MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Discriminative Genre-Independent Audio-Visual Scene Change Detection

Kevin Wilson, Ajay Divakaran

TR2009-001 January 2009

Abstract

We present a technique for genre-independent scene-change detection using audio and video features in a discriminative support vector machines (SVM) framework. This work builds on our previous work by adding a video feature based on the MPEG-7 ”scalable color” descriptor. Adding this feature imporoves our detection rate over all genres by 5% to 15% for a fixed false positive rate of 10%. We also find that the genres that benefit the most are those with which the previous audio-only was least effective.

SPIE Electronic Imaging

Discriminative Genre-Independent Audio-Visual Scene Change Detection

Kevin W. Wilson[1] and Ajay Divakaran[2]
[1]Mitsubishi Electric Research Lab, Cambridge, MA;
[2]Sarnoff Corp., Princeton, NJ

ABSTRACT

We present a technique for genre-independent scene-change detection using audio and video features in a discriminative support vector machine (SVM) framework. This work builds on our previous work[1] by adding a video feature based on the MPEG-7 “scalable color” descriptor. Adding this feature improves our detection rate over all genres by 5% to 15% for a fixed false positive rate of 10%. We also find that the genres that benefit the most are those with which the previous audio-only was least effective.

Keywords: Multimedia Content Segmentation, Multimedia Content Classification

1. INTRODUCTION

In our previous paper[1] we addressed the problem of scene change detection. In broadcast video content, scene changes provide structure that can be useful for understanding, organizing, and browsing the content. Our primary motivation for studying scene change detection is to improve the video-browsing capabilities of consumer electronics devices to allow users to more quickly and effectively manage their content. Thus, in this paper, the term “scene change” refers to a semantically meaningful change (in filming location, subject matter, etc.) that may or may not have an obvious manifestation in the low-level video and/or audio. Furthermore, we choose a definition of “scene change” that results in an average of one scene change every few minutes, which we believe is a useful granularity for content browsing.

Our work depends on hand-labeled ground truth, so the operational definition of a scene change depends on the opinion of the human who located scene changes in our video corpus. In sitcoms and dramas, scene changes typically correspond to changes in filming location or to the entrances of significant new characters. For news, scene changes correspond to boundaries between news stories. For talk shows, scene changes correspond to changes from one guest or skit to another. Similar judgements are made for other genres. In all genres, the transitions between program content and commercials and the transitions from one commercial to the next are also considered scene changes.

Detecting these scene changes using simple audio and video features is challenging because scene changes for different genres, and even scene changes within one genre, do not necessarily have obvious similarities. Detecting shot changes, the change from one continuous sequence filmed by a single camera to another such sequence, is a much-studied problem[2] that can largely be solved using simple, low-level video features. We will use such a shot-change detector as a component in our scene change detector, but it is important to note that our semantic scene change detection task is a distinct and more challenging problem.

Detecting video scene changes over scripted and unscripted content has been explored in the past using image differences and visual motion vectors, as well as differences in audio distributions.[3–6] Usually, after a feature extraction step, a comparison with a set threshold is required. In other cases, research has focused on developing audio and visual models but without a framework to compare the effectiveness of features easily. Our work provides a more thorough performance analysis, and in addition, our testing and training are done on a much more diverse range of content than any previous work of which we are aware.[3–6] (Jiang et al[3] used only news content. Lu et al[4] used sitcoms and short scenes from a few movies. Sundaram[5,6] used an hour-long segment of a single movie.)

There has also been work on the shot change (as opposed to scene change) problem that parallels our work. Boreczky and Wilcox[7] propose an HMM model for genre-independent shot change detection using audio and video features that does not rely on hand-tuned thresholds; however, they differ from our approach because they use a generative, as opposed to

Authors’ note: Ajay Divakaran performed this work while at Mitsubishi Electric Research Lab.

