Chapter 1

MOVIE CONTENT ANALYSIS, INDEXING AND SKIMMING VIA MULTIMODAL INFORMATION

Ying Li
IBM T.J. Watson Research Center
30 Saw Mill River Road, Hawthorne, NY 10532
yingli@alumni.usc.edu

Shrikanth Narayanan, and C.-C. Jay Kuo
Integrated Media Systems Center and Department of Electrical Engineering
University of Southern California, Los Angeles, CA 90089
shri,ckuo@sipi.usc.edu

Abstract A content-based movie analysis, indexing and skimming system is developed in this research. Specifically, it includes the following three major modules: 1) event detection module, where three types of movie events, namely, 2-speaker dialogs, multiple-speaker dialogs, and hybrid events are extracted from the content. Multiple media sources such as audio, speech, visual and face cues have been exploited; 2) speaker identification module, where an adaptive speaker identification scheme is proposed to recognize interested movie casts for the content indexing purpose. Both audio and visual sources are exploited in the identification process, where the audio source is analyzed to recognize speakers using a likelihood-based approach, and the visual source is examined to locate talking faces with face detection/recognition and mouth tracking techniques; 3) movie skimming module, where an event-based skimming system is developed to abstract movie content in the form of a short video clip for the content browsing purpose. Extensive experiments on integrating multiple media cues for movie content analysis, indexing and skimming have yielded encouraging results.

Keywords: Movie content analysis, movie event detection, speaker identification, talking face detection, movie skimming, multimodal analysis
1. Introduction

With the fast growth of multimedia information, content-based video analysis, indexing and representation have attracted increasing attention in recent years. Many applications have emerged in the areas such as video-on-demand, distributed multimedia systems, digital video libraries, distance education and entertainment. The need for content-based video indexing and retrieval was also recognized by ISO/MPEG, and a new international standard called “Multimedia Content Description Interface” (or in short, MPEG-7) has been initialized since 1998 (MPEG-7, 1999), and finalized in September 2001.

Content-based video analysis aims at obtaining a structured organization of the original video content and understanding its embedded semantics like humans do. Content-based video indexing is the task of tagging semantic video units obtained from content analysis to enable convenient and efficient content retrieval. Although content understanding is an easy task for a human, it remains to be a very complicated process for a computer because of the limitations of machine perception under unconstrained environments and the unstructured nature of video data. Robust techniques are still lacking today despite a large amount of effort in this area (Yeung et al. 1997, Sundaram et al. 2000, Tsekeridou et al. 2001).

So far, the predominant approach to this problem is to first extract some low- to mid-level audiovisual features such as color, texture, shape, motion, shots, keyframes, object trajectories, human faces, and classified audio classes; then partially derive or understand the video semantics by analyzing and integrating these features. Although encouraging results have been reported in previous work, a semantic gap still exists between the real video content and the video contexts derived from these features.

Content-based video skimming aims at abstracting the video content and presenting its essence to users in form of a short video clip. Video skimming is mainly adopted for the video browsing purpose, which forms an inseparable part of a video indexing and retrieval system. Currently, most video skimming work develops its systems based on pre-extracted keyframes, which may result in undesired discontinuities in the skim’s audio and visual content.

This work first extracts semantic video events from the content. Feature film is the major focus of this work. The extracted event information can be utilized to facilitate movie content browsing, abstraction and indexing. In the second stage, interested movie casts (or target speakers) are identified from the content based on the speech and face information. Finally, a movie skimming system is proposed which summarizes movie
content based on pre-obtained event structures as well as optional user preferences.

The rest of this chapter is organized as follows. We first review some related previous work and give an overview of our approach in Section 2. Low-level audiovisual content analysis is briefly reviewed in Section 3. Section 4 elaborates on the work of movie event extraction and characterization. In Section 5, we give details on the proposed adaptive speaker identification scheme. In Section 6, we describe our work on the event-based movie skimming system. Experimental results are reported and discussed in Section 7, and finally, concluding remarks are drawn in Section 8.

2. Approach Overview

2.1 Event Detection and Extraction

By *event*, we mean a video paragraph which contains a meaningful theme and progresses under a consistent environment. Because event is a rather subjectively defined concept, below we will review some previous work that addresses similar concepts or has similar research goals.

Shot detection, where a *shot* is defined as a set of contiguously recorded image frames, is usually the first step towards video content understanding. However, while the shot forms the building block of the video content, this low-level structure does not correspond to the underlying video semantics in a direct and convenient way. Thus most recent work tends to understand the video semantics by extracting the underlying video *scenes*, where a scene is defined as a collection of semantically related shots that depict and convey a high-level concept or story. For instance, Rui and Yeung (Rui et al. 1998, Yeung et al. 1996) proposed to extract video scenes by grouping visually similar and temporally adjacent shots. Huang et al. developed a scene detection scheme based on the integration of audio, visual and motion information. Similar ideas were also explored in the Informedia project (Hauptmann et al. 1995) where three types of media cues including audio, image and text keywords, were combined to determine scenes.

However, while a scene does provide a higher-level video context, not every scene contains a meaningful thematic topic, especially for movies where progressive scenes, which are frequently inserted to establish the story situation, are actually unimportant for content understanding. Therefore, we propose to extract event, which operates at a higher semantic level, from movies in this work so as to better reveal, represent and abstract the content. The reason we choose the movie application is that movie has a clear story structure which can be well exploited by
our approach. Moreover, movie has many special characteristics such as the complex film editing techniques (Reisz et al. 1968), thus it is not only interesting but also challenging to work with movies.

Because a movie plot is usually developed through either dialogs or actions, we focus on identifying the following three types of events in this research: 2-speaker dialogs, multiple-speaker dialogs, and hybrid events which accommodate for events with less speech and more visual action. The detection of dialogs has been explored by some previous work. For instance, (Yeung et al. 1997) characterized a temporal event into either dialog, action or others, where a dialog could be either an actual conversation or a conversation-like montage presentation of two or more concurrent processes. In particular, they proposed to detect a dialog by searching a shot sequence with a repetitive nature of two dominant shots such as “A B A B A B”. A similar periodic analysis transform was also employed in (Sundaram et al. 2000) for dialog detection. However, since the arrangement of shot sequences in a dialog basically varies with the film genre and also heavily depends on the directorial style, strict periodic analysis appears to be too restrictive for a general scenario. In addition, the problem becomes more complex when multiple speakers are present. Finally, the speech information, which is an important indicator for dialogs, was not considered in both work.

In this work, we aim to fulfill this task by analyzing the movie content structure and exploiting film’s special editing features. Moreover, visual cues such as human faces, will be effectively integrated with speech cues to obtain robust results.

2.2 Speaker Identification

Automatic speaker identification has been an active research topic for many years with bulk of the progress facilitated by work on standard speech databases such as YOHO, HUB4, and SWITCHBOARD (Chagnolleau et al. 1999, Johnson, 1999). Recently, with the increase of the accessibility to other media sources, researchers have attempted to improve system performance by integrating knowledge from all available media cues. For instance, (Tsakiridou et al. 2001) proposed to identify speakers by integrating cues from both speaker recognition and facial analysis schemes. This system is, however, impracticable for generic video types since it assumes there is only one human face in each video frame. Similar work was also reported by (Li and Wei, 2001), where TV sitcoms were used as test sequences. In (Li et al. 2002), a speaker identification system was proposed for movie content indexing purpose, where
both speech and visual cues were employed. This system, however, has certain limitations since it only identifies speakers in movie dialogs.

