Adaptive and Multimodal Approach to Multimedia Content Analysis

D I S S E R T A T I O N

for the Degree of

Doctor of Philosophy (Electrical Engineering)

Zhu Liu

January 2001

ADAPTIVE AND MULTIMODAL APPROACH TO MULTIMEDIA CONTENT ANALYSIS

D I S S E R T A T I O N

Submitted in Partial Fulfillment
of the REQUIREMENTS for the

Degree of

DOCTOR OF PHILOSOPHY (Electrical Engineering)

at the

POLYTECHNIC UNIVERSITY

by

Zhu Liu

January 2001

Approved:

____
Department Head

____
Date

Copy No. ____

Approved by the Guidance Committee:

Major: Electrical Engineering

Yao Wang
Professor of
Electrical Engineering

Date

Qian Huang
Principal Technical Staff Member of
AT&T Labs - Research

Date

Onur G. Guleryuz
Assistant Professor of
Electrical Engineering

Date

Minor: Computer Science

Edward K. Wong
Associate Professor of
Computer and Information Science

Date

VITA

Zhu Liu was born in China on June 21, 1972. He received the B.S. and M.S. degrees in Electronic Engineering from Tsinghua University, Beijing, China, in 1994 and 1996, respectively. He was two times recipient of Motorola Scholarship, and won the honor of Excellent Graduate in Tsinghua University. Since September 1996, he has been a Ph. D. candidate at Electrical Engineering Department in Polytechnic University, Brooklyn, NY, under the guidance of Prof. Yao Wang. From May 1998 to August 1999, he was with the Multimedia Processing Department, AT&T Labs - Research, Red Bank, NJ as a consultant. During his thesis study, he published 18 technical papers, and had 3 patents pending. He has conducted research in the fields of audio/video signal processing, multimedia database, and pattern recognition. He is a student member of IEEE and member of Tau Beta Pi.

To my wife Chun Jin and my grateful parents
for their love and support.

ACKNOWLEDGEMENT

First, I would like to thank my thesis advisor, Professor Yao Wang, for her invaluable support and guidance throughout the course of this dissertation. Her extensive knowledge, enlightening direction, and continuous encouragement make my thesis work smooth and positive.

I am also indebted to other members of my dissertation committee. Dr. Qian Huang and Professor Edward K. Wong have closely followed my research and many ideas come from discussions with them. I am also benefited very much from Dr. Qian's thorough writing abilities and acute insights in academic researches. I wish to thank Professor Onur G. Guleryuz for many constructive suggestions in my thesis work.

I wish to acknowledge the members of the image processing group, including Jincheng Huang, Ru-Shang Wang, Zhong Ke Wang, Yu Chen, and Doo-man Chung. I thank them for many discussions, cooperation, and assistance. I am also very grateful to my colleagues at AT&T Labs - Research: Dr. Behzad Shahraray, David Gibbon, Dr. Aaron E. Rosenberg, and Dr. Atul Puri for their valuable financial and technical supports. My 18 month work at AT&T makes up a very important part of my thesis and provides me many inspirations on later research.

In particular, I want to thank my dear wife, Chun Jin. Without her meticulous caring and continuous support, the achievements of my thesis are impossible. Finally, I thank my parents for their persistent emotional support.

Part of this work was supported by the National Science Foundation through its STIMULATE program under Grant No. IRI-9619114.

AN ABSTRACT

ADAPTIVE AND MULTIMODAL APPROACH TO MULTIMEDIA CONTENT ANALYSIS

by

Zhu Liu

Advisor: Yao Wang

Submitted in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy (Electrical Engineering)

January 2001

The volume of multimedia data generated nowadays is exploding. To efficiently access and retrieve desired information, tools that enable automated analysis based on content are becoming indispensable. Multimedia content is defined at both perceptual and conceptual levels. The former refers to the content characterized purely by intrinsic perception properties such as color, motion, or acoustic features. The latter refers to the content that is specified based on concepts or semantics such as sunset, anchors, or news headline stories. At both levels, the content is embedded in multiple forms that are usually complimentary to each other. The main objective of this thesis is to adaptively analyze the multimedia content by integrating cues from multiple modalities, including audio, video, and text, mainly in the scope of news broadcast.

At the perceptual level, news broadcast data is segmented and classified into different video events such as news reporting and commercials. Audio and visual features are developed and integrated, aiming at discriminating different events effectively. Various classification mechanisms, including linear fuzzy threshold, maximum likelihood using Gaussian Mixture Model and Hidden Markov Model, Neural Network, as well as Support Vector Machine, are benchmarked.

At the conceptual level, algorithms and demonstration systems for three applications are developed. In *News Broadcast Browsing System*, recovering and presentation of the embedded hierarchy structure of news broadcast are addressed. Important semantic objects such as hosting characters and headline news stories are adaptively extracted using the audio/visual models that are bootstrapped from on-line data. The problem of efficient search and retrieval of segmented multimedia objects based on audio is discussed in *Query-by-example in Audio System*. A distance metric framework is proposed to determine the difference of mixture type Probability Density Functions, and is applied in measuring the dissimilarity of audio segments based on their model parameters. In *Major Cast Detection System*, we developed an algorithm to detect the major casts in video, for example, anchor persons in news broadcasts and major characters in movies. The algorithm integrates both speaker and face information and constructs a ranked list of major casts based on their temporal and spacial presence.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Multimedia content analysis refers to computerized understanding of the semantic meanings of a multimedia document, such as a video sequence with accompanying audio track and closed caption. Tools that enable automated content analysis are becoming indispensable as we enter the digital multimedia information era, when we need to efficiently access, digest, and retrieve multimedia information. This dissertation tackles this problem at different semantic levels by integrating cues from multiple modalities, mainly in the scope of news broadcast data.

1.1 Observations and Approaches

Most of our work is driven by the following two observations. First, the semantics of a multimedia document are embedded in multiple forms that are usually complimentary to each other. For example, a live coverage on TV about an earthquake conveys information that is far beyond what we hear from the reporter. We can see and feel the effect of the earth quake, while hearing the reporter talking about the statistics. Therefore, it is necessary to analyze all types of data: image frames, sound tracks, texts that can be extracted from image frames, and spoken words that can be deciphered from the audio track. Second, multimedia content is defined at both perceptual and conceptual levels. The former refers to the content characterized purely by intrinsic perception properties such as color, motion, or acoustic features. The latter refers to the content that is specified based on concepts or semantics such as sunset, anchors, or news headline stories. Multimedia content analysis should be done at both of these levels, and we may integrate different modalities at one or both levels depending on the application. The novelty of this thesis also lies in the combination of these two points of view.

Perceptual level indexing serves as the foundation for the conceptual level analysis. Two major tasks at this level are the parsing of a multimedia document and the detection of basic multimedia events. For a video, this usually means to segment the entire video into shots, within which the audio and/or visual characteristics are coherent, and scenes, which correspond to semantically meaningful units, and to classify each scene or shot into predefined categories of events. Multimeida events may be defined at different levels, e.g. news or commercial at high level, and speech or music at low level. Beyond such "labeled" indexes, some audio and visual descriptors may also be useful as low-level indexes, so that a user can retrieve a video clip that is aurally or visually similar to a query example.

Conceptual level analysis is more application driven, and the approaches are determined by the interested semantics of different applications. In this thesis, we propose various algorithms utilized in three distinct multimedia content analysis systems: news broadcast browsing system, audio query system, and major cast detection system. The addressed techniques include anchor person detection, face detection and tracking, news story generation, audio content description and comparison, speaker and face clustering, and etc. Different conceptual level objects, for example, face and speaker identification, news summary, and major casts are extracted. While perceptual level indexes are helpful, conceptual level "labels" are more meaningful and intuitive for the users to understand the main idea of a video.

With the huge amount of multimedia data available, audio-visual summary is also essential in building a video retrieval system, to enable a user to quickly browse through a large set of returned items in response to a query. Beyond a text summary of the video content, an audio-visual presentation will give the user a better grasp of the characters, the settings, and the style of the video. Such issues are also studied accordingly in this thesis.

1.2 Previous Work

Earlier research in this field has focused on using visual features for segmentation, classification, and summarization of video content. Recently, researchers have begun to realize that audio characteristics are equally, if not more, important when it comes to understanding the semantic content of a video. This applies not just to the speech information, which obviously provides semantic information, but also generic acoustic properties. For example, we can tell whether a TV program is a news report, a commercial or a sports game, without actually watching the TV or understanding the words being spoken, because the background sound characteristics in these scenes are very different. When either audio or visual information alone is not sufficient in determining the scene content, combining audio and visual cues may resolve the ambiguities in individual modality, and thereby help to obtain more accurate answers.

Visual based approach started from computer vision [1], which aims to understand the content and structure of image. Although the state of art in this field is still far away from its ultimate objective - emulation of the functionality of human eyes, techniques in restricted application fields do gain great success. Among them are applications in medicine domain, object detection, face recognition [2], and etc. Since Zhang *et al.* [3, 4, 5] introduced the concept of content-based video processing, there has been much work done in this field [6, 7]. Flickner *et al.* [8] developed QBIC system which allows users to find pictorial information in image and video database based on color, shape, texture and sketches. Yeung and Yeo [9, 10] proposed to use video posters to compactly present and fast browse pictorial content. Rui *et al.* [11] explored the automatic extraction of video structures from both the physical shots and the semantic scenes and developed tools that can construct table of content (TOC) to assist user's access. Ferman and Tekalp [12] proposed a clustering-based framework to segment video sequence and generate visual summary for video management. In face detection and recognition field, many algorithms can be found in [13, 2, 14]. Rowley *et al.* proposed a neural network based face detection algorithm in [15], where a 20×20 region is fed into a neural network after light correction and histogram equalization. The structure of the neural network is elaborately designed with three types of hidden neurons aiming to detect different facial features.

As a complimentary effort, audio based content analysis has existed for decades. The traditional approaches are speech and speaker recognition, which answer the question, who says what. Among the various well established techniques developed in this field, Hidden Markov Model (HMM) based framework is successful in speech recognition [16] and text dependent speaker recognition [17], and Gaussian Mixture Model (GMM) is suitable for text-independent speaker recognition [18]. Until recently, broader types of audio content, including different types of music, background noise, and other general sounds like water, animals, etc. are studied. Saunders [19] proposed a technique to discriminate speech and music in radio broadcast. Saraceno and Leonardi [20] further classified audio into four types: silence, speech, music, and noise. A more elaborate audio content categorization was proposed by Wold *et al* [21], which divided audio content into 10 groups: animal, bells, crowds, laughter, machine, instrument, male speech, female speech, telephone, and water. Furthermore, instrument sound was classified into altotrombone, cellobowed, oboe, percussion, tubularbells, violinbowed, and violinpizz. Zhang and Kuo [22] classified audio content in a hierarchical way. At the coarse level, audio data is classified into speech, music, environmental sounds, and silence, and at the fine level, environmental sounds are further classified into applause, rain, birds' sound, etc. Different sets of audio features aiming to simulate the human hearing perception and reflect the characteristics of audio content are also developed in these works.

Approaches combining multiple modalities in video content analysis are relatively new [23, 24, 25, 26, 27, 28, 29]. Boreczky [30] used HMM framework for video segmentation using both audio and image features. Saraceno and Leonardi [31] considered segmenting a video into the following basic scene types: dialogs, stories, actions, and generic. This is accomplished by first dividing a video into audio and visual shots independently, and then grouping video shots so that audio and visual characteristics within each group follow some predefined patterns. In [32], a hierarchical segmentation approach was proposed that can detect scene breaks and shot breaks. The algorithm is based on the observation that a scene change is usually associated with simultaneous changes of color, motion, and audio characteristics, whereas a shot break is only accompanied with visual changes. Lienhart *et al.* [33] proposed to use different criteria to segment a video into scenes with similar audio characteristics and scenes with similar settings, and dialogs. The scheme considers audio features, color features, orientation features, and face information. In [34], Liu and Huang reported their approach to adaptively detect unknown anchor person using on-line trained audio and visual models. *Name-It* [35] is a project aiming at automatically associating faces detected from video frames and names detected from the closed caption in news. It does not rely on any pre-stored face templates for selected names, which is both the challenge and novelty of the system.

1.3 Dissertation Outline

This thesis is organized as follows. In Chapter 2, perceptual level multimedia content analysis is described. Then in Chapters 3, 4, and 5, three different conceptual level multimedia analysis applications are presented. Demonstration systems are developed to show the effectiveness of proposed techniques, and corresponding simulation results are shown and discussed. Finally, in Chapter 6, we draw our conclusions and indicate possible future works.

In Chapter 2, we introduce the video indexing at the perceptual level. Specifically, a video is first segmented into clips and then each clip is classified into interested events. Audio and visual features are developed aiming at discriminating different types of events effectively. Feature space reduction techniques are also studied. Various classifiers, including linear fuzzy classifier, neural network, maximum likelihood using Gaussian Mixture Model and Hidden Markov Model, and Support Vector Machine are benchmarked.

In Chapter 3, we first present the hierarchy of news broadcast, and then show the approach we adopted to recover the structure automatically. Addressed issues include how to adaptively detect anchor person and how to extract headline news stories by integrating both audio and text information. A Table-of-Content is generated to guide the user efficiently browse interested news document.

In Chapter 4, an audio-based query system is described. The audio signal is segmented into homogeneous segments whose features are characterized using Gaussian Mixture Models. A new metric framework for measuring the similarity between two Probability Density Functions of mixture type is described and applied to GMMs. Beyond passive skimming the multimedia summaries, the developed techniques enable the user to actively search the multimedia document based on audio property.

In Chapter 5, we present an approach for automatically generating the list of major casts for video based on both audio and visual information. A template-based face detection and tracking algorithm is proposed. After speakers and faces are detected and clustered, the major casts are chosen relying on face and speaker correlation values, and are sorted based on the importance scores determined by their temporal and spatial presence. Based on the list of major casts, the user is able to find the attractive portion of video based on both audio and visual characteristics.

Chapter 2

Perceptual Level Multimedia Content Indexing

This chapter discusses video indexing at perceptual level, which provides the basis for conceptual level content analysis. Specifically, video is first segmented into clips that are about 2 seconds long, and then each clip is classified into different events, such as news reporting and commercials based on their intrinsic signal properties. The task is formalized as a pattern recognition problem, where two important issues are features utilized and classification mechanism. Audio and visual features are exploited aiming at discriminating different types of events effectively. Feature dimension reduction techniques are also considered. Various classifiers including linear fuzzy classifier, neural network, maximum likelihood using Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM), as well as Support Vector Machine (SVM) are benchmarked.

2.1 Feature Extraction

Video events are characterized by accompanying audio and visual properties. Different categories of video events may be effectively separated by different sets of audio and/or visual features. For example, audio information is feasible to differentiate speech from music since this two types of events are defined mostly based on their acoustic difference. Color information may be helpful for extracting football games from basketball games, since the color patterns of playgrounds for these two kinds of games are distinct. In this section, we first describe the audio and visual features we developed, and then introduce some feature space reduction techniques based on feature correlation.

2.1.1 Audio Features

Audio signal has been studied for video shot/scene segmentation and video scene classification [20, 36, 37, 38]. We developed a set of audio features [39, 40, 41, 42], which have been found to be quite effective in characterizing different types of video events, e.g. commercials, games, and news reporting.

As illustrated in Figure 2.1, a digitized audio waveform is segmented into clips, which may or may not overlap with previous clips and cover constant or variable duration. Then each clip is divided into frames of 512 samples, each overlapping with the previous frame by 256 samples. Eight features are computed for each frame. Based on these frame level features, we extract 14 features for each audio clip.

To clearify the motivation of audio feature extraction procedure, we plot some of the features for audio clips from distinctive video events: commercial, basketball game, and news reporting. These graphs show that each feature is designed to be capable of discriminating different video events.

Time Domain Features Volume distribution of an audio signal reveals the temporal variation of the signal magnitude. We use the root mean square of the magnitude within each frame to approximate the volume of that frame. The volume of the n^{th} frame is defined as

$$v(n) = \sqrt{\frac{1}{N} \sum_{i=0}^{N-1} s_n^2(i)}, \tag{2.1}$$

where $s_n(i)$ is the i^{th} sample in the n^{th} audio frame and N is the frame length.

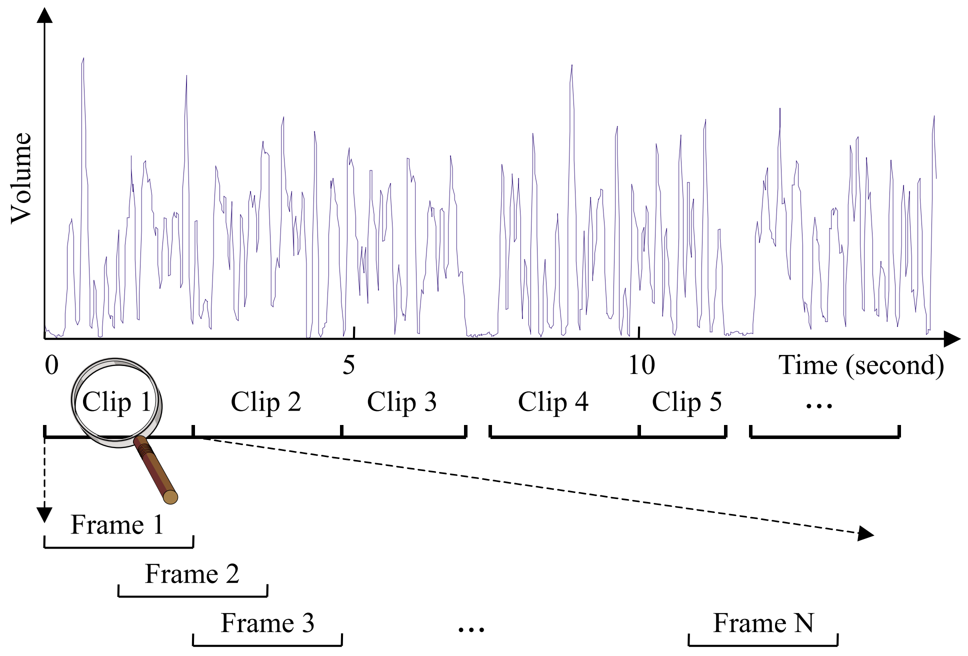
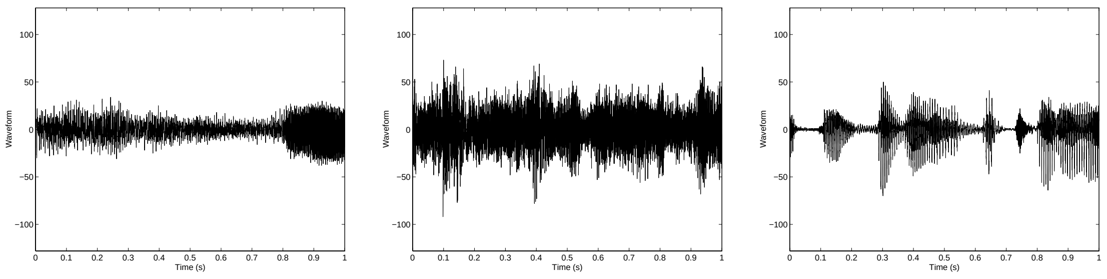


Figure 2.1: An audio clip used in video segmentation and classification.

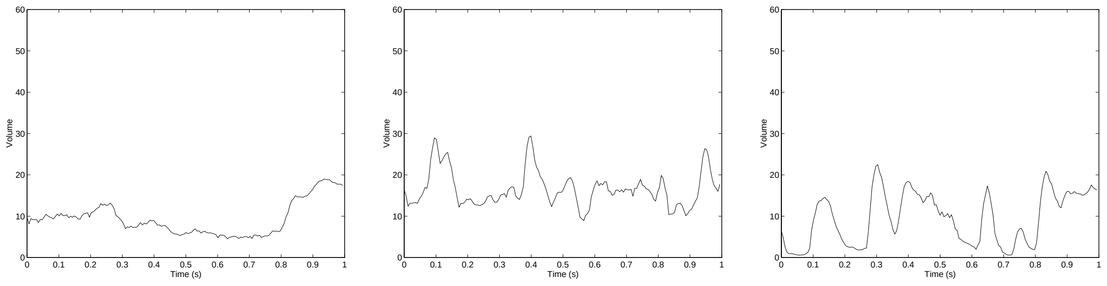
Figures 2.2 and 2.3 show the waveforms and volume contours of three different video events. From these plots, we know that the volumes of these three audio clips have different distribution. To measure the volume variation of an audio clip, we define three time-domain features based on the volume distribution. The first one is the *volume standard deviation* (VSTD), which is the standard deviation of the volume over a clip. The second one is the *volume dynamic range* (VDR), which is the difference between the maximum volume and the minimum volume in the clip. The last one is the *volume undulation* (VU), which is the accumulation of the difference of neighboring peaks and valleys of the volume contour in the clip. These three features are normalized by the maximum volume in the clip.

Zero crossing rate (ZCR) is the number of times that an audio waveform crosses the zero axis. To detect silence periods, we compare the volume and ZCR of each frame to some preset thresholds. Using both volume and ZCR can prevent misclassfying the low energy unvoice speech as silent signal. Based on the result of



(a) *Commercial* (b) *Basketball* (c) *News*

Figure 2.2: Waveforms of three audio clips.



(a) *Commercial* (b) *Basketball* (c) *News*

Figure 2.3: Volumes of three audio clips.

silence detection, we calculate the *non-silence ratio* (NSR), which is the ratio of the non-silent interval to the entire clip. We also calculate the *standard deviation of zero crossing rate* (ZSTD).

Frequency Domain Features The volume contour of a speech waveform typically peaks at 4Hz [43]. We define the *4-Hz modulation energy* (4ME) as

$$4ME = \frac{\int_0^\infty W(\omega)|C(\omega)|^2 d\omega}{\int_0^\infty |C(\omega)|^2 d\omega} \tag{2.2}$$

where $C(\omega)$ is the Fourier transform of the volume contour of an audio clip and $W(\omega)$ is a triangular window function centered at 4 Hz. Clips composed of speech tend to have higher 4ME values than those composed of music or noise.

Pitch is the fundamental period of an audio waveform. We use the short time *Average Magnitude Difference Function* (AMDF) to determine the pitch of each frame. The AMDF is defined as

$$\gamma(l) = \frac{\sum_{i=0}^{N-l-1} |s_n(i+l) - s_n(i)|}{N - l} \tag{2.3}$$

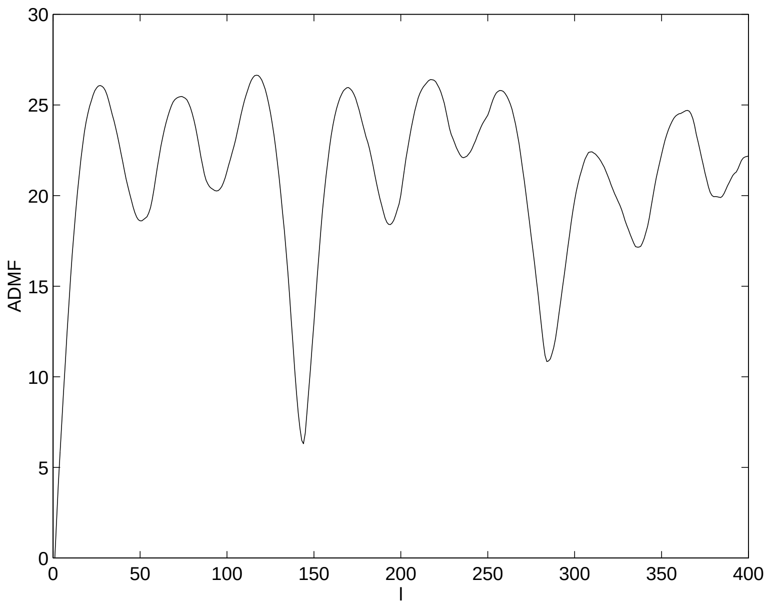
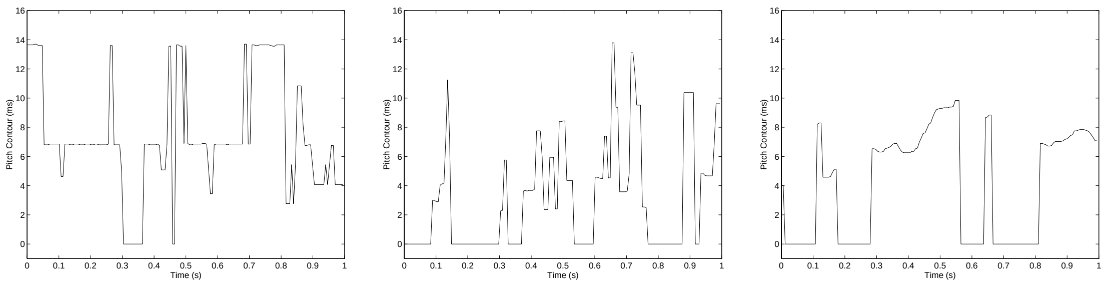


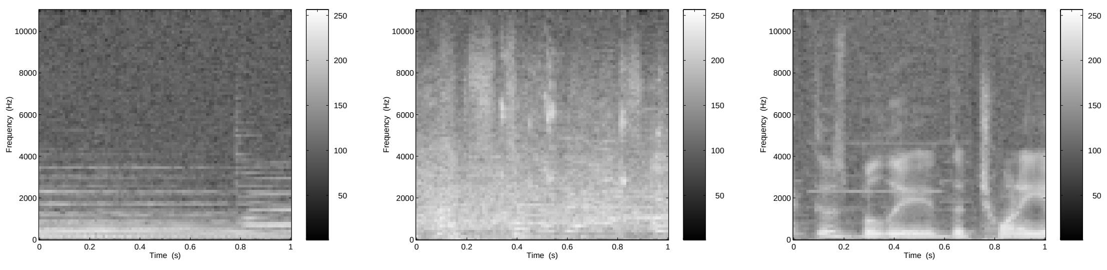
Figure 2.4: The AMDF of one speech frame.