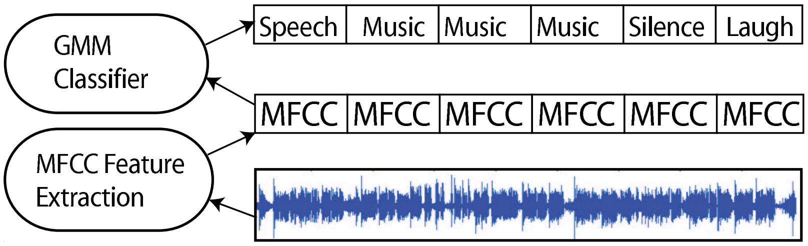


Figure 1. Audio feature streams: Low level MFCC spectral coefficients, high level semantic labels.

discriminative, model. In addition, our previous paper[1] investigates the relative performance of several different features, both individually and in combination. Camara Chavez et al[8] use an SVM framework for shot change detection. Their work focuses more on video features, and it trains and tests on a much smaller corpus than that of our current work.

In summary, detecting semantic scene changes is challenging due to factors including (1) the lack of training data; (2) difficulty in defining scene changes across diverse genres; (3) absence of a systematic method to characterize and compare performance of different features; (4) difficulty in determining thresholds in hand-tuned systems. In our previous paper, we addressed issues (1) and (2) by hand-labeling several hours of data comprising content from several genres, and addressed issues (3) and (4) by using an SVM framework which automatically determines decision boundaries during training and which can accommodate a variety of audio- or video-derived features. We obtained decent results using audio features alone or in combination with shot detection. However, we have found during subsequent informal experimentation that human scene-change detection performance using short audio snippets is only about 85 % accurate. When listening to these audio snippets, the human listener is using his spoken language understanding skills and auditory scene analysis skills to understand what is being said and done. Even state-of-the-art speech recognition and auditory scene analysis systems are far worse than human performance, so we assume it would be difficult for an audio-only automated system to surpass human audio-only performance on our scene change detection task. We are thus motivated, in this paper, to add new visual features to raise the performance ceiling above the upper bound for our audio-only approach.

2. FEATURE DESCRIPTION

We use a discriminative Gausssian-kernel SVM framework[9] for detecting video scene changes. During the training phase, the classifier requires input vectors for scene changes as well as non-scene changes, and constructs the optimal (possibly non-linear) decision boundary separating the vectors in the input space. Our goal is to find good features for distinguishing scene boundaries from non-scene boundaries in diverse video content. Because of our finite amount of training and test data, we also require that our input vectors to the SVM be relatively low-dimensional. Finally, we base our choice of features on the fact that certain feature streams are readily available, computationally efficient, and amenable to our product platform.

The audio feature streams are shown in Fig. 1. We start with an MPEG video source and extract a single-channel audio stream at 44.1 KHz. We compute 12 Mel-frequency cepstral coefficients (MFCCs) over 20 ms frames. Based on the low-level MFCC features, we classify each second of audio into one of four semantic classes: {music, speech, laughter, silence} using maximum likelihood estimation over Gaussian Mixture Models (GMMs).[10] The mixture models for each semantic class were estimated from separate data. These semantic labels help us to detect, for example, the brief snippets of music that accompany scene changes in some content or the laughter that often comes at the end of a scene in a sitcom.

For video, we base one feature on the MPEG-7 Scalable Color descriptor for each frame, and we base another feature on shot changes as determined by the algorithm from Lienhart.[2]

Using the above audio and video features, we define an SVM input vector X_i for scene(+) and non-scene(-) boundaries as follows: $X_i = \{x_1, x_2, x_3, \ldots, x_{14}\}$. In our experiments, our best-performing feature vector contained 14 dimensions, but we experimented with various features and subsets of varying dimensionality.

The input vectors X_i describe the local information about a particular time position t (in seconds) within the video. We compute an X_i at the hand-labeled time positions for scene boundaries and (randomly chosen) non-scene-boundaries. The first 9 components of X_i are histograms of semantic labels as explored in recent work,[10] the next two components represent the difference between the audio distribution before and after a particular time t. The next component is based on video shot cut counts, and the final two components represent the difference between the color distribution before and after a particular time t. The components are defined as follows:

1. **Pre-histogram:** variables x_1, x_2, x_3
   The pre-histogram tallies the number of semantic labels in the set $\{$music, speech, laughter, silence$\}$ within a window of $[t - W_L, t]$, where W_L is a chosen window size. The histogram is normalized to sum to 1. We discard one dimension from the 4D histogram because it is fully determined by the remaining three histogram values.

