Feature Article

Learning Video Preferences Using Visual Features and Closed Captions

Darin Brezeale
University of Texas at Arlington

Diane J. Cook
Washington State University, Pullman

An approach to identifying a viewer's video preferences uses hidden Markov models by combining visual features and closed captions.

People today have access to more video than at any time in history. Sources of video include television broadcasts, movie theaters, movie rentals, video databases, and the Internet. While many videos come from the entertainment domain, other types of video, including medical videos[1] and educational lectures (see http://www.videolectures.net), are becoming more common. As the number of video choices increases, the task of searching for videos of interest is becoming more difficult for users.

One approach that viewers take is to search for video within a specific genre. In the case of entertainment video, the distributors provide the genre when the video is released. However, many video types aren't classified, giving rise to research in automatically assigning genres.[2] While knowing a video's genre is helpful, the large number of video choices within many genres still makes finding video of interest a time-consuming process. This problem is even greater for people who enjoy video from a variety of genres. For these reasons, it's useful to have systems that can learn a particular person's preferences and make recommendations on the basis on these preferences.

There have traditionally been two approaches to identifying videos of interest to a viewer. The first is the case-based approach, which uses descriptions of video content including the genre, director, actors, and plot summary.[3,4] The advantage of the case-based approach is that it relies strictly on the viewer's profile. Once a viewer's preferences are known, they can be matched with video content descriptions. However, one weakness of this approach is that it takes effort to produce content descriptions, and there is much video in databases and on the Internet for which there are no descriptions. Another weakness is that the viewer must devote time and effort to seed the system with a substantial amount of initial preference details.

The second approach is collaborative filtering, in which users attempt to identify viewers that are considered similar by some measure. Recommendations for the current viewer are drawn from the positively rated video of these similar viewers. Collaborative filtering doesn't require the content descriptions used by the case-based approach. However, a weakness of this approach is that it takes effort to gather enough information about other viewers to determine which viewers are similar to the current viewer. A second weakness of collaborative filtering is the latency for a new video; a video can't be recommended if no one has seen and rated it yet.

The approach we describe in this article to handle video preferences is to extract visual features and closed captions from video to learn a viewer's preferences. We combine visual features and closed captions to produce observation symbols for training hidden Markov models (HMM). We define a video as a collection of features in which the order that the features appear is important, which suggests that an HMM might be appropriate for classification. We believe that visual features and closed captions are complementary. Visual features represent what is being seen, but miss much of the social interaction. Video dialogue typically doesn't describe what is being seen, but represents the social interaction.

While we believe our approach is most useful in those situations that preclude the use of collaborative filtering or case-based methods, it's also applicable in situations for which the

1070-986X/09/$25.00 © 2009 IEEE Published by the IEEE Computer Society

other approaches are appropriate, and it can be used to supplement those approaches. Even so, the approach we describe here does have certain limitations. Individually, each feature of the approach has a weakness. For example, some methods for representing text or images suffer from a lack of context. The bag-of-words model, which is a common method for representing documents, does not maintain word order and, as a result, two documents with essentially the same words but different word order can have different meanings but appear similar when comparing their term-feature representations. Likewise, two different images might appear similar when represented as color histograms. By combining text and visual features, we believe we can sidestep these limitations.

Textual features

Closed captioning is a method of letting hearing-impaired people know what is being said in a video by displaying text of the speech on the screen. Closed captions are found in line 21 of the vertical blanking interval of a television transmission and require a decoder to be seen on a television. On a DVD, the closed captions are stored in sets with display times. For example, the 1,573rd and 1,574th sets of closed captions for the movie *Star Trek: Close Contact* appear as

1573
01:34:21,963 → 01:34:23,765
RELAX, DOCTOR. I'M
SURE THEY'RE JUST HERE
1574
01:34:23,765 → 01:34:25,767
TO GIVE US A SENDOFF.