Figure 2.4 shows the AMDF graph for a speech frame. We use the algorithm in [44] to determine pitch from AMDF. The algorithm searches the first valley point in the AMDF starting from the origin. The valley is defined as a local minimum which satisfies additional constraint in terms of its value relative to the global minimum and its curvature. For example, the AMDF in Figure 2.4 has two valleys. The pitch frequency is the reciprocal of the time period between the origin and the first valley. The search range is from 50 Hz to 450 Hz, which is the pitch range of normal human speech. We assume the pitch frequency is zero when no pitch is found. A median filter is used to eliminate falsely detected pitches. Figure 2.5 gives the pitch tracks of the same three audio clips as used in Figure 2.2. In the commercial clip, there exists music background with overlapping notes and the detected pitch at a particular frame depends on which note is stronger. Therefore, the pitch track stays flat for short intervals and there exist both high and low pitch periods. In the basketball clip, since there is significant background noise, the pitch track is very rough, rarely with a smooth region. In the news clip, the pitch track is smooth and lasts relatively long. The intervals between two smooth tracks correspond to silence/unvoice period. From the detected pitch contours, we derive three features: *standard deviation of pitch* (PSTD), *smooth pitch ratio* (SPR), and *non pitch ratio* (NPR). The SPR is the percentage of frames in a clip that have similar pitches as the previous frames. The NPR is the percentage of frames with no pitch detected.



(*a*) *Commercial* (*b*) *Basketball* (*c*) *News*

Figure 2.5: Pitch contours of three audio clips.



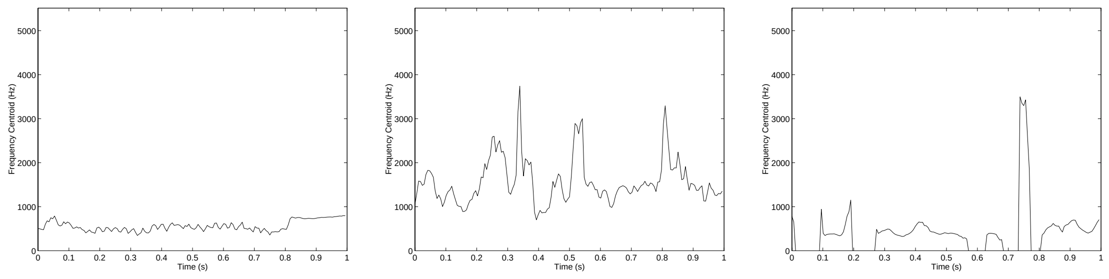
(*a*) *Commercial* (*b*) *Basketball* (*c*) *News*

Figure 2.6: Spectrograms of three audio clips.

To obtain frequency domain features, we first calculate the spectrogram of an audio clip, which is a 2D plot of the short-time Fourier transform (over each audio frame) along the time axis. Figure 2.6 shows the spectrograms of the three audio clips given in Figure 2.2. Let $S_i(\omega)$ represents the short-time Fourier transform of the i^{th} frame. The frequency centroid, $C(i)$, and the bandwidth, $B(i)$, are defined as

$$\begin{aligned} C(i) &= \frac{\int_0^\infty \omega|S_i(\omega)|^2 d\omega}{\int_0^\infty |S_i(\omega)|^2 d\omega}, \\ B^2(i) &= \frac{\int_0^\infty (\omega - C(i))^2 |S_i(\omega)|^2 d\omega}{\int_0^\infty |S_i(\omega)|^2 d\omega} \end{aligned} \tag{2.4}$$

Figures 2.7 and 2.8 show the contours of the frequency centroid and bandwidth computed based on the spectrograms. The zero regions in the contour correspond to silent frames. From these figures, we can see that the basketball clip's frequency centroid is high and has bigger dynamic range, on the other hand, the news clip has low frequency centroid and bandwidth during the voice period and high centroid and bandwidth during the unvoice period. In the commercial clip, there is a continuous music background, so the frequency centroid and bandwidth contours are quite smooth.

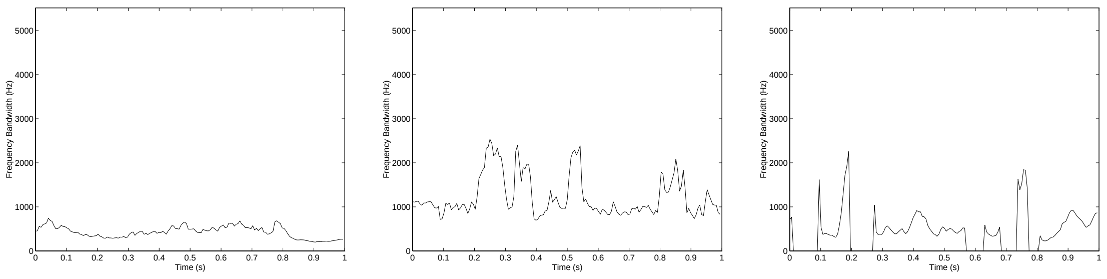


(a) *Commercial* (b) *Basketball* (c) *News*

Figure 2.7: Contours of frequency centroid of three audio clips.

The clip level *frequency centroid* (FC) and the *bandwidth* (BW) are calculated as the energy-weighted means of frequency centroid and bandwidth of each frame, respectively. Because the frame with high energy has more influence to human ears, the weighting for a frame is proportional to the energy of the frame.

The energy distribution in different frequency bands also varies quite significantly among different types of audio signals. We divide the entire spectrum into



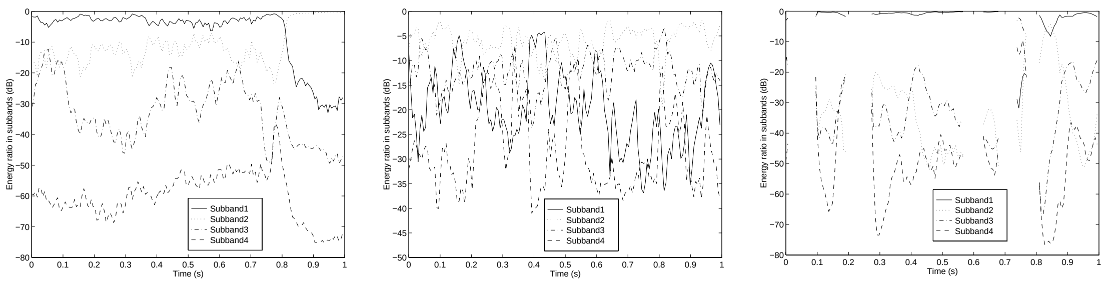
(*a*) *Commercial* (*b*) *Basketball* (*c*) *News*

Figure 2.8: Contours of bandwidth of three audio clips.

four subbands. Each subband consists of six critical bands which represent cochlear filter in the human auditory model [45]. The frequency ranges of the four subbands are 0 – 630 Hz, 630 – 1720 Hz, 1720 – 4400 Hz, and 4400 – 11025 Hz. We calculate subband energy ratios, which are the ratios of the energies in the four subbands to the total energy, for each frame. Figure 2.9 shows the 4 subband energy ratio contours of the three audio clips given in Figure 2.2. The four contours in the commercial clip are rather smooth, on the other hand, the contours in basketball clip vary a lot. The energy ratio of subband 1 in the news clip is much higher than those of the other subbands. Since the four subband energy ratios sum to 1, we only consider the first three ratios as features. From the frame level energy ratios, we calculate clip level ratios by using a weighted average, where the weightings are proportional to the energy of the frames. We refer the three clip level subband features as ERSB1, ERSB2, and ERSB3.

To summarize, we develop fourteen clip level audio features:

- NSR: non-silence-ratio
- ZSTD: standard deviation of zero crossing rate
- VSTD: volume standard deviation
- VDR: volume dynamic range



(a) *Commercial* (b) *Basketball* (c) *News*

Figure 2.9: Energy ratio in 4 subbands of three audio clips.

- VU: volume undulation
- 4ME: 4-Hz modulation energy
- PSTD: pitch standard deviation
- SPR: smooth pitch ratio
- NPR: non pitch ratio
- FC: frequency centroid
- BW: frequency bandwidth
- ERSB1-3: energy ratios of subbands 1-3.

2.1.2 Visual Features

Several excellent papers [46, 11] have appeared recently, summarizing and reviewing various visual features, within the categories of color, texture, shape, and motion, which are useful for image/video indexing. The visual features are chosen based on the observed signal properties of the underlying classes. In broadcast news, the dominant classes of events are news reporting and commercials. Due to the cost of air time, a piece of commercial tends to have more actions, corresponding to more shot transitions and faster motion within each shot. To reflect such difference, color and motion features are extracted to capture (1) higher frequency of shots (using color) and (2) higher motion energy (using motion information).

Color Features The effectiveness of the color histogram feature depends on the color coordinate used and the quantization method. Wan and Kuo [47] studied the effect of different color quantization methods in different color spaces including RGB, YUV, HSV, and CIE L*u*v*. Here we use RGB space for its simplicity and effectiveness. Based on color histogram, we choose the most dominant color (DC) and its percentage (DCP) as color features. The dominant color represents the overall color perception of one frame, and the percentage shows how dominant the color is. Both of them are important for discriminating video events. For example, the dominant color for basketball game is orange, and that of football games is green. The dominant color of in-studio news report frames may have higher percentage than that of live news report since the color of studio environment is much more concentrated. Means and standard deviations of the above features within one clip are used as clip level color features. While color histogram may not necessarily directly reflect the video style (e.g., both news reporting and commercials may have similar color distributions), the change rate of color histogram does. We compute the difference between the color histograms (DCH) from two adjacent frames by the χ^2 distance, and use its mean and variance within a video clip as two extra color features.

Motion Features We extract two types of motion features: frame-wise dominant motion (DM) and block-wise motion energy (ME). We use the phase correlation function (PCF) between two frames [37] to compute the dominant motion vector. When one frame is the translation of the other, the PCF has a single peak at a location corresponding to the translation vector. When there are multiple objects with different motions in the scene, the PCF tends to have multiple peaks, each with a magnitude proportional to the number of pixels experiencing a particular motion. In this sense, the PCF reveals similar information as the motion histogram. But it can be computed from the image intensity directly, and therefore is not affected by motion estimation inaccuracy. Similar to color features, we also use the percentage of dominant motion (PDM) as a frame feature. The motion energy between two adjacent frames is computed as follows. First, each image is divided into 8×8 blocks, and for each block, motion based block matching [48] is performed within a search range of 16×16. Let D_i denote the total intensity difference between two blocks. The motion energy of a block k is the weighted motion magnitude $e_k = \lambda \times \sqrt{\delta x^2 + \delta y^2}$, where δx and δy represent the best displacement (with minimum D_i) along x and y axes, λ is the weight calculated as $\lambda = 1 + \lfloor \frac{D_i}{T_i} \rfloor$. Here, T_i is a pre-determined constant. When D_i is smaller than T_i, the score is the motion magnitude itself. The higher the D_i is, the larger the λ is and so is the motion energy e_k. When the changes are caused by small motion, the motion energy is small. When a sudden scene change occurs, the motion energy becomes large. We then use the sum of the motion energy from all blocks $E = \sum_k e_k$ as the overall motion energy between two adjacent frames. Means and variances of all above motion features within one video clip are used to characterize the motion property.

Overall, we develop eighteen clip level visual features:

- DCR, DCG, DCB: mean red, green, and blue components of dominant color
- PDC: mean percentage of dominant color
- DCRSTD, DCGSTD, DCBSTD: standard deviation of red, green, and blue components of dominant color
- PDCSTD: standard deviation of percentage of dominant color
- DCH: mean of difference between the color histograms
- DCHSTD: standard deviation of difference between the color histograms

- DMX, DMY: mean X and Y components of dominant motion
- PDM: mean percentage of dominant motion
- DMXSTD, DMYSTD: standard deviation of X and Y components of dominant motion
- PDMSTD: standard deviation of percentage of dominant motion
- ME: mean motion energy
- MESTD: standard deviation of motion energy

2.1.3 Correlation Between Audio and Visual Features and Feature Space Reduction

Given a long list of audio and visual features that one can come up with, a natural question to ask is whether they provide independent information about the scene content, and, if not, how to derive a reduced set of features that can best serve the purpose. One way to measure the correlation among features within the same modality and across different modalities is by computing the covariance matrix,

$$\mathbf{C} = \frac{1}{N}\sum_{\mathbf{x}\in\chi}(\mathbf{x}-\mathbf{m})(\mathbf{x}-\mathbf{m})^T \qquad \text{with} \qquad \mathbf{m} = \frac{1}{N}\sum_{\mathbf{x}\in\chi}\mathbf{x}, \tag{2.5}$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_K)^T$ is a K-dimensional feature vector, χ is the set containing all feature vectors derived from training sequences, N is the total number of feature vectors in χ. The normalized correlation between features i and j is defined by

$$\tilde{C}(i,j) = \frac{C(i,j)}{\sqrt{C(i,i)C(j,j).}} \tag{2.6}$$

where $C(i,j)$ is the (i,j)-th element in $\mathbf{C}$.

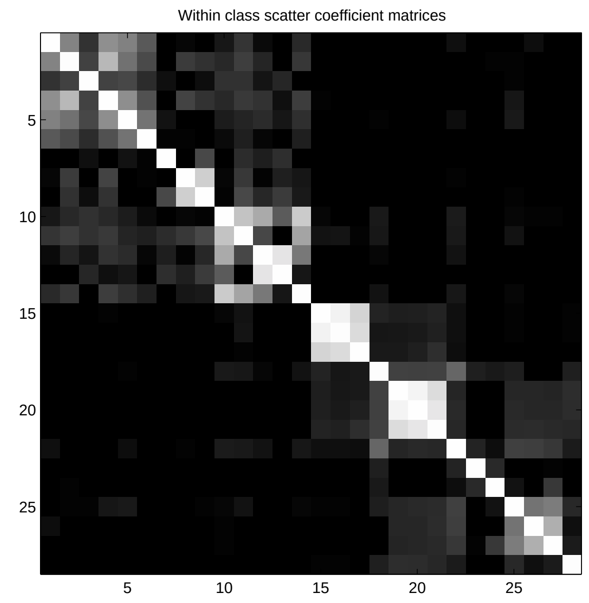


Figure 2.10: The normalized correlation matrix of features from different modalities.

Figure 2.10 shows the normalized correlation matrix in absolute value derived from a training set containing five types of TV programs: commercials, news, live basketball games, live football games, and weather forecast. About ten minutes of each scene type are included in the training set. A total of 28 clip level features are considered: 14 audio features, 8 color features and 6 motion features. The color features include DCR, DCG, DCB, DCP, DCRSTD, DCGSTD, DCBSTD, and DCPSTD; and the motion features are DMX, DMY, DMP, DMXSTD, DMYSTD, and DMPSTD. Figure 2.10 shows the normalized correlation matrix of features from different modalities, where the first 14 features are audio features, the next eight are color features, and the last six are motion features. It is clear that the correlations among different modalities are very low. Within the same modality, high correlation exists among some features, such as NSR, VSTD, and VDR, SPR and NPR, FC and BW, and ERSB1 and ERSB2 among audio features, the means and variances of three color components in the dominant color.

High correlation among certain features in the above example suggests that the feature dimension can be reduced through proper transformations. Two powerful feature space reduction techniques are Karhunen Loeve Transform (KLT) [49] and Multiple Discriminant Analysis (MDA) [50], both using linear transforms. With KLT, the transform is designed to decorrelate the features, and only those features with eigen-values larger than a threshold will be retained. With MDA, the transform is designed to maximize the ratio of between inter-class scattering and intra-class scattering. The maximum dimension of the new feature space is the number of classes minus one. Figure 2.11 shows the 2D projection of the 14 audio feature vectors from news and commercial clips using Karhunen Loeve transformation. The separability is evident, which shows that the integrated clip level features can capture the characteristics of underlying audio events of interest.

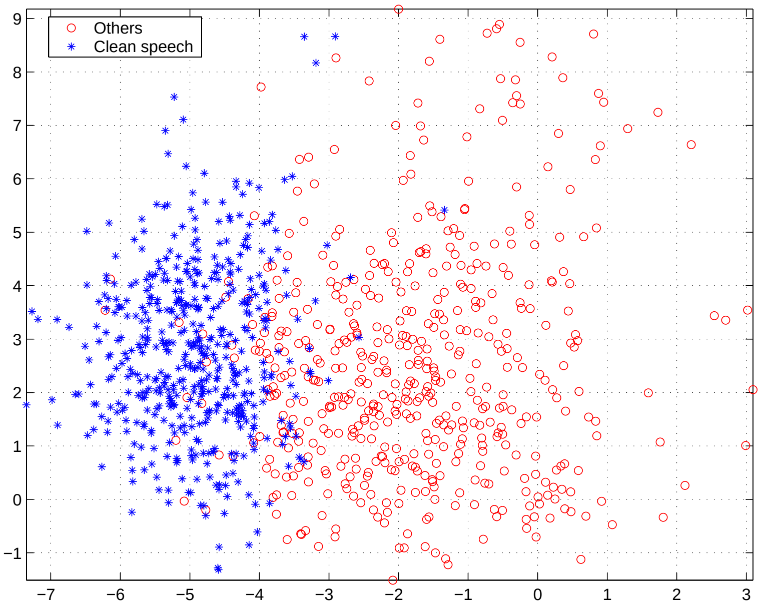


Figure 2.11: 2D projection of feature vectors using Karhunen Loeve Transformation.

Figure 2.12 shows the distribution of feature points from five scene classes (denoted by different symbols and colors). The original feature space consists of the same 28 features used in Figure 2.10. The left plot is based on two original features: FC and the mean of the most dominant color (red component). The middle plot is

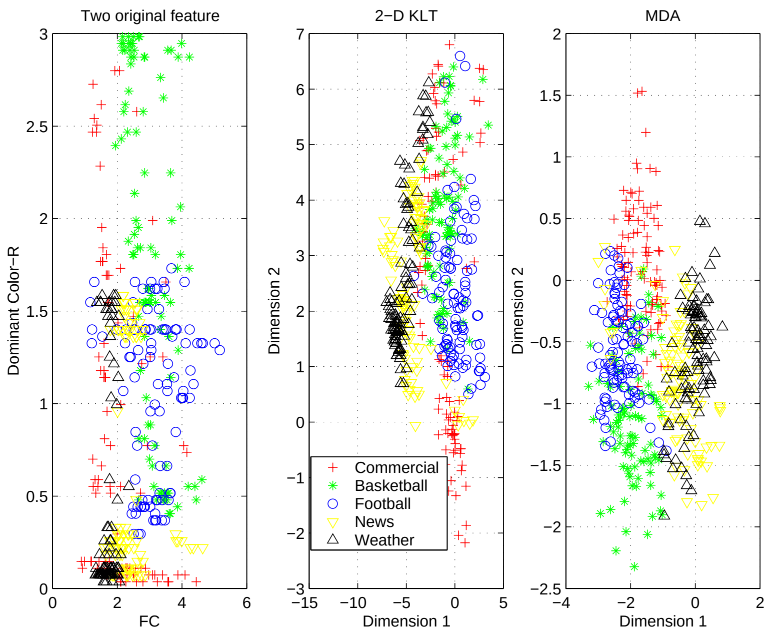


Figure 2.12: Distribution of two features in the original feature vector, after KLT, and after MDA.

based on the first two features obtained after applying KLT on the original feature vector. The right plot is based on the first two features obtained with MDA. We can easily see that there is the least amount of inter-class overlapping in the feature space obtained with MDA. This means that the two new features after MDA have the best scene discrimination capability.

2.2 Video Event Classification

Besides feature extraction, classification method is also an important issue. Five types of classifiers are benchmarked in this work. Specifically, we used linear Fuzzy classifier, Neural Network classifier, Gaussian Mixture Model (GMM) classifier, Support Vector Machine (SVM) classifier, and Hidden Markov Model (HMM) classifier.

2.2.1 Linear Fuzzy Classifier

In fuzzy classifier, each feature is associated with a fuzzy membership function for each type of events. The impact that each feature attribute to the overall decision is realized in the form of a weighted sum, where each weight is derived from the fuzzy membership function of that feature. An overall threshold value is then applied to the weighted sum to reach the final decision of the classification. The membership function we used is a piece-wise linear function shown in Figure 2.13, where the solid line and dash line are two types of functions for specific features. Suppose we want to classify two events: news report and commercial, then, for news report, the membership function of audio feature NSR follows the dashed line in the figure. The lower NSR is, the more likely the corresponding clip is news report.

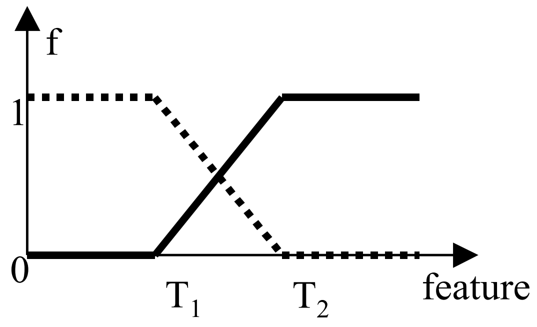
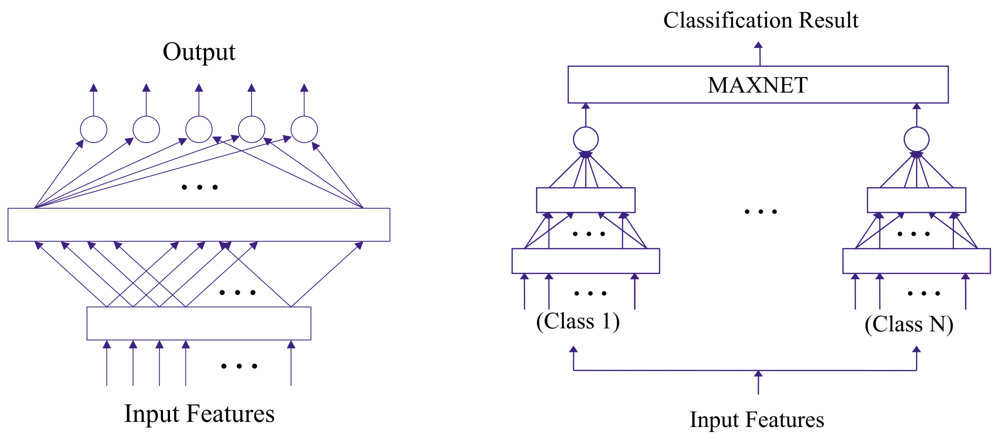


Figure 2.13: Membership function used in linear fuzzy classifier.

2.2.2 Neural Network Classifier

Artificial neural networks have been used successfully as pattern classifiers in many applications for their ability to implement nonlinear decision boundaries and their capability to learn complicated rules from training data [51, 52]. Conventional multi-layer perceptron (MLP) use the all-class-in-one-network (ACON) structure, which is shown in Figure 2.14(a). Such a network structure has the burden of having to simultaneously satisfy all the desired outputs for all classes, so the required number of hidden units tends to be large. Besides, if one wants to adapt the network to new training data or add new classes, all the weights need to be re-computed. On the other hand, in the one-class-one-network (OCON) structure, one subnet is designated for recognizing one class only [53]. The structure is illustrated in Figure 2.14(b). Each subnet is trained individually using the back-propagation algorithm so that its output is close to 1 if the input pattern belongs to this class, otherwise the output is close to 0. Given an input audio clip, it is classified to the class whose subnet gives the highest score. An advantage of the OCON structure is that one can accommodate a new class easily by adding a subnet trained for that class. Given the network structure, we still can adjust the classifier performance by one parameter: the number of hidden neurons. The choice of this number is a tradeoff between the capability and generality of the network.