From the other point of view, most existing work in this field deals with supervised identification problem, where speaker models are not allowed to change once they are trained. Two drawbacks arise when this approach is applied to feature films. First, we may not have sufficient training data. Because a speaker's voice can have distinct variations along time, especially in feature films, thus a model built with limited training data cannot model a speaker well for the entire sequence. Second, since we have to go through the movie at least once to collect and transcribe the training data before the actual identification process can be started, it wastes time and decreases system efficiency.

An adaptive speaker identification system is proposed in this work which offers a better solution for identifying speakers in movies. Specifically, after building coarse models for target speakers during system initialization, we will continuously update them on the fly by adapting to speakers’ newly contributed data. It is our claim that, by adapting models to incoming speech data, we can achieve higher identification accuracy as they can better capture speakers’ voice variations along time. Both audio and visual sources will be exploited in the identification process, where the audio source is analyzed to recognize speakers using a likelihood-based approach, and the visual source is parsed to find talking faces using face detection/recognition and mouth tracking techniques.

2.3 Video Skimming

Video skimming is used to summarize video content and present users with its essence in the form of a moving storyboard. So far, many research efforts have been reported on video skimming such as the VAbstract system (Pfeiffer et al. 1996), which generated trailers for feature films, and the Informedia project developed in the Carnegie-Mellon University (Smith et al. 1997) which generated short video synopsis by exploiting text keywords and image information. Various techniques such as textual content analysis (Toklu et al. 2000), speech transcription analysis (Taskiran et al. 2002), and dynamic sampling schemes (Tseng et al. 2002) have been adopted to obtain meaningful skims.

However, while acceptable results were reported in these work, most of them develop their skimming schemes based on pre-developed summarization schemes and, as a result, the generated video skims are actually the by-products of those summarization systems. For instance, a general approach in these work is to first locate all keyframes using certain summarization techniques, then either shots or other video segments that
contain these keyframes are assembled to form the skim. Two major
drawbacks exist in these systems: 1) the skim’s semantic flow is dis-
continuous. This is because that keyframes are usually visually different
and temporally apart, thus when we generate a skim by expanding these
keyframes, all visual, audio and motion content continuities might be
lost; 2) the embedded audio cue is ignored in the skim generation pro-
cess. The accompanying audio track usually contains very important
information, especially for movies, but unfortunately, skimming of the
audio source has not been well exploited yet.

This work proposes an event-based video skimming system for fea-
ture films which aims to produce better skims by avoiding the above
drawbacks. Specifically, given the set of extracted movie events, we first
compute six types of low- to high-level features for each event; then we
use these features to evaluate its importance when integrated with user
preference. Finally, important events are assembled to generate the final
skim.

3. Audio and Visual Content Pre-analysis

The first step towards visual content analysis is shot detection. a color
histogram-based approach is employed to perform this task. Specifically,
once a distinct peak is detected in the frame-to-frame histogram diffe-
rence, we declare it as a shot cut (Li and Kuo, 2003). An average of 92.5%
precision and 99% recall rates have been achieved in the current work.
In the second step, we proceed to extract one or more keyframes from
each shot to represent its underlying content. For simplicity, currently
we assign the first and last frames of each shot as its keyframes.

The audio content analysis mainly deals with audio content classi-
fication, where each shot is classified into one of the following four classes:
silence, speech, music, and environmental sounds (Zhang et al. 2001).
Five audio features are extracted for the classification purpose which
include the short-time energy function, the short-time average zero-
crossing rate, the short-time fundamental frequency, the energy band
ratio and the silence ratio. An average of 88% classification accuracy
has been achieved in the current work.

Facial analysis is mainly performed to detect human faces in the
frontal view or faces rotated by plus or minus 10 degrees from the vertical
direction. Currently, we use the face detection and recognition library
provided by the HP Labs (HP Labs, 1998), which reports 85% detection
and recognition accuracy. However, due to the complex motions of movie
casts, we may get both false negatives and false positives. Because false
negatives do not severely affect the system performance, we have thus
developed a simple face tracking scheme to reduce the false positives. Specifically, only faces which appeared in several consecutive frames are retained.

4. Movie Event Extraction

Movie, known as a recording art, is practical, environmental, pictorial, dramatic, narrative and musical (Monaco, 1977). Since a film operates in a limited period of time, all movie shots are efficiently organized by a film-maker in such a way that audiences will follow his or her own way of story-telling. Specifically, this goal is achieved by presenting audiences a sequence of cascaded events that gradually develop the movie plot.

There are basically two ways to develop a thematic topic in an event: through actions or through dialogues. Although they differ a lot in the way to convey the story, both of them may present repetitive visual structures at certain points. This is due to the so-called montage effect as described in (Tarkovsky, 1986), “One of the binding and immutable conditions of cinema is that actions on the screen have to be developed sequentially, regardless of the fact of being conceived as simultaneous or retrospective .... In order to present two or more processes as simultaneous or parallel, you have to show them one after the other, they have to be in sequential montage.” This means that, in order to convey conversations, innuendos or reactions, film-makers have to repeat important shots to express the content and motion continuity.

Our objective in this work is to extract three types of events: 2-speaker dialogues, multiple-speaker dialogues and hybrid events, where a dialogue refers to an actual conversation between two or more people. Figure 1.1 gives two movie dialogue models which are constructed based on the analysis of its editing styles (Reisz et al. 1968). Particularly, Figure 1.1 (a) models a 2-speaker dialogue, and (b) models a multiple-speaker dialogue (here we use three speakers as an example). Each node in the figure represents a shot that contains the indicated speaker(s), and arrows are used to denote the switches between two shots. As we can see from these models that there are certain shot repeating patterns in both cases, although the former one presents more periodic patterns than the latter one. Based on this observation, we propose to extract movie events in the following four steps: 1) shot sink computation, where temporally close and visually similar shots are pooled into a sink; 2) sink clustering and characterization, where each sink is recognized to be either periodic, partly-period, or non-periodic; 3) event extraction and classification; and 4) post-processing with integrated speech and face cues. Each of these steps is detailed in the following subsections.
4.1 Computing Shot Sinks Using Visual Information

Since an event is generally characterized by a repetitive visual structure, our first step is to extract all video paragraphs that possess this feature. A new concept called shot sink is defined for this purpose. Particularly, a shot sink contains a pool of shots which are temporally close and visually similar. Shot sinks are generated using the proposed window-based sweep algorithm as described below.

4.1.1 Window-based Sweep Algorithm. Given shot \( i \), this algorithm will find all shots that are visually similar to \( i \). Considering that an event practically occurs within a certain temporal locality, we naturally restrict this search range to a window of length \( \text{winL} \). To compare the visual similarity of two shots, in principle we should compare every pair of video frames, with each taken from one shot. However, due to the inherent complexity in such an operation, keyframes are usually used in the place of regular frames. This is acceptable since keyframes are shot representatives.