In addition to representing the dialog occurring in the video, closed captioning displays information about other types of sounds, such as onomatopoeias (for example, ''grrrr''), sound effects (for example, ''bear growls''), and music lyrics (enclosed in music note symbols). At times, the closed captions might also include the marks ≫ to indicate a change of speaker or ⋙ to indicate a change of topic.

One advantage of text-based approaches is that they can use the large body of research conducted on document text classification.[5] Another advantage is that the relationship between the features (that is, words) and specific genre is easy for humans to understand. For example, few people would be surprised to find the words ''stadium,'' ''umpire,'' and ''shortstop'' in a transcript from a baseball game.

However, using closed captions for classification does have some disadvantages. One is that the text available in closed captions is largely dialog; there is little need to describe what is being seen. For this reason, closed captions don't capture much of what is occurring in a video. A second is that not all video has closed captions. A third is that while extracting closed captions is not computationally expensive, generating the feature vectors of terms and learning from them can be expensive because the feature vectors can have tens of thousands of terms.

A common method for representing text features is to construct a feature vector using the bag-of-words model. In the bag-of-words model, each feature vector has a dimensionality equal to the number of unique words present in all sample documents (or closed-caption transcripts) with each term in the vector representing one of those words. Each term in a feature vector for a document will have a value equal to the number of times the word represented by that term appears in the document. To reduce the dimensionality of the data, we can apply stop lists of common words to ignore (for example, ''and'' and ''the'') and stemming rules (for example, replace ''independence'' and ''independent'' with ''indepen'') prior to constructing a term feature vector.

Visual features

Several features can be obtained from the visual part of a video, as demonstrated by the video retrieval and classification fields.[6] Some feature choices are color, texture, objects, and motion. Visual features might correspond to cinematic principles or concepts from film theory. For example, horror movies tend to have low light levels while comedies are often well lit. Motion might be a useful feature for identifying action movies, sports, or music videos; low amounts of motion are often present in dramas. The type of transition from one video shot to the next not only can affect mood but also can help indicate the type of movie.[7]

Visual features are often extracted on a per-frame or per-shot basis. While a shot is all of the frames within a single camera action, a scene is one or more shots that form a semantic unit. For example, a conversation between two

people might be filmed so that only one person is shown at a time. Each time the camera appears to stop and move to the other person represents a shot change; the collection of shots that represent the entire conversation is a scene. A single frame, or keyframe, can represent a shot.

In addition, shots are associated with some cinematic principles. For example, movies that focus on action tend to have shots of shorter duration than those that focus on character development.[8] One problem with using shot-based methods, though, is that the methods for automatically identifying shot boundaries don't always perform well.[9] Identifying scenes is even more difficult and there are few video-classification approaches that do so.

Color-based features are simple to implement and inexpensive to process. They are useful in approaches wishing to use cinematic principles, for example, amount and distribution of light and color set mood.[10] Color histograms are frequently used to compare frames. However, histograms don't retain information about the color placement in the frame, and the color channel bands might need to be normalized to account for different lighting conditions between frames.

Motion within a video consists primarily of movement on the part of the objects being filmed and movement due to camera actions. The quantity of motion in a video is useful in a broad sense, but it's not sufficient by itself in distinguishing between the video types that typically have large quantities of motion, such as action movies, sports, and music videos. Measuring specific types of motion, such as object or camera, also presents a problem because of the difficulty in separating the two.

One of the more popular video formats is MPEG. During the encoding of MPEG-1 video, each pixel in each frame is transformed from the RGB color space to the YC_bC_r color space, which consists of one luminance (Y) and two chrominance (C_b and C_r) values. The values in the new color space are then transformed in blocks of 8×8 pixels using the discrete cosine transform (DCT). Each frame in the MPEG-1 format is classified as either an I-frame, a P-frame, or a B-frame depending on how it is encoded. I-frames contain all of the information needed to decode the frame. In contrast, P-frames and B-frames make use of information from previous or future frames.