(*a*) *All class one network Structure* (*b*) *One class one network Structure*

Figure 2.14: Two different structures of Neural Network

2.2.3 Gaussian Mixture Model Classifier

Another approach is to build models for the underlying classes using labeled training data. Based on such trained models, a test sample can be classified using maximum likelihood (ML) method. Here we use Gaussian Mixture Model. A GMM model consists of a set of weighted Gaussian's:

$$f(\mathbf{x}) = \sum_{i=1}^{K} \omega_i \times g(M_i, \Sigma_i, \mathbf{x}), \text{where } g(M_i, \Sigma_i, \mathbf{x}) = \frac{exp\{-\frac{(\mathbf{x}-M_i)^T \Sigma_i^{-1} (\mathbf{x}-M_i)}{2}\}}{(\sqrt{2\pi})^n \sqrt{det(\Sigma_i)}}, \quad (2.7)$$

where K is the number of mixtures, M_i and Σ_i are the mean vector and covariance matrix of the i^{th} mixture, respectively, and ω_i is the weight associated with the i^{th} Gaussian. Based on training data, the parameter set $\lambda = (\omega, M, \Sigma)$ are optimized such that $f(\mathbf{x})$ best fits the given data. The initial parameters are estimated from a clustering algorithm, then expectation maximization (EM) method is used to iteratively refine the parameters until some preset conditions are met.

It has been proven that ML based estimation method for Gaussian mixture model has no solution [54] due to the fact that, theoretically, there is no upper bound for the likelihood value during the training. Therefore, we limit the covariance of each feature within a specified range as a constraint.

The decision about the number of mixtures to be used in the model is empirical, relating to both the data characteristic and the amount of training data available. If the amount of training data is small, a model with a large number of mixtures is not reliable. On the other hand, a model with a small number of Gaussian's constructed based on sufficient amount of training data may not adequately approximate the data to a desirable precision.

2.2.4 Support Vector Machine Classifier

Support vector machine maps an input space into a high-dimensional feature space denoted by Z (a Hilbert Space) [55] through some non-linear mapping Φ chosen a priori and then construct the optimal separating hyperplane in the feature space, making it possible to construct linear decision surfaces in the feature space which correspond to the nonlinear decision surfaces in the input space [54, 56, 57].

To construct the optimal separating hyperplane in the feature space, there is no need to consider the feature space in explicit form. Without knowing the mapping function Φ, we can express the inner product of vectors $\mathbf{z_1}$ and $\mathbf{z_2}$ in feature space Z as $(\mathbf{z_1} \cdot \mathbf{z_2}) = K(\mathbf{x_1}, \mathbf{x_2})$, where $\mathbf{z_1}$ and $\mathbf{z_2}$ are the images in the feature space of vector $\mathbf{x_1}$ and $\mathbf{x_2}$ in the input space. The kernel function $K(\mathbf{x}, \mathbf{y})$ can be any symmetric function that satisfies the Mercer condition [54]. In this work, we experimented dot product, polynomial and radial basis function (RBF) as kernel functions. They are defined as:

$$K_{dot}(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}, \tag{2.8}$$

$$K_{poly}(\mathbf{x}, \mathbf{y}) = ((\mathbf{x} \cdot \mathbf{y}) + 1)^d, d = 1, ... \tag{2.9}$$

$$K_{RBF}(\mathbf{x}, \mathbf{y}) = exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2), \tag{2.10}$$

where d is the order of polynomial kernel and γ is a parameter of RBF.

The pattern recognition problem in SVM can be formulated as: For a set of samples $(\mathbf{z_i}, y_i)$, $\mathbf{z_i} \in$ Z, $y_i \in \pm 1$, $i = 1, ..., N$, we want to find the optimal hyperplane $f(\mathbf{z}) = (\mathbf{w} \cdot \mathbf{z}) + b$, that satisfies $sign(f(\mathbf{z_i})) = y_i$. The embedded idea introduced by SVM is to minimize an upper bound on the generalization error. Considering the freedom to scale $\mathbf{w}$ and b simultaneously, there is another requirement for a canonical pair:

$$\min_{i=1,...N} |(\mathbf{w} \cdot \mathbf{z_i}) + b| = 1 \tag{2.11}$$

In the separable case, all samples satisfy the following separation constraints:

$$y_i((\mathbf{z_i} \cdot \mathbf{w}) + b) \geq 1, \quad i = 1, ..., N \tag{2.12}$$

During training, we want to maximize the hyperplane margin (see Figure 2.15), that is the sum of the shortest distance from the hyperplane to the closest positive and negative samples, so that to maximize the generalization ability of SVM classifier. This is equivalent to minimizing $\tau(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2$, since the shortest distance

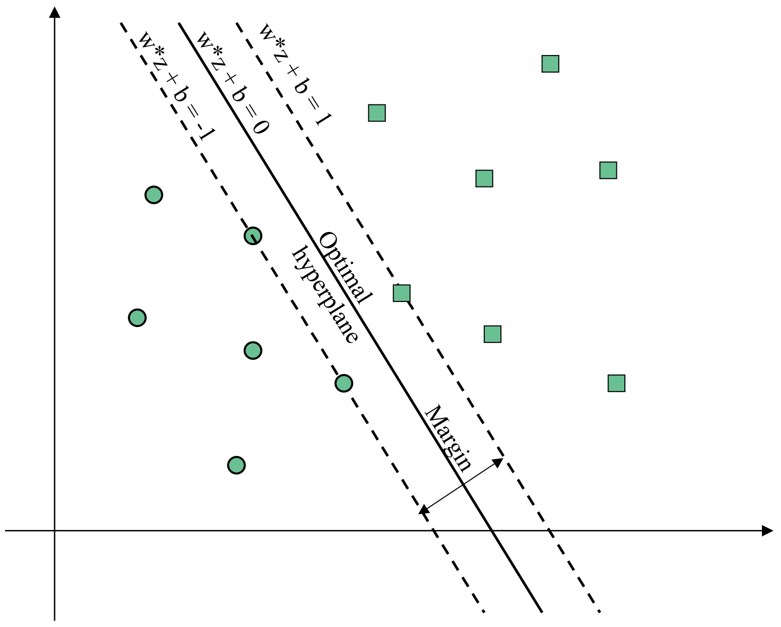


Figure 2.15: The optimal separating hyperplane in SVM.

from positive and negative samples to the hyperplane is $1/\|\mathbf{w}\|$. The corresponding Lagrangian is

$$L_P = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i (y_i((\mathbf{z_i} \cdot \mathbf{w}) + b) - 1) \tag{2.13}$$

with $\alpha_i \geq 0$. The requirement that the gradient of L_P with respect to $\mathbf{w}$ and b must vanish leads to conditions:

$$\mathbf{w} = \sum_{i=1}^{N} \alpha_i y_i \mathbf{z_i}, \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{2.14}$$

Substituting these conditions to the original Lagrangian, we get the dual form that we need to maximize:

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{N} \alpha_i \alpha_j y_i y_j \mathbf{z_i} \mathbf{z_j}, \tag{2.15}$$

subject to constraints $\alpha_i \geq 0$, and $\sum_{i=1}^{N} \alpha_i y_i = 0$. This quadratic programing problem can be solved by numerical methods to get α_i. The samples $\mathbf{z_i}$ with positive α_i are called support vectors. b is compute by

$$b = -\frac{1}{2}\mathbf{w}[x_1 + x_{-1}], \tag{2.16}$$

where x_1 is any support vector belonging to the class labeled 1, and x_{-1} is any support vector belonging to the class labeled -1. Together with their α_i values and b, support vectors compose the parameters of the SVM.

During testing, we assign the class of $\mathbf{z}$, $f(\mathbf{z})$ to be $sign(\mathbf{wz}+b)$, which is

$$f(\mathbf{z}) = sign(\sum_{i=1}^{N} \alpha_i y_i (\mathbf{z} \cdot \mathbf{z_i}) + b). \tag{2.17}$$

In the non-separable case, we need to introduce slack variables: $\xi_i \geq 0, i = 1, ..., N$. The separation constraints are relaxed to $y_i((\mathbf{z_i} \cdot \mathbf{w})+b) \geq 1-\xi_i, \quad i = 1, ..., N$. Now, we need to minimize, $\tau(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i$, where $\sum_{i=1}^{N} \xi_i$ is an upper bound on the number of training errors, and C is the penalty to errors. Following the similar procedure in the separable case, we can find the coefficients α_i.

Due to its nature of locality, results of the above classifiers are often quite noisy. Contextual information is used in a smoothing mechanism that reaches a final classification by considering the overall classification within a predetermined neighborhood. One effective smoothing method is to use a median filter to post-process the clip-based classification labels.

2.2.5 Hidden Markov Model Classifier

The above methods make decision based on the feature vector from one incoming clip. This is sometimes not feasible, since there may be overlapping in feature space for different video events. To reliably detect video events, the temporal information need also be considered. This means that we need to classify the video input based on a sequence of clips, and utilize the embedded temporal variation pattern. Hidden Markov Model is very powerful in such kind of task and has been successfully used in speech recognition domain.

There are two types of HMM, discrete and continuous models. A discrete HMM is characterized by the following parameters[16]:

- N, the number of states in the model. $S = \{s_1, s_2, \ldots, s_N\}$ is the set of the states. The state at time t is given by $q_t \in S, 1 \le t \le T$, where T is the length of the observation sequence.

- M, the number of different observation symbols. $V = \{v_1, v_2, \ldots, v_M\}$ is the collection of all the possible observation symbols. Here, we assume that all the possible realizations of the observation vector $\mathbf{o}$ are quantized into a finite set of symbols, using a pre-designed vector quantizer with M codewords.

- $A = \{a_{i,j}\}$, the state transition probability matrix, where $a_{i,j} = P[q_{t+1} = S_j | q_t = S_i]$, $1 \le i, j \le N$, with the state transition coefficients satisfying $0 \le a_{i,j} \le 1$, $\sum_{j=1}^{N} a_{i,j} = 1$, $1 \le i, j \le N$.

- $B = \{b_i(k)\}$, the observation symbol probability matrix with $b_j(k) = P[O_t = v_k | q_t = S_j]$, $1 \le j \le N, 1 \le k \le M$.

- $\Pi = \{\pi_i\}$, the initial state distribution, where $\pi_i = P[q_1 = S_i], \quad 1 \le i \le N$.

We use the notion $\lambda = (A, B, \Pi)$ to indicate the complete parameter set of the model. For a continuous HMM, the observation probabilities of the feature vectors are characterized in a parameterized form.

The HMM training process follows the Baum-Welch method. The initial parameters of A and B are chosen randomly and the initial values of Π are uniformly distributed for each state. After training we have $\lambda_1, \lambda_2, \ldots, \lambda_C$, where C is the number of target classes. In classification stage, we use maximum likelihood method.

2.3 Simulation Results and Discussion

Here we present two sets of results under different applications. The first one is to separate news from commercial [58], and the second one is to classify five TV programs: commercial, basketball games, football game, news report, and weather forecast [59, 60]. Different feature sets and classifiers are studied in both simulation settings. All the results reported in this section are raw error rate without any smoothing. By raw error rate, we refer to the rate calculated from the initial classification performed on each clip without any smoothing.

2.3.1 Classification of News and Commercial

Overall, we acquired 4 hours of data from NBC Nightly News, of which, 2 hours of the data is used for training and the rest for testing. Audio track is sampled at 16 KHz with 16 bits per sample, visual track is digitized at 10 frames per second, with size 160×120. Most commercials (the only non-news reporting portion in our task) are speech mixed with music background but some contain conversation only. The news reporting includes clean speech from studio and noisy live report. All the data used are manually labeled as news or commercial. In linear fuzzy classifier, we use nine audio clip features: NSR, VSTD, ZSTD, VDR, VU, 4ME, SMR, NUR, and ERSB2. In GMM and SVM classifiers, we test three cases: fourteen audio clip features; four visual clip features, specifically, DCH, DCHSTD, ME, and MESTD; and combination of these audio and visual features.

The linear fuzzy classifier achieves 11.8% accuracy. This also shows that the nine audio features used are well-separated for the two video events.

The benchmarking results of GMM classifier on five different mixtures (from 2 to 32 in octave step) is shown in Table 2.1. In the table, we give the classification results in three cases that use 1) audio features only, 2) visual features only, and 3) the two types of features combined. To combine audio and visual features, they are simply concatenated into a big feature vector. When the mixture number is too small, or too big, the error rates are high. This is because that simple models can not precisely approximate the distribution of real data. On the other hand, complex models do not have good generalization property. Visual only approach gives poor performance. The lowest error rate of 7.97% is reached when the number of mixture is 8 with combined features. Compared with the results using audio features only, about 8% improvement is achieved by exploiting additional visual information. From our experiments, it can be seen that although visual features help, it is not very significant with the current integration framework. This may be due to the fact that the integration at clip level may be at too fine resolution, making it more difficulty to capture the dynamics of the visual difference between the underlying classes of events. If a larger context is considered, say several adjacent clips, the effect of adding visual features may become more obvious.

Number of mixture	Features used		
	Audio	Visual	Combined
2	9.29	38.2	8.83
4	8.60	31.6	8.43
8	8.66	30.3	7.97
16	10.1	31.2	8.26
32	9.98	35.7	8.20

Table 2.1: Error rates of GMM classifiers. (unit: %)

Kernel type	Features used		
	Audio	Visual	Combined
Dot product	7.34	26.8	6.89
2nd polynomial	8.03	26.2	8.37
3rd polynomial	9.98	26.2	8.20
Radial basis function	10.67	26.4	7.51

Table 2.2: Error rates of SVM classifiers. (unit: %)

Table 2.2 presents the classification results from SVM classifiers based on audio only, visual only and combined features. Four kernel functions are tested: dot product, 2nd, and 3rd order polynomials, and radial basis function. The γ coefficient used in radial basis function is set to 0.1. Best performance occurs when combined features are used with dot product as the kernel function. With SVM approach, adding visual information introduces about 6% of improvement compared to using audio information only.

Although linear fuzzy threshold classifier gives poorer performance, it is simple and light weight in computation. SVM classifiers give the best performance, but the computation cost is higher in both training and testing stages compared to those of GMM classifiers.

2.3.2 Classification of Commercial, Basketball, Football, News, and Weather Forecast

We have collected 20 minutes of broadcast video for each of the five classes of TV programs. The audio data is sampled at 22.05 KHz and 16 bits per sample, and the visual data is acquired at 10 frames per second with resolution 240×180. We separate data into training set and testing set arbitrarily. Each scene class in both sets contains about 10 minutes of video. The video data is divided into 1.5 seconds long clips and each clip overlapped with the previous clip by 1 second. For each clip, we extracted fourteen audio clip features and fourteen visual clip features described in Section 2.1. The visual features include DCR, DCG, DCB, DCRSTD, DCGSTD, DCBSTD, DCP, DCPSTD, DMX, DMY, DMXSTD, DMYSTD, DMP, and DMPSTD. The feature vectors of every 20 successive clips form one observation sequence (11 seconds long) for the HMM classifier. To improve the efficiency of limited data, we generate the next sequence by shifting one clip from the previous one. This yield about 1,000 training sequences and 1,000 testing sequences for each scene class.

Since there is no simple theoretical way to choose the number of states and the number of observation symbols, we test different combinations. Table 2.3 gives the average classification accuracies for different choices of state/observation symbol combinations using audio features alone. The average accuracy is defined as the average of the percentages of a class being correctly classified, averaged over five classes. As seen from the table, the HMM classifiers perform best when the number of states is 5 and the number of symbols is 512. The average accuracy varies slightly as the number of states changes while the observation size is fixed at 256, whereas the variation is more significant when the observation size is 512. In general, the performance improves initially as the number of observation symbols increases, but then starts to drop as the number of symbols exceeds a certain limit. This is partly because the number of training sequences becomes insufficient as the number of model parameters to be trained increases.

Number of Symbols	Number of States				
	4	**5**	**6**	**7**	**8**
64	74.94	76.45	74.19	74.35	76.94
128	76.19	75.85	77.66	78.36	76.48
256	80.53	79.71	78.11	80.69	80.07
512	79.27	81.08	77.68	76.32	80.28

Table 2.3: Average classification accuracy using audio features based on HMM with different numbers of states and symbols

Table 2.4 shows the detailed classification results for the HMM with 5 states and 256 symbols using audio features only. The classifier can accurately classify basketball games, football games and weather forecast. However, a high percentage of news report is misclassified as weather because pure speech is dominant in both of these events. The average accuracy of classifying the three super-classes (commercials, basketball/football games, and news/weather reports) is 93.37%.

	Output Class				
	ad	*bskb*	*ftb*	*news*	*wth*
ad	75.66	7.36	0.38	15.66	0.94
bskb	1.46	91.79	5.29	1.46	0.00
ftb	1.82	13.28	83.64	1.26	0.00
news	0.00	0.19	4.58	57.55	37.68
wth	0.00	0.00	0.00	10.08	89.92

Table 2.4: Classification accuracy of 5-state HMM with 256 observation symbols using audio features only (Average accuracy: 79.71 %)

We also test the neural network approach [41] and GMM classifier on the same data sets. For neural network classifier, the overall classification accuracy is 72.8% and the classification accuracy of commercial, games, news/weather report is 86.8%. We get 7.0% improvement of overall classification accuracy by using HMM. For GMM classifier, we test four cases of mixture numbers: 8, 16, 32, and 64. The average accuracies obtained with different number of mixtures are about 60%, and the results are not very sensitive to the number of chosen mixtures.

By Multiple Discriminant Analysis, we reduce the dimension of audio features from 14 to 4. Classification results using the reduced set of features are similar to the original set of features. Table 2.5 gives the results for 5-state HMM with 256 observation symbols.

	Output Class				
	ad	*bskb*	*ftb*	*news*	*wth*
ad	86.03	9.34	0.75	3.11	0.75
bskb	8.76	85.22	5.66	0.36	0.00
ftb	5.14	8.06	86.40	0.40	0.00
news	1.17	0.00	4.28	66.12	28.43
wth	0.00	0.00	0.00	20.64	79.36

Table 2.5: Classification accuracy of 5-state HMM with 256 observation symbols using the reduced audio feature set by MDA (Average accuracy: 80.63 %)

Table 2.6 shows the results for 5-state HMM with 256 observation symbols using both audio and visual features. They are simply concatenated into a big feature vector. The results are much better than those using audio features alone. The reason may be due to the discrimination capability of visual features in separate news and weather, as well as basketball games and football games.

2.4 Summary

In this chapter, we present a solution for perceptual level video indexing, specifically, we solve the problem of detecting different video events in broadcast

	Output Class				
	ad	*bskb*	*ftb*	*news*	*wth*
ad	91.23	7.08	0.00	1.60	0.09
bskb	2.55	86.13	8.21	3.10	0.00
ftb	1.58	1.34	94.31	2.77	0.00
news	2.63	1.66	3.02	64.95	27.75
wth	0.00	0.00	0.00	4.17	95.83

Table 2.6: Classification accuracy of 5-state HMM with 256 observation symbols using both audio and visual features.(Average accuracy of 86.49 %).

news. Based on the audio/visual features, several different classification schemes are tested. While all of them give reasonably good results, the Gaussian Mixture Model based Maximum Likelihood classifier is attractive for computational efficiency and good performance. Different combinations of audio/visual feature sets and classifiers are suitable for different tasks. The indexing results at perceptual level provide fundamental content units for the conceptual level multimedia content analysis.

Chapter 3

News Story Hierarchy Generation

This chapter addresses the conceptual level content analysis on structured multimedia data such as broadcast news. Extraction of semantically meaningful content for the purpose of information indexing and retrieval has long been a strong interest in the research community. Such tasks are mainly achieved, in the past, through data analysis in individual media [61, 62]. There are difficulties associated with these existing techniques, especially the performance suffered from lack of either reliability in single media processing or reinforcement from other media. This chapter presents how to utilize the new dimensions that multimodal data represents and to explore novel solutions that can achieve much more than what one can achieve from one media alone.

Using the techniques described in the chapter, a hierarchy of different types of content can be automatically identified. Examples of such content include different speakers (e.g., anchor), news reporting (correspondences or interviews), news stories, news summaries, or commercials. From such extracted semantics, a Table-of-Content (ToC) can be constructed that provides a compact yet meaningful abstraction of the data, serving as an effective index table. Compared with conventional linear information browsing or keywords based search with a flat layer, the enabled ToC facilitates non-linear browsing capability that is especially desired when the amount of information is huge.

3.1 News Story Hierarchy and Involved Techniques

We observe that a typical national news program consists of news and commercials. News consists of several headline stories, each of which is usually introduced and summarized by the anchor prior to and following the detailed report by correspondents, quotes, and interviews from news makers. Commercials are usually found between different news stories. With this observation, we propose an integrated solution to recover this content hierarchy by utilizing cues from different media whenever it is appropriate. Figure 3.1 shows the hierarchy we intend to recover.

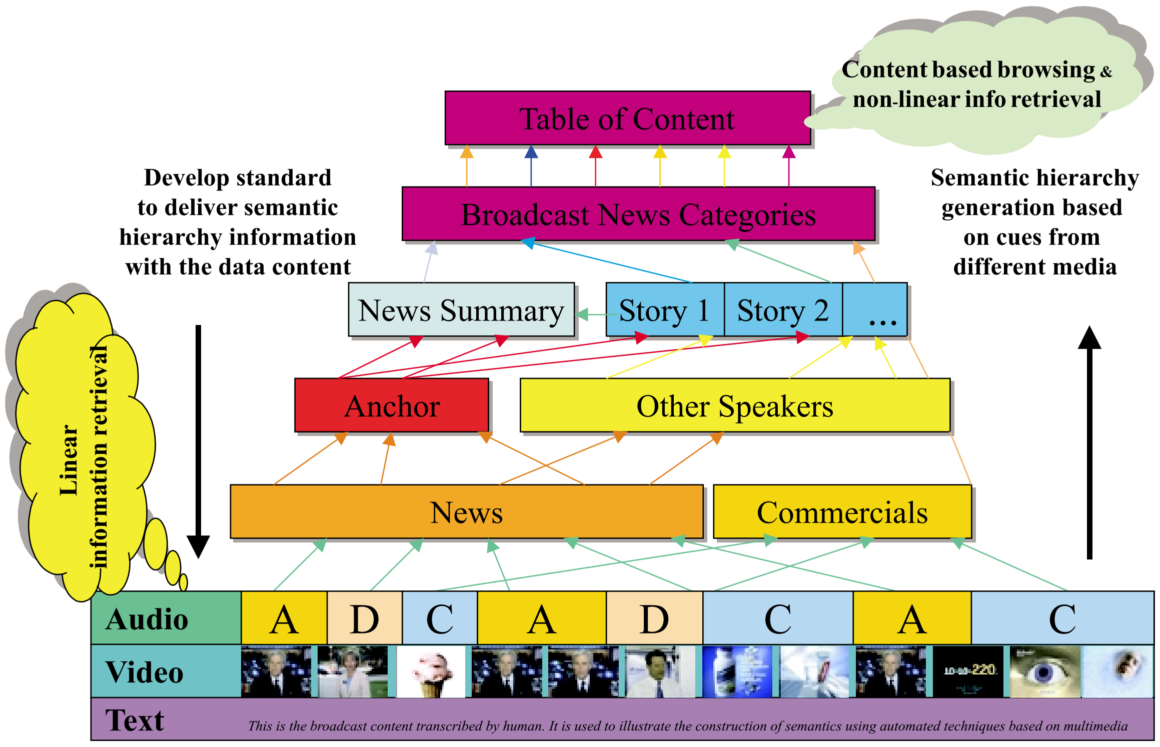

Content based browsing & non-linear info retrieval
Table of Content
Develop standard to deliver semantic hierarchy information with the data content
Broadcast News Categories
Semantic hierarchy generation based on cues from different media
News Summary
Story 1
Story 2
...
Linear information retrieval
Anchor
Other Speakers
News
Commercials
Audio
A
D
C
A
D
C
A
C
Video
Text
This is the broadcast content transcribed by human. It is used to illustrate the construction of semantics using automated techniques based on multimedia

Figure 3.1: Content hierarchy of broadcast news programs.