2. **Mid-histogram:** variables x_4, x_5, x_6
   The mid-histogram is similar to the pre-histogram and tallies semantic labels within $[t - \frac{W_L}{2}, t + \frac{W_L}{2}]$.

3. **Post-histogram:** variables x_7, x_8, x_9
   The post-histogram tallies labels within $[t, t + W_L]$.

4. **Audio Bhattacharyya Shape+Distance:** variables x_{10}, x_{11}
   We calculate the Bhattacharyya shape and Mahalanobis distance between single Gaussian models estimated from the low level MFCC coefficients for region $[t - W_L, t]$ and region $[t, t + W_L]$.

$$D_{shape} = \frac{1}{2} \ln \frac{|\frac{C_i + C_j}{2}|}{|C_i|^{\frac{1}{2}} |C_j|^{\frac{1}{2}}} \tag{1}$$

$$D_{mahal} = \frac{1}{8} (\mu_i - \mu_j)^T (\frac{C_i + C_j}{2})^{-1} (\mu_i - \mu_j) \tag{2}$$

   The covariance matrices C_i and C_j and the means μ_i and μ_j represent the (diagonal) covariance and mean of the MFCC vectors before and after a time position t.

   Bhattacharyya shape and Mahalanobis distance are sensitive to changes in the distributions of the MFCCs, so these features provide much lower-level cues about changes. For example, a scene change accompanied by a change from a male speaker to a female speaker would generate a large MFCC Mahalanobis distance even though the semantic histograms would show that both scenes contained primarily speech. (Our speech class is trained on both male and female speech.)

5. **Average Shot Count:** variable x_{12}
   The final component is the average number of shot cuts per second present in the video within a window $[t - W_L, t + W_L]$. Shot cuts are detected using the algorithm from Lienhart.[2]

6. **Color Bhattacharyya Shape+Distance:** variables x_{13}, x_{14}
   This feature is based on the MPEG-7 Scalable Color Descriptor[11] which is derived from a color histogram defined in the Hue-Saturation-Value color space. It uses a Haar transform encoding thus allowing scalable representation as well as scalable computation for increasing or decreasing the accuracy of the matching and extraction. It is exceptionally compact and easy to compute. We use the coarsest level, 16 parameters per frame, in the interest of computational simplicity. As for the audio Bhattacharyya features, we compute the scalable color descriptor over a window before the point of interest and a window after the point of interest, and after computing diagonal Gaussians, carry out the Bhattacharyya shape and distance comparisons described above. One difference is that we have 30 color descriptors per second because the video is at 30fps, while we have 92 audio feature vectors per second.

Since we use a kernel-based SVM with a smoothing bandwidth that is equal along all dimensions, we normalize all of the variables in X_i have approximately the same variance. After experimenting with different window sizes, we found that an optimal window length of $W_L = 14$ seconds gives the best overall performance.