Dimensionality reduction

Samples in a data set are often represented by a large number of features, which might make learning difficult or be computationally infeasible to process. One approach to finding a new smaller representation of video signals is to perform wavelet analysis, which decomposes a signal into two signals: a trend (or weighted average) signal and a details signal, each having half the terms of the original signal.[11] Wavelet analysis can be applied to 2D data, such as an image, by first applying the wavelet transform to each row (or column) of the image and then to the transformed columns (or rows). By keeping only the trend signal values, the dimensionality of the original signal can be reduced.

Random projection can reduce dimension by projecting a set of points in a high-dimensional space to a randomly selected lower-dimensional subspace using a random matrix.[12] An advantage of random projection is that it's not computationally expensive compared to principal component analysis.[13]

Clustering is a method of unsupervised learning that partitions the input data into closely related sets or clusters. In our work, we use clustering to reduce an image's feature vector by clustering image features and representing images as a vector of cluster memberships. We also use the same approach for closed-caption sets, and represent textual information as a vector of closed-caption cluster membership frequencies.

Methodology

Our goal is to learn a user's video preferences by constructing a model whose input is a set of textual and visual features drawn from videos that this user has viewed and rated. Our approach maintains the temporal relationship between individual features in the video clip. To do this, we need to determine how to combine the text and visual features and how to capture the temporal relationship of features.

A common approach to dealing with the first issue is to segment a video into shots and then represent each shot by the features that occur during this shot. However, automating shot detection is difficult and unreliable. Closed captions displayed onscreen at the

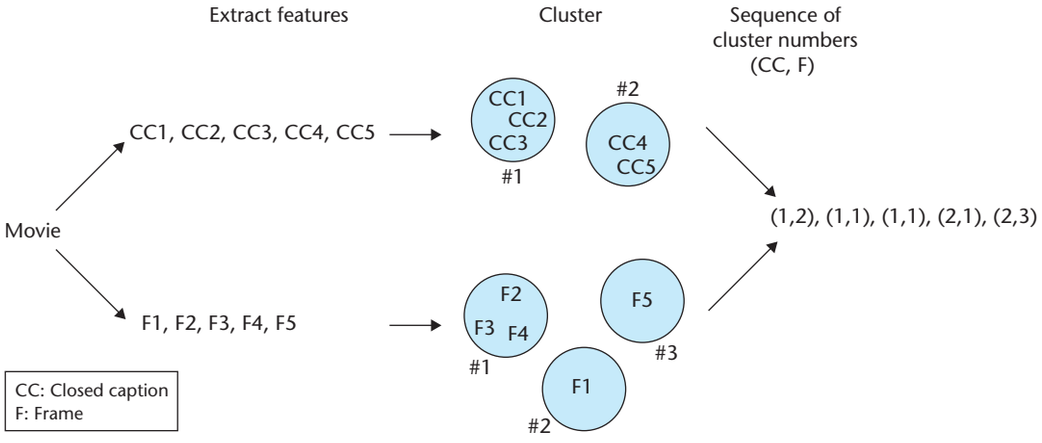

Figure 1. Example of observation symbol production.

same time, which we call *closed-caption sets*, are stored along with the time period for which the closed caption will be displayed. By using these closed-caption set display times, we know the time that certain text and visual features occur. Therefore, we extract the closed-captions sets and use the corresponding times to segment the video and to find a corresponding video frame from which to extract visual features.

To prevent oversegmenting the video, we combine consecutive closed-caption sets to form a single feature vector. Once we determine the segmentation times, we extract the visual features from a single video frame that occurs during this time period.

To capture the temporal relationship of features, we constructed an HMM for each targeted class of videos. To generate the set of observation symbols, we cluster the closed-caption features for all of the movies a viewer has rated and repeat the process for the visual features. We hypothesize that feature vectors from movies the user liked and disliked will tend to fall in different clusters. We generate observation symbols by combining the cluster number of a closed-caption set and the cluster number of its corresponding video frame (the video frame for the time period that the closed-caption set was displayed) in the form shown in Figure 1.