In this hierarchy, the lowest level contains the continuous multimedia data stream (audio, video, and text). At the next level, we separate news from commercials (C). The news is then segmented into the anchor person's speech (A) and the speech from others (D). The intention of this step is to use detected anchor's identity to hypothesize a set of story boundaries that consequently partition the continuous text (synchronized using speech text alignment technology) into adjacent blocks of text. Higher levels of semantic units can then be extracted by grouping the text blocks into news stories and news introductions. In turn, each news story can consist of either the story by itself or augmented by the anchor person's introduction. Detailed semantic structure at the story level is shown in Figure 3.2.

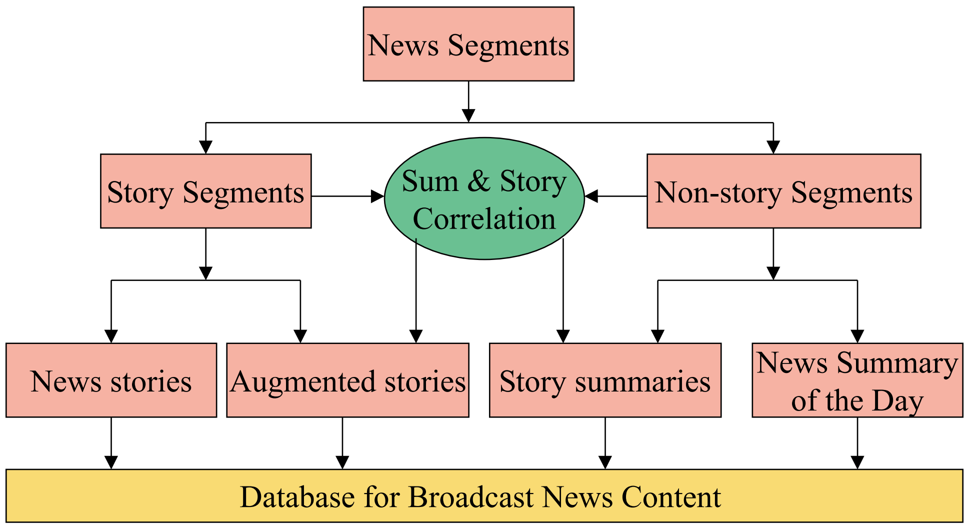


Figure 3.2: Relationship among the semantic structures at story level.

In this Figure, input consists of news segments with boundaries determined by the location of anchor person segments. Commercial segments are not included. Using duration information, each news segment is initially classified as either the story body (having longer duration) or news introduction or non-story segments (having shorter duration). Further text analysis verifies and refines the story boundaries, the introduction associated with each news story, and the news summary of the day.

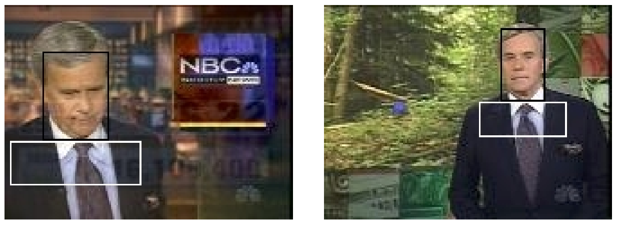
The news data is segmented into multiple layers in a hierarchy to meet different needs. For instance, some users may want to retrieve a story directly; some others may want to listen to the news summary of the day in order to decide which story sounds interesting before making further choices; yet others (e.g., a user employed in the advertising sector) may have a totally different need: to monitor commercials from competitors in order to come up with a competing commercial. Our segmentation mechanism partitions the broadcast data in different ways so that direct indices to the events of different interests can be automatically established.

3.2 Anchor Person Detection

Automatically detecting a specific person is often instrumental in automated video indexing tasks. For instance, identifying the anchor persons in broadcast news can help to recover various kinds of content such as news stories and news summary [24, 63, 64]. Most of the existing approaches to this problem are based on either acoustic [65, 24] or visual properties [64] alone. Some targeted at detecting a predefined anchor (supervised). Some aimed at detecting whoever the anchor is from the given data (unsupervised). While supervised detection can be useful in identity verification tasks, it is usually not adequate in detecting unspecified anchors. In this section, we address the problem of unsupervised anchor detection. Given a broadcast news program, we like to accurately identify the segments corresponding to whoever the anchor is.

Most of the work in detecting a particular host are based on either visual (appearance) or acoustic (speech) cues only. In visual based detection, there are two classes of approaches. One is model based and the other is clustering based. The former often uses a visual template as the model that usually includes both the target as well as the background. Such models are not flexible and not scalable. Depending on what is being used in the model (anchor or anchor shot), this class of methods can be very sensitive to (1) the appearance of the anchor (especially when different anchors appear on different dates of the same program), (2) the studio background (color and the visual content in the background), and (3) the location and the size of the anchor.

With an unsupervised clustering approach, keyframes are clustered and the anchor keyframes may be identified as the ones from the largest cluster. This kind of methods will work only when the visual appearance of the studio scenes within the same program basically remains the same. From the recent data that we acquired from different news broadcasters, this property is often not true. Figure 3.3(a) and Figure 3.3(b) show two anchor scenes from NBC Nightly News program on the same day. From there, we can see that the location and scale of the anchor are very different and the background change is more dramatic. When the assumption of similar appearance does not hold, anchor keyframes are conceivably scattered across several clusters. Another problem is that there is sometimes no anchor appearance when the anchor is speaking. Obviously, such anchor segments can only be recovered when the audio information is simultaneously utilized in anchor detection.



(a) $keyframe$ 379 (b) $keyframe$ 467

Figure 3.3: Two anchor keyframes from NBC Nightly News on April, 14, 1999.

In audio based anchor detection, there are two parallel categories of techniques. One is model based and the other is unsupervised clustering based. The model based methods have similar weakness as in the visual domain. On the other hand, clustering based methods are usually very sensitive to background noise in the audio track such as music or environmental sounds. If visual information is considered at the same time, the noisy anchor speech segments may be recovered by relying on the visual cues.

The approach proposed in this section is precisely to exploit both types of cues and utilize them to compensate each other. Although our goal is to perform unsupervised detection, our approach is model based (supervised) with the distinction (compared with conventional off-line model based method) that our audio/visual models will be built on-the-fly. Simultaneous exploitation of both audio and visual cues enables the initial on-line collection of appropriate training data which will be subsequently used to build the adaptive audio/visual models for the current anchor. The adapted models can then be used, in the second scan, to more precisely extract the segments corresponding to the anchor.

3.2.1 Adaptive Anchor Detection Using On-Line Audio/Visual Models

To adaptively detect an unspecified anchor, we present a new scheme depicted in Figure 3.4. There are two main parts in this scheme. One is visual based detection (top part) and the other is integrated audio/visual based detection. The former serves as a mechanism for initial on-line training data collection where possible anchor video frames are identified by assuming that the personal appearance (excluding the background) of the anchor remains salient within the same program.

Two different methods of visual based detection are described in this diagram. One is along the right column where audio cues are first exploited that identify the theme music of the given news program. From that, an anchor frame can be reliably located, from which a feature block is extracted to build an on-line visual model for the anchor. Figure 3.3 illustrates the feature blocks for two anchor frames. From this figure, we can see that the feature blocks capture both the style and the color of the clothes and they are independent of the image background as well as the location of the anchor. By properly scaling the features extracted from such blocks, the on-line anchor visual model built from such features are invariant to location, size, scale, and background. With the model, all other anchor frames can be identified by matching against it.

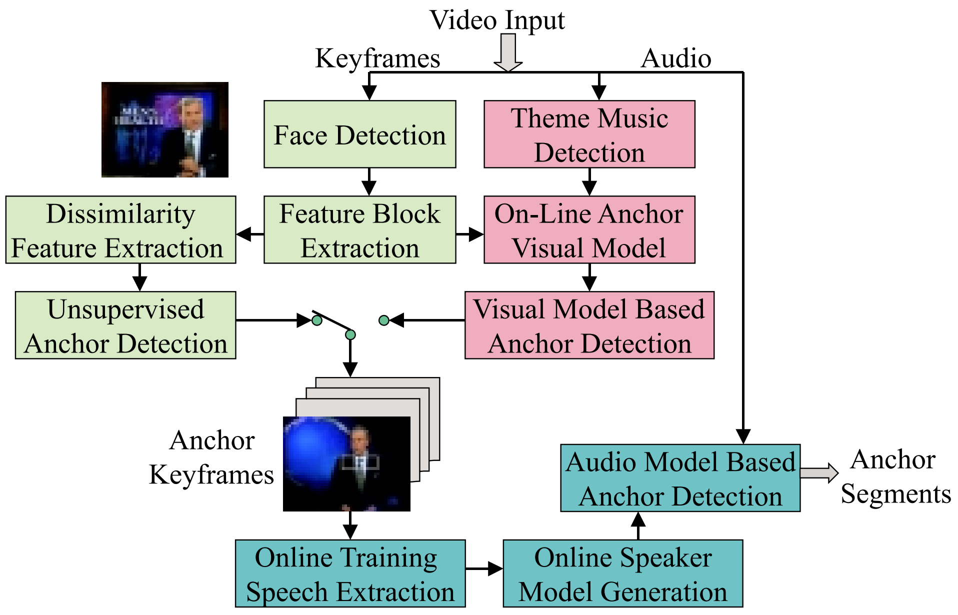

Video Input
Keyframes
Audio
Face Detection
Theme Music Detection
Dissimilarity Feature Extraction
Feature Block Extraction
On-Line Anchor Visual Model
Unsupervised Anchor Detection
Visual Model Based Anchor Detection
Anchor Keyframes
Audio Model Based Anchor Detection
Anchor Segments
Online Training Speech Extraction
Online Speaker Model Generation

Figure 3.4: Diagram of proposed integrated algorithm for anchor detection.

The other method for visual based anchor detection is for when there is no acoustic cues such as theme music present so that no first anchor frame can be reliably identified to build an on-line visual model. In this scenario, face detection is applied and then feature blocks are identified in a similar fashion for every detected human face. Once invariant features are extracted from all the feature blocks, dissimilarity measures are computed among all possible pairs of detected persons. An agglomerative hierarchical clustering is applied to group faces into clusters that possess similar features (same cloth with similar colors). Given the nature of the anchor's function, it is clear that the largest cluster with the most scattered appearance time corresponds to the anchor class. Both methods described above enable an adaptive anchor detection in visual domain.

Visual based anchor detection only is not adequate because there are situations where the anchor speech is present but not the anchor appearance. To precisely identify all anchor segments, we need to recover these segments as well. This is achieved by combining with audio based anchor detection. The visually detected anchor keyframes from the video stream identify the locations of the anchor speech in audio stream. Acoustic data at these locations can be gathered as the training data to build an on-line speaker model for the anchor, which can then be applied, together with the visual detection results, to extract all the segments from the given video where the anchor is present.

3.2.2 Theme Music Detection

One salient landmark in a news program is the theme music. The anchor in news usually appears right after the theme music. Therefore, identifying the theme music in audio stream will help to extract an on-line model of the current anchor, from which remaining anchor frames can be recovered via similarity matching.

To detect theme music, we extract seven frame-level auido features. They are Root Mean Square (RMS) Energy, Zero Crossing Rate (ZCR), Frequency Centroid (FC), Bandwidth (BW), and SubBand Energy Ratio (SBER) in three subbands. Detailed description of these features can be found in Section 2.1.1. A template is built against a particular chosen theme music. Since the playback rate for theme music is always constant, there is no need to apply expensive dynamic programming in template matching. In such situations, linear correlation evaluation works adequately well. Let $\mathbf{T} = (\mathbf{t}_1, ...\mathbf{t}_N)$ be the target theme music template, and $\mathbf{O} = (\mathbf{o}_1, ...\mathbf{o}_M)$ be the testing sequence, where $\mathbf{t}_i$ and $\mathbf{o}_i$ are extracted feature vectors (column wise) from corresponding i^{th} frame and M and N are the frame number of two sequences. The similarity between the template and the testing sequence at n^{th} frame is defined as:

$$S(n) = \frac{\sum_{i=1}^{N}(\mathbf{t}_i - \overline{\mathbf{t}})^T \cdot (\mathbf{o}_{i+n} - \overline{\mathbf{o}_n})}{\sqrt{\sum_{i=1}^{N} \|\mathbf{t}_i - \overline{\mathbf{t}}\|^2}\sqrt{\sum_{i=1}^{N} \|\mathbf{o}_{i+n} - \overline{\mathbf{o}_n}\|^2}}, \quad n = 0, ..., M - N \tag{3.1}$$

where $\overline{\mathbf{t}}$ is the mean feature vector of the template, $\overline{\mathbf{o}_n}$ is the mean of the testing frames $\mathbf{o}_n$, ... $\mathbf{o}_{n+N}$, $\| * \|$ is norm. When $S(n)$ is found to be a local maximum and its value is higher than a preset threshold, it is declared as the beginning of the theme music. Figure 3.5 shows the similarity values of one theme music for a half hour news program. The actual beginning time of the target theme music is 96 second, which can be easily detected by simple thresholding. Once the theme music location is specified, a keyframe can be chosen as the anchor using a fixed off-set in time. From the chosen keyframe, anchor face and its feature block (cloth part) can then be localized which serves as the model for visual based matching.

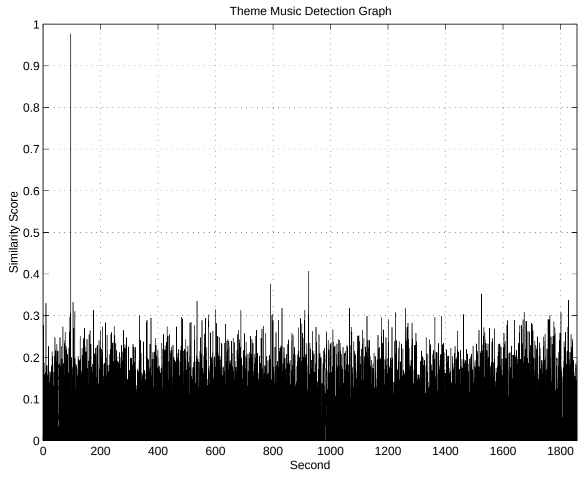


Figure 3.5: Similarity graph of theme music detection

3.2.3 Face Detection

Instead of using expensive face detection algorithms, such as neural network based approach [15], we adopt a light weight detection scheme that uses a skin color model [66] with verifications of facial features based on face projection profiles in X and Y directions. We reasonably assume that the anchor mostly appears as front views. Figure 3.6 illustrates the steps in color based face detection algorithm. There are two major parts: (1) locating the face candidate regions and (2) verifying the face candidates. The first part is composed of three steps: skin tone likelihood computation (against the skin color model), morphological smoothing operation, and region growing. The second part verifies the face candidates using four criteria: shape symmetry, aspect ratio, horizontal, and vertical profiles. Some of the intermediate processing results are shown on the right of the figure.

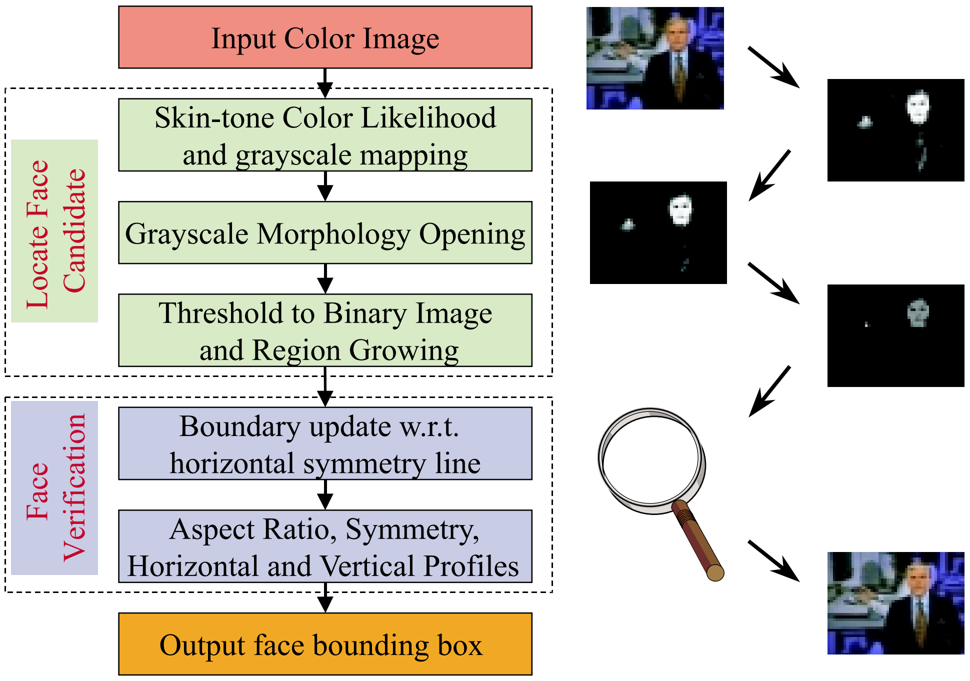

Input Color Image
Skin-tone Color Likelihood and grayscale mapping
Grayscale Morphology Opening
Threshold to Binary Image and Region Growing
Locate Face Candidate
Boundary update w.r.t. horizontal symmetry line
Aspect Ratio, Symmetry, Horizontal and Vertical Profiles
Face Verification
Output face bounding box

Figure 3.6: Diagram of face detection algorithm.

Chroma Chart of Skin Tone

To effectively model skin color, we use the Hue Saturation Value (HSV) color system [67]. Compared with standard Red Green Blue (RGB) color coordinate, HSV produces more concentrated distribution for skin color. Most humans, despite the race and age, have similar skin hue, even though they may have different saturation and values. As value more depends on image acquisition setting, we use hue and saturation only to model human skin color. Figure 3.7 gives the distribution of 2000 training data points, in hue-saturation space, which are extracted from different face samples. Clearly, it is appropriate to use a Gaussian with full covariance matrix to model this distribution. The hue of skin-color centroid is about 0.07, indicating that skin color is somehow between red and yellow. To reduce the boundary effect, we shift the hue-saturation coordinate before computing the skin color likelihood value so that the mean of the Gaussian model is located at the center $(0.5, 0.5)$.

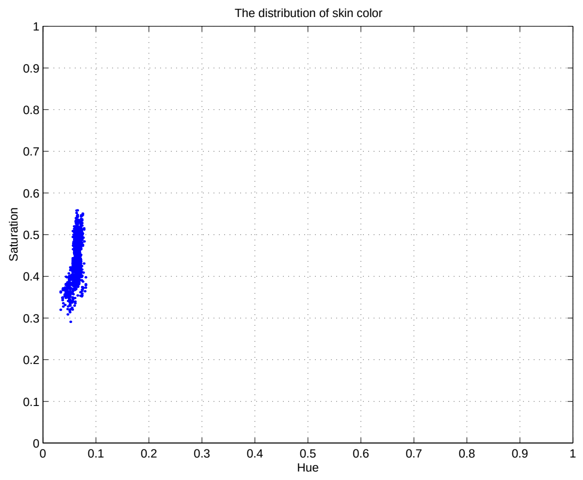


Figure 3.7: The distribution of skin color from selected training data.

Locating Face Candidates

Based on the trained skin color model, a likelihood value can be computed for each pixel. To reduce the noise effect so that connected candidate face regions can be more reliably obtained, we (1) first linearly map the log likelihood value to the range of 0 to 255 and (2) apply gray scale morphology opening operation on the likelihood values. A 3×3 structuring element is applied with amplitude of 64. After thresholding, a blob coloring algorithm [1] is performed on the binary image so that each connected component corresponds to one candidate face region, which can be described by a rectangular box as the bounding box.

Face Verification

Non-face objects (regions) can have similar human skin-like color. Face verification step is designed to further test that the candidate regions detected using color only have other distinct visual features of a human face. A common approach in the literature is to match the candidate region with a face template so that facial configuration can be identified. This method is not only sensitive to lighting condition but also can be computationally expensive. We propose a different method that verifies a face region by testing four criteria. First of all, a face should be symmetric with respect to the center line of the region. Second, a face should be elongated with an acceptable aspect ratio. Although these two simple rules eliminate many fake face candidates, they are not sufficient. The symmetric region may be round or square or totally different from real face shape, we still need to strengthen the verification criteria. Third, since the symmetric shape of face is pretty much an ellipse, the intensity projection profile in X direction should present a nice parabolar shape (see Figure 3.8(a)). Forth, due to distinct facial features (eyes, nose, mouth, and their spatial configurations), the intensity variations projected along Y direction should obey certain characteristic profiles. This can be illustrated in Figure 3.8(b) where three valley points on the curve, denoted as v_1, v_2, and v_3, correspond to eyes (v_1), mouth (v_3), and possibly (not as obvious) shadow of nose (v_2). The last two tests can be done by matching the projected X and Y profiles from the candidate region with the model profiles.

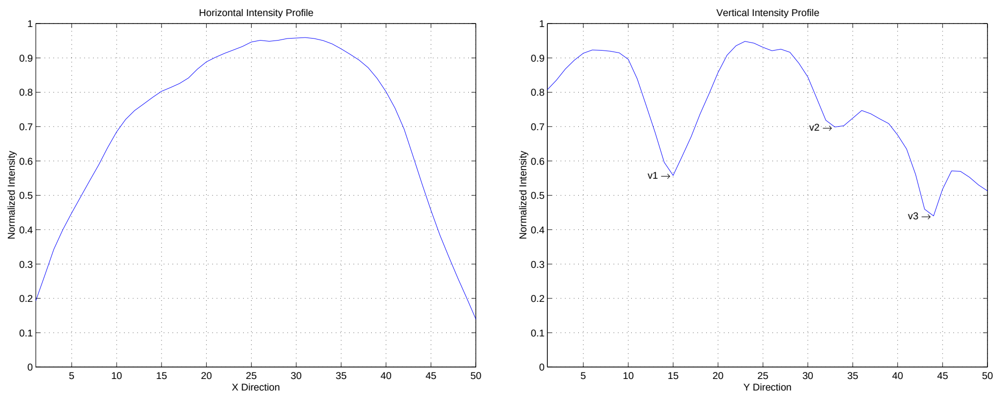
Formally, let FC be the intensity image of candidate region, with height M and width N, and $FC(i,j), 0 \le i < M, 0 \le j < N$, is the intensity value at pixel (i,j). To find the line of symmetry, we need to search all possible horizontal positions to identify the point with maximum symmetric degree, defined as

$$SD(k) = 1 - \frac{\sum_{i=0}^{M-1} \sum_{j=0}^{w_k} |FC(i,k-j) - FC(i,k+j)|}{\sum_{i=0}^{M-1} \sum_{j=0}^{w_k} (FC(i,k-j) + FC(i,k+j))}, \quad \frac{N}{4} \le k \le \frac{3N}{4}, \qquad (3.2)$$

where $w_k = min(k, N-k)$. Suppose the maximum of $SD(k)$ happens when $k = kc$, the left and right boundary of face candidate region is adjusted to $kc - w_{kc}$ and $kc + w_{kc}$. When $SD(kc)$ is adequately high, we further compute the aspect ratio

N/M based on the updated boundary and compare it with preset up bound and low bound thresholds. If the face candidate region passes both symmetry and aspect ratio tests, we move on to the next step.

To generate the facial feature projection model profiles, a set of face images are chosen as training data. To ensure consistent scaling, the projection profiles for individual training faces are scaled properly using certain points on the curves as registration points. For horizontal (X axis) model profile, the registration point is the center of symmetry. For vertical (Y axis) model profile, two most widely separated valley points (eyes and mouth) are identified and used as registration points. Fixing the registration points, individual profiles can be re-scaled so that they all cover the same projection range. Figure 3.8 shows the model profiles. Since the contour of human face is like ellipse, the horizontal profile of face has maximum value around the middle and decreases when approaching both sides. As explained before, the three valley points in Figure 3.8(b) (v_1, v_2,, and v_3) as well as their spatial configurations correspond to distinct facial features.



(a) *X direction profile* (b) *Y direction profile*

Figure 3.8: The projection profiles used in face verification.

Let the horizontal model profile be $M_{HP}(n), 0 \leq n < P$, and the testing horizontal profile from a face candidate region be $T_{HP}(n), 0 \leq n < P$, where P is the length of model profile, the linear correlation between $M_{HP}(n)$ and $T_{HP}(n)$ is computed as

$$Corr(M_{HP}, T_{HP}) = \frac{\sum_{n=0}^{P-1}(M_{HP}(n) - \overline{M_{HP}})(T_{HP}(n) - \overline{T_{HP}})}{\sqrt{\sum_{n=0}^{P-1}(M_{HP}(n) - \overline{M_{HP}})^2}\sqrt{\sum_{n=0}^{P-1}(M_{HP}(n) - \overline{M_{HP}})^2}}. \quad (3.3)$$

where $\overline{M_{HP}}$ and $\overline{T_{HP}}$ are the mean value of $M_{HP}(n)$ and $T_{HP}(n)$. The linear correlation between vertical model profile and the testing profile can be similarly obtained. If the correlation values are higher than preset thresholds, the candidate region is then verified as a face region.

