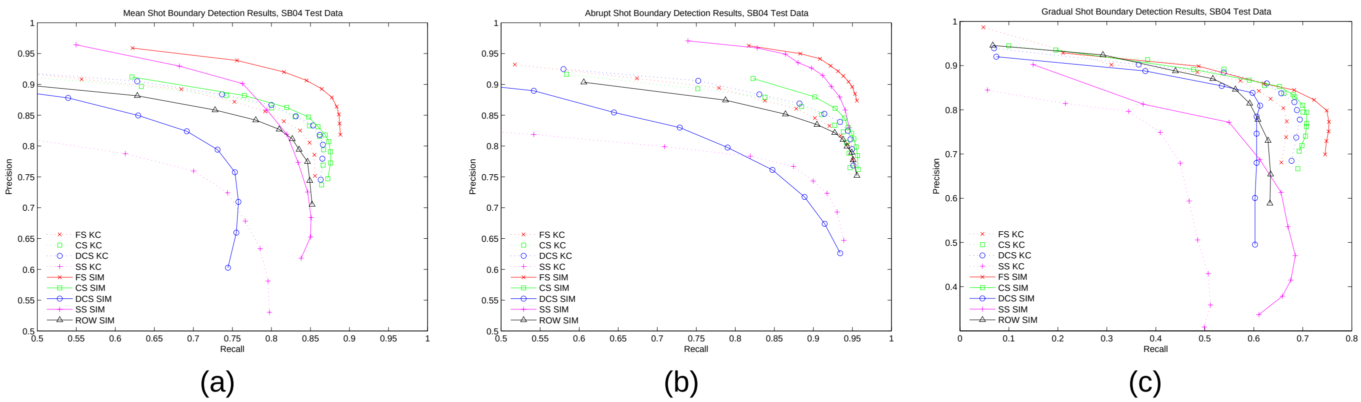


Fig. 4. Mean experimental results for shot boundary detection (a), abrupt (cut) boundary detection (b) and gradual boundary detection (c).

first set, cuts are labeled positively and all other frames are labeled negatively. In the second training set, cuts are discarded, gradual transition frames are labeled positively, and the non-transition frames are labeled negatively.

B. Experimental setup

For each frame, we extract a global YUV histogram, and block YUV histograms to compute separate (partial) similarity matrices $\mathbf{S}^{(G)}$ and $\mathbf{S}^{(B)}$ as before. We form intermediate features for classification using elements of $\mathbf{S}^{(G)}$ and $\mathbf{S}^{(B)}$. We control the sensitivity of the classifier using the integer parameter $\kappa : 1 \leq \kappa \leq k$. If at least κ out of the k nearest neighbors of the test vector X_n in the training data are from the "transition" class, we label frame n as a transition and otherwise label it as a non-transition. κ thus determines how the system trades off false positive versus false negative classification errors. The performance curves below are produced by varying κ between 1 and k. We use the same value of κ for both classification steps, although this is unlikely to be optimal. The kernel width L is set to tradeoff performance and computational complexity. As previously noted, increasing L produces features $\mathbf{X}$ that better represent local temporal structure. Generally, larger values of L provide improved gradual boundary detection performance, with additional computational cost. Throughout, $k = 11$.

The only post-processing is the application of simple temporal heuristics. We require detected transitions to be separated by at least 60 frames (2 seconds). If multiple transitions are detected within a 60 frame interval, we retain the transition with the most positively labeled frames among its nearest neighbors breaking ties arbitrarily. We also require gradual transitions to have a minimum duration of 11 frames. For evaluation, we use the common figures of merit of precision and recall [2]:

$$\begin{aligned}\text{Precision} &= \frac{\#(\text{Boundaries correctly detected})}{\#(\text{Total boundaries detected})}\;, &(8)\\ \text{Recall} &= \frac{\#(\text{Boundaries correctly detected})}{\#(\text{Total ground truth boundaries})}\;. &(9)\end{aligned}$$

During testing, we monitored the required computation time. The systems below process 90 dimensional intermediate features for classification using training sets with between 55,000 and 65,000 frames. Thus, the run time of the various systems below are all similar, operating at about twice real-time. That is, the segmentation requires compute time equal to twice the duration of the input video [3]. Decoding the MPEG stream to extract individual frames and their corresponding histogram features requires slightly more time than the two kNN classification steps combined.