Applying this process to each closed-caption set and video-frame pair produces a sequence of observation symbols for the movie. By generating observation symbols that combine these types of features, we capture some element of context. To classify an unseen movie, we generate a sequence of observation symbols for each HMM. We assign to the movie the HMM classification that generates the sequence with the highest probability.

Experiments

We obtained the user ratings for the experiments described here from the following two publicly available data sets: the MovieLens 1-million ratings (see http://www.cs.umn.edu/Research/GroupLens) and the Netflix Prize (see http://www.netflixprize.com). The MovieLens data set includes titles and genre for 3,883 movies as well as over one million viewer ratings using the range 1 (strongly disliked) to 5 (strongly liked). The Netflix data set consists of over 100 million ratings from 480,189 users for a set of 17,770 movies. The range of rating values is also 1 to 5. We acquired the DVD version of 90 movies represented in the MovieLens data set; 88 of these movies are also in the Netflix data set. We selected these movies from 18 entertainment genres, with many having multiple genre labels.

Using text and visual features separately

Our initial experiment assesses the viability of using closed captions and visual features independent of temporal relationships. We tested closed captions and visual features separately for the tasks of classification by genre, classification by user using a 1 through 5 video rating, and classification by grouped-user ratings using a rating of 1 through 3 to indicate dislike and 4 and 5 as like. We performed all tests using the support-vector-machine classifier available in the Weka data-mining software.[14] SVMs are well suited to problems for which there are few training examples but where the feature vectors have many terms.[15] We chose 81 movies represented in the MovieLens project that

Table 1. Summary of preliminary results using closed captions.

Experiment classification	Classification accuracy (%)	95 percent confidence interval
By genre	89.71	(84.34, 95.09)
Individual ratings	38.45	(37.40, 39.50)
Grouped ratings	64.04	(63.02, 65.05)

had been rated by at least 20 users. There were 1,116 users who had rated at least 10 of these 81 movies. For each type of experiment we calculated the mean classification accuracy.

We initially evaluated movie classification with closed captions alone using a bag-of-words model. We converted each movie's closed captions to a feature vector using the bag-of-words model after applying a standard stop list and stemming algorithm. The feature vectors contained up to 15,254 terms. Table 1 summarizes the results from these experiments.

When classifying by video feature alone, we hypothesize that movies with similar shot types should have similar feature vectors and therefore we represent each movie as a set of video features for each of its shots. Because of the computational and storage requirements, we extracted video features from the first five minutes of each video. We determined shot boundaries by comparing color histograms, then modified MPEG Java to extract the DCT coefficients from the first frame of each shot. These frames had a resolution of 240×352 pixels.

Next, we represented each frame as a histogram of the DCT coefficients. A term-by-term comparison of histograms determines that two frames are similar if they have similar color distributions, even if the exact color locations in each frame differ. We clustered the histograms in order to group similar shots. After the clustering, a feature vector represented each movie with a term for each of the k-clusters. For example, when $k \times 5$, a movie with the feature vector [1, 0, 5, 50, 0] contains one shot in cluster one, five in cluster three, and 50 in cluster four.

The total number of shots for all 81 movies was 46,311. Table 2 summarizes the results for the experiments that used DCT coefficients with 95 percent confidence intervals.

Comparing Tables 1 and 2, we can see that the results were virtually the same regardless of whether we used closed captions or DCT coefficients. We expected classification by genre of a movie to be easier than learning an individual's preferences, but also noticed that the results were similar using 20 or 40 clusters. The results from using individual ratings are better than a random guess, but there is still much room for improvement. One reason for this poor performance could be that the number of training examples for each user was too small to learn a user's rating preferences.

Table 2. Summary of preliminary results using DCT coefficients.