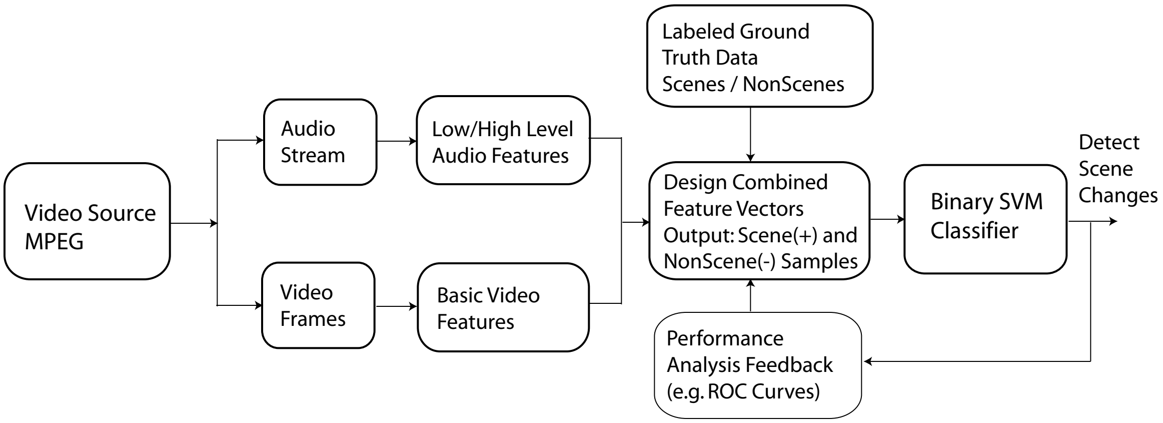

Labeled Ground Truth Data Scenes / NonScenes
Audio Stream
Low/High Level Audio Features
Video Source MPEG
Video Frames
Basic Video Features
Design Combined Feature Vectors Output: Scene(+) and NonScene(-) Samples
Binary SVM Classifier
Detect Scene Changes
Performance Analysis Feedback (e.g. ROC Curves)

Figure 2. SVM Classifier Framework.

3. SVM CLASSIFIER FRAMEWORK

A support vector machine (SVM)[9] is a supervised learning algorithm that attempts to find the maximum margin hyperplane separating two classes of data. Given data points $\{X_0, X_1, \dots X_N\}$ and class labels $\{y_0, y_1 \dots y_N\}, y_i \in \{-1, 1\}$, the SVM constructs a decision boundary for the two classes that generalizes well to future data. For this reason, the SVM has been used as a robust tool for classification in complex, noisy domains. In our case, the two classes are scene(+) versus non-scene(-) boundaries. The data points X_i are up to 14D vectors as described in Section 2. We expect that an SVM using our 14D feature input vector will be easily implementable on our product platform.

One advantage of the SVM framework is that the data **X** can be transformed to a higher dimensional *feature space* via a kernel function. Data may be linearly separable in this space by a hyperplane that is actually a non-linear boundary in the original input space. In our implementation, we found a radial basis kernel worked well:

$$K(X_i, X_j) = e^{-\gamma D^2(X_i, X_j)} \tag{3}$$

We use L_2 distance although various distance functions are possible. We fixed the value of the kernel bandwidth $\gamma = 2.0$, but could adjust this value for less smoothing if more training data were available. With limited training samples, we would like a smooth boundary to account for noise. Noise is introduced in various ways such as inaccuracies in the audio or video feature streams (misclassified semantic labels, missed/false shot cuts, alignment of streams), and in incorrect hand-labeled boundaries.

We used over 7.5 hours of diverse content to generate training and test samples for the classifier. This amounted to 530 scene(+) sample points. For non-scene(-) samples, we automatically generated twice as many random non-scene boundaries chosen at time positions outside a specific W_L of scene(+) positions. Fig. 2 shows a block diagram of the overall SVM framework.

4. EXPERIMENTS

In our experiments, we tested (1) the ability of our framework to compare different sets of features in terms of ROC performance; and (2) the ability of our framework to detect scene changes over a wide variety of broadcast genres. We used the OSU SVM Toolbox (`http://sourceforge.net/projects/svm/`), and results are based on 5-fold cross-validation. The SVM classification runs much faster than real-time on a standard 2GHz Pentium 4.

In order to generate ROC curves, we varied the SVM cost penalty for misclassifying a scene(+) boundary versus misclassifying a non-scene(-) boundary. Based on the cost ratio, the SVM produces a different separating hyperplane, yielding a performance result with different true and false positive rates. The true positive rate is the percentage of scene changes correctly detected by our system. The false positive rate is the percentage of non-scene boundaries that were classified incorrectly as scene boundaries. Ideally, we wish to achieve high true positive rates and low false positive rates. In classifying a new video piece, it may be necessary to achieve a false positive rate of 5% and as high a true positive rate as possible. In other cases, we can lower the false positive rate by other means such as pre-processing, only choosing candidate locations to test for scene changes.