3.2.4 Feature Block Extraction

The features used in visual based anchor detection should be invariant to location, scale, and background scenes. We devise a feature extraction scheme that satisfy these conditions. A rectangular feature block, covering the neck-down clothing part of a person, is localized with a fixed aspect ratio with respect to the detected human face. The reason to use this area is two folds. The appearance of a face is sensitive to both lighting and orientation, making it difficult to be used for recognition or even verification. On the other hand, from the detected faces, we can easily locate the neck-down cloth section as a salient feature block where the color combination of the clothes a person is wearing is fairly robust within one news program. This can be seen from Figure 3.3 where the two keyframes from the same person from the same program indicate that using the detected face information to verify that the two are the same person is very difficult. However, the visual appearances of the two feature blocks are extremely similar if proper scaling and normalization are performed. In addition, by localizing the feature blocks via face detection, the background scenes (even though they can be very different as evidently shown in Figure 3.3) become irrelevant to the detection process.

Assume the rectangular area of a detected face region is $N \times M$, where $N = x_{max} - x_{min}$, $M = y_{max} - y_{min}$, x_{max} and x_{min} are the left and right boundaries,

y_{min} and y_{max} are the top and bottom boundaries. A feature block is then defined as the rectangular $\delta_x \times \delta_y$ where

$$\delta_x = X_{max} - X_{min}, \delta_y = Y_{max} - Y_{min},$$

with

$$X_{min} = max\{0, x_{min} - \frac{1}{2} \times N\}, X_{max} = min\{W - 1, x_{max} + \frac{1}{2} \times N\}$$

and

$$Y_{min} = min\{H - 1, y_{max} + 1\}, Y_{max} = min\{H - 1, Y_{min} + \frac{1}{2} \times M.\}$$

where H and W are the height and width of the input image. Such defined feature block correspondes to the area on a person from neck down. This is illustrated in Figure 3.3 with the feature block superimposed on the anchor image. Since the ultimate objective is to detect anchor person keyframes, we only consider those face regions whose sizes fall into a reasonable range which is true for normal news programs.

3.2.5 Invariant Feature Extraction

The intention of identifying feature blocks is to extract, within the blocks, the features that are useful in identifying the anchor class. Two features are computed from each feature block. Both designed as dissimilarity measures, one measuring the dissimilarity between existing color components and the other measuring the difference in intensity distributions in space. The former is computed based on color histograms, capturing the dominance of color components (but ignoring the spatial information). The latter is derived via motion compensated block matching where the more similar the two feature blocks, the smaller the intensity difference there is. Such matching is performed with proper scaling and normalization of the dynamic range of the intensity values.

We experimented with 3D color histograms. Each of the color channel Red, Green, and Blue is quantized into K bins by performing a mapping $r^q_{x,y} = Q(R_{x,y})$,

$g^q_{x,y} = Q(G_{x,y})$, and $b^q_{x,y} = Q(B_{x,y})$ where Q is the quantization function. Then a 3D color histogram with $K \times K \times K$ bins can be constructed by increasing, for every pixel (x, y) in the feature block, the vote in bin $(r^q(x, y), g^q(x, y), b^q(x, y))$. This forms a sparse histogram in 3D space. To measure the dissimilarity d_h between two feature blocks F_i and F_j with respect to their 3D histograms H^i and H^j, χ^2 is adopted:

$$d_h(F_i, F_j) = \chi^2(H^i, H^j) = \sum_k \frac{(H^i_k - H^j_k)^2}{H^i_k + H^j_k}.$$

In motion compensated block matching, for a corresponding pair of small $n \times n$ regions, each within its feature block, the best matching score is defined as the lowest absolute difference in intensity values and is identified during the search performed in a pre-defined small neighborhood. Since the motion compensated block matching is performed between two feature blocks with most likely different sizes, proper scaling needs to be done. Assume (x, y) is the coordinate of a pixel point within a feature block with size $dx \times dy$ and $dx = x_{max} - x_{min}$ and $dy = y_{max} - y_{min}$. To match this feature block with another feature block $dx' \times dy'$ with $dx' = x'_{max} - x'_{min}$ and $dy' = y'_{max} - y'_{min}$, the scaled counter point of (x, y) is (x', y'), computed as

$$x' = x'_{min} + \frac{dx'}{dx} \times (x - x_{min}), y' = y'_{min} + \frac{dy'}{dy} \times (y - y_{min}).$$

The dissimilarity measure from block matching between two feature blocks, denoted by d_m, is the average absolute intensity difference per pixel after motion compensation.

While color histogram based matching examines the dissimilarity in color composition of the involved feature blocks, it does not indicate that the existing color components are similarly configured in space. Motion compensated block matching provides a measure that can compensate in this regard. Therefore, we combine both features to ensure that both aspects are simultaneously considered in grouping. That is, dissimilarity $D(F_i, F_j)$ between two feature blocks F_i and F_j is defined as: $D(F_i, F_j) = w_h \times d_h(F_i, F_j) + w_m \times d_m(F_i, F_j)$ where w_h is the weight on d_h and w_m is the weight on d_m.

3.2.6 Visual Based Anchor Detection

As described earlier, there are two ways to detect the anchor keyframes in the visual domain. On-line model based approach is enabled when theme music is present. Unsupervised clustering is applied when there is no on-line visual model can be established.

In model based method, given the visual model M_v for the anchor, a feature block F_i is considered as the anchor if $D(M_v, F_i)$ is lower than a pre-defined threshold. In unsupervised method, an agglomerative hierarchical clustering [68] is performed. Initially, each feature block is a cluster on its own. During each iteration, two clusters with minimum dissimilarity value are merged, where the dissimilarity between two clusters is defined as the maximum dissimilarity among all possible pairs of two feature blocks from each cluster. This procedure continues until the minimum cluster dissimilarity is larger than a preset threshold. Due to the fact that anchor is the host of the program, hence with continuous appearances, the largest cluster is finally identified as the anchor class.

Compared with existing unsupervised anchor detection algorithms where the entire image is usually used in clustering, our approach is more accurate, more adaptive, and more robust. The localized feature blocks allow our approach to discard irrelevant background information so that misclassification caused by using such information can be minimized. In addition, as the features are invariant to location, scaling, and certain degree of rotation, the clustering method is able to group anchor frames together despite the fact that the images appear very differently.

3.2.7 Audio/Visual Integrated Anchor Detection

In broadcast news data, there are situations where anchor speech and anchor appearance do not co-exist. To use the anchor as the landmark to index content, we need to extract all video segments where anchor's speech is present. Therefore, visual based detection result is not adequate. In our scheme, it serves initially as the mechanism to adaptively collect appropriate audio training data so that an on-line acoustic model for the anchor can be dynamically established. Detected anchor keyframes identify the audio clips where the anchor is present, that can be used to train a speaker model. The on-line trained acoustic model is then applied back to the video stream, for the second scan, to extract anchor speech segments.

Rose and Reynolds [69, 18] reported that maximum likelihood method based on Gaussian Mixture Model is suitable for robust text-independent speaker recognition task. We also use GMM to model speakers. The features we used are 13 order Mel-frequency cepstral coefficients (MFCCs) [16], pitch period, 13 delta MFCCs, and delta pitch period. These 28 features are computed every 16 msec. Based on them we build a target GMM for anchor person and also a background GMM for non-anchor audio, which includes environmental noise, music, and speech of other persons. The number of mixtures of both models is chosen to be 64 based on our benchmark studying. There are two types of anchor models: off-line and on-line models. To train off-line model for known anchor, we use training speech collected for the specified anchor. To build the on-line anchor model for unknown anchor, we use the audio signal accompanying the anchor keyframes.

During detection step, we compute the log likelihood ratio (LLR) of input frame regarding to anchor GMM and background GMM. To smooth out the grainy effect of frame based LLR value, we consider the average LLR value within a clip. When the average LLR is higher than certain threshold, we classify the corresponding clip as anchor speech. Three tap Median filter is used as post-processing to further rectify and smooth the recognition results. Finally we remove all anchor segments which are shorter than 6 seconds and merge neighboring anchor segments which are less than 6 seconds away. This heuristic rule is commonly true for news programs.

3.3 News Story Extraction

Up to this point, we have a set of hypothesized story boundaries as shown in Figure 3.9. The segments with label "A" indicates that they are anchor segments, "D" detailed news reporting, and "C" commercials. With identified "A" segments, the synchronized text can be partitioned into two sets of text blocks:

$$T_1 = \{T_1^1, T_1^2, ..., T_1^n\},$$

$$T_2 = \{T_2^1, T_2^2, ..., T_2^n\},$$

where T_1^i is a block of text that starts with anchor speech and T_2^i is a subblock of T_1^i containing only the text from the anchor speech. Based on the structure of the broadcast news, each news story consists of one or more T_1^i's.

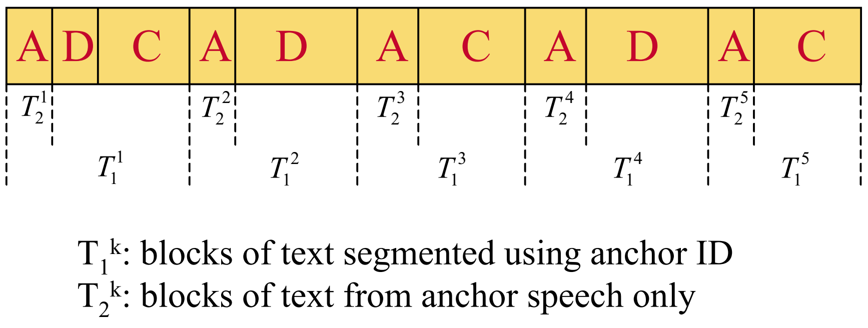

A D C A D A C A D A C
T_2^1 T_2^2 T_2^3 T_2^4 T_2^5
T_1^1 T_1^2 T_1^3 T_1^4 T_1^5
T_1^k: blocks of text segmented using anchor ID
T_2^k: blocks of text from anchor speech only

Figure 3.9: Illustration of how the detected anchor segments lead to initial text partition for story segmentation.

Our goal is to extract three classes of semantics: news stories, augmented stories (augmented by the introduction of the story by the anchor), and news summary of the day. At this stage, text cues are further integrated with the cues from audio and video in performing the analysis to (1) separate news stories and news introductions, (2) verify story boundaries, (3) for each detected story, identifies the news introduction segment associated with that story, and (4) form news summary of the day by finding a minimum set of news introduction segments that cover all the detected stories.

With blocks of text available at this point, the task is to determine how these blocks of text can be merged to form semantically coherent content based on appropriate criteria. Since news introductions are to provide a brief and succinct message about the story, they naturally have much shorter durations than the detailed news reports. Based on this observation, we initially classify each block of text as a story candidate or an introduction candidate based on duration. Such initial labels are shown in Figure 3.10(b) where I stands for introduction and S stands for story. The remaining tasks are to verify the initial classification of news introductions and stories and to form three classes of semantics indicated in Figure 3.10(c): individual news stories, augmented news stories, and a news summary. A news story represents merely the story body itself. An augmented story consists of the introduction that previews the story and the story body. The news summary of the day is composed of the introductions for each and every news story reported on that day. For example, in Figure 3.10(c), the second augmented story is formed by the third introduction section and the second story body. The news summary of the day does not necessarily include all the introduction sections. What we are seeking is a mimimum set of anchor speech that previews all the headline stories. For example, in Figure 3.10(c), the second introduction section is not included in news summary of the day.

Formally, our input data for text analysis is two sets of blocks of text: $T_1 = \{T_1^1, .., T_1^i, .., T_1^m\}$ where each T_1^k, $1 \leq k \leq m$, begins with the anchor person's speech (corresponding to the blocks shown in Figure 3.10(b)) and $T_2 = \{T_2^1, .., T_2^j, .., T_2^n\}$ where each T_2^k, $1 \leq k \leq n$, contains only the anchor's speech. Since the blocks in both sets are all time stamped, we have $m = n$ and $T_2^k \subseteq T_1^k$. To verify story boundaries, we evaluate similarity measure $sim()$ between every pair (T_{b_1}, T_{b_2}) of adjacent blocks [70]:

$$sim(T_{b1}, T_{b2}) = \frac{\sum_w f_{w,b_1} \times f_{w,b_2}}{\sqrt{\sum_w f_{w,b_1}^2 \times \sum_w f_{w,b_2}^2}}.$$

Here, w enumerates all the token words in each text block; f_{w,b_i} is the weighted

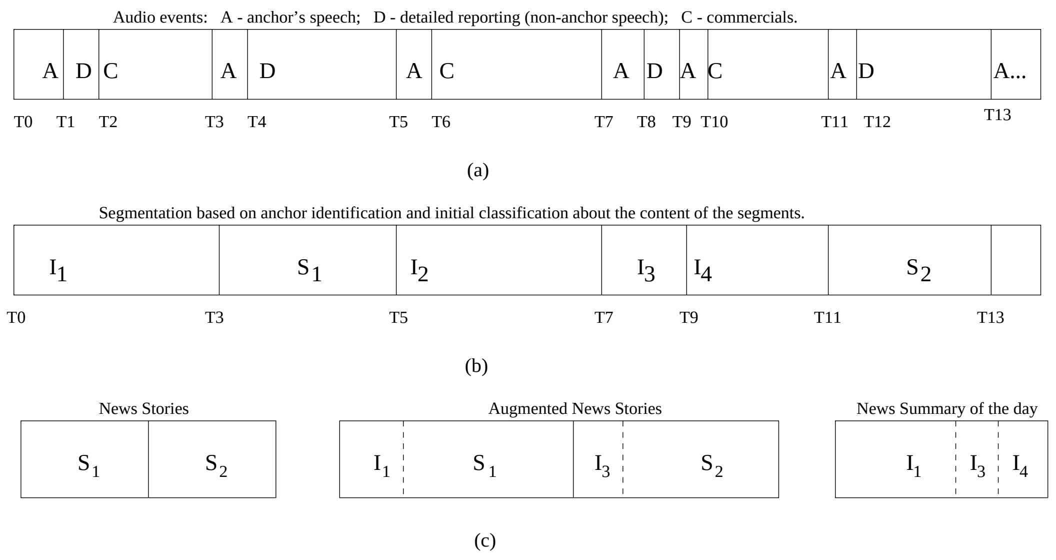

Audio events: A - anchor's speech; D - detailed reporting (non-anchor speech); C - commercials.
A D C A D A C A D A C A D A...
T0 T1 T2 T3 T4 T5 T6 T7 T8 T9 T10 T11 T12 T13
(a)
Segmentation based on anchor identification and initial classification about the content of the segments.
I1 S1 I2 I3 I4 S2
T0 T3 T5 T7 T9 T11 T13
(b)
News Stories
S1 S2
Augmented News Stories
I1 S1 I3 S2
News Summary of the day
I1 I3 I4
(c)

Figure 3.10: Illustration of the process of story boundary identification.

frequency of word w in block b_i, $i \in 1, 2$; and $0 \leq sim() \leq 1$. Here, token words are extracted by excluding all the stop words from the text. The frequency of each token word is then weighted by the standard frequency of the same word computed from a corpus of broadcast news data collected from NBC Nightly News in 1997. The higher the frequencies of the common words in the two involved blocks are, the more similar the content of the blocks is. We experimentally set up a threshold to determine the story boundaries.

The output of the story boundary verification is a set of text blocks

$$S = \{S_1, S_2, ..., S_m\},$$

where $S_i = T_1^j - T_2^j, 1 \leq i, j \leq n$. With news stories segmented, we take set T_2 and the story set S as input to further extract other classes. For each story, we extract its introduction by finding a T_2^k that has the highest similarity to that story (T_2^k is not necessarily connected) Merging each story with its introduction, we form an augmented story. That is, using S and T_2, augmented news stories set

$$S^a = \{S_1^a, S_2^a, ..., S_m^a\}$$

can be generated by identifying each

$$S_i^a = S_i \bigcup T_2^j, 1 \leq i \leq m, 1 \leq j \leq n$$

such that $sim(S_i, T_2^j)$ is maximized. Notice here, different S_i may associate with the same T_2^j.

The news summary of the day is extracted with the criterion that it has to provide the minimum coverage for all the stories reported on that day. Therefore, it is a minimum set of T_2^k's that together introduces all the stories of the day without overlap (i.e., each story has to be introduced but only once). Based on this, a set of text blocks from T_2 is chosen to form *news summary of the day* by using the following criterion:

$$NS = \bigcup_{1 \leq k_i \leq n} T_2^{k_i},$$

such that $\sum_{i=1}^{m} sim(S_i, T_2^{k_i})$ is maximized. With such a higher level of abstraction, users can browse desired information in a very compact form without losing primary content.

In contrast to conventional discourse segmentation methods, our grouping criterion is simultaneously based on audio/visual cues. Since anchor-based segmentation provides the initial grouping of text, in effect, (1) adaptive granularity that is directly related to the content is achieved, (2) the hypothesized boundaries are more natural than those obtained using a fixed window, (3) blocks formed in this way not only contain enough information for similarity comparison but also have natural breaks of chains of repeated words if true boundaries are present, (4) the original task of discourse segmentation is achieved by boundary verification, and (5) once a boundary is verified, its location is far more precise than what conventional discourse segmentation algorithms can achieve. This integrated multimodal analysis provides an excellent starting point for the similarity analysis and boundary detection.

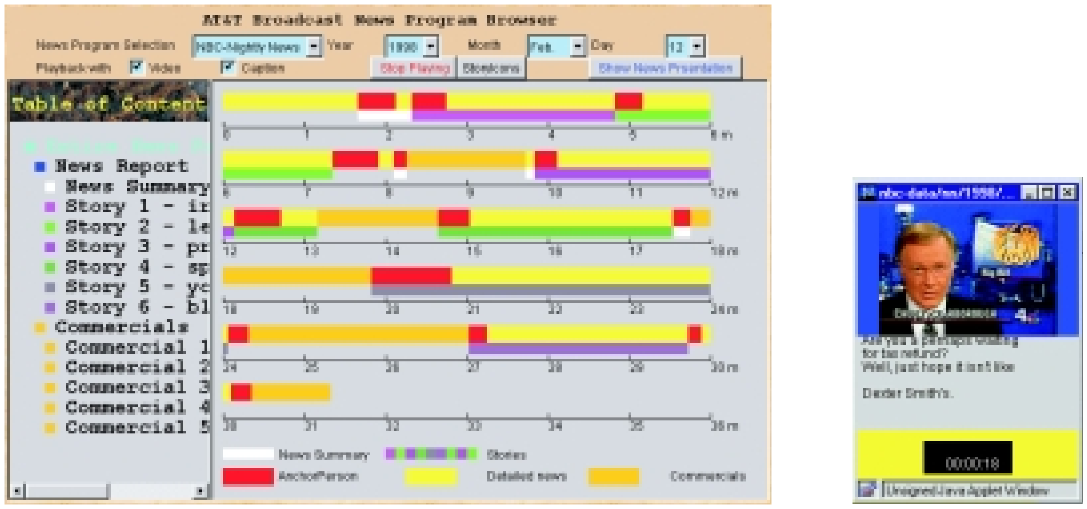
Differing from most studies in the literature where the processing is applied only to adjacent blocks of text, some of the semantics we attempt to extract require merging of disconnected blocks of text. One example is the news summary of the day (because the anchor's introductions to different headline stories are scattered throughout the half-hour program).

3.4 News Story Presentation

In the previous sections, we addressed a mechanism to recover the semantic structure of the data so that it can be used for creating a table of content for the news. For effective retrieval, there is another equally important task related to human machine interface: how to present the extracted semantic units in a form that is compact, concise, easy to understand, and at the same time visually pleasing. Now, we examine three aspects of this task. First, how to present the semantic structure to the users; second, how to represent the particular semantics based on the content of the news story; and third, how to form the representation for news summary of the day.

3.4.1 Representation for News Semantic Structure

A commonly used presentation for semantic structure is in the form of a table of content. Since this concept is familiar to most users, we employ it in our representation as well. In addition, in order to give users a sense of time, we also design a streamline representation for the semantic structure. Figure 3.11(a) shows our presentation for the semantic structure of a news program. On the left of the screen, different semantics are categorized in the form of a table of content (commercials, news, and individual news stories, etc.). It is in a familiar hierarchical fashion which indexes directly into the time stamped media data. Each item listed is color coded by an icon of a button. To play back a particular item, a user simply clicks on the button of the desired item in this hierarchical table. On the right of this interface is the streamline representation where the time line runs from left to right and top to bottom. The time line has two layers of categorization. The first layer is event based (anchor speech, others' speech, and commercials) and the second layer is semantics based (stories, news introduction, and news summary of the day). Each distinct section is marked by a different color and the overall color codes correspond to the color codes used in the table of content. Obviously, the content categorized in this representation is aligned with time simultaneously.



(a) *Representation for semantic structures* (b) *Playback interface*

Figure 3.11: Representation for extracted semantic structures.

These two representations are directly related to each other, although one (table) is more conceptual and the other more visual. When users click on a particular segment in the streamline representation, it triggers the same effect as clicking on a particular item in the table of content. When an item in the table is chosen to be played back, the corresponding segment in the streamline becomes active (flash), which also gives users a sense of time. For example, if a user chooses to play the second story by clicking on the second item under story category in the table of content, the corresponding segment in the streamline representation will blink during the play back. Therefore, while the table of content provides a conceptual abstraction of the content (without the structure along time), the streamline representation gives a description of how content is distributed in the news program. With these two complementary representations, users can quickly get a sense of both the semantic structure of the data and the timing. Through this representation, users can easily perform non-linear retrieval.

Figure 3.11(b) is the wondow which displays streaming playback. It is triggered when users click on a particular item. In this playback window, the upper portion shows the video and the lower portion the text synchronized with the video and audio. Currently, we display only the key frames (as opposed to the original video stream). The text scrolls up with time. In the black box at the bottom, the timing with respect to the starting point of the program is given.

3.4.2 Representation for News Stories

For each extracted news story, we developed two forms of representation. One is textual and the other is combination of text with visual. Our goal is to automatically construct the representation in a form that is most relevant to the content of the underlying story. For textual representation, keywords are chosen from the story according to their importance computed as weighted frequency. In the table of content shown in Figure 3.11(a), next to each story listed, a set of 10 keywords are given. The intention is that users will get a feeling about the content of the story. Another more detailed representation for a story is called "story icon". To invoke it for a particular story, users can click on the "StoryIcon". Figure 3.13 gives one example of such story representation. We designed a content based method to automatically construct this visual story representation.

Within the boundary of each story, a keyword histogram is first constructed as shown in Figure 3.12 where the X axis is the keyframe numbers and the Y axis is the frequency of the keywords. In the figure, the solid curve is the keyword histogram. A fixed number of key frames within the boundary are chosen so that they (1) are not within anchor speech segments and (2) yield maximum covered area with respect to the keywords histogram. The peak points marked on the histogram in Figure 3.12 indicate the positions of the chosen frames and the shaded area underneath them defines the total area coverage on the histogram by the chosen key frames.