Denote shots \( i \) and \( j \)'s keyframes by \( b_i, e_i \) and \( b_j, e_j \) \( (i < j) \), extracted as described in Section 3, we compute shots \( i \) and \( j \)'s similarity as

\[
\text{Dist}_{i,j} = \frac{1}{4} (w_1 \times \text{dist}(b_i, b_j) + w_2 \times \text{dist}(b_i, e_j)
+ w_3 \times \text{dist}(e_i, b_j) + w_4 \times \text{dist}(e_i, e_j))
\]

where \( \text{dist}(b_i, b_j) \) could be either the Euclidean distance or the histogram intersection between \( b_i \) and \( b_j \)'s color histograms. \( w_1, w_2, w_3 \) and \( w_4 \) are
four weighting coefficients computed as
\[
\begin{align*}
    w_1 &= 1 - \frac{L_i}{\text{winL}}, & w_2 &= 1 - \frac{L_i + L_j}{\text{winL}}, \\
    w_3 &= 1, & w_4 &= 1 - \frac{L_j}{\text{winL}},
\end{align*}
\]

(1.1)

where \(L_i\) and \(L_j\) are shot lengths in the unit of frames. The derivation of these four coefficients is explained as follows. First, since we want to find all similar shots within the window (hence the name “sweep”), we shall not lower their visual similarity because of their physical separation. Thus, we set \(w_3\) to be 1 since \(e_i\) and \(b_j\) form the closest frame pair. Second, due to motion continuity, the similarity between \(b_i\) and \(b_j\) becomes smaller as shot \(i\) gets longer so that we set \(w_1\) to be \(1 - \frac{L_i}{\text{winL}}\) where \(\text{winL}\) is introduced for the normalization purpose. We can derive the formulas for \(w_2\) and \(w_4\) in similar ways.

Now, if \(\text{Dist}_{i,j}\) is less than a predefined threshold \(\text{shotT}\), we consider shots \(i\) and \(j\) to be similar, and put shot \(j\) into shot \(i\)'s sink. Basically we will run this algorithm for every shot. However, if one shot has already been included in a sink, we will skip this shot and continue with the next.

### 4.2 Clustering Shot Sinks Using K-means Algorithm

In this stage, we will cluster and characterize each sink into one of the following three predefined classes: periodic, partly-periodic and non-periodic, based on the evaluated shot repetition degree. For instance, if shot \(i\)'s sink contains shots \(i, \ i+2, \ i+4\) and \(i+6\), we will classify it into the first class since a very strict shot repetition pattern is observed. If shot \(i\)'s sink only contains itself, this sink will be discarded and excluded from further consideration. To quantitatively determine the sink periodicity, we apply the following three steps.

1. For each sink, calculate the relative temporal distance between each pair of neighboring shots. For example, if shot \(i\)'s sink contains shots \(i, \ i+2, \ i+4, \ i+7\) and \(i+10\), then the distance sequence would be 2, 2, 3, 3.

2. Compute mean \(\mu\) and standard deviation \(\sigma\) for each sink’s distance sequence and set them as its features. Thus, for the sink in the above example, it will have mean 2.5 and standard deviation 0.5. Intuitively, a sink belonging to the periodic class will have a smaller standard deviation than the one belonging to the non-periodic class.
3. Group all sinks into the three desired classes using K-means algorithm in terms of their features. With the K-means algorithm which deals with unsupervised clustering, we can circumvent the trouble of determining thresholds. Furthermore, the K-means algorithm is a least-squares partitioning method that naturally divides a collection of objects into $K$ groups. Hence, it is more tolerant to “noisy” data as compared to other approaches.

Figure 1.2 shows the clustering results for two test movies. As we can see, all shot sinks have been well categorized into three groups, where the leftmost group belongs to the periodic class and the rightmost belongs to the non-periodic class.

4.3 Extracting and Classifying Events

At this step, we proceed to extract events by grouping all temporally overlapped sinks into one event. This is because that, no shots which are semantically inter-related with each other will belong to different events, since different events shall have different thematic topics. Moreover, shots that do not belong to these sinks but are physically covered by their temporal ranges will be included into the same event as well. For example, if shot $i$’s sink contains shots $i$, $i + 2$, $i + 4$ and $i + 7$, and shot $i + 1$’s sink contains shots $i + 1$, $i + 3$, and $i + 8$, then they will be grouped into one event ranging from shot $i$ to shot $i + 8$.
After extracting all events, we proceed to classify them into the three desired classes based on the following three heuristically derived rules.

1. If an event contains at least two periodic, at most one partly-periodic, and no non-periodic shot sinks, it is declared as a 2-speaker dialog. This rule is quite intuitive as the camera will basically track the speakers back and forth during a typical movie conversation, thus producing a series of alternating close-up shots.

2. If the event contains several partly-periodic sinks, or if the periodic and non-periodic shot sinks coexist, we label it as a multiple-speaker dialog. The reason to tolerate non-periodic sinks is that, when there are multiple speakers present, we have no control of who will be the next speaker since everyone has an equal opportunity to talk.

3. All remaining events are labeled as the hybrid.

4.4 Integrating Speech and Face Information

Due to the limitation of the pure color information, we have observed the following two types of false alarms in the coarse-level event results: 1) Type I: *Misdetect a conversation-like montage presentation as a spoken dialog.* In these scenarios, the camera usually shuttles back and forth between two silent objects (or humans), thus resulting in a series of repeated shots. Nevertheless, since no dialogs actually goes on in these events, we shall not declare them as 2-speaker dialogs; 2) Type II: *Misclassify a multiple-speaker dialog as a 2-speaker dialog.* This type of false alarm usually occurs when the camera frequently switches between two couples instead of among individual speakers. Errors will also occur in the scenario where one person dominates the dialog while the rest of speakers talk less.

To reduce the Type I false alarm, we integrate the embedded audio information into the detection scheme. Specifically, to be qualified as a spoken dialog, an event should contain a higher ratio of speech content. Detailed processing steps are as follows. First, we classify every shot in the candidate dialog into one of the four audio classes described in Section 3. Then, we calculate the ratio of its contained speech shots. If the ratio is above a certain threshold, we confirm the event to be a dialog; otherwise, we label it as a hybrid event.

To reduce the Type II false alarm, we include the facial cue into the detection scheme. Specifically, for each shot in a 2-speaker dialog, we first perform a face detection on its underlying frames and output the
average of its detected face numbers. Thus if there are \( n \) shots in the
dialog, we will get \( n \) output values. Then, we go check if more than half
of these values are larger than one. If yes, we re-label this event as a
multiple-speaker dialog. This is because that, a 2-speaker dialog should
not have more than one face in most of its component shots if it presents
a periodic shot repeat pattern.

5. Adaptive Speaker Identification

In this module, we aim at identifying target movie casts for the con-
tent indexing purpose. Figure 1.3 shows the proposed system framework
that consists of the following six major blocks: (1) shot detection and
audio classification, (2) face detection, recognition and mouth tracking,
(3) speech segmentation and clustering, (4) initial speaker modeling,
(5) audiovisual (AV)-based speaker identification, and (6) unsupervised
speaker model adaptation. As shown, given an input video, we first split
it into audio and visual streams, then a shot detection is performed on
the visual source. Following this, a shot-based audio classification is
carried out which categorizes each shot into either environmental sound,
silence, music, or speech. Next, with non-speech shots being discarded,
all speech shot are further processed in the speech segmentation and
clustering module where the same person’s speeches are grouped into ho-
ogeneous cluster. Meanwhile, a face detection/recognition and mouth
tracking process is performed on speech shots to recognize talking faces.
Both of the speech and face cues are then effectively integrated to fi-
nalize the speaker identities in the AV-based identification module. Ei-
ther initial or updated speaker models will be needed in this process.
Finally, the identified speaker’s model is updated in the unsupervised
model adaptation module, which will turn into effect in the next round
of identification process.

Because the first module has been discussed in Section 3, below we
will mainly focus on the rest of five modules.

5.1 Face Detection, Recognition and Mouth
 Tracking

The goal of this module is to detect and recognize talking faces in
speech shots.