Figure 3 (a) and (b) show that there is a substantial improvement over all genres with the addition of the scalable color descriptor to the audio features proposed in our previous paper. (a) shows results across all scene-change events, including commercial boundaries. In this case, adding the color Bhattacharyya yields a substantial improvement over an audio-only feature vector. The color bhattacharyya feature by itself is substantially better than our previous visual feature based on shot changes, and the combination of shot change, color bhattacharyya, and audio features yields the best overall performance. Figure 3(b) shows the results specifically for scene changes that do not involve commercial transitions, which is the most interesting case for us. In this case, the color Bhattacharyya feature is far more useful than our previous shot change feature. Additionally, it is clear that the non-commercial scene changes are more difficult since all ROC curves are substantially lower in (b). Still, we achieve reasonable performance even on the more challenging non-commercial scene changes.

Figure 4 shows a genre-wise breakup with three different combinations of features. (All of Figure 4 is for all scene changes including commercial boundaries.) A comparison of Figures 4(a) and 4(b) with Figure 4(c) shows that the combined audio-video feature vector performs at least as well as audio or video alone in all cases. In almost all cases, it is substantially better. For "How-to," audio-video performance is nearly identical to audio-only, which is already quite good. The genres on which the audio-only approach is the least effective, such as news and sitcoms, benefit substantially from the addition of the visual features. In other words, the audio and video features combine in a complementary fashion to yield improved accuracy across all genres. Note that this result is achieved with the coarsest color descriptor. This result therefore is a significant advance over our previous work.[1]

Figure 3(a) shows that with our best feature, we achieve an equal-error rate of 15% (15% false positive rate at 85% detection rate). This is comparable to human audio-only performance according to informal experiments we have performed, and given the gap between human and machine performance in other areas of audio perception, we believe this level of performance will be impossible to achieve in the forseeable future for an audio-only automated system. Our combination of audio and video features has achieved this performance at reasonable computational expense, and in combination with a user interface that facilitates rapid recovery from errors, we feel that this level of performance is sufficient for use in practical systems.

5. CONCLUSION

In this paper, we presented a visual enhancement of our previously proposed SVM kernel-based classifier framework that is useful for comparing sets of features for scene change detection. We have established that at the small computational expense of addition of the MPEG-7 scalable color descriptor to our previous audio-only framework, we are able to substantially improve the overall as well as genre-wise scene change detection accuracy. In particular, there is considerable improvement with genres that caused poor performance with the previously proposed audio-only technique. We compared results on all scene changes including commercial boundaries to scene changes excluding commercial boundaries and found that results are better for scene changes including commercials, presumably because commercial boundaries are among the easiest to detect. Still, performance on the more challenging non-commercial scene boundaries was above 80% detection for 15% false positive rate, which we believe will be useful for practical applications.

In future work, we will explore further improvement of the scene change detection accuracy by both increasing the fineness of the color descriptor as well as adding other computationally simple visual descriptors.

We would like to thank Naveen Goela for his critique of an early draft of this paper.

REFERENCES

[1] Goela, N., Wilson, K., Niu, F., Divakaran, A., and Otsuka, I., "An svm framework for genre-independent scene change detection," in [*Proc. IEEE ICME*], (2007).

[2] Lienhart, R. W., "Comparison of automatic shot boundary detection algorithms," *Storage and Retrieval for Image and Video Databases VII* **3656**(1), 290–301, SPIE (1998).

[3] Jiang, H., Lin, T., and Zhang, H., "Video segmentation with the support of audio segmentation and classification," in [*Proc. IEEE ICME*], (2000).

[4] Lu, S., King, I., and Lyu., M., "Video summarization by video structure analysis and graph optimization," in [*Proc. IEEE ICME*], (2004).

[5] Sundaram, H. and Chang, S., "Video scene segmentation using video and audio features," in [*Proc. IEEE ICME*], (2000).

[6] Sundaram, H. and Chang, S., "Audio scene segmentation using multiple models, features and time scales," in [*IEEE ICASSP*], (2000).