Experiment classification (number of clusters tested)	Classification (number of clusters) accuracy (%)	95 percent confidence interval
Genre (20)	88.48	(82.66, 94.30)
Individual ratings (20)	33.26	(32.33, 34.19)
Grouped ratings (20)	59.23	(58.28, 60.19)
Genre (40)	87.24	(81.17, 93.31)
Individual ratings (40)	32.54	(31.63, 33.45)
Grouped ratings (40)	58.76	(57.83, 59.69)

Using hidden Markov models with textual and visual features

In our preliminary experiments, we were able to classify video by genre with promising results. This suggests that text and visual features are viable for learning preferences. However, the results were much less favorable when we used each of these types of features for predicting that a viewer would like or dislike a movie, or in predicting the specific viewer rating of a movie.

To address the limitations of the approach taken in our preliminary experiments, we wanted to combine the text and visual features as well as represent the temporal relationship of the features. Our approach follows:

- extract the closed captions and visual features and cluster each separately,
- generate observation symbols for an HMM by combining the cluster assignments of the features, and
- construct an HMM for each of the two classes (that is, like or dislike) that we are interested in predicting.

In this experiment, we increased the average number of movies rated per user by switching to the Netflix data set. In our earlier

Table 3. Comparison of features from HMM and k-means clustering.

Features	Accuracy (%)	95 percent confidence interval for accuracy	Precision (%)	95 percent confidence interval for precision	Recall (%)	95 percent confidence interval for recall
Closed caption and visual	61.7	(60.2, 63.2)	51.2	(48.0, 54.4)	53.4	(49.1, 57.6)
Closed caption only	60.9	(59.4, 62.4)	49.0	(46.0, 52.0)	50.8	(46.9, 54.7)
Visual only	61.5	(60.1, 63.0)	50.9	(47.8, 54.0)	50.1	(46.3, 54.0)

experiments using the MovieLens data set, there were 357 users who matched our criteria. There are 334 users in the Netflix data set that meet our new criteria of rating at least 45 movies. Because we were able to effectively reduce the dimensionality of our feature vectors, we extracted 20 minutes of video from each movie. Specifically, we extracted minutes five through 25 so as to skip the credits and introductory graphics typically found in the first five minutes.

To capture temporal relationships between the features, we first segment the video and then extract the text and visual features from each segment. As described earlier, we used the begin and end times associated with each closed-caption set to create video segments, and gathered closed captions and visual features from the corresponding time in the video. Instead of using a single closed-caption set as a segmentation mechanism, we created a window of consecutive closed-caption sets with a length of 20 sets, with a new window beginning every 10th closed-caption set. That is, we combined closed-caption sets 1–20, 10–30, 20–40, and so forth. We combined all words from all closed-caption sets within the window to represent that entire time that these closed captions are displayed. We derived visual features from a single frame within this period to represent the entire time frame. The frame we chose is the first frame from each closed-caption window.

We applied random projection to reduce the vector dimensions from 4,003 terms to 363 terms. To produce the visual features, we represented the first frame of each video segment by a concatenation of the pixel RGB values. We reduced the vectors from 253,440 terms to 363 terms by applying five levels of a 2D, Daubechies 4 wavelet[11] separately to the R, G, and B components.

As described earlier, we cluster closed-caption terms and visual feature terms separately and use the text cluster number and visual cluster number pairs as observation symbols for the HMMs. We constructed two HMMs: one from the training samples of the movies the viewer rated as liked and one from the movies the viewer rated as disliked. The HMMs initially have randomized probabilities for start states, transitions, and observation symbols. The states of the HMM represent the high-level concepts that occur in the movies that the user has rated. In theory, these concepts could be things like ''car chases'' or ''two people talking.'' However, it is difficult to look at the extracted features to discern the actual concepts being represented, especially in light of the fact that the constructed HMMs aren't unique. Also, different viewers will have different preferences and therefore different models will be constructed.