C. *Inter-frame similarity versus kernel correlation*

The goal of the first set of experiments is to compare intermediate features based on the various kernels of Section II. First, we examine performance using the raw pairwise similarity data (without kernel correlation) as input to the kNN classifier. We define the intermediate features $\mathbf{X}$ as in Section III using raw chi-squared similarity values for $L = 5$. These features form the training and testing sets for classification. The results appear as solid curves in Figure 4 for the FS kernel ($\times$, "FS SIM"), the CS kernel ($\Box$, "CS SIM"), the SS kernel ($+$, "SS SIM"), the ROW kernel ($\triangle$, "ROW SIM"), and the DCS kernel ($\circ$, "DCS SIM"). The additional information in the FS features produces the best performance. These results exhibit a clear tradeoff between the dimensionality of the intermediate representation $\mathbf{X}$ and segmentation performance. Results for all experiments are summarized in Table I.

For comparison, we produce intermediate features using (1) and kernels with maximal lag $L = 2, 3, 4, 5$. For each L, we compute kernel correlations separately using $\mathbf{S}^{(G)}$ and $\mathbf{S}^{(B)}$. We concatenate these novelty scores across scale to construct the vector X_n:

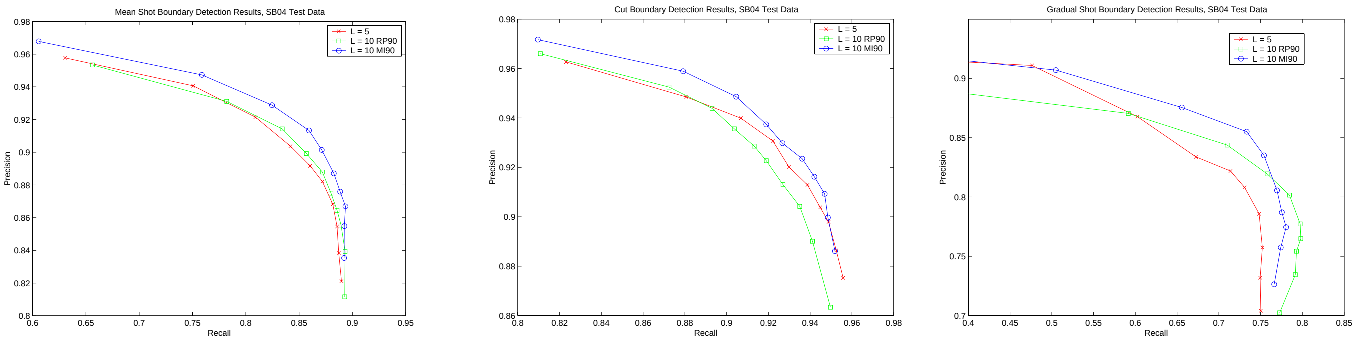
$$X_n = \left[\nu_2^{(G)}(n) \quad \cdots \quad \nu_5^{(G)}(n) \quad \nu_2^{(B)}(n) \quad \cdots \quad \nu_5^{(B)}(n) \right] .$$

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 $\mathbf{S}^{(B)}$. In this case, the size of the intermediate vectors X_n is the same for all kernels.

The results appear as dashed curves in Figure 4 for the FS kernel ($\times$, "FS KC"), the CS kernel ($\Box$, "CS KC"), the SS kernel ($+$, "SS KC"), and the DCS kernel ($\circ$, "DCS KC"). Among these systems, the best performance is achieved by the CS and the DCS kernels. As noted previously, the CS kernel is the matched filter for a cut segment boundary in $\mathbf{S}$. Both the CS and DCS kernels emphasize dissimilarity between segments at multiple time scales.

Panel (b) shows performance for abrupt boundary detection. Kernels that emphasize similarity comparisons between adjacent frames, $\mathbf{S}(n, n \pm 1)$, detect abrupt transitions well, and the FS SIM system performs best of all variations. Panel (c) shows gradual transition detection performance. Recall is generally lower than for abrupt

[3] More time complexity details appear in [6], the machine used for testing has an Athlon 64 3500+ processor. These results apply to the systems denoted FS SIM, L=5, L=10RP90, and L=10MI90.