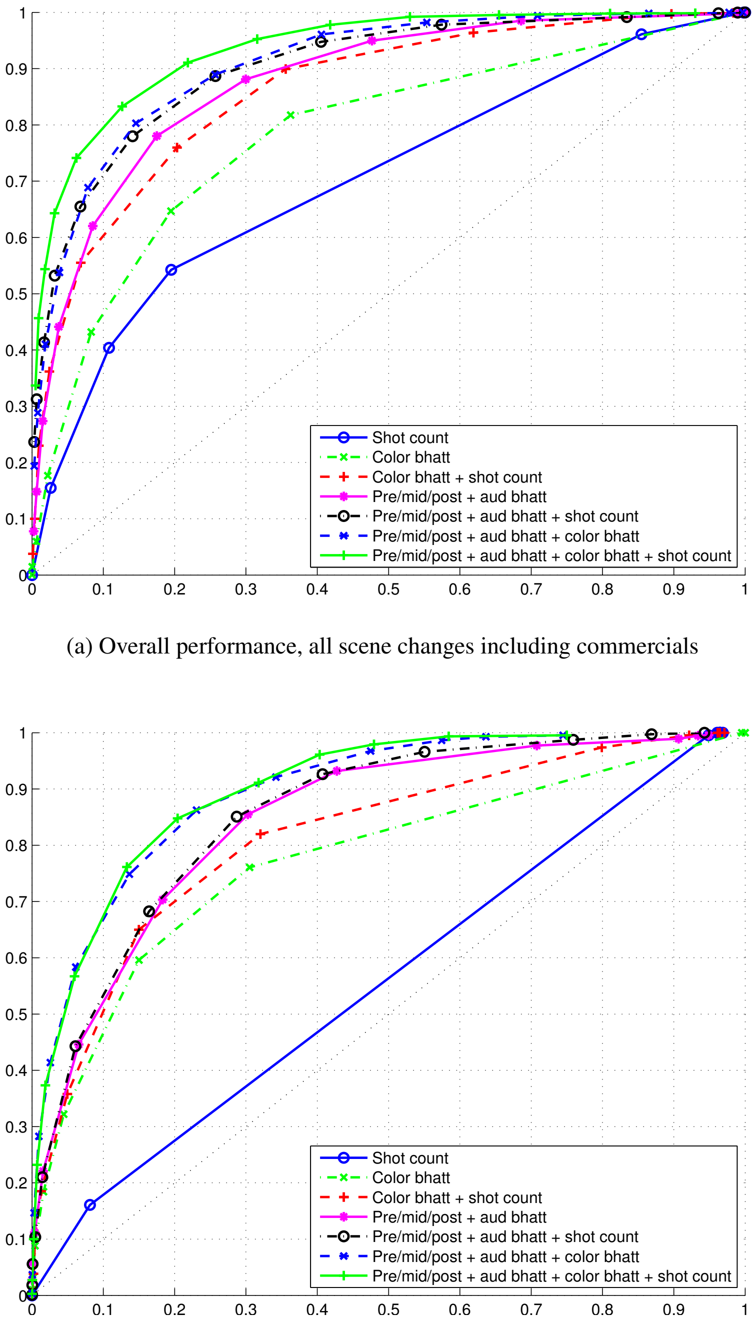
[7] Boreczky, J. S. and Wilcox, L. D., "A hidden markov model framework for video segmentation using audio and image features," in [*Proc. IEEE ICASSP*], (1998).

[8] Camara Chavez, G., Cord, M., Precioso, F., Philipp-Foliguet, S., and de A. Araujo, A., "Video segmentation by supervised learning," in [*Computer Graphics and Image Processing, 2006. SIBGRAPI '06*], 365–372 (Oct 2006).

[9] Hastie, T., Tibshirani, R., and Friedman, J., [*The Elements of Statistical Learning: Data Mining, Inference, and Prediction*], Springer (August 2001).

[10] Niu, F., Goela, N., Divakaran, A., and Abdel-Mottaleb, M., "Audio scene segmentation for video with generic content," *Multimedia Content Access: Algorithms and Systems II* **6820**(1), 68200S, SPIE (2008).