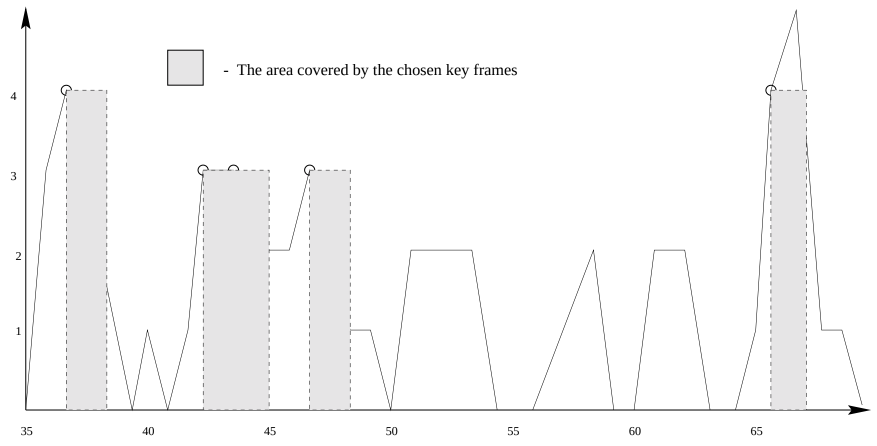
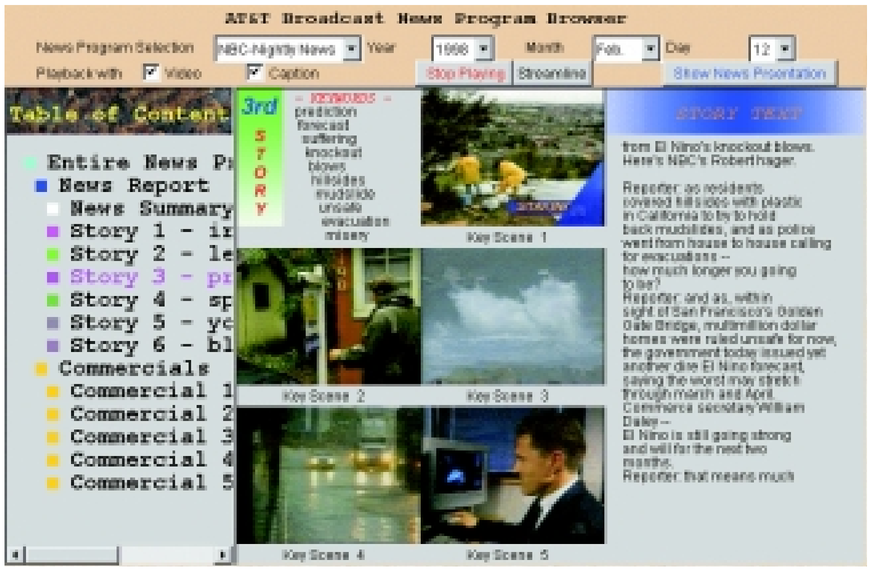


Figure 3.12: Histogram of keywords within a story.



from El Nino's knockout blows.
Here's NBC's Robert hager.

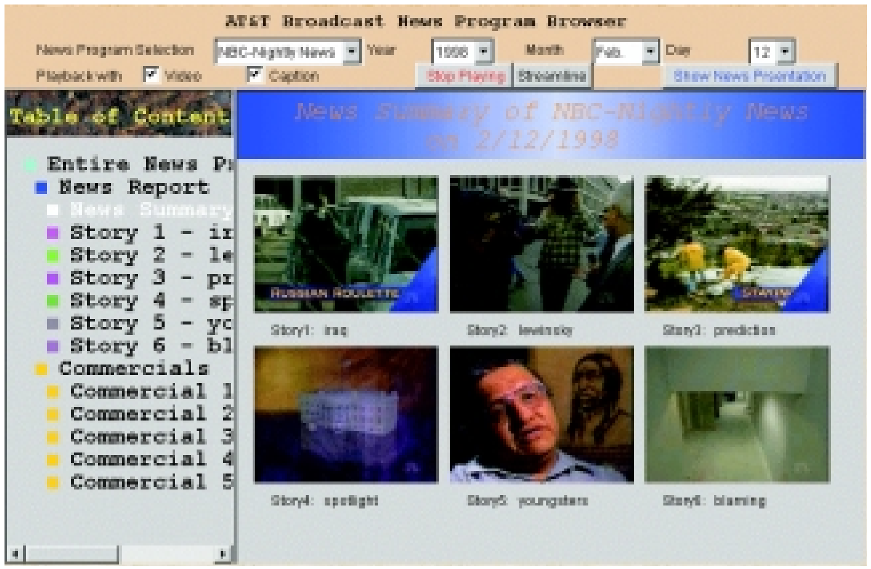
Reporter: as residents
covered hillsides with plastic
in California to try to hold
back mudslides, and as police
went from house to house calling
for evacuations --
how much longer you going
to be?
Reporter: and as, within
sight of San Francisco's Golden
Gate Bridge, multimillion dollar
homes were ruled unsafe for now,
the government today issued yet
another dire El Nino forecast,
saying the worst may stretch
through march and April.
Commerce secretary William
Daley --
El Nino is still going strong
and will for the next two
months.
Reporter: that means much

Figure 3.13: Visual representations for stories about El Nino.

The representation of one story is shown in Figure 3.13. The chosen story is the third (which can be seen in the table of content on the left portion of the interface). The presentation for each story has three parts: the upper left corner is a set of 10 keywords automatically chosen from the segmented story based on the relative importance of the words; the right part displays the text of the story; the rest is the visual presentation of the story consisting of five images chosen from video in the content based manner described above. Figure 3.13 is the visual representation about El Nino story. We can see from this figure that the story representation constructed this way is compact, semantically revealing, and visually informative with respect to the content of the story. Compared with linear browsing or low level shot cut browsing, our system allows a more effective content based non-linear information retrieval.

3.4.3 Representation for News Summary of the Day

Finally, we construct the representation for the news summary of the day. It is composed of k images, where k is the number of headline stories on a particular day. The k images are chosen so that they are the most important in each story, as measured by the covered area size in the keyword histogram. Figure 3.14 gives the visual presentation for the news summary of the day for the NBC Nightly News on 12th of February, 1998. From this presentation, a user can see immediately that there are a total of six headline stories on that particular day. Below the representative image for each story, the list of its keywords is displayed dynamically so that users can get a sense of the story from the keywords. In this example, the first story is about the weapon inspection in Iraq where Russians are suspected to tip Saddam. The second story is about Clinton scandal. The third one is about El Nino. The fourth one is about whether secret service workers should testify against the president. The fifth is about the high suicide rate among youngsters in an Indian village. The sixth is about government's using tax dollars to pay the rent for empty buildings. From these examples, the effectiveness of this story-telling visual representation for the news summary is evident.


AT&T Broadcast News Program Browser

News Program Selection NBC-Nightly News Year 1998 Month Feb. Day 12

Playback with ✓ Video ✓ Caption

Table of Content

Entire News Pr
News Report
News Summary
Story 1 - ir
Story 2 - le
Story 3 - pr
Story 4 - sp
Story 5 - yo
Story 6 - bl
Commercials
Commercial 1
Commercial 2
Commercial 3
Commercial 4
Commercial 5

News Summary of NBC-Nightly News on 2/12/1998

Figure 3.14: Representation for news summary of the day.

3.5 Simulation Results and Discussion

A total of seven half hour broadcast news programs are used for our experiments, collected from NBC Nightly News Broadcast from February to April of 1999. The targeted anchor person is Tom Brokaw. The seven days are February 18, 19, 23, March 3, 8, 9 and April 14, 1999. To simplify the notation, these testing sequences are denoted as 990218, 990219, 990223, 990303, 990308, 990308, 990309, and 990414 respectively. Each program covers about 5 minutes anchor speech, scattered in the program. The audio signal is sampled at 16kHz and 16 bits per sample. Due to the size of raw visual data, only keyframes are retained after real-time shot change detection operation [71]. The image size of each keyframe is 160 by 120. As the keyframes are compressed in JPEG format, the quality is degraded, which poses a challenge to our face detection algorithm.

In separating news and commercials, we use the trained model in Section 2.3 with median filter smoothing as post-processing. The experimental results from GMM model-based approach is shown in Table 3.1 which illustrates that the best average classification error rate is 2.9% when four component mixtures are used. Table 3.2 gives the classification results using the SVM method. Three kernels, dot product, second order polynomial, and third order polynomial, are tested and the former two provided better performance. Comparing GMM with SVM, it can be seen that the results are comparable with GMM result (2.9% error rate) slightly better than that of SVM (3.4% error rate).

# of components	2	4	8	16	32
990218	1.1%	0.7%	0.9%	0.8%	0.8%
990219	1.2%	2.0%	1.6%	1.7%	1.7%
990223	4.3%	3.7%	5.3%	4.3%	3.7%
990303	2.3%	2.4%	2.8%	2.8%	2.3%
990308	4.3%	4.0%	4.1%	2.9%	2.9%
990309	7.0%	7.4%	7.1%	7.5%	7.5%
990414	2.4%	0.0%	2.7%	1.8%	1.8%
Overall	3.2%	**2.9%**	3.5%	3.1%	3.0%

Table 3.1: Classification error rates using GMM model based classifiers.

Kernel Type	Dot	Poly2	Poly3
990218	1.8%	1.4%	1.5%
990219	1.1%	3.4%	3.2%
990223	3.0%	3.1%	2.2%
990303	2.5%	2.6%	5.3%
990308	3.1%	2.9%	2.7%
990309	7.8%	9.4%	8.9%
990414	4.8%	2.3%	5.4%
Overall	**3.4%**	3.6%	4.2%

Table 3.2: Classification error rates using SVM based classifiers.

Prior to further experimentations, we built several off-line models. The skin color model and the human face model profiles are trained based on 30 face keyframes of a set of different people. These are generic models and not specific to any particular person. In order to compare our approach with conventional audio based anchor detection, we also built, off-line, the acoustic speaker model for our target anchor as well as the acoustic model for background audio. To train these models, we labeled a data set containing 20 minute clean speech from Tom Brokaw and 50 minute non-target audio data, including speech, environmental sound, and music.

Table 3.3 provides the detailed results on face detection on the seven testing programs. The second column of the table gives the total number of keyframes for each program. Considering the length of each program (around 30 minutes), the average duration of a keyframe is about 3 seconds, although the actual duration may vary greatly. The duration of a keyframe from commercials may be as short as one half of a second and that of an anchor keyframe can be as long as one half of a minute. The third column of Table 3.3 is the ground truth, the real number of keyframes where the anchor is present within each program. The number of detected face images is listed in the fourth column. The fifth column gives the number of anchor faces among all detected faces (also identified manually). The last column is the visual-based anchor detection result given as the number of faces in the anchor cluster.

Test Sequence	Keyframe	Anchor Keyframe	Detected Face	Detected Anchor	Anchor Cluster Size
990218	587	14	39	10	9
990219	551	11	29	9	9
990223	555	16	38	12	12
990303	545	12	42	9	9
990308	572	11	37	9	8
990309	583	12	41	10	9
990414	552	17	31	12	11
Total	3945	93	257	71	67

Table 3.3: Face detection results

There are two types of detection error: false rejection and false acceptance. It is usually true that reducing one error rate will increase the other. Since the main purpose of visual based anchor detection is to exploit the visual cues to locate on-line audio training data of the target speaker to build an adaptive acoustic model, it is obviously necessary for us to minimize the false acceptance rate to ensure the quality of the collected training data.

During the experiments, face detection is followed by feature block localization and invariant feature extraction. A matrix of dissimilarity vectors are formed for clustering purpose. In color histogram based feature extraction, a 3D histogram is built with the resolution of $16 \times 16 \times 16$. Because feature d_h and d_m have different dynamic ranges, we set the weights w_h and w_m to be 1.0 and 0.2 so that both measures fall into the similar range. After the clustering, the largest cluster is classified as the anchor class. In our experiments, we set up the thresholds so that the false alarm rate can be kept minimum during both face detection and anchor detection. Computed from the results in Table 3.3, the statistics yielded are: detection accuracy - 72%; false rejection rate - 28%, and false acceptance rate - 0%. Examining the falsely rejected anchor frames, it was found that they fall into mostly two categories: poor quality of anchor facial color (due to fade in/out, they are missed during face detection) and side views of the anchor (when the rotation is severe, the corresponding feature block does not possess the similar visual features as the ones from frontal views). In simulation, we also experimented with using histogram or motion based measure only for clustering. The performance is not as satisfactory which indicates that the combined feature vector is more effective. Some of our experimental results for testing sequence 990218 are visualized in Figure 3.15, where the upper part gives a set of detected faces and their corresponding feature blocks and the lower part shows the final cluster for the anchor.

When theme music is present and can be detected, an on-line visual model based approach can be used. In our experiments, all test data contains distinct NBC Nightly News theme music and all such segments in our testing data are accurately detected. Using them as cues, an anchor keyframe can be precisely identified and used as the on-line visual model for the anchor. However, depending on the shot

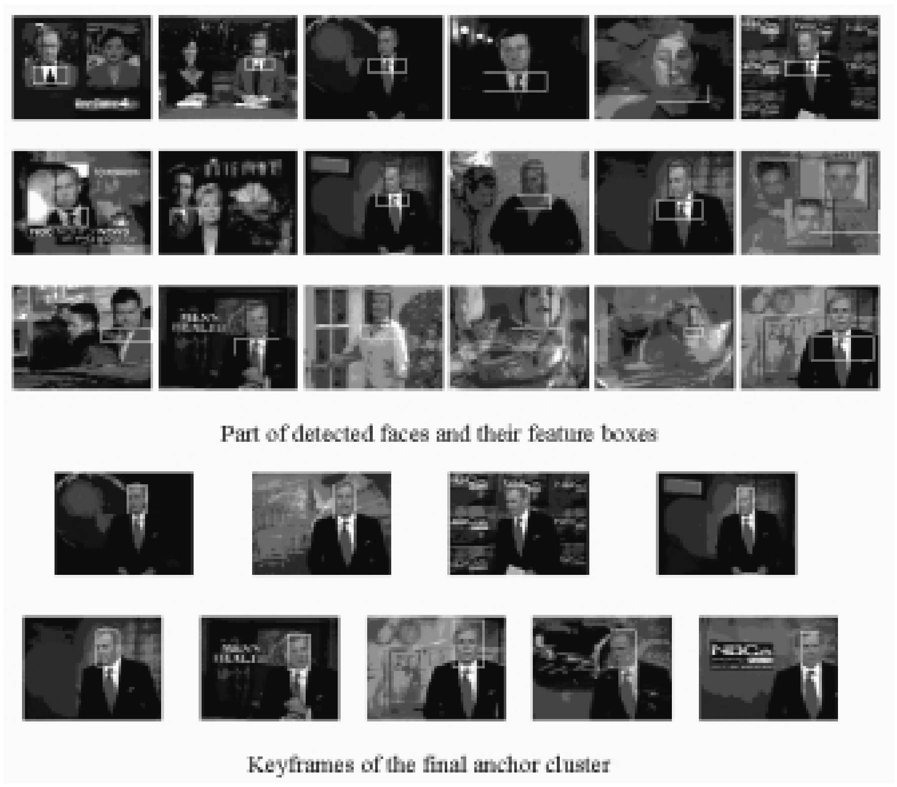

Part of detected faces and their feature boxes
Keyframes of the final anchor cluster

Figure 3.15: Results of anchor keyframe detection.

cut algorithm, the quality of the first anchor frame extracted this way varies because a shot cut algorithm may sometimes cut in the middle of the fade in/out, yielding a keyframe with poor visual quality. In this case, the on-line visual model based anchor detection may fail. Among seven testing programs, two failed using this approach. For the other five testing data, it yielded comparable anchor detection results as clustering method, with yet much less computation (no need to compute the dissimilarity matrix).

In our experiments, on an average, around 70% of the anchor speech data can be successfully collected on-line with the help of the visual cues (visual based anchor detection). This is more than adequate amount of data needed to train the on-line acoustic model for the anchor. For each testing program, a speaker model is built and applied back to the audio stream to extract all the segments where the anchor speech is present. Currently, we measure the performance at segment level. Figure 3.16 illustrates the relation of detected anchor segments and ground truth, where detected segments are denoted by solid horizontal lines, and the target speaker segments are denoted by pairs of dashed vertical lines. Four measures are used: Segment Hit Rate (SHR), Segment False-alarm Rate (SFR), Difference of segment starting time ($Diff_{st}$), and Difference of segment ending time ($Diff_{end}$). $Diff_{st}$ is defined as the difference of starting time of detected anchor segment and that of the corresponding real anchor segment. $Diff_{end}$ is defined in a similar way.

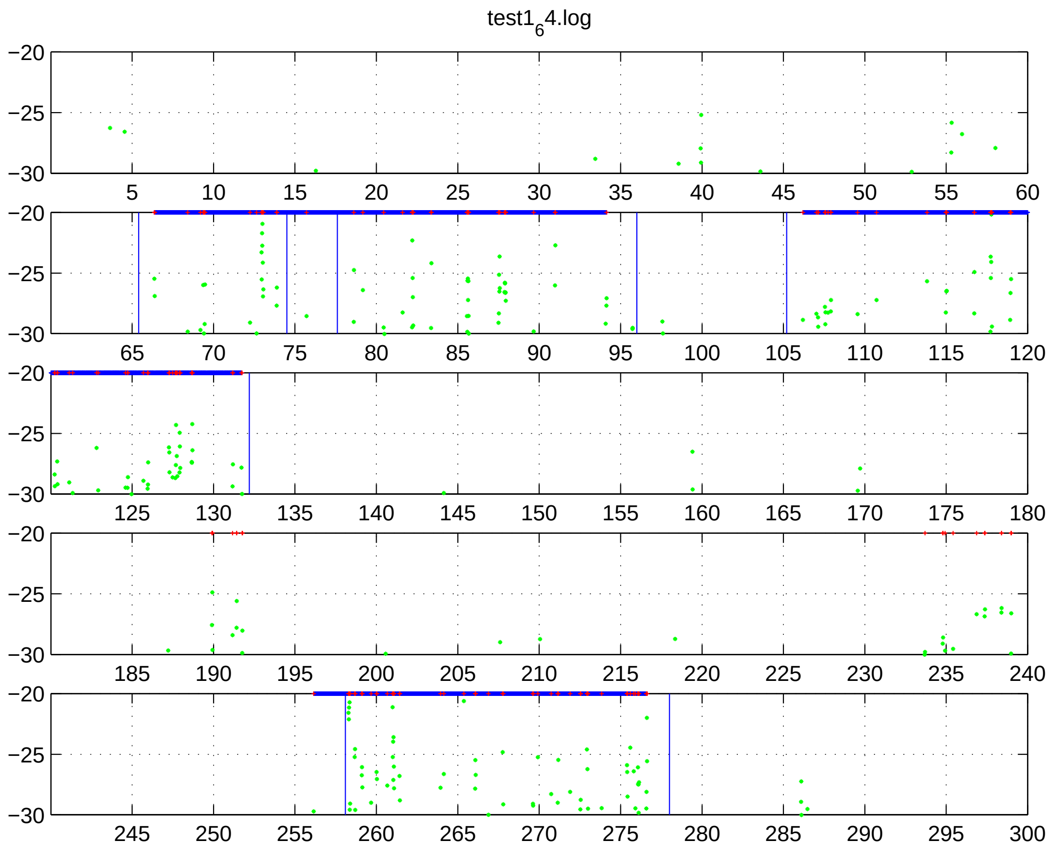


Figure 3.16: Normalized score in a portion of a test broadcast showing actual segment boundaries of the target speaker (dashed vertical lines) and estimated target segment boundaries (solid horizontal lines).

In audio based anchor detection, we set up to compare the performance of both off-line and on-line model based detection results. Tables 3.4 and 3.5 show the experimental results using each method. In both tables, the second column gives the real anchor segments manually labeled. The third and forth columns give the number of hit segment and false detected segment. The SHR of off-line approach is 95.4% while on-line approach gives 90.8%. For SFR, on-line approach is 2.3%, better than off-line - 8.0%. The fifth and sixth columns give the mean and standard deviation of $Diff_{st}$. Those of $Diff_{end}$ are shown on the last two columns. Overall, the experimental results from both approaches showed similar performance, with obviously the on-line method having the full flexibility of detecting arbitrary anchors while the off-line approach can not.

Testing Sequence	True Segment	Hit	False	$Diff_{st}$ Mean	$Diff_{st}$ STD	$Diff_{end}$ Mean	$Diff_{end}$ STD
990218	12	12	0	632	263	503	2154
990219	13	13	1	798	1718	-536	722
990223	12	10	0	1567	1487	-1148	1313
990303	12	11	1	2137	2547	-663	641
990308	12	11	2	651	1300	-469	1354
990309	12	12	1	661	2396	-778	4264
990414	14	14	2	174	1233	121	1988
Total/AVerage	87	83	7	946	1563	-424	1777

Table 3.4: Anchor person detection using off-line speaker model (Unit of $Diff_*$ is msec)

The text analysis generates four classes: stories, augmented stories, story introduction, and the news summary of the day. Story segmentation results are shown in Table 3.6. The second column (N_g) is the known number of stories (ground truth) in a program, the third column (N_s) is the number of segmented stories, the fourth column (N_{off}) is the number of segmented stories whose boundaries differ from the ground truth, and the last column (W_{off}) is the average number of words contained in the shifted boundaries. We examine the quality of story segmentation from two

Testing Sequence	True Segment	Hit	False	$Diff_{st}$ Mean	$Diff_{st}$ STD	$Diff_{end}$ Mean	$Diff_{end}$ STD
990218	12	12	1	632	263	1503	2538
990219	13	12	0	1069	674	-1116	1288
990223	12	11	0	729	183	-1236	1368
990303	12	10	0	1094	1028	-410	1699
990308	12	10	0	1692	1763	-1407	1106
990309	12	11	0	669	278	-658	1014
990414	14	13	1	479	848	-455	4167
Total/AVerage	87	79	2	909	720	-540	1883

Table 3.5: Anchor person detection using on-line speaker model (Unit of $Diff_*$ is msec)

aspects. One is the segmentation itself. That is, how many stories are indeed correctly extracted. Another aspect is the precision of the story boundary. As we can see from the table, the only missegmentation occurred for 990218 where two adjacent blocks of text are mistakenly merged due to the high similarity score. By examining the content, it was found that the two merged adjacent headline stories are both about murder cases so that many words used in both stories are the same (e.g., justice, killing, murder, police, testified, trial, lawyers, etc.). The high rate of word overlap leads to high similarity between the two stories. On the precision of the segmented stories, even though 62% of the stories do not have the exact boundary as the ground truth, the average number of *words* deviated from the ground truth boundary is as low as 5.7, excluding the result from 990309. This precision is very high compared with reported results in the literature. Since result from 990309 has unusually poor precision (in one of the five segmented stories), we analyze it separately. For 990309, there are only two stories (out of five) having their boundaries not exactly aligned with the true boundary. One shifted by 5 words (which is normal) and the other shifted by 32 words. The reason for this severe shift is due to the fact that the text in closed caption for the beginning of that story is totally different from what is being said in the audio track.

One common contributor to all the boundary shifts is due to the quality of text-speech alignment. We found that most of the words shifted around boundary are within the finishing sentence of reporters. For example, in the end of reporting a story, the correspondence often concludes the story by, say, "Robert Hager, NBC News, New York". Almost all the boundary shifts occurred in this particular type of sentence.

Test data	N_g	N_s	N_{off}	W_{off}
990218	6	5	4	4.75
990219	5	5	4	3.5
990223	5	5	3	4.3
990303	5	5	3	11.3
990308	6	6	4	5.25
990309	5	5	2	(32+5)/2
990414	6	6	3	5.3

Table 3.6: Performance of story segmentation based on integrated audio and text processing.

3.6 Summary

This chapter proposes an integrated approach to automatically generate indices for news broadcast at conceptual level. The experimental results show that the integrated anchor person detection algorithm achieves similar performance as audio based approach, but with much more flexibility. Text and audio based story extraction algorithm produces more accurate story boundaries. The embedded hierarchical structure of news program is successfully recovered, and the automatically generated table of content allows users to browse through large amounts of multimedia data with convenience and efficiency.

Chapter 4

Query-by-example in Audio

This chapter addresses the issues in (1) automatic index generation based on acoustic content in an unsupervised way and (2) efficient search and retrieval based on given audio query examples [72, 73]. Audio content analysis for information indexing and retrieval is a relatively new field that has attracted more attention in recent years. In [21], Wold et al. proposed to classify the audio into more than 10 different audio content. Although such supervised classification can be useful when there are a fixed number of known categories, it is not adequate to index general audio content, where content category is not a prior defined. How to efficiently query the audio content is also a very important issue, especially for database of middle and large size.

4.1 Overview of Audio Based Query

For unconstrained browsing and query, it is often impossible to plan ahead in terms of what can or can not be retrieved. For example, a user may want to find the audio clips that sound similarly to an audio sample in hand. In this case, to retrieve the desired clips, the audio data in the database has to be segmented (but not necessarily labeled), each segment has to be matched against the sound of interest, then the clips that are similar to the given example can be returned to the user. Here, the intention is to find similar clips but not necessarily to understand what the clips are. In an unsupervised manner, we segment an audio stream into homogeneous audio events, which provide a set of primitives so that higher level of grouping and clustering can be further performed. Another advantage is that this allows a free form browsing and query-by-example.