(a) (b) (c)

Fig. 5. Panel (a) shows mean results for three-class shot boundary determination. The L=5 results correspond to a maximum temporal kernel lag of $L = 5$ frames. The L=10 RP90 results correspond to a maximal temporal lag of $L = 10$. These features are randomly projected down to 90 dimensions from 380. The L=10 MI90 results correspond to a maximal temporal lag of 10. These features are ranked using information theoretic measures, and the top 90 are selected. Cut boundary and gradual boundary detection results appear in panels (b) and (c), respectively.

transitions, which reflects the relative difficulty of this task. The kernels that combine larger lag comparisons between non-adjacent frames perform best. Again, the FS SIM system provides the best performance, due to its more complete representation of local temporal structure.

D. *Segmentation with feature selection*

We conclude from the previous section that building intermediate features corresponding to the FS kernel provides excellent overall performance. The performance of the various systems demonstrate that adding information to the intermediate features generally benefits performance. At the same time, larger intermediate feature vectors increase the computational requirements of classification. The goal of this section is to explore this tradeoff between performance and complexity using feature selection. We first increase the lag parameter to L=10, producing 380 dimensional features $\mathbf{X}$ for the FS kernel. We use feature selection to then reduce dimensionality, while improving performance.

We compare two approaches to feature selection using the “FS SIM” system of the previous section as a baseline. For the baseline, the intermediate features $\mathbf{X}$ are generated with L=5. We select feature subsets from FS similarity features $\mathbf{X}$ generated with L=10. The first feature selection method is random projection (RP) [31]. This approach generates a subspace for projection randomly with the constraint that it be orthogonal. The method is proven to preserve distances in the original high-dimensional space, and thus naturally complements nearest-neighbor classification. Figure 5 shows the corresponding results using RP (□ “L=10 RP90”). The second set of results use information-theoretic measures to select 90 features greedily as in subsection III-C (○ “L=10 MI90”). The curves for the baseline system are denoted (× “L=5”).

Figure 5 shows that both systems using feature selection outperform the original L=5 system using the same feature dimensionality for classification. The system using information-theoretic feature selection performs best of the three. The L=5 system outperforms L=10 RP90 in abrupt boundary detection (panel (b)), while the opposite is true for gradual boundary detection (panel (c)). This reaffirms our intuitions. First, information critical to cut boundary detection is in the pairwise similarities closest to the boundary frame, while the extra information in the L=10 features is superfluous. Random projection mixes these features which degrades cut detection performance. All the features are beneficial for gradual boundary detection, and mixing them does not hurt accuracy.

The L=10 MI90 system better exploits the discriminative power of the L=10 features. It selects the important features for cut detection by design, and selects combinations of complementary features for gradual boundary detection. In this way, it outperforms the L=10 RP90 system overall. The selected features appear in the dark elements of the kernels of Figure 6. The top row shows features selected for abrupt boundary detection in the first classification step. These feature subsets emphasize similarity comparisons near the current frame (center of the kernel). In contrast, the subsets on the bottom row for gradual boundary detection emphasize comparisons between frames farther from the current frame. The block histogram comparisons are better represented in all subsets. Visually, the feature subsets in the first and second classification steps are largely disjoint. This supports the use of feature selection to reduce complexity by separately optimizing the two steps.

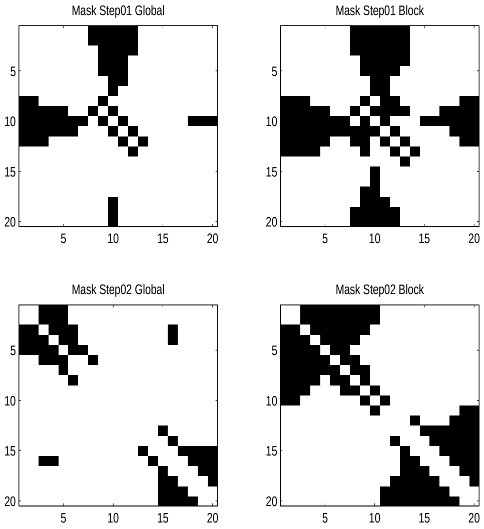


Fig. 6. The figure depicts the kernels selected using the greedy information theoretic procedure of Section III. The feature subsets used for abrupt boundary detection are shown in the top row for global image histograms (top left) and block histograms (top right). The feature subsets used for gradual boundary detection appear in the bottom row for global image histograms (bottom left) and block histograms (bottom right). The kernels are symmetric for visualization although only half the elements are used in practice.