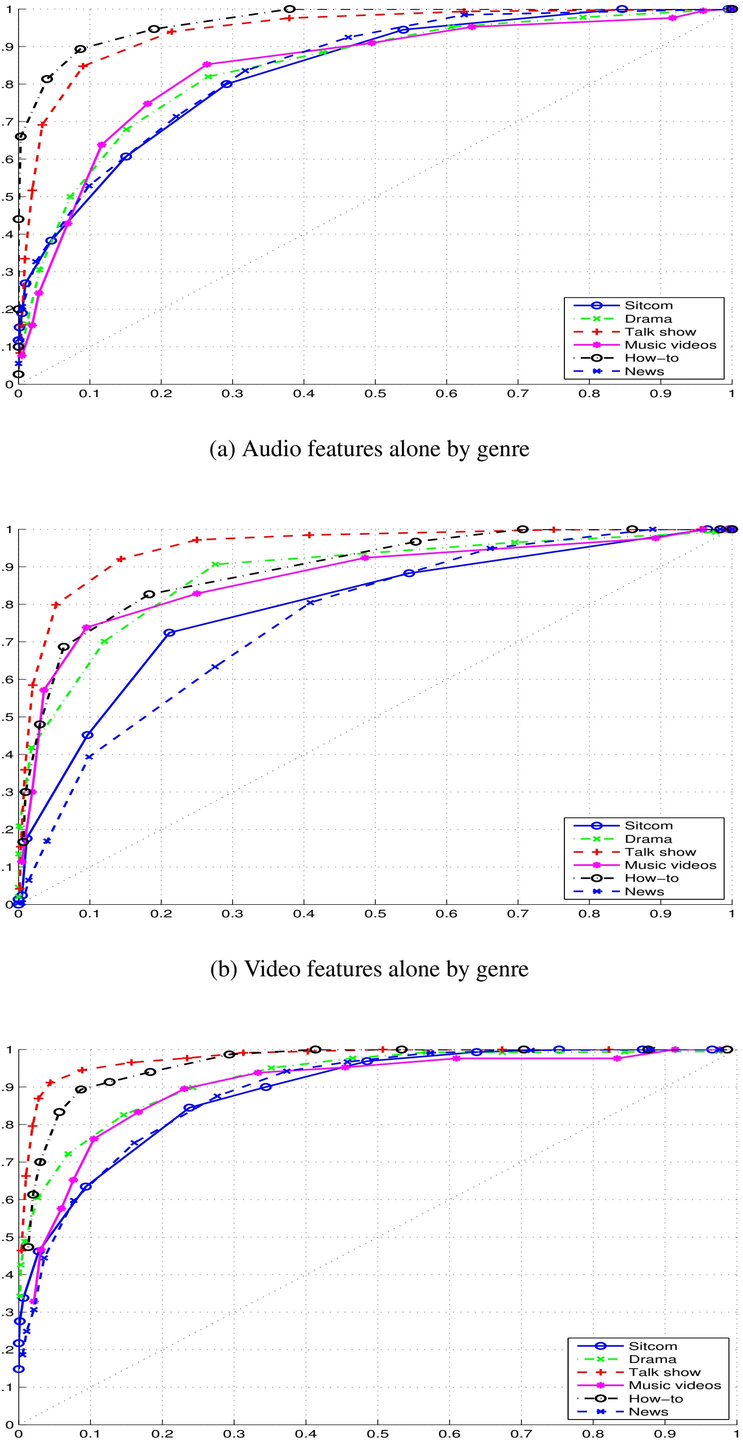
[11] Manjunath, B. S., Salembier, P., and Sikora, T., eds., [*Introduction to MPEG-7 Multimedia Content Description Interface*], Wiley (2002).



(a) Overall performance, all scene changes including commercials

(b) Overall performance, excluding commercials

Figure 3. ROC results: All curves in each panel are generated with a single classifier, and in all cases the horizontal axis is false positive rate and the vertical axis is true positive rate. (a) and (b) show performance averaged across all genres with and without commercial transitions included, respectively.



(a) Audio features alone by genre

(b) Video features alone by genre

(c) Combined audio and video features by genre

Figure 4. ROC results: All curves in each panel are generated with a single classifier, and in all cases the horizontal axis is false positive rate and the vertical axis is true positive rate. (a) and (b) show performance by genre for audio features only and for combined audio and video features, respectively.