The goal of our work is to provide users the means of organizing the audio stream, to derive some useful structure of the data, and then to allow users to quickly browse or query about the content they need. Figure 4.1 illustrates the block diagram of the audio query system. Our strategy is that we first extract a set of audio events via unsupervised segmentation, and then use a Gaussian Mixture Model (GMM) to represent each homogeneous segment. To meet the query needs, we further propose a parametric distance metric so that the acoustic similarity between different audio events can be measured. To recover the content structure, an unsupervised clustering method is applied to group the segments that possess similar acoustic properties into the same cluster. A query system is then built to (1) present the content structure, (2) provide a search/browsing interface, and (3) enable query by audio example.

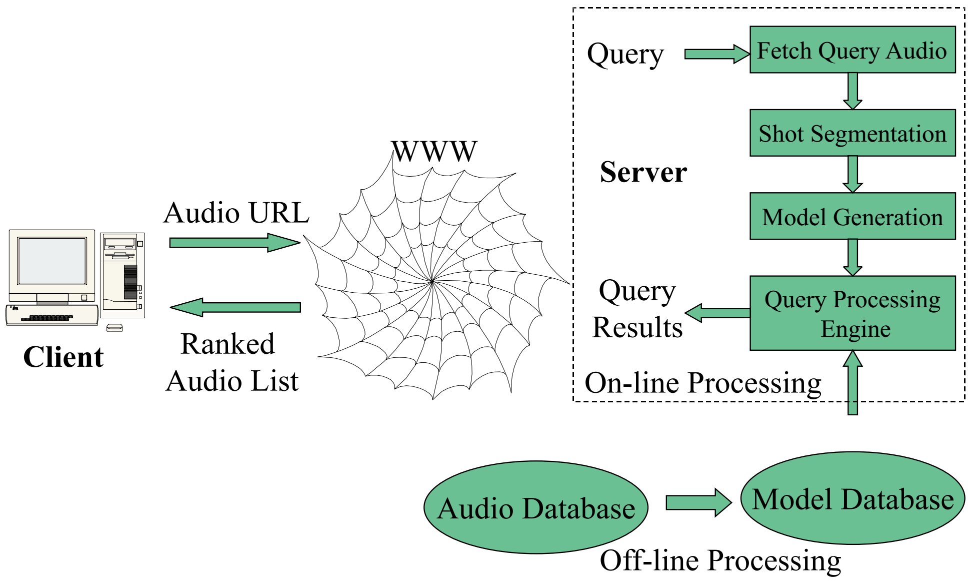


Figure 4.1: Illustration of audio query system.

4.2 Audio Event Segmentation and Modeling

Segmentation is the first step to explore the content structure of an audio stream. Parallel to the shot cut in visual domain, the objective is to identify the boundaries of changes in terms of some acoustic properties. A succinct and accurate representation - modeling of each segment is also indispensable for efficient audio storage and query.

4.2.1 Audio Event Segmentation

The audio event segmentation algorithm consists of three steps: feature extraction, splitting, and merging. We employ 13 order Mel-frequency cepstral coefficient (MFCC) features for each frame. During splitting, we identify possible scene change boundaries. During merging, neighboring scenes are merged if their contents are similar.

MFCC is widely used in speech domain and it provides a smoothed version of spectral that considers the non-linear human hearing property. The degree of smoothness depends on the order of MFCC being employeed. Two properties of MFCC, one being that the first coefficient is proportional to the audio energy and the other being that there is no correlation among different coefficients, make MFCC attractive.

In the second step, low energy frames, which are local minimum points of the volume contour, are located as boundary candidates. Figure 4.2 shows the volume contour of an audio file, where all low energy frames are indicated by circles. For each boundary candidates, the difference (see definition below) between its neighbors (both left and right) is computed. The definition of neighbors is illustrated in Figure 4.2, where for frame X, two dotted rectangular windows $W1$ and $W2$ are the neighbors of X and each with length of L seconds. If the difference is higher than certain threshold and it is the maximum in surrounding range, we declare that the corresponding frame is a scene boundary.

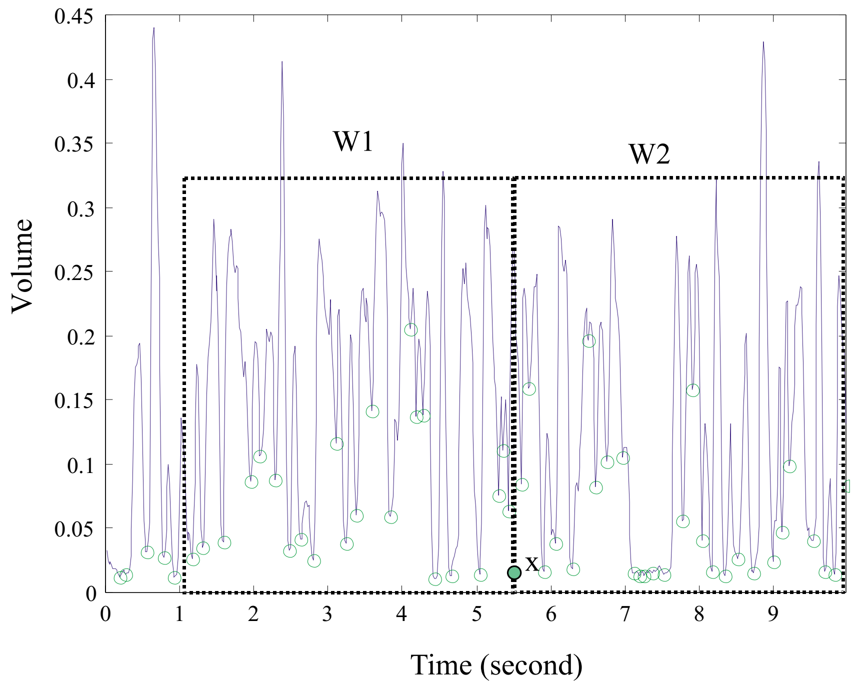


Figure 4.2: Illustration of audio segmentation.

Exteneded Kullback Leibler distance (KLD) [74] is adopted to measure the difference. For two 1-dimension Gaussian's $G(m_1, \sigma_1)$ and $F(m_2, \sigma_2)$, the extended KLD between G and F can be directly computed from the model parameters

$$D_P(G, F) = \frac{\sigma_1^2}{\sigma_2^2} + \frac{\sigma_2^2}{\sigma_1^2} - 2 + \left(\frac{\sigma_1^2 + \sigma_2^2}{\sigma_1^2 \sigma_2^2}\right)(m_1 - m_2)^2 \tag{4.1}$$

Since we assume that different audio features are independent, the overall distance is simply the summation of extended KLD of each feature.

The above described splitting process yields, in general, over-segmentation. A merging step is necessary to group similar neighboring segments together to form homogeneous audio events. This is done by comparing the statistical properties of adjacent segments. The extended KLD of adjacent segments is computed. If it is lower than a threshold, the two segments are grouped and the corresponding feature vectors are updated.

After the segmentation, each boundary point is located at the transition from one homogeneous audio event to another. While this may describe the primitive structure of the audio content, more can be done to identify its higher level structure. For example, for an audio stream containing a dialog, the speech of the same speaker distributed at different time instances can be grouped as one cluster. We achieve this using an agglomerative hierarchical clustering algorithm [68].

4.2.2 Audio Event Modeling

Using all the feature data corresponding to a speaker segment to represent the speaker is the most accurate way, since that is all information we can collect. Unfortunately, because of the huge storage space required and high computation load involved, we can not afford to do that when the number of speakers and the length of speaker segments are large. An alternative way is to approximate the feature distribution by certain models, and then use corresponding parameters to determine the difference of speakers. In this way, we may sacrifice accuracy, but the succinct representation will save much computation and storage. We choose GMM in this task due to its capability to approximate any distribution within required accuracy. For an audio segment, a model is generated by fitting the model with the features of that segment. The derived model parameters serve as the representation of the segment. We may choose the number of mixtures as the tradeoff between computation load and model accuracy. In certain situations, prior knowledge of data characteristics will also help to make the choice.

One immediate question is how to measure the dissimilarity between two segments based on their model parameters, which is very important in both audio segment clustering and audio query. There is no existing metric in the literature that can effectively measure the dissimilarity between two mixture type probability density functions (PDFs), like GMMs. In the next section, we propose a new distance metric, in its closed form solution, that measures the distance between two PDFs of mixture type, directly from their parameters.

4.3 Model Difference Measurement

The distance between each pair of audio events is defined as the distance of corresponding GMMs. Generally, the distance is required to satisfy three properties: non-negativeness, symmetry, and triangular inequality [75]. Let $G(x)$, $F(x)$, and $H(x)$ be three PDFs. Denote $D(G, F)$ as the distance between $G(x)$ and $F(x)$, the three properties can be formalized as,

$$D(G,F) \geq 0, \quad \text{and} \quad D(G,F) = 0 \quad \text{iff.} G = F \tag{4.2}$$

$$D(G,F) = D(F,G) \tag{4.3}$$

$$D(G,H) + D(H,F) \geq D(G,F) \tag{4.4}$$

4.3.1 Need for New Metric for Model Distance

There are several approaches available to measure the difference between two PDFs, which may or may not satisfy the three properties of distance measure. One approach defines the distance in $\mathbf{L}^r$ space by

$$D_{L^r}(G,F) = \left(\int_{x \in \mathbf{X}} |G(x) - F(x)|^r dx \right)^{1/r}, \tag{4.5}$$

where commonly used values of r may be 1 or 2. Although satisfying all three distance properties, D_{L^r} is usually computed by numerical method and the complexity can easily go beyond control with the increased dimension.

Another approach is the relative entropy or Kullback Leibler distance (KLD).

It is defined as [74],

$$D_{KL}(G,F) = \int_{x \in \mathbf{X}} G(x) \log \frac{G(x)}{F(x)} dx \tag{4.6}$$

It is obvious that KLD satisfies only the first property. By extending the original KLD to $D_{KL}(G,F) + D_{KL}(F,G)$, the second condition can be met. Although the third property still does not hold, the extended KLD is popular in many applications due to the lack of other alternatives. There are practical ways to approximate $D_{KL}(G,F)$. For example, data sequences T_G and T_F can be generated from models G and F and then the average log-likelihood ratio of the sequences with respect to $G(x)$ and $F(x)$ can be used to approximate the extended KLD. That is,

$$D_{Seq}(G,F) = \frac{1}{N}(|\log \frac{p(T_G|G)}{p(T_G|F)}| + |\log \frac{p(T_F|F)}{p(T_F|G)}|), \tag{4.7}$$

where N is the length of the data sequences T_G and T_F. The performance of D_{Seq} is a function of both the value of N as well as the data generation procedure. The bigger the N is, the more reliable is the approximation. At the same time, it makes the estimation more expensive.

Since GMMs are characterized by their component model parameters, the most desirable solution is to compute the distance directly from their respective parameters. Ideally, it is hoped that such a method can achieve at least comparable performance with a precise closed form solution which consequently can lead to a more efficient computational procedure. The existing method in this category can handle only simplified cases. For example, the extended KLD between two Gaussian's in one dimension can be simply computed by Formula 4.1. While the computation of D_P is simple and it can be extended to handle higher dimension Gaussian, it can not handle multiple mixture Gaussians. Even with the possibility of simplifying the models so that (4.1) can be applied, the outcome often indicates that it is not effective. This can be illustrated in a simple example. Consider two GMMs $G = 1/3 \times N(-2,1) + 2/3 \times N(1,1)$ and $F = 1/3 \times N(2,1) + 2/3 \times N(-1,1)$, where $N(m,\sigma)$ is Gaussian distribution with mean m and standard deviation σ. Obviously, both G and F have two components that are distributed very differently. Hence, the distance between them is clearly not zero. To apply (4.1), both G and F have to be simplified into one mixture Gaussian, denoted by $G'(m_G, \sigma_G)$ and $F'(m_F, \sigma_F)$, where the new mean and standard deviation can be derived as the weighted average of mean and standard deviation from their components. This yields the same mean ($m_G = m_F = 0$) and standard deviation ($\sigma_G = \sigma_F$) for both G' and F' which leads to $D_P(G', F') = 0$. Evidently, the measure derived using extended KLD from the simplified model fails to capture the obvious difference between the two original PDFs.

Therefore, there is a need to develop other alternatives that can effectively measure the difference between GMMs directly from their model parameters. In the following section, we proposes such an alternative.

4.3.2 Proposed New Metric

Suppose $G(x)$ and $H(x)$ are two PDFs of mixture type,

$$G(x) = \sum_{i=1}^{N} \mu_i g_i(x), \quad H(x) = \sum_{k=1}^{K} \gamma_K h_k(x), \tag{4.8}$$

where $G(x)$ is a mixture of N element PDFs $g_i(x)$, $H(x)$ is a mixture of K element PDFs $h_k(x)$, and μ_i and γ_k are corresponding weights that satisfy $\sum_{i=1}^{N} \mu_i = 1$ and $\sum_{k=1}^{K} \gamma_k = 1$. For simplicity, in the rest of this chapter, we will not use x explicitly in other formulae. Denote the distance between any two element PDFs g_i and h_k by $d(g_i, h_k)$, the overall distance between G and H is defined as

$$D_M(G, H) = \min_{\mathbf{w}=[w_{ik}]} \sum_{i=1}^{N} \sum_{k=1}^{K} w_{ik} d(g_i, h_k), \quad s.t. \tag{4.9}$$

$$w_{ik} \geq 0, \quad 1 \leq i \leq N, 1 \leq k \leq K \tag{4.10}$$

$$\sum_{i=1}^{N} w_{ik} = \gamma_k, 1 \leq k \leq K, \quad \sum_{k=1}^{K} w_{ik} = \mu_i, 1 \leq i \leq N \tag{4.11}$$

According to the definition, any component g_i in one mixture can interact with any other component h_k in the other mixture via weighted element distance $w_{ik}d(g_i, h_k)$. The degree of interaction is inversely proportional to the element distance and proportional to the mixture weights μ_i and γ_k. The weights w_{ik} are ultimately determined through optimizing with respect to the given constraints in (4.10, 4.11). The proposed framework can be visualized in Figure 4.3.

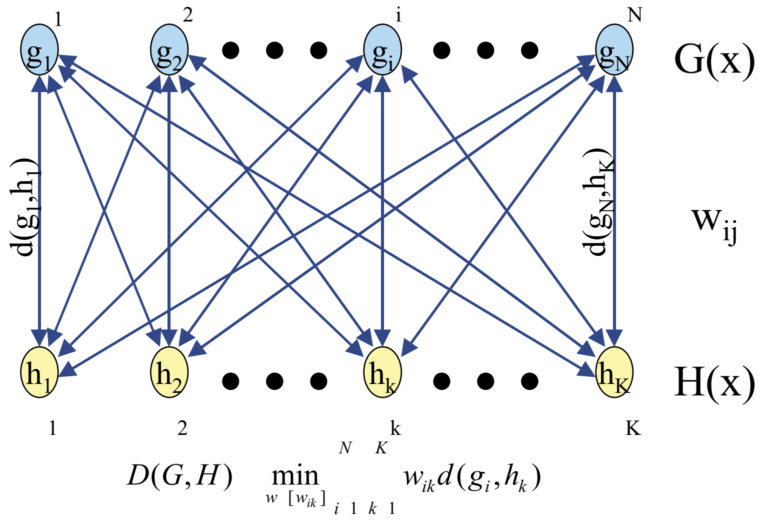


Figure 4.3: Distance between two mixture type PDFs.

Clearly, the solution is posed as a linear programming problem. There are many algorithms available to solve it efficiently, such as simplex tableau method [76]. We have a total of $N \times K$ free parameters (w_{ik}'s) and $N+K$ equality constrains, where only $N+K-1$ of them are independent. By the optimization theory, at most $N+K-1$ of the $N \times K$ parameters will not vanish. The above problem has solution because (1) we can easily find a feasible vector that satisfy all the constrains: $w_{ik} = \mu_i \times \gamma_k$ and (2) the upper bound for the objective function exists: $MD = \max_{ik} d(g_i, h_k)$, which is proved as follows,

$$D_M(G,H) = \sum_{i=1}^{N}\sum_{k=1}^{K} w_{ik}d(g_i,h_k) \leq MD\sum_{i=1}^{N}\sum_{k=1}^{K} w_{ik} = MD\sum_{i=1}^{N}\mu_i = MD \qquad (4.12)$$

4.3.3 Property of the New Metric

The proposed metric is defined as a general framework, constructed based on element distances. Its generality is due to the fact that the element distance measure is left unspecified. Depending on different application needs, appropriate element distance measures, which may even be non-parametric, can be plugged in and the overall distance between two mixture PDFs can be computed using the same framework. Furthermore, there is no requirement about the specific type of element distribution or that each element PDF should be the same type. If the element distance satisfies the three general distance property, the overall distance also does. The proof of the first two properties is straightforward. We here focus on the proof of the third property. For any three mixture PDFs, G, H, and F, we need to show that,

$$D_M(G,H) + D_M(H,F) \geq D_M(G,F) \tag{4.13}$$

The definitions of G and H are the same as (4.8). F is similarly defined as $F = \sum_{j=1}^{M} \xi_j f_j$, satisfying $\sum_{j=1}^{M} \xi_j = 1$.

Applying the definition in (4.9) to both pairs (G,H) and (H,F), we have their distances as,

$$D_M(G,H) = \sum_{i=1}^{N}\sum_{k=1}^{K} w_{ik} d(g_i,h_k) \tag{4.14}$$

$$D_M(H,F) = \sum_{k=1}^{K}\sum_{j=1}^{M} v_{kj} d(h_k,f_j) \tag{4.15}$$

where w_{ik} and v_{kj} satisfy $\sum_{i=1}^{N} w_{ik} = \sum_{j=1}^{M} v_{kj} = \gamma_k$, $\sum_{k=1}^{K} w_{ik} = \mu_i$, and $\sum_{k=1}^{K} v_{kj} = \xi_j$. Then

$$\begin{aligned} & D_M(G,H) + D_M(H,F) \\ = \; & \sum_{i=1}^{N}\sum_{k=1}^{K} w_{ik} d(g_i,h_k) + \sum_{k=1}^{K}\sum_{j=1}^{M} v_{kj} d(h_k,f_j) \\ = \; & \sum_{k=1}^{K} [\sum_{i=1}^{N}\sum_{j=1}^{M} \frac{w_{ik}v_{kj}}{\gamma_K}(d(g_i,h_k) + d(h_k,f_j))] \end{aligned}$$

$$\begin{aligned} &\geq \sum_{k=1}^{K}[\sum_{i=1}^{N}\sum_{j=1}^{M}\frac{w_{ik}v_{kj}}{\gamma_k}d(g_i,f_j)] \\ &= \sum_{i=1}^{N}\sum_{j=1}^{M}(\sum_{k=1}^{K}\frac{w_{ik}v_{kj}}{\gamma_k})d(g_i,f_j) \end{aligned} \tag{4.16}$$

Let $\alpha_{ij} = \sum_{k=1}^{K}\frac{w_{ik}v_{kj}}{\gamma_k}$ then (4.16) can be rewritten as,

$$D_M(G,H) + D_M(H,F) \geq \sum_{i=1}^{N}\sum_{j=1}^{M}\alpha_{ij}d(g_i,f_j) \tag{4.17}$$

On the other hand, for any set α_{ij} that satisfies the equation constraints in (4.11), the following inequality is also true since $D_M(G,F)$ is the outcome of optimization,

$$D_M(G,F) \leq \sum_{i=1}^{N}\sum_{j=1}^{M}\alpha_{ij}d(g_i,f_j) \tag{4.18}$$

Actually the variables α_{ij} indeed satisfy the required constrains.

$$\sum_{i=1}^{N}\alpha_{ij} = \sum_{i=1}^{N}\sum_{k=1}^{K}\frac{w_{ik}v_{kj}}{\gamma_k} = \sum_{k=1}^{K}v_{jk} = \xi_j \tag{4.19}$$

Similarly we have $\sum_{j=1}^{M}\alpha_{ij} = \mu_i$. Putting (4.17) and (4.18) together, we proved (4.13).

4.3.4 Performance of the New Metric

While we have proved that the new metric proposed possesses certain properties, we also like to demonstrate that it has similar behavior as other existing measures in experimentation. In this section, we compare it with the two previously defined measures: D_{L^2} and D_{Seq} based on synthetic data. For simplicity, we perform the comparison on 2 dimensional GMMs F and G, each with two mixtures. The element distance used is KLD defined in (4.1). Specifically, F is

$$F = 0.5N\left(\begin{bmatrix}1\\0\end{bmatrix},\begin{bmatrix}1\\1\end{bmatrix}\right) + 0.5N\left(\begin{bmatrix}-1\\0\end{bmatrix},\begin{bmatrix}1\\1\end{bmatrix}\right) \tag{4.20}$$

where $N(\mu, \delta)$ is a 2-D gaussian with mean vector μ and diagonal covariance δ. The comparison is conducted in four settings, in each of which, by perturbing the model parameters in G we observe how the three different measures (D_{L^2}, D_{Seq}, and D_M) react to the changes.

In setting one, G has exactly the same component Gaussians as F with yet variable mixture weights,

$$G = \mu N\left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) + (1-\mu)N\left(\begin{bmatrix} -1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) \tag{4.21}$$

where μ varies between 0 and 0.5.

In setting two, the two component Gaussians of G have the same weights and covariances as those of F but with variable mean vectors, changed along a circle of radius one.

$$G = 0.5N\left(\begin{bmatrix} \cos\alpha \\ \sin\alpha \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) + 0.5N\left(-\begin{bmatrix} cos\alpha \\ \sin\alpha \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) \tag{4.22}$$

where α is in the range 0 to π.

Setting three is similar to setting two except we vary the mean vectors of G symmetrically in the first dimension.

$$G = 0.5N\left(\begin{bmatrix} m \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) + 0.5N\left(-\begin{bmatrix} m \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) \tag{4.23}$$

where m is from 0.5 to 1.5.

In setting four, G has the same weights and mean vectors for both components but with the covariance changing along both dimension simultaneously.

$$G = 0.5N\left(\begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} \delta \\ \delta \end{bmatrix}\right) + 0.5N\left(\begin{bmatrix} -1 \\ 0 \end{bmatrix}, \begin{bmatrix} \delta \\ \delta \end{bmatrix}\right) \tag{4.24}$$

where δ ranges from 0.5 to 1.5.

Figure 4.4 shows the plotted behavior of the three measures under four different settings. All the curves are normalized so that the maximum distance is 1.

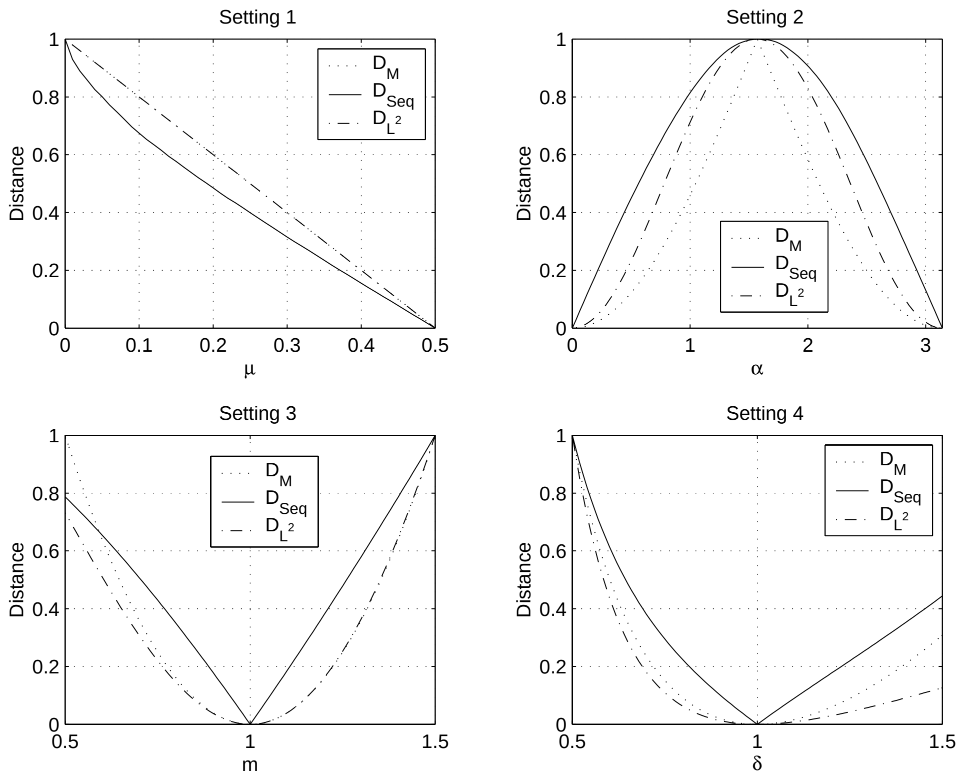


Figure 4.4: Behaviors of three different measures under four testing settings.