We investigated HMMs with 10–70 states and found the highest classification accuracy occurred with 60 states. Table 3 shows the results achieved when we used *k*-means clustering with HMM models with 60 states. The table shows the results in terms of accuracy, precision, and recall. Precision and recall are more commonly found in information retrieval; we include them here for comparison with other video-recommender research. In the case of a movie recommender system, the recommender might recommend a subset of the total movies available. Precision is a measure of how many of the recommended movies the user actually prefers, while recall is a measure of how many of the movies in the total data set the user would prefer to end up in the recommendations.

We can see that the results from combining features were approximately the same as those achieved when generating observation symbols for either type of feature alone. An analysis of the models shows that, for many users, one of the models would perform well while the other model would perform poorly. In particular, this disparity would happen when the user's ratings were not close to being evenly distributed between liked and disliked ratings.

Table 4. Results per number of movies rated in hybrid approach using k*-means.*

Number rated	Number of users	Accuracy	95 percent confidence interval
45 ≤ movies rated < 50	167	60.0	(57.8, 62.3)
50 ≤ movies rated < 60	131	62.5	(60.3, 64.8)
60 ≤ movies rated < 70	26	64.6	(59.7, 69.5)
70 ≤ movies rated ≤ 88	10	72.3	(59.8, 84.8)

This disparity could be a consequence of having too few training examples to learn from for one of the classes.

Table 4 shows the classification accuracy by number of movies rated. The precision and recall for each range of ratings was approximately 59 and 53 percent, respectively. We can see here that, as would be expected, the classification accuracy improved as the number of ratings increased. However, an analysis of the individual predictions shows that even for the users for which there were a large number of rated movies, in most cases only one of the HMMs performed well. Because this was the HMM constructed from the majority of the user's ratings, it showed a measured improvement in performance.

Much of the other researchers in video recommendation report results in terms of precision and recall. Ardissono et al. achieved a precision of 80 percent and a mean absolute error rate of 30 percent in their case-based approach to recommendation.[3] Basu et al. achieved precision and recall values of 83 percent and 34 percent, respectively, in their system that combined the case-based and collaborative filtering approaches.[16] Their approach focused on achieving high precision at the expense of recall.

In comparison, the precision of our results was less than what either of these other approaches achieved. Neither of the other approaches reports overall classification accuracy, so we can't compare our results to theirs using that metric. While both of their approaches achieve higher precision than our approach, they are still restricted to those situations in which substantial information is available, such as hand-constructed video information and extensive ratings from the same or similar viewers. Our results are promising for identifying preferred videos with little a priori information.

Conclusions

Traditional approaches to video recommendation have proved to have relatively good performance. However, these approaches aren't always applicable. To address this need, we have explored the use of visual features and closed captions extracted from video for learning a viewer's preferences. We believe this approach to be a viable alternative to traditional approaches. While the approach yielded promising results, we found that in many cases one of the learned models tended to not perform well. For most viewers, the number of liked and disliked movies was far from even, resulting in an insufficient number of training examples for one of the classes.

While our experiments focused on entertainment video, other domains, such as education, should be explored to determine the viability of this approach to that type of video. For example, as more educational video becomes available, students will have a variety of choices for learning a particular topic. Given a robust recommendation system, videos that are similar to those that resulted in the best performance by the student could be recommended. Moreover, we believe this work can be applied to video classification at the shot or scene level. Applications for such systems could include content filtering to identify violent or important scenes. **MM**

References

1. J. Fan et al., ''Semantic Video Classification and Feature Subset Selection Under Context and Concept Uncertainty,'' Proc. 4th ACM/IEEE-CS Joint Conf. Digital Libraries (JCDL), ACM Press, 2004, pp. 192-201.
2. D. Brezeale and D.J. Cook, ''Automatic Video Classification: A Survey of the Literature,'' *IEEE Trans. Systems, Man, and Cybernetics, Part C,* vol. 38, no. 3, 2008, pp. 416-430.
3. L. Ardissono et al., ''User Modeling and Recommendation Techniques for Personalized Electronic Program Guides,'' *Personalized Digital Television: Targeting Programs to Individual Viewers,* L. Ardissono, A. Kobsa, and M. Maybury, eds., Kluwer, 2004, pp. 3-26.
4. J. Zimmerman et al., ''TV Personalization System: Design of a TV Show Recommender Engine and Interface,'' *Personalized Digital Television: Targeting Programs to Individual Viewers,* L. Ardissono, A. Kobsa, and M. Maybury, eds., Kluwer, 2004, pp. 27-52.