5.1.1 Face Detection and Recognition. To speed up the
face detection process, here we will only carry out the detection on speech
shots as shown in Figure 1.3. Figure 1.4(a) shows one detection example
where the detected face is boxed by a rectangle and eyes are indicated
Figure 1.3. Block diagram of the proposed adaptive speaker identification system.

by crosses. Also, to facilitate the subsequent recognition process, we organize the detection results into a set of face sequences, where all frames within each sequence contain the same number (nonzero) of human faces.

Figure 1.4. (a) A detected human face, (b) the coarse mouth center, the mouth search area, and two small squares for skin-color determination, and (c) the detected mouth region.

To construct the face database for the recognition purpose, we first ask users to select their $N$ interested casts (i.e., target speakers) during the system initialization by choosing frames that contain the casts’ faces. These faces are then detected, associated with the casts’ names, and stored into the face database.

During the face recognition process, each detected face in the first frame of each face sequence is recognized. The result is returned as a face vector $\mathbf{f} = [f_1, \ldots, f_N]$, where $f_i$ is a value in $[0, 1]$ which indicates the confidence of being target cast $i$.

5.1.2 Mouth detection and tracking. In this step, we will first apply a weighted block matching approach to detect the mouth for the first frame of a face sequence, then we track the mouth for the rest of the frames. Note that if more than two faces are present in the sequence, we will virtually split it into a number of sub-sequences with
each focusing on one face.

1. Mouth detection

According to the facial biometric analogies, we know that there is a certain ratio between the interocular distance and the distance $dist$ between eyes and mouth. Thus, once we obtain the eyes positions from the face detector, which are denoted by $(x_1, y_1)$ and $(x_2, y_2)$, we can subsequently locate the coarse mouth center $(x, y)$ for an upright face. However, when a face is rotated, we need re-calculate its mouth center as (Li et al. 2003)

$$
\begin{align*}
    x &= \frac{x_1 + x_2}{2} \pm dist \times \sin(\theta), \\
    y &= \frac{y_1 + y_2}{2} + dist \times \cos(\theta),
\end{align*}
$$

(1.2)

where $\theta$ is the head rotation angle.

We then expand the coarse mouth center $(x, y)$ into a rectangular mouth search area as shown in Figure 1.4(b), and perform a weighted block-matching process to locate the target mouth. The criterion we used to detect the mouth is that, the mouth area should present the largest color difference from the skin color, which is determined from the average pixel color in the two small under-eye squares as shown in Figure 1.4(b). An example of a correctly detected mouth is shown in Figure 1.4(c). For the rest of the discussion, we denote the detected mouth center by $(cx, cy)$.

2. Mouth tracking

To track the mouth for the rest of frames, we assume that for each subsequent frame, the centroid of its mouth mask can be derived from that of the previous frame as well as from its eye positions. Moreover, we assume that the distance between the coarse mouth center $(x, y)$ and the detected mouth center $(cx, cy)$ remains the same for all frames. Figure 1.5 shows the mouth detection and tracking results on a face sequence containing ten consecutive frames.

Finally, a color histogram-based approach is applied to determine if the tracked mouth is talking. Particularly, if the normalized accumulated histogram difference in the mouth area of the entire or part of the face sequence $f$ exceeds a certain threshold, we label it as a talking mouth; and correspondingly, we mark sequence $f$ as a talking face sequence.

5.2 Speech segmentation and clustering

For each speech shot, the two major speech processing tasks are speech segmentation and speech clustering. In the segmentation step, all indi-
vital for the success of the clustering process. In the clustering step, we group the same speaker’s segments into homogeneous clusters so as to facilitate the successive identification process.

5.2.1 Speech segmentation. A 2-step process is applied to separate speech from the background: 1) given the audio signal of a speech shot, we first sort all audio frames into an array based on their energies, then we quantize all frames into $N$ bins. Next, the threshold $T$ which separates speech and silence is determined from the average energies in the first and last three bins (Li et al. 2003); 2) a 4-state transition diagram (Li and Zheng, 2001) is then employed to extract the speech segments as shown in Figure 1.6. Particularly, the transition conditions between two states are labelled on each edge, and the corresponding actions are described in parentheses. As we can see, this state machine basically groups blocks of continuous silence/speech frames as silence/speech segments while removing impulsive noises at the same time.

5.2.2 Speech clustering. Speech clustering has been studied for a long time, and many approaches have been proposed. In this work, we use Bayesian Information Criterion (BIC) to measure the similarity between two speech segments (Chen et al. 1998).

When comparing two segments using the BIC, the distance measure can be stated as a model selection criterion where one model is represented by two separate segments $X_1$ and $X_2$, and the other model represents the joined segment $X = \{X_1, X_2\}$. The difference between these two modeling approaches equals

$$\Delta BIC(X_1, X_2) = \frac{1}{2}(M_2 \log |\Sigma| - M_1 \log |\Sigma_1|)$$
where $\Sigma_1$, $\Sigma_2$, $\Sigma$ are $X_1$, $X_2$ and $X$’s covariance matrices, and $M_1$, $M_2$, $M_{12}$ are their respective feature vector numbers. $\lambda$ is a penalty weight and equals 1 in this case. $d$ gives the dimension of the feature space. According to the BIC theory, if $\Delta BIC(X_1, X_2)$ is negative, the two speech segments, $X_1$ and $X_2$, can be considered from the same speaker.

Now, assume cluster $C$ contains $n$ homogeneous speech segments, then given a new speech segment $X$, we compute $Dist(X, C)$ as: $Dist(X, C) = \Sigma_{i=1}^{n} w_i \times \Delta BIC(X, X_i)$, where $w_i = M_i / \sum_{j=1}^{n} M_j$. Finally, if $Dist(X, C)$ is less than 0, we merge $X$ to cluster $C$; otherwise, if none of existing clusters is matched, a new cluster will be initialized.

### 5.3 Initial speaker modeling

To bootstrap the identification process, we need initial speaker models as shown in Figure 1.3. This is achieved by exploiting the inter-relations between the face and speech cues. Specifically, for each target cast $A$, we first find a speech shot that $A$ is talking based on the face detection and recognition result. Then, we collect all of its speech segments as described in Section 5.2.1, and build $A$’s initial model. The Gaussian Mixture Model (GMM) has been employed here for the modeling purpose. Note that at this stage, the initial model will only contain...
one Gaussian component with its mean and covariance computed as the global ones due to the limitation of training data.

5.4 Likelihood-based speaker identification

At this stage, we will identify speakers based on pure speech information. Specifically, given a speech signal, we first decompose it into a set of overlapped audio frames; then 14 Mel-frequency cepstral coefficients (Reynolds et al. 1995) are extracted from each frame to form an observation sequence $X$. Finally, we calculate the likelihood $L(X; M_i)$ between $X$ and all speaker models $M_i$, and obtain a speaker vector $\hat{\theta}$.

5.4.1 Likelihood calculation. Because the Gaussian mixture density can provide a smooth approximation to the underlying long-term sample distribution of a speaker’s utterances (Reynolds et al. 1995), we choose to use GMM to model speakers in this work. Particularly, a GMM model $M$ can be represented by the notation $M = \{p_j, \mu_j, \Sigma_j\}, j = 1, \ldots, m$, where $m$ is the total number of components in $M$, and $p_j$, $\mu_j$, $\Sigma_j$ are the weight, mean vector and covariance matrix of the $j$th component, respectively.