Table I shows the results using information theoretic feature selection for three additional system variations. L=10 MD90 uses greedy feature selection following [7]. This approach ignores redundancies in the selected feature subsets, and we see the resulting overall performance is relatively poor. L=10 MI25 and L=10 MI45 show the results of using the first 25 and 45 features, respectively from the 90 element feature subset from L=10 MI90. The results indicate that the L=10 MI45 system approximately matches the overall performance of the L=10 RP90 system

TABLE I

The table documents performance of the various systems tested for three class shot boundary detection. P and R denote precision and recall, respectively. The F1 columns show the F-score which is defined as $F1 = (2 \cdot P \cdot R)/(P + R)$. Each row shows the results using the value of κ that maximizes the overall F-score for the corresponding system. The system denoted "FS SIM" in Figure 4 is the system denoted "L=5" in the Table and in Figure 5. The row denoted "DCS THRESH" shows the top result from TRECVID 2004 using DCS kernel features and adaptive thresholds. The row denoted "TV MEAN" shows the mean results from TRECVID 2004. The row denoted "TV BEST" shows the top overall run from TRECVID 2004.

Experimental results (best runs for each)									
	MEAN			ABRUPT			GRADUAL		
SYS	R	P	F1	R	P	F1	R	P	F1
ROW SIM	0.8267	0.8115	0.819	0.9381	0.8105	0.8696	0.5916	0.8146	0.6854
SS KC	0.7439	0.7241	0.7339	0.8746	0.7669	0.817	0.468	0.594	0.5233
SS SIM	0.763	0.901	0.826	0.8642	0.949	0.905	0.5495	0.772	0.642
DCS KC	0.8533	0.8334	0.8432	0.9341	0.8391	0.8841	0.6828	0.8172	0.7440
DCS SIM	0.731	0.7941	0.7612	0.7902	0.7976	0.7939	0.6058	0.7846	0.6837
CS KC	0.8489	0.8327	0.841	0.9273	0.8337	0.8780	0.6835	0.8295	0.7495
CS SIM	0.8475	0.8471	0.847	0.9399	0.8453	0.890	0.6524	0.8528	0.7392
FS KC	0.8369	0.8252	0.831	0.9206	0.8327	0.874	0.6602	0.8038	0.725
L=5	0.8718	0.8821	0.87692	0.9387	0.9129	0.9256	0.7307	0.8082	0.7675
L=10 RP90	0.8718	0.8879	0.8798	0.9132	0.9286	0.9208	0.7845	0.8016	0.7930
L=10 MD90	0.8596	0.8417	0.8506	0.9387	0.8428	0.8882	0.6926	0.8386	0.7586
L=10 MI25	0.8439	0.8908	0.8667	0.9233	0.9169	0.9201	0.6764	0.8235	0.7427
L=10 MI45	0.8583	0.8975	0.8775	0.9224	0.925	0.9237	0.723	0.8311	0.7733
L=10 MI90	0.8712	0.9014	0.8860	0.9267	0.9298	0.9282	0.754	0.8351	0.7925
DCS THRESH	0.814	0.846	0.830	0.895	0.885	0.890	0.644	0.748	0.692
TV MEAN	0.725	0.727	0.709	0.831	0.763	0.776	0.502	0.578	0.565
TV BEST	0.884	0.896	0.890	0.928	0.931	0.929	0.792	0.82	0.806

using half as many features.

Table I also includes selected results from TRECVID 2004. We include the best run by the system in [20] which uses DCS kernel correlation features in combination with multiple thresholds to provide a comparative baseline for thresholding methods. Comparing this result with the most similar variant of our system, "DCS KC", we conclude that the kNN improves performance over thresholding, especially for gradual boundary detection. The mean results for all evaluated systems appear in the row denoted "TV MEAN". The best single run in the evaluation ranked by mean F-score was produced by the system in [26] and is denoted "TV BEST." Our system denoted "L=10MI90" achieves virtually the same performance. The Table suggests that further optimization may be possible by using fewer features for abrupt boundary detection, and a larger feature set for gradual boundary detection.

V. Conclusion

In this paper, we have presented a general system for temporal media segmentation with two central components: pairwise similarity and supervised classification. We proposed a set of generic intermediate features to represent local temporal structure in a longer source stream via comparisons between extracted frame features. Although the low-level features and similarity measures are standard, we've demonstrated high-performance shot boundary detection by combining the intermediate representation with non-parametric supervised classification. We also integrated information-theoretic analysis to select features enhancing performance significantly. This step is consistent with the general approach; it improves performance without introducing limiting assumptions about the source content. Furthermore, it extends naturally to classification of a larger number of boundary classes.