Related Work

Research related to the work discussed in the main text falls primarily into two categories: video recommendation and automatic classification of video by genre. Several researchers have found collaborative filtering outperforms case-based classification in explicit comparison, but note that combining both sources of information performs best.[1,2] Other researchers have fused multiple user models and thus combine demographic information with general interest and observed TV viewing habits to improve overall recommendation accuracy.[3,4]

We believe that our approach has several advantages over existing methods. It doesn't require that a viewer provide any information about his or her preferences other than a rating for a viewed video. This saves time and avoids poor recommendations that might occur due to omissions in the preference description. Another benefit is that it's unnecessary to identify similar viewers. A third is that there are situations in which neither case-based nor collaborative filtering approaches are applicable and the only choice is to analyze the video itself. On the other hand, our approach does require the existence of closed-caption information and some initial video preference information to form the models.

Approaches to classification of video by genre use three feature modalities: audio, visual, or text.[5] Zhu et al. classify news stories using the first 20 unique keywords that are obtained from closed captions.[6] Lin and Hauptmann,[7] Wang et al.,[8] and Qi et al.[9] have combined two or more feature modalities successfully to categorize news videos into story types. In this approach, unlike in our approach, the temporal relationships between features aren't used.

Dimitrova et al. classify four types of TV programs using a hidden Markov model (HMM) for each class and face counts and extracted text.[10] Lu et al. classify a video by first summarizing it.[11] A hierarchical clustering algorithm segments the video into scenes; the keyframes from the scenes represent the summarized video. One HMM is trained for each video genre with the keyframes as the observation symbols.

References

1. N. Karunanithi and J. Alspector, ''A Feature-Based Neural Network Movie Selection Approach,'' *Proc. Int'l Workshop on Applications of Neural Networks to Telecommunications,* Lawrence Erlbaum Assoc., 1995, pp. 162-169.
2. B. Smyth and P. Cotter, ''Surfing the Digital Wave: Generating Personalised Television Guides Using Collaborative, Case-Based Recommendation,'' *Proc. Int'l Conf. Case-Based Reasoning,* Springer, 1999, pp. 561-571.
3. L. Ardissono et al., ''User Modeling and Recommendation Techniques for Personalized Electronic Program Guides,'' *Personalized Digital Television: Targeting Programs to Individual Viewers,* L. Ardissono, A. Kobsa, and M. Maybury, eds., Kluwer, 2004, pp. 3-26.
4. J. Zimmerman et al., ''TV Personalization System: Design of a TV Show Recommender Engine and Interface,'' *Personalized Digital Television: Targeting Programs to Individual Viewers,* L. Ardissono, A. Kobsa, and M. Maybury, eds., Kluwer, 2004, pp. 27-52.
5. D. Brezeale and D.J. Cook, ''Automatic Video Classification: A Survey of the Literature,'' *IEEE Trans. Systems, Man, and Cybernetics, Part C,* vol. 38, no. 3, 2008, pp. 416-430.
6. W. Zhu, C. Toklu, and S.-P. Liou, ''Automatic News Video Segmentation and Categorization Based on Closed-Captioned Text,'' *Proc. IEEE Int'l Conf. Multimedia and Expo,* IEEE Press, 2001, pp. 829-832.
7. W.-H. Lin and A. Hauptmann, ''News Video Classification Using SVM-Based Multimodal Classifiers and Combination Strategies,'' *Proc. ACM Multimedia,* ACM Press, 2002, pp. 323-326.
8. P. Wang, R. Cai, and S.-Q. Yang, ''A Hybrid Approach to News Video Classification Multimodal Features,'' *Proc. Joint Conf. Int'l Conf. Information, Communications, and Signal Processing and the 4th Pacific Rim Conf. Multimedia,* vol. 2, IEEE Press, 2003, pp. 787-791.
9. W. Qi et al., ''Integrating Visual, Audio, and Text Analysis for News Video,'' *Proc. IEEE Int'l Conf. Image Processing,* IEEE Press, 2001.
10. N. Dimitrova, L. Agnihotri, and G. Wei, ''Video Classification Based on HMM Using Text and Faces,'' *Proc. European Signal Processing Conf.,* 2000.
11. C. Lu, M.S. Drew, and J. Au, ''Classification of Summarized Videos Using Hidden Markov Models on Compressed Chromaticity Signatures,'' *Proc. ACM Int'l Conf. Multimedia,* ACM Press, 2001, pp. 479-482.