Now, let $M_i$ be the GMM model corresponding to the $i$th enrolled speaker with $M_i = \{p_{ij}, \mu_{ij}, \Sigma_{ij}\}$, and let $X$ be the observation sequence consisting of $T$ cepstral vectors $\bar{x}_t$, $t = 1, \ldots, T$, under the assumption that all observation vectors are independent, the log likelihood $\ell(X; \mu_{ij}, \Sigma_{ij})$ between $X$ and $M_i$ can be computed as (Mardia et al. 1979)

$$\ell(X; \mu_{ij}, \Sigma_{ij}) = -\frac{T}{2} \log |2\pi \Sigma_{ij}| - \frac{T}{2} \text{tr}(\Sigma_{ij}^{-1} S) - \frac{T}{2} (\bar{X} - \mu_{ij})' \Sigma_{ij}^{-1} (\bar{X} - \mu_{ij}),$$

(1.3)

where $S$ and $\bar{X}$ are $X$’s covariance and mean, respectively.

Based on this identification scheme, a speaker vector $\hat{\theta} = [v_1, \ldots, v_N]$ can be obtained for each cluster $C$, where $v_i$ is a value in $[0, 1]$ which equals the normalized log likelihood value $\ell(X; \mu_{ij}, \Sigma_{ij})$, and indicates the confidence of being target speaker $i$.

5.5 Audiovisual integration for speaker identification

This step aims at finalizing the speaker identification task for cluster $C$ (in shot $S$) by integrating the audio and visual cues obtained in Sections 5.1, 5.2 and 5.4. Specifically, given cluster $C$ and all recognized talking face sequences $F$ in $S$, we examine if there is a temporal overlap between
$C$ and any sequence $F_i$. If yes, we assign $F_i$’s face vector $\tilde{f}$ to $C$ if the overlap ratio exceeds a threshold. Otherwise, we set $C$’s face vector to null. However, if $C$ is overlapped with multiple $F_i$ due to speech clustering or talking face detection errors, we choose the one with the highest overlap ratio.

Now, we determine the speaker’s identity in cluster $C$ as

$$\text{speaker}(C) = \arg \max_{1 \leq j \leq N} (w_1 \cdot f[j] + w_2 \cdot v[j]),$$

(1.4)

where $\tilde{f}$ and $\tilde{v}$ are $C$’s face and speaker vectors, respectively. $N$ is the total number of target speakers. $w_1$ and $w_2$ are two weights that sum up to 1.0. Currently we set them to be equal.

5.6 Unsupervised Speaker Model Adaptation

Now, after we identify speaker $P$ for cluster $C$, we will update his model using $C$’s data in this step. Meanwhile, a background model will be either initialized or updated to account for all non-target speakers. Specifically, when there is no a priori background model, we use $C$’s data to initialize it if the minimum of $L(C; M_i), i = 1, \ldots, N$ is less than a preset threshold. Otherwise, if the background model produces the largest likelihood, we denote the identified speaker as “unknown” and use $C$’s data to update the background model.

The following three approaches are investigated to update the speaker model: Average-based model adaptation, MAP-based model adaptation, and Viterbi-based model adaptation.

5.6.1 Average-based Model Adaptation. In this approach, $P$’s model is updated in the following three steps.

Step 1: Compute BIC distances between cluster $C$ and all of $P$’s mixture component $b_j$. Denote the component that gives the minimum distance $d_{\text{min}}$ by $b_0$.

Step 2: If $d_{\text{min}}$ is less than an empirically determined threshold, we consider $C$ to be acoustically close to $b_0$, and use $C$’s data to update this component. Specifically, let $N(\mu_1, \Sigma_1)$ and $N(\mu_2, \Sigma_2)$ be $C$ and $b_0$’s Gaussian models, respectively, we update $b_0$’s mean and covariance as (Mokbel, 2001)

$$\mu_2 = \frac{N_1}{N_1 + N_2} \mu_1 + \frac{N_2}{N_1 + N_2} \mu_2,$$

(1.5)

$$\Sigma_2 = \frac{N_1}{N_1 + N_2} \Sigma_1 + \frac{N_2}{N_1 + N_2} \Sigma_2 + \frac{N_1 N_2}{(N_1 + N_2)^2} (\mu_1 - \mu_2)(\mu_1 - \mu_2)^T,$$

(1.6)
where \( N_1 \) and \( N_2 \) are the numbers of feature vectors in \( C \) and \( b_0 \), respectively.

Otherwise, if \( d_{\text{min}} \) is larger than the threshold, we will initialize a new mixture component for \( P \) with its mean and covariance equaling to \( \mu_1 \) and \( \Sigma_1 \). However, once the total number of \( P \)'s components reaches a certain value (which is set to 32 in this work), only component adaptation is allowed. This is adopted to avoid having too many Gaussian components in each model.

Step 3: Update the weight for each of \( P \)'s mixture component.

5.6.2 MAP-based Model Adaptation. MAP adaptation has been widely and successfully used in speech recognition, yet it has not been well explored in speaker identification. In this work, due to the limited speech data, only Gaussian means will be updated. Specifically, given \( P \)'s model \( M_p \), we update component \( b_i \)'s mean \( \mu_i \) via

\[
\mu'_i = \frac{L_i}{L_i + \tau} \tilde{\mu} + \frac{\tau}{L_i + \tau} \mu_i, \tag{1.7}
\]

where \( \tau \) defines the "adaptation speed" and is currently set to 10.0. \( L_i \) gives the occupation likelihood of the adaptation data to component \( b_i \), and is defined as

\[
L_i = \sum_{t=1}^{T} p(i|\vec{x}_t, M_p), \tag{1.8}
\]

where \( p(i|\vec{x}_t, M_p) \) is the a posteriori probability of \( \vec{x}_t \) to \( b_i \). Finally, \( \tilde{\mu} \) gives the mean of the observed adaptation data, and is defined as

\[
\tilde{\mu} = \frac{\sum_{t=1}^{T} p(i|\vec{x}_t, M_p) \vec{x}_t}{\sum_{t=1}^{T} p(i|\vec{x}_t, M_p)}. \tag{1.9}
\]

Unlike the previous method, this MAP adaptation is applied to every component of \( P \) based on the principle that every feature vector has a certain possibility of occupying every component. Thus, MAP adaptation provides a soft decision on which feature vector belongs to which component.

5.6.3 Viterbi-based Model Adaptation. Similar to the MAP-based approach, this approach also allows different feature vectors belonging to different components. Nevertheless, while the MAP approach provides a soft decision, this approach implies a hard decision, i.e., for any one particular feature vector \( \vec{x}_t \), it can either occupy component \( b_i \) or not. Therefore, the probability function \( p(i|\vec{x}_t, M_p) \) in Equation 1.8 is now replaced by an indicator function which is either 0 or
1. Now, given any feature vector $\tilde{x}_t$, the mixture component it occupies will be determined by

$$m_0 = \text{arg} \max_{1 \leq i \leq m} p(d|\tilde{x}_t, M_p).$$

Finally, Equations 1.5 and 1.6 are used to update $P$’s components after we assign every feature vector to its belonged component. As one can see, this approach is actually a compromise between the previous two methods.

To summarize, based on the proposed model adaptation approaches, a speaker model will grow from 1 Gaussian mixture component up to 32 components as we go through the entire movie sequence.

6. Event-based Movie Skimming

From the discussion in Section 4, we know that the entire movie content can be compactly represented by a set of events, with each of them containing an individual thematic topic and having continuous audiovisual contents. These events thus form the candidate skimming components.