There are several possible directions for future work. Algorithmic efficiency could be significantly improved by adapting the method to operate directly on compressed streams, or by developing schemes to avoid sequentially processing every frame. This will necessarily incur a performance loss, but will also reduce complexity. More complete analysis of inter-feature redundancies in the framework of [8] can also be performed. Since the feature selection step is off-line, the complexity of this analysis will not impact the system at run time, but can be expected to provide further improvements in performance. Finally, we believe the method can be applied to other modalities such as text and audio, and combinations of these modalities, given appropriate measures of similarity and low-level features.

ACKNOWLEDGMENT

The authors would like to thank Andrew Moore for making the kNN software developed at the Auton Lab at CMU available to us. We thank Jonathan Foote for comments on an early draft of this paper. We also thank the anonymous reviewers whose comments greatly improved our manuscript.

REFERENCES

[1] A. Smeaton, W. Kraaij, and P. Over, "The trec 2003 video track report," in *Proceedings of the TREC Video Retrieval Evaluation (TRECVID)*. Washington D.C.: NIST, 2003.

[2] W. Kraaij, A. Smeaton, P. Over, and J. Arlandis, "Trecvid 2004 - an introduction," in *Proceedings of the TREC Video Retrieval Evaluation (TRECVID)*. Washington D.C.: NIST, 2004, pp. 1–13.

[3] J. M. Martinez, R. Koenen, and F. Pereira, "Mpeg-7: The generic multimedia content description standard," *IEEE Multimedia*, vol. 9, pp. 78–87, 2002.

[4] M. Cooper and J. Foote, "Scene boundary detection via video self-similarity analysis." in *IEEE Intl. Conf. on Image Processing (3)*, 2001, pp. 378–381.

[5] J. Foote, "Visualizing music and audio using self-similarity." in *ACM Multimedia (1)*, 1999, pp. 77–80.

[6] J. Adcock, A. Girgensohn, M. Cooper, T. Liu, L. Wilcox, and E. Rieffel, "Fxpal experiments for trecvid 2004," in *Proceedings of the TREC Video Retrieval Evaluation (TRECVID)*. Washington D.C.: NIST, 2004, pp. 70–81.

[7] N. Vasconcelos, "Feature selection by maximum marginal diversity: optimality and implications for visual recognition." in *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (1)*, 2003, pp. 762–772.

[8] N. Vasconcelos and M. Vasconcelos, "Scalable discriminant feature selection for image retrieval and recognition." in *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (2)*, 2004, pp. 770–775.

[9] J. S. Boreczky and L. A. Rowe, “Comparison of video shot boundary detection techniques.” in *Storage and Retrieval for Image and Video Databases (SPIE)*, 1996, pp. 170–179.

[10] U. Gargi, R. Kasturi, and S. H. Strayer, “Performance characterization of video-shot-change detection methods.” *IEEE Trans. Circuits Syst. Video Techn.*, vol. 10, no. 1, pp. 1–13, 2000.

[11] F. Arman, A. Hsu, and M.-Y. Chiu, “Image processing on compressed data for large video databases,” in *MULTIMEDIA ’93: Proceedings of the first ACM international conference on Multimedia*. New York, NY, USA: ACM Press, 1993, pp. 267–272.

[12] B. Yeo and B. Liu, “A unified approach to temporal segmentation of motion jpeg and mpeg compressed video,” in *Proc. International Conference on Multimedia Computing and Systems*, 1995, pp. 81–89.

[13] J. Bescos, “Real-time shot change detection over online mpeg-2 video,” *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 14, no. 4, pp. 475–484, 2004.

[14] K. Hoashi, M. Sugano, M. Naito, K. Matsumoto, F. Sugaya, and N. Y, “Shot boundary determination on mpeg compressed domain and story segmentation experiments for trecvid 2004,” in *Proceedings of the TREC Video Retrieval Evaluation (TRECVID)*. Washington D.C.: NIST, 2004, pp. 109–120.

[15] J. Puzicha, T. Hofmann, and J. M. Buhmann, “Non-parametric similarity measures for unsupervised texture segmentation and image retrieval,” in *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR’97)*, 1997, pp. 267–272.