5. F. Sebastiani, ''Machine Learning in Automated Text Categorization,'' *ACM Computing Surveys,* vol. 34, no. 1, 2002, pp. 1-47.
6. Y.A. Aslandogan and C.T. Yu, ''Techniques and Systems for Image and Video Retrieval,'' IEEE Trans. Knowledge and Data Engineering (TKDE), special issue on multimedia retrieval, vol. 11, no. 1, 1999, pp. 56-63.
7. G. Oldham, *First Cut: Conversations with Film Editors,* Univ. of Calif. Press, 1992.
8. N. Vasconcelos and A. Lippman, ''Statistical Models of Video Structure for Content Analysis and Characterization,'' *IEEE Trans. Image Processing,* vol. 9, no. 1, 2000, pp. 3-19.
9. R. Jadon, S. Chaudhury, and K. Biswas, ''A Fuzzy Theoretic Approach for Video Segmentation

Using Syntactic Features,'' *Pattern Recognition Letters,* vol. 22, no. 13, 2001, pp. 1359-1369.
10. Z. Rasheed, Y. Sheikh, and M. Shah, ''Semantic Film Preview Classification Using Low-Level Computable Features,'' *Proc. 3rd Int'l Workshop Multimedia Data and Document Engineering,* 2003; http://www.cs.cmu.edu/~yaser/RasheedSheikhShah_MDDE2003.pdf.
11. J.S. Walker, *A Primer on Wavelets and their Scientific Applications,* CRC Press, 1999.
12. S. Dasgupta, ''Experiments with Random rojection,'' *Proc. 16th Conf. Uncertainty in Artificial Intelligence,* Morgan Kaufmann, 2000, pp. 143-151.
13. E. Bingham and H. Mannila, ''Random Projection in Dimensionality Reduction: Applications to Image and Text Data,'' *Proc. ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining,* ACM Press, 2001, pp. 245-250.
14. I.H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations,* Morgan Kaufmann, 2000.
15. K.P. Bennett and C. Campbell, ''Support Vector Machines: Hype or Hallelujah?'' *SIGKDD Explorations,* vol. 2, no. 2, 2000, pp. 1-13.
16. C. Basu, H. Hirsh, and W. Cohen, ''Recommendation as Classification: Using Social and Content-Based Information in Recommendation,'' *Proc. Nat'l Conf. Artificial Intelligence,* AAAI Press, 1998, pp. 714-720.

Darin Brezeale is a lecturer in the computer science and engineering department at the University of Texas at Arlington. His research interests include artificial intelligence, math, and statistics. Brezeale has a PhD in machine learning and video preferences from the University of Texas at Arlington. Contact him at darin.brezeale@uta.edu.

Diane J. Cook is a Huie-Rogers Chair Professor in the School of Electrical Engineering and Computer Science at Washington State University. Her research interests include artificial intelligence, machine learning, graph-based relational data mining, smart environments, and robotics. Cook has a PhD in computer science from the University of Illinois. Contact her at cook@eecs.wsu.edu.