The proposed movie skimming system consists of the following two steps: 1) event feature extraction. Six types of low- to high-level features are extracted from each event using the content analysis techniques described earlier. These features are then used to evaluate the event importance when integrated with user’s preference; 2) movie skim generation. Selected important events are assembled to generate the final skim at this stage based on either user’s preference or a set of system-defined judging rules. Each of these two steps are detailed below.

6.1 Event Feature Extraction

For simplicity, we denote the current event by $EV$, and the number of its component shots by $L$.

1. Music ratio. The music ratio $MR$ of an extended event is computed as

$$MR = \frac{\sum_{l=1}^{L} M_l}{L},$$

where $M_l$ is a binary number. $M_l$ equals 1 when shot $l$ in $EV$ contains music; otherwise, it equals 0. Based on this ratio, we classify $EV$ into the following three categories:

- low music ratio, $MR < 0.3$
- medium music ratio, $0.3 \leq MR \leq 0.6$
- high music ratio, $MR > 0.6$
2. **Speech ratio.** The speech ratio \( SR \) is computed as

\[
SR = \frac{\sum_{l=1}^{L} s_l}{L},
\]

where \( s_l \) is a binary number. \( s_l \) equals 1 when shot \( l \) in \( EV \) contains speech signals; otherwise, it is set to 0. Similarly, \( EV \) is also classified into the three categories based on this ratio.

3. **Sound loudness.** The sound loudness \( SL \) is used to indicate if the current event has a loud and noisy background. Because a shot containing loud speeches or loud music cannot be claimed as a noisy shot, thus \( SL \) is only computed as the average sound energy for all environmental shots. Moreover, for the sake of event importance ranking, \( SL \) is further normalized to the range of \([0,1]\) by dividing the largest \( SL \) value of the entire movie. \( SL \) is a very effective parameter for identifying events of high actions since they usually contain loud background sound. The sound loudness is classified into low, medium and high three levels as well.

4. **Action level.** The action level \( AL \) is used to indicate the amount of motions involved in the current event, and is computed as the average motions contained in its component frames (Li and Kuo, 2003). For the same purpose, the value of \( AL \) is also normalized to the range of \([0,1]\) over the entire movie. This parameter can be used to roughly distinguish a low-motioned scene such as a dialog from a highly-motioned scene such as an explosion. The action level is also classified into low, medium and high three levels.

5. **Present cast.** The present cast \( PC \) not only includes the identified speakers but also contains the cast whose face is recognized in the current event. This parameter is used to indicate if certain user-interested movie characters are present in the event. Currently, we only provide the top five user-interested casts.

6. **Theme topic.** The theme topic \( TT \) in the current event corresponds to one of the three event types: the 2-speaker dialog, the multiple-speaker dialog, and other general events.
Based on these features, we define an attribute matrix $A$ for an incoming movie as

$$A = \begin{bmatrix}
a_{1,1} & a_{1,2} & \cdots & a_{1,M} \\
a_{2,1} & \cdots & \cdots & a_{2,M} \\
\vdots & \cdots & \cdots & \vdots \\
a_{N,1} & a_{N,2} & \cdots & a_{N,M}
\end{bmatrix},$$

where $M$ is the total number of features extracted from each event (which equals 6 in the current case), $N$ is the total number of events in the current movie, and $a_{i,j}$ is the value of the $j$th feature in the $i$th event. For example, if $a_{1,1} = 0.4$, $a_{2,5} = 0 \cup 1 \cup 3$, and $a_{4,6} = 0$, it means that the first event has a music ratio of 0.4, the second one has 3 present casts with casts 0, 1 and 3, and the fourth one is a 2-speaker dialog.

### 6.2 Movie Skim Generation

At this step, we generate the movie skim by choosing important events from the candidate set based on user’s preference. A user can specify his or her preference on all features using a preference vector $\vec{p} = [p_1, p_2, \ldots, p_M]^T$. For example, the user’s preference on the music ratio, speech ratio, sound loudness and action level could be “low”, “medium”, “high” and “no preference”. For feature $PC$, $p_5$ can be a combination of all desired casts such as “0 $\cup$ 3”, or simply “no preference”. As for feature $p_6$ on the theme topic, it can be a union of the numbers between 0 to 3, with 0 being “the 2-speaker dialog”, and 3 being “no preference”. Finally, the user can also specify his or her desired skim length.

Given the preference vector $\vec{p}$, we then compute the event importance vector $\vec{e}$ as

$$\vec{e} = A \odot \vec{p}$$

$$\begin{bmatrix}
a_{1,1} & a_{1,2} & \cdots & a_{1,M} \\
a_{2,1} & \cdots & \cdots & a_{2,M} \\
\vdots & \cdots & \cdots & \vdots \\
a_{N,1} & a_{N,2} & \cdots & a_{N,M}
\end{bmatrix} \odot \begin{bmatrix} p_1 \\
p_2 \\
p_3 \\
p_M
\end{bmatrix} = \begin{bmatrix} e_1 \\
e_2 \\
e_3 \\
e_M
\end{bmatrix}$$

where

$$e_i = a_{i,1} \odot p_1 + \ldots + a_{i,M} \odot p_M, \quad (1.13)$$

with “$\odot$” being a mathematical operator that functions as a logical “AND”. For instance, if $a_{1,1} = 0.2$, and $p_1 = low$, we have $a_{1,1} \odot p_1 = 1$ since $a_{1,1}$ denotes a low music ratio which is consistent with $p_1$. Otherwise, we have $a_{1,1} \odot p_1 = 0$. Similarly, we can define the operations for $a_{i,2} \odot p_2$, $a_{i,3} \odot p_3$, and $a_{i,4} \odot p_4$. For the operation between $a_{i,5}$ and
p_5$, we define it as, if $a_{i,5}$ contains at least one of the casts listed in $p_5$,

$a_{i,5} \odot p_5 = 1$. Otherwise, it is 0. The same definition applies to $a_{i,6} \odot p_6$
as well. Finally, in case that $p_j$ is set to be “no preference”, we have

$a_{i,j} \odot p_j = -1$.

Now, given that each event has a score $e_i$ that ranges from -6 to 6, we

\[ e_i \in [-6, 6] \]

can generate the final skim by selecting the events that maximize the cumulative score in the resulting skim while preserving the desired skim length at the same time. This could be viewed as an example of the 0-1 knapsack problem defined as (Martello et al. 1990)

\[
\text{To maximize } \sum_{i=1}^{N} w_i e_i, \quad \text{subject to } \sum_{i=1}^{N} w_i T_i \leq T, \quad (1.14)
\]

where $w_i$ is a binary variable, which equals 1 when the $i$th event is selected for the skim and 0, otherwise. $T_i$ is its temporal duration and $T$ is the desired skim length.

To solve this knapsack problem, we first sort all events based on their scores in a descending order. Then, a greedy selection algorithm is applied which starts by selecting the toppest event. We then keep on selecting the next event as long as its duration is less than the remaining skim time. This process stops when the cumulative skim duration exceeds the target length. Finally, all selected events are concatenated to form the final skim. Note that, in order to make the final skim more appealing to users, we can use certain predefined transition effects such as fade in/out, checkerBoard, dissolve or wipe, to gracefully concatenate events. Moreover, to include as many events as possible, we could remove silence shots from the selected dialogs since they usually do not contain important messages.

6.3 Discussion

Up to now, a theoretically complete movie skimming system has been proposed. There are, however, practical considerations to be included for more satisfactory results.