[16] N. Vasconcelos and A. Lippman, “Statistical models of video structure for content analysis and characterization,” *IEEE Trans. on Image Processing*, vol. 9, no. 1, pp. 3–19, 2000.

[17] Y. Qi, A. Hauptmann, and T. Liu, “Supervised classification for video shot segmentation,” in *Proc. IEEE Intl. Conf. on Multimedia & Expo (II)*, 2003, pp. 689–692.

[18] A. Hanjalic, “Shot-boundary detection: Unraveled and resolved?” *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 12, no. 2, pp. 90–105, 2002.

[19] H. Zhang, A. Kankanhalli, and S. W. Smoliar, “Automatic partitioning of full-motion video,” *Multimedia Syst.*, vol. 1, no. 1, pp. 10–28, 1993.

[20] D. Heesch, P. Howarth, J. Magalhães, A. May, M. Pickering, A. Yavlinsky, and S. Rüger, “Video retrieval using search and browsing,” in *Proceedings of the TREC Video Retrieval Evaluation (TRECVID)*. Washington D.C.: NIST, 2004, pp. 92–102.

[21] A. Witkin, “Scale-space filtering: A new approach to multi-scale description,” in *Proc. IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, Mar. 1981, pp. 39A.1.1–39A.1.4.

[22] M. Slaney, D. Ponceleon, and J. Kaufman, “Multimedia edges: finding hierarchy in all dimensions,” in *MULTIMEDIA ’01: Proceedings of the ninth ACM international conference on Multimedia*. ACM Press, 2001, pp. 29–40.

[23] D. Pye, N. Hollinghurst, T. Mills, and K. Wood, “Audio-visual segmentation for content-based retrieval,” in *Proc. Intl. Conf on Spoken Language Processing*, 1998.

[24] M. Cooper, “Video segmentation combining similarity analysis and classification,” in *MULTIMEDIA ’04: Proceedings of the 12th annual ACM international conference on Multimedia*. ACM Press, 2004, pp. 252–255.

[25] T. Volkmer, S. Tahaghoghi, and H. Williams, “Rmit university at trecvid 2004,” in *Proceedings of the TREC Video Retrieval Evaluation (TRECVID)*. Washington D.C.: NIST, 2004, pp. 171–178.

[26] J. Yuan, W. Zheng, Z. Tong, L. Chen, D. Wang, D. Ding, J. Wu, J. Li, F. Lin, and B. Zhang, “Tsinghua university at trecvid 2004: Shot boundary detection and high-level feature extraction,” in *Proceedings of the TREC Video Retrieval Evaluation (TRECVID)*. Washington D.C.: NIST, 2004, pp. 184–196.

[27] C. Petersohn, “Fraunhofer hhi at trecvid 2004: Shot boundary detection system,” in *Proceedings of the TREC Video Retrieval Evaluation (TRECVID)*. Washington D.C.: NIST, 2004, pp. 64–69.

[28] B. Gunsel, M. Ferman, and A. M. Tekalp, "Temporal video segmentation using unsupervised clustering and semantic object tracking," *Journal of Electronic Imaging*, vol. 7, pp. 592–604, July 1998.

[29] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer, 2001.

[30] T. Liu, A. Moore, and A. Gray, "Efficient exact k-nn and nonparametric classification in high dimensions," in *Advances in Neural Information Processing Systems 16*, S. Thrun, L. Saul, and B. Schölkopf, Eds. Cambridge, MA: MIT Press, 2004.

[31] D. Fradkin and D. Madigan, "Experiments with random projections for machine learning," in *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM Press, 2003, pp. 517–522.

Matthew Cooper received the B.S., M.S., and D.Sc. degrees in electrical engineering from Washington University in St. Louis in 1993, 1994, and 1999 respectively. He joined FX Palo Alto Laboratory in 2000 where he is presently a senior research scientist. His research interests are in multimedia analysis and retrieval, statistical inference, information theory, and computer vision. He is a member of the IEEE.

Ting Liu received the B.E. degree in computer science from Tsinghua University in 2001. She received the Ph.D. degree in computer science from Carnegie Melon University in 2006. She is presently a software engineer at Google, Inc. in Mountain View, CA.

Eleanor Rieffel is a Senior Research Scientist at FX Palo Alto Laboratory, where she has been working since 1996. A mathematician by training, she has performed research in fields as diverse as geometric group theory, hypertext, bioinformatics, video analysis, evolutionary computation, modular robot control, and quantum computation.