6.3.1 When More Judging Rules Are Needed. Since

the event score $e_i$ has a relatively narrow value range, different events may share the same score. Consequently, new rules are needed to distinguish their importance. Moreover, when the user has no preference on any features, we need a set of default rules to generate the skim. Possible solutions are given below to each of these two scenarios: 1) ask more user preferences such as the cast’s ranking from the most interested to the least, or apply more strict rules so that no event that
contains user-uninterested casts is selected; 2) derive judging rules from the movie genres. We know that different types of movies have different, yet stereotyped ways to attract their audience. For instance, romances usually attract audience with touching love stories, while actions tend to use many thrilling scenes with deafening sounds and crazy actions. Thus dialog or music scenes are more informative in romances while action scenes are more important to thrillers than to comedies.

6.3.2 Sub-sampling the Video Skim. To include as many contents as possible within a limited skimming time, we have added a video sub-sampling scheme to this work. Specifically, given the desired skim length \( L \), we first generate an \( \alpha L \)-long skim with \( \alpha \) greater than 1. Then, we play 1 out of every \( \alpha \) frames during the skim playback, thus to maintain its target duration. This fast forward-like display is acceptable since there is a certain temporal redundancy between neighboring video frames. Currently, we choose \( \alpha \) between 1 and 2.5.

To maintain a comprehensible audio quality in the skim, and also to make the accompanying audio track be consistent with the compressed image sequence, we have also compressed the audio track with the same compression ratio. The MWSOLA (Modified Waveform Similarity Overlap-And-Add) technique proposed in (Liu et al. 2001) is employed for this purpose. Specifically, MWSOLA is a time-domain scaling approach that modifies the signal's time duration while preserving its important acoustic attributes such as pitch and timber.

6.3.3 Discovering the Story Structure. Because feature films have elaborately designed and meticulously edited story structures, we are able to generate more informative movie skims if we could discover them. According to (Block, 2001), a movie's story structure is usually instantiated by its visual structure, which could be detected and extracted by analyzing seven basic visual components including space, line, shape, tone, color, movement, and rhythm.

A typical movie story contains three basic parts: the beginning (exposition), the middle (conflict), and the end (resolution). The exposition part gives the facts that are needed to begin the story such as the main characters. The middle part contains the rising actions or the conflict, and as the story develops, the conflict increases in intensity. The most intense part of the movie is the climax where the conflict gets resolved. The resolution part wraps up incomplete story elements and ends the story.

It is apparent that, if we can extract the story structure as discussed, we are able to include more informative contents in the skim. For in-
stance, when the plot reaches the climax, we shall give it more weights since it usually attracts audience attention. In contrast, for the exposition and resolution parts, a brief explanation would be enough.

7. Experimental Results

7.1 Event Detection Results

For all the experiments reported in this section, video streams are compressed in MPEG-1 format with a frame rate of 29.97 frames/sec. To validate the effectiveness of the proposed approach, representatives of various movie genres were tested. Specifically, the test set includes Movie1 ("The Legend of the Fall", a tragic romance), Movie2 ("When Harry Met Sally", a comedic drama) and Movie3 ("Braveheart", an action movie). Each movie clip is approximately 1 hour long.

Due to the inherent subjectivity of the event definition, we do not attempt to discuss the appropriateness of extracted events since people's opinions may differ. Instead, we will only examine the correctness of the event classification results, for which it is easier to reach a consensus. Experimental results are shown in Table 1.1 for all 3 movies which contain 80 events in total. Also, because the hybrid class contains all events excluding the dialogs, it is omitted from these tables. Precision and recall rates are computed to evaluate the system performance, where

\[
\text{Precision} = \frac{\text{hits}}{\text{hits} + \text{false alarms}}, \quad \text{Recall} = \frac{\text{hits}}{\text{hits} + \text{misses}}.
\]

As shown in this table, encouraging event extraction results have been achieved. When the speech and facial cues are integrated, both precision and recall rates reach 83% in all three movies, which is very encouraging. Regarding the misses observed in the table, the missed 2-speaker dialog in Movie1 was misclassified as a hybrid event, where one of the speakers was walking all time, which resulted in a frequent background change and therefore an irregular periodicity. In Movie3, a multiple-speaker dialog was misdetected due to the reason that people were talking in a too random fashion in that scene, thus an irregular shot repeat pattern was resulted.

7.2 Speaker Identification Results

To evaluate the performance of the proposed adaptive speaker identification system, studies have been carried out on above three movies. However, due to the space limit, only the results on Movie2 will be reported here.
Table 1.1. Event detection results for Movie1, Movie2 and Movie3

<table>
<thead>
<tr>
<th>Movie1 – Tragic Romance</th>
<th>Combining Speech &amp; Face Cues</th>
<th>Without Speech/Face Cues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event</td>
<td>Hit</td>
<td>Miss</td>
</tr>
<tr>
<td>M-spk</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>2-spk</td>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Movie2 – Comedic Drama</th>
<th>Combining Speech &amp; Face Cues</th>
<th>Without Speech/Face Cues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event</td>
<td>Hit</td>
<td>Miss</td>
</tr>
<tr>
<td>M-spk</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>2-spk</td>
<td>14</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Movie3 – Action</th>
<th>Combining Speech &amp; Face Cues</th>
<th>Without Speech/Face Cues</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event</td>
<td>Hit</td>
<td>Miss</td>
</tr>
<tr>
<td>M-spk</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>2-spk</td>
<td>13</td>
<td>0</td>
</tr>
</tbody>
</table>

Three interested characters were chosen for Movie2, and totally 952 speech clusters were generated. An average of 90% clustering purity was achieved which is defined as the ratio between the number of segments from the dominant speaker and the total number of segments in a cluster. Regarding the talking face detection, we have achieved 83% precision and 88% recall rates on 425 detected face sequences. However, the ratio of talking face-contained frames over the total number of video frames is as low as 11.5%. This is because movie casts are always in constant moving status, thus making it difficult to detect their faces.

The identification results for all obtained speech clusters are reported in the form of a confusion matrix as shown in Table 1.2. The three speakers are indexed by A, B, C, and their corresponding movie characters are denoted by A’, B’ and C’. “Unknown” is used for all non-target speakers. The number in each grid, say grid (A’, B), indicates the number of speech segments where character A’ is talking yet actor B is identified. Obviously, the larger the number in the diagonal, the better the performance. Three parameters, namely, false acceptance (FA), false rejection (FR) and identification accuracy (IA) are calculated to evaluate the system performance. Particularly, for each cast or character, we have

\[
FR = \frac{\text{sum of off-diagonal numbers in the row}}{\text{sum of all numbers in the row}},
\]
\[ FA = \frac{\text{sum of off-diagonal numbers in the column}}{\text{sum of all numbers in the column}} , \]

\[ IA = 1 - FR. \]

Table 1.2(a) gives the identification result when the average-based model adaptation is applied. An average of 75.3\% IA and 22.3\% FA are observed. Result obtained from the MAP-based approach is given in Table 1.2(b) where we have an average 78.6\% IA and 21\% FA. This result is slightly better than that in (a), yet at the cost of a higher computation complexity. Table 1.2(c) shows the result for the Viterbi-based approach. As we can see, this table presents the best performance with an average 82\% IA and 20\% FA. The fact that this approach outperforms the MAP approach may imply that, for speaker identification, a hard decision would be good enough.

By carefully studying the results, we found two major factors that degrade the system performance: (a) imperfect speech segmentation and clustering, and (b) inaccurate facial analysis results. Due to the various sounds/noises existing in movies, it is extremely difficult to achieve perfect speech segmentation and clustering. Besides, incorrect facial data can result in mouth detection and tracking errors, which will further affect the identification accuracy.

To determine the upper limit of the number of mixture components in each speaker model, we examined the average identification accuracy in terms of 32 and 64 components for all three adaptation methods and plotted them in Figure 1.7(a). As shown, except for the average-based method where a similar performance is observed, the use of 32 Gaussian mixture components has produced a better performance.

Finally, the average identification accuracy obtained by using or without using face cues is compared in Figure 1.7(b). Clearly, without the assistance of face cue, the system performance has been significantly degraded, especially for the average-based adaptation approach. This indicates that the face cue plays an important role in model adaptation.

Figure 1.8 gives a detailed description of a speaker identification example. Specifically, the upper part shows the waveform of an audio signal recorded from a speech shot where two speakers take turns to talk. The superimposed pulse curve illustrates the speech-silence separation result where all detected speech segments are bounded by the passband of the curve. The speaker identity for each speech segment is given right below this sub-figure, where speakers A and B are represented by dark- and light-colored blocks, respectively. The likelihood-based speaker identification result is given right below the ground truth, where only the toppest speaker's identity is shown. Besides, since two of the speech
Table 1.2. Adaptive speaker identification results for Movie2 using: (a) the average-based, (b) the MAP-based, and (c) the Viterbi-based model adaptation approaches.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Ukwn</th>
<th>FR</th>
<th>IA</th>
</tr>
</thead>
<tbody>
<tr>
<td>A’</td>
<td>228</td>
<td>42</td>
<td>6</td>
<td>32</td>
<td>26%</td>
<td>74%</td>
</tr>
<tr>
<td>B</td>
<td>37</td>
<td>281</td>
<td>35</td>
<td>24</td>
<td>29%</td>
<td>75%</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
<td>9</td>
<td>115</td>
<td>16</td>
<td>23%</td>
<td>77%</td>
</tr>
<tr>
<td>Ukwn</td>
<td>10</td>
<td>15</td>
<td>4</td>
<td>88</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FA</td>
<td>20%</td>
<td>19%</td>
<td>28%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(a)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Ukwn</th>
<th>FR</th>
<th>IA</th>
</tr>
</thead>
<tbody>
<tr>
<td>A’</td>
<td>239</td>
<td>25</td>
<td>22</td>
<td>22</td>
<td>22%</td>
<td>78%</td>
</tr>
<tr>
<td>B’</td>
<td>59</td>
<td>302</td>
<td>5</td>
<td>11</td>
<td>20%</td>
<td>80%</td>
</tr>
<tr>
<td>C’</td>
<td>10</td>
<td>8</td>
<td>117</td>
<td>15</td>
<td>22%</td>
<td>78%</td>
</tr>
<tr>
<td>Ukwn</td>
<td>19</td>
<td>16</td>
<td>7</td>
<td>75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FA</td>
<td>27%</td>
<td>14%</td>
<td>22%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(b)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>Ukwn</th>
<th>FR</th>
<th>IA</th>
</tr>
</thead>
<tbody>
<tr>
<td>A’</td>
<td>246</td>
<td>29</td>
<td>13</td>
<td>20</td>
<td>20%</td>
<td>80%</td>
</tr>
<tr>
<td>B’</td>
<td>41</td>
<td>317</td>
<td>13</td>
<td>6</td>
<td>16%</td>
<td>84%</td>
</tr>
<tr>
<td>C’</td>
<td>10</td>
<td>10</td>
<td>123</td>
<td>7</td>
<td>18%</td>
<td>82%</td>
</tr>
<tr>
<td>Ukwn</td>
<td>18</td>
<td>22</td>
<td>14</td>
<td>63</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FA</td>
<td>22%</td>
<td>14%</td>
<td>24%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(c)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(a) Identification Accuracy Comparison vs. 32-component
(b) Identification Accuracy Comparison vs. 64-component

Figure 1.7. Identification accuracy comparison for the average-based, the MAP-based, and the Viterbi-based approaches with: (a) 32-component vs. 64-component for speaker models, and (b) using vs. without using face cues.

segments are too short (indicated by the circles) for speaker identification, we will disregard them in later processes. As shown, there are two false alarms in this result where B is falsely recognized as A twice. The
talking face detection result is shown in the next sub-figure including both talking and non-talking cases. Finally, the last sub-figure shows the ultimate identification result obtained from integrating both speech and face cues as discussed in Section 5.5. As shown, although the first error still exists as the face cue cannot offer help, the second error has been corrected.

7.3 Movie Skimming Results

Since it is difficult to qualitatively evaluate a video skimming system, we have carried out a preliminary user study to quantitatively measure the system performance. Specifically, we have designed the following six statements, and asked each participant to assess them on a 5-point scale (1-5), where 1 stands for “strongly disagree”, and 5 for “strongly agree”: 1) visual comprehension: “the visual quality of the skim is good, no jerky motions”; 2) audio comprehension: “the audio quality is good, no staccato speeches”; 3) semantic continuity: “the embedded semantic flow is continuous and understandable”; 4) well abstraction: “the generated skim can well summarize the movie”; 5) quick browsing: “this skimming system can help me quickly browse the movie content”; and 6) video skipping. “I can skip watching the original movie by only viewing the skim”.

We performed the study on both Movie1 and Movie2. Two graduate students were invited to participate in the experiment who are familiar with one movie on the average. The survey results are reported in Table 1.3.

<table>
<thead>
<tr>
<th>Questions</th>
<th>Score</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Std. deviation</td>
</tr>
<tr>
<td>Visual comprehension</td>
<td>4.8</td>
<td>0.21</td>
</tr>
<tr>
<td>Audio comprehension</td>
<td>4.62</td>
<td>0.17</td>
</tr>
<tr>
<td>Semantic continuity</td>
<td>4.4</td>
<td>0.36</td>
</tr>
<tr>
<td>Well abstraction</td>
<td>4.5</td>
<td>0.22</td>
</tr>
<tr>
<td>Quick browsing</td>
<td>4.7</td>
<td>0.15</td>
</tr>
<tr>
<td>Video skipping</td>
<td>4.15</td>
<td>0.78</td>
</tr>
</tbody>
</table>

From this table, we see that encouraging results have been obtained in the study. Both participants were very satisfied with the skim’s audio and visual quality. Moreover, since the event forms the building block of the skim, the underlying semantic flow is continuous and un-
Figure 1.8. A detailed description of a speaker identification example.
REFERENCES

nderstandable. Overall, they agreed that the generated skims have well summarized the movie content, and can help them quickly browse the sequence. However, when we asked them if they would skip watching the original movie by only viewing the skim, they were not definitive since the original movie obviously contains much richer information. Finally, they suggested that a scalable video skim would be interesting and helpful.

8. Conclusion

A content-based movie analysis, indexing and skimming system is presented in this chapter. Various media cues such as face, audio and visual information, have been employed to extract high-level video semantics (event and speaker identity), as well as represent movie contents in a compact, yet meaningful manner. Although feature films have been our major focus, the methodology presented here could be easily extended to other types of generic videos. More robust event extraction results could be achieved by integrating other image/video processing techniques such as human/object tracking. Finally, representing the system output information in MPEG-7 standardized description format would be another interesting topic.

Acknowledgments

This research has been partially funded by the Integrated Media Systems Center, a National Science Foundation Engineering Research Center, under Cooperative Agreement Number EEC-9529152 and partially funded by the Hewlett-Packard Company. The authors would also like to acknowledge the HP Labs in Palo Alto, CA, for providing the face detection and recognition library.

References

REFERENCES