From these plots, one can see that the overall behaviors of all three are consistent in all settings. D_M curve overlaps with D_{L^2} in setting one and part of setting three. In setting four, D_M falls between D_{L^2} and D_{Seq}. These plots show that the proposed new metric behaves similarly in different scenarios as the existing measures that have been widely used in practice. But the proposed metric is obviously more efficient in terms of computation. In addition, with this metric, there is no need to store or to generate data points in order to compare the difference between two PDFs. This is significant, particularly in content based search and retrieval where large amounts of data is pre-indexed, stored, preferably, in a succinct parametric form, and searched in real-time. For example, to retrieve the speech segments of a particular speaker given as an query example, all the pre-stored speaker segments in a database have to be matched against the given query sample. In this case, having a measure that can compare the similarity directly from the speaker model parameters will be much more efficient than the ones that require to generate the data points from the models first and then compare, especially when the search range is large, a realistic scenario in almost all content based retrieval tasks.

4.4 Query Processing Engine

Traditional query is based on text information. With the fast development of multimedia applications, not only the demand has grown out of needs in text, but also the manual annotation is no longer feasible. Query based on acoustic characteristics is one alternative to text based retrieval. For example, to retrieve some audio clips based on text, one has to know exactly how this clip is labelled. But there are many cases where users know only how the content (speakers, music, or songs) sounds like but not what semantics it has been identified previously. Therefore, retrieval by audio example is an alternative to conventional text based retrieval. The user can simply provide a sample audio stream and ask to retrieve the audio segments that posess similar acoustic properties.

Technically, the sample audio is first segmented, and each segment is fitted by a GMM based on the procedure previously described. Normally, the sample audio is expected to be short and the entire example audio stream is one homogeneous audio event. In this case, sample GMM is compared with those saved in the database. When there are several audio events in the example, there are more alternatives to process the query. One is to choose the dominant segment (longest one) and then make the query based on its GMM. Another possibility is to use all the segments in query without considering their temporal order. The query results may be similar to any of the segments in the query example. If temporal order is considered, measures that can characterize both the duration and the order simultaneously are needed. Although dynamic programming (DP) is a good candidate for this kind of matching procedure, we currently do not use DP approach due to its expensive computation and our requirement to perform efficient on-line query against a large audio database. We instead adopt a simpler method.

Denote the query sequence Q by $(Q_1, Q_2, ..., Q_N)$ and an audio sequence S in database by $(S_1, S_2, ..., S_M)$, where Q has N segments and S has M segments. Without loss of generality, we assume that $M \geq N$. Q_i and S_i are GMMs, representing the corresponding segments. The distance between Q and a portion of S started at t is computed as,

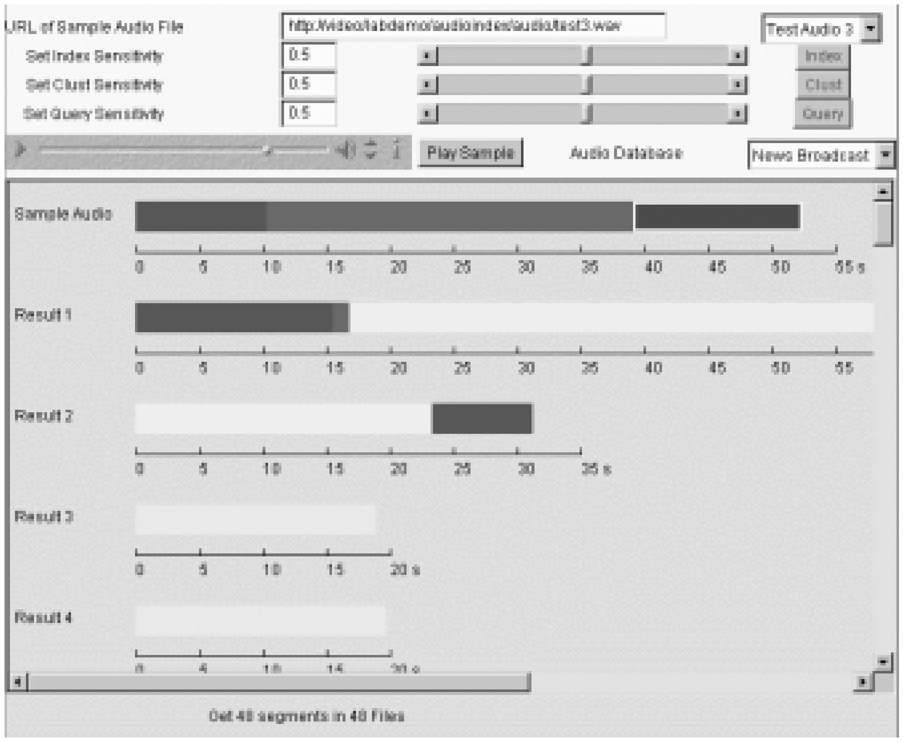
$$D_{Q,S}(t) = \sum_{i=1}^{N} D_M(Q_i, S_{t+i-1}) w(Q_i, S_{t+i-1}) \tag{4.25}$$

where $w(Q_i, S_{t+i-1})$ is a weight related to the duration of Q_i and S_{t+i-1}. If the duration difference is not considered, w can be assigned to 1. Otherwise, it should reflect the duration effect. Suppose the duration of segment Q_i is T_i^Q and that of S_j is T_j^S, one possible definition of w is,

$$w(Q_i, S_j) = \frac{T_i^Q}{\sum_{k=1}^{N} T_k^Q} \times (2 - \frac{min(T_i^Q, T_j^S)}{max(T_i^Q, T_j^S)}) \tag{4.26}$$

4.5 Query-by-Example in Audio System

To evaluate the proposed approach in audio indexing and retrieval, experiments are performed on an audio query system. The interface of the system is shown in Figure 4.5. Users can provide their audio sample by specifying the URL of the audio file. The user may trig three tasks: segmenting the audio sample, clustering the segments, and querying the database. The indexed audio segments are presented as blocks of different colors in Figure 4.5. When clustering is performed, segments within the same cluster will be painted with the same color. The query results are painted using a designated highlight yellow color where different hit segments will have different brightness, depending on the similarity scores (the more similar, the brighter the color is). When the query is performed across different audio streams, the order of the hit streams will be returned according to the degree of similarity measured as the minimum distance between the sample segment and any hit segment of an audio stream. As can be seen from the top of Figure 4.5, the interface offers users the means to adjust the sensitivity values in different tasks based on their application needs.



URL of Sample Audio File: http://video/labdemo/audioindex/audio/test3.wav

Test Audio 3

Set Index Sensitivity: 0.5

Set Clust Sensitivity: 0.5

Set Query Sensitivity: 0.5

Audio Database: News Broadcast

Get 48 segments in 48 Files

Figure 4.5: The interface of audio query system.

4.6 Simulation Results and Discussion

All the audio data in our database is in the format of mono raw data with 16 KHz sampling rate and 16 bit resolution per sample. To measure the segmentation performance we test on two TV news broadcast sequences, denoted by test1 and test2, each about 30 minutes. The ground truth of the segment boundaries are generated manually, where speaker changes, speech/music changes, and news/commercial changes are annotated. Overall, test1 has 157 segments and test2 has 159 segments. The length of window used in segmentation is 3 seconds. Four measurements are designed to evaluate the performance based on identified event boundaries: 1) Hit Rate (HR) - the ratio of the number of correctly detected boundary points (within two seconds deviation) to the true number of boundary points, 2) False Alarm Rate (FR) - the ratio of the number of falsely detected boundary points to the number of detected boundary points, 3) Mean Difference (MD) - the average difference (in ms) between correctly detected boundaries and the true boundaries, and 4) Standard Deviation of boundary difference (SD). Table 4.1 gives the results from the two test sequences. The high FR is due to the variation in the background sound.

Sequence	HR (%)	FR (%)	MD (ms)	SD (ms)
test1	92.26	20.56	-336	736
test2	93.63	22.22	-235	658

Table 4.1: Audio segmentation results.

A database containing 278 audio events is constructed, each is about 10 to 30 seconds long. Every event is an acoustically homogeneous segment such as a segment of speech from a particular speaker or a piece of music, whose parameters are stored in the database. During query, an audio segment is provided by users as the query example and the retrieval process is to find all the audio segments in the database that have the similar acoustic properties as the query example. For example, if the query sample is a piece of speech from president Clinton, the task is to find all Clinton speech segments from the database.

Using the given query example, a 4 mixture GMM is built and compared with all other GMMs stored in the database. Two categories of measures are used to perform the comparison of GMM models. One is the distance measure by sequence D_{Seq} and the other category is the proposed distance measure. Since the proposed distance measure uses element distance measure as building block, we choose, in this experiment, two types of element distance measures to show that the proposed framework has the flexibility of adapting to different application needs. One element distance measure is L_1 norm and the other is L_2 norm, both satisfy all three distance properties. Formally the distance between f and g can be written as,

$$d_{L_r}(f,g) = \left(\sum_{i=1}^{N} |\mu_i^f - \mu_i^g|^r + \sum_{i=1}^{N} |\sigma_i^f - \sigma_i^g|^r \right)^{1/r}, r = 1, 2 \tag{4.27}$$

where N is feature dimension, μ_i^f, μ_i^g, σ_i^f and σ_i^g are the the i^{th} means and standard deviations of f and g.

Even though the mean and standard deviation may have very different dynamic ranges, the choice is reasonable for this application because when one range is much larger than the other, the impact from the smaller one is negligible in the overall distance value. Plugging in the two chosen element distance measures, it yields two measures, denoted by D_{ME1} and D_{ME2}. Using each of the three measures, we compute distance between the given query example and each of the audio event in the database. When the distance is smaller than a threshold (can be set by user), the corresponding audio event is considered as a hit.

To evaluate the retrieval performance, we use Recall Rate (RR) and False detection Rate (FR). Specifically, they are defined as follows. Assume that there are T recorded events in database. Given a query example, there are Q events in the database that are the true match. If the retrieval process returns R events as query results, among which C events are the correct match, then RR is defined as C/Q, and FR is defined as $(R - C)/(T - Q)$. Similar to the Receiver Operating Characteristic (ROC) in classical detection theory [77], we can plot a FR-RR graph (similar to the PF-PD graph in detection theory) by varying the similarity threshold of query processing engine in Figure 4.6 to visualize the retrieval performance.

The query is for a particular speaker, the anchor of NBC Nightly News, Tom Brokaw. In the database, there are 55 segments that are Tom Brokaw's speech. We use each of them as a query example and compute the corresponding FR-RR graph. Figure 4.6 shows the average FR-RR graph of all the query performance. As it can be seen from the figure, D_{ME1} and D_{ME2} display similar performance as D_{Seq}. When $FR < 0.11$, D_{ME2} is slightly worse than D_{Seq}, and D_{ME2} is slightly better than D_{Seq} when $FR > 0.11$. While computing D_{Seq}, we choose the length of the testing sequence as 5000. For each query, the computation time of D_{Seq} is 25 times more than D_{M1} and D_{M2}. Taking into account the significant reduction in computation, the proposed new metric outperforms the existing ones.

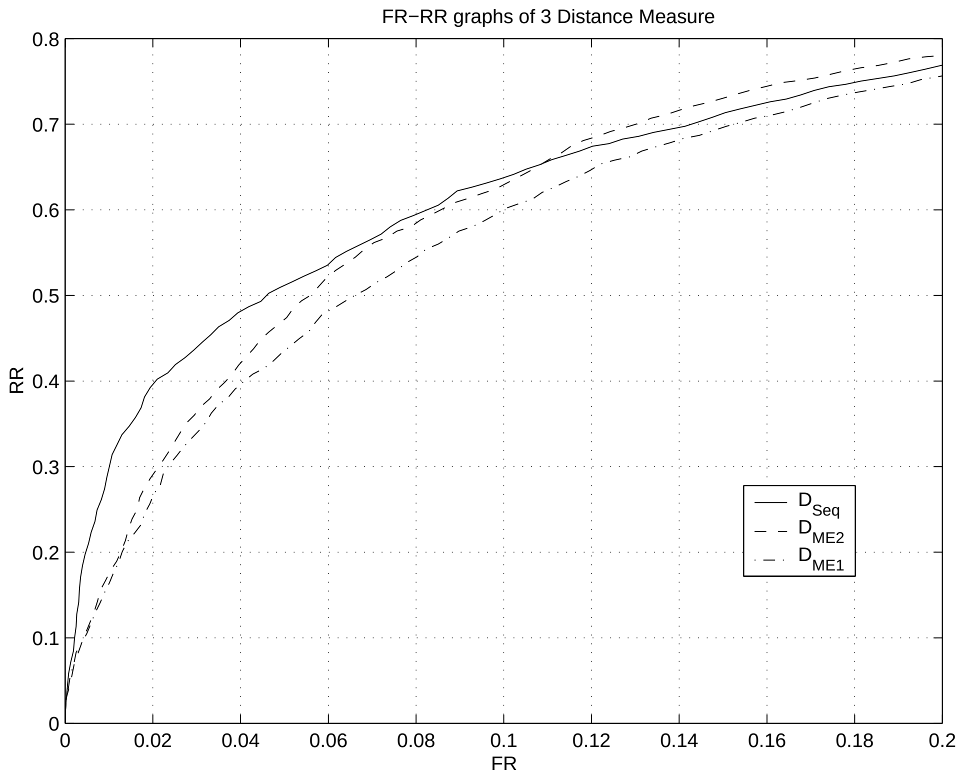


Figure 4.6: Performance comparison using FR-RR curves.

4.7 Summary

This chapter proposes a new approach for content based audio indexing and query. The audio signal is segmented into homogeneous events whose features are characterized using GMM models. A new metric for measuring the similarity between two PDFs of mixture type is described and applied to GMMs. Both audio clustering and on-line query are based on this new metric. An audio retrieval system is presented as a demonstration of the effectiveness of the proposed techniques. From the experimental results on both synthetic and real data sets, it can be seen that our proposed approach is promising for audio content indexing, description, search, and retrieval.

Chapter 5

Major Cast Detection

Major casts, for example the anchor persons or reporters in news programs and principal characters in movies play an important role, and their occurrences provide good indices for organizing video content. The users may easily digest the main scheme by skimming the list of major casts and sampling related video clips. This chapter presents an approach for automatically generating the list of major casts for video based on both audio and visual information.

5.1 The Diagram of Major Cast Detection Algorithm

Figure 5.1 illustrates the major cast detection algorithm we proposed. Each major cast is characterized by two attributes: face and speech. The detection procedure is to find corresponding face occurrences and speech segments by analyzing video in two levels. Audio and visual information is utilized separately in low level, and in high level cues from different modalities are combined [78].

In the first level, video sequence is segmented independently in audio and visual tracks. In audio track, clean speech chunks are first extracted using techniques similar to those mentioned in Chapter 2, within which speaker boundaries are identified. Video is segmented into homogeneous shots, and face detection and tracking

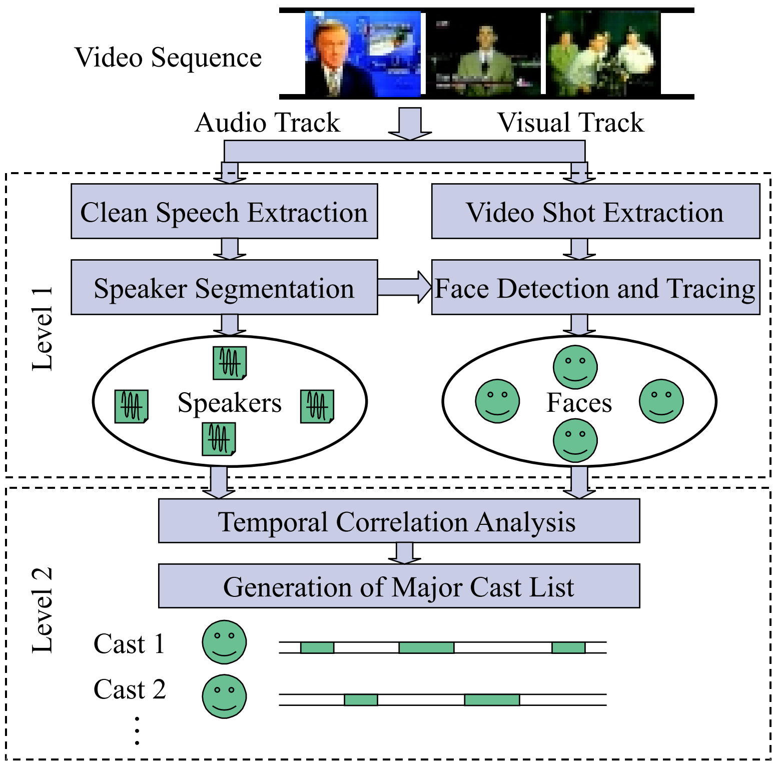


Figure 5.1: Major cast detection algorithm.

are applied within and among shots. After all speaker segments and face tracks are detected, they can be grouped by independent or integrated clustering method so that segments containing the same speaker and tracks consisting of the same face are merged. In the second level, we analyze the temporal correlation among different faces and speakers and link them to certain characters. A list of major casts is then constructed, and we present each cast's face and corresponding speech segments in an order that reflects the characters' importance. The importance score is determined based on the accumulative temporal and spatial presence of each cast.

5.2 Speaker Segmentation and Clustering

5.2.1 Speaker Segmentation

Besides speech signal, there are other kinds of sound in audio track, for example, music, speech with music, noise, speech with noise, and etc. To recognize and cluster different speakers, we want to extract speaker information based on clean speech only. Noisy speech may deteriorate the accuracy of speaker model and further reduce the speaker segmentation and clustering performance. Therefore the proposed speaker segmentation algorithm includes two steps: 1) Extract the clean speech chunks from the audio track. 2) Locate the speaker boundaries in clean speech audio chunk.

Clean speech can be extracted by techniques introduced in Chapter 2. Here we use the 14 clip-level audio features and compare three classification mechanisms: neural network, GMM, and SVM classifiers.

Speaker segmentation follows the procedure presented in Section 4.2. We modify the algorithm a little bit such that it is better targeted for segmenting speakers within clean speech chunk, but not general audio. When we compare the difference of two audio blocks, we use MFCCs and delta MFCCs, and only consider those frames for which pitch is detectable. These frames normally correspond to voice, which reflects the characteristics of speaker's vocal track. By this way, speaker boundaries are more reliably detected. Similar to Section 4.2.2, each speaker segment is fit by a GMM. Again, only those frames that have pitch values are used to train the GMM.

5.2.2 Speaker Clustering

Distance matrix of all speaker segments is computed by the proposed distance metric defined in Section 4.3.2. Note we use extended KLD as the element distance. Based on the distance matrix, we apply a clustering algorithm to group speaker segments into different speakers. Considering that the used distance does not satisfy the triangular inequality property of general distance, we employ a two step hierarchical clustering algorithm [68]. Initially in the first step, each segment is a cluster on its own. During each iteration, two clusters with the minimum dissimilarity value are merged, where the dissimilarity between two clusters is defined as the maximum dissimilarity among all possible pairs of segments, one from each cluster. This procedure continues until the minimum inter-cluster dissimilarity is larger than a preset threshold T_1. Then in the second step, we define the dissimilarity between two clusters as the minimum dissimilarity among all possible pairs of segments, one from each cluster. The two clusters with minimum dissimilarity merge in each iteration until the minimum dissimilarity is larger than another threshold T_2, where $T_2 < T_1$.

After clustering, speaker segments with similar audio property are grouped together. The thresholds $T1$ and $T2$ are intentionally set low to make sure few segments from different speakers are grouped together. Although it is possible that segments from the same speaker are scattered into different clusters, it is possible to merge them based on corresponding face similarity.

5.3 Face Detection and Tracking

Parallel to speaker segmentation and clustering, face information are also recovered from the visual track. Based on the proposed face detection algorithm, face tracks are extracted and clustered.

5.3.1 Face Detection in Still Image

This section describes the procedure of detecting faces in a still image of complex scene [79]. The basic procedure is first described, and applying the basic procedure in different locations we can find faces with certain sizes confined by the size of the face template. Then we show how to detect faces with different sizes by applying the basic procedure in multi-resolutions. Training procedure for average face template and consideration of performance improvement are also discussed.

Basic Procedure of Face Detection

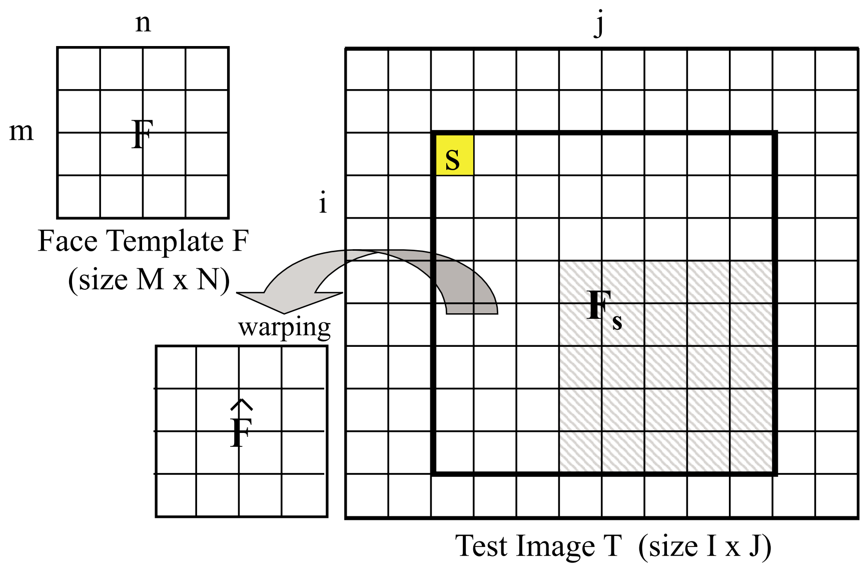


Figure 5.2: Illustration of template matching.

In Figure 5.2, F is the face template image of size $M \times N$, T is the test image of size $I \times J$, and each small block represents a pixel. The task is to find a region in the test image T that is best matched with the template by some warping functions that map the columns/rows in the region to those of the template. We make use of two constraints for the warping functions. The global constraint is that the height and width of the face in test image are no less than those of the face template, and no bigger than twice of the face template. For a given top-left pixel position, s, of a candidate region, the regions for which we need to examine are all rectangles that end at any pixel within the shaded area. The largest candidate region is illustrated by a bold rectangle in the figure. The local constraint is that one or two rows/columns in candidate region are mapped to each row/column of the face template. Let F_s

represent a current candidate region,with size $I' \times J'$, the row and column mapping functions f and g from F_s to F can be formalized as,

$$\begin{cases} f(i) = m, \quad i = 1, ..., I', m = 1, ..., M \\ g(j) = n, \quad j = 1, ..., J', n = 1, ..., N \end{cases}, \tag{5.1}$$

subject to

$$\begin{cases} f(1) = 1, f(I') = M \\ g(1) = 1, g(J') = N \\ f(i) + 1 \geq f(i+1) \geq f(i), \quad f(i+2) > f(i) \\ g(j) + 1 \geq g(j+1) \geq g(j), \quad g(j+2) > g(j) \end{cases} \tag{5.2}$$

The mapped image $\hat{F}$ can be computed by,

$$\hat{F}(m, n) = \underset{\{(i,j)|f(i)=m,\ g(j)=n\}}{AVERAGE} F_s(i, j) \tag{5.3}$$

where $m = 1, ..., M, n = 1, ..., N$.

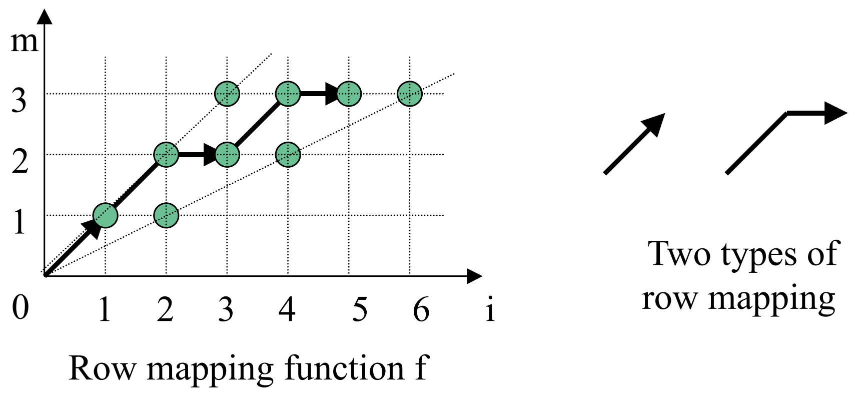


Figure 5.3: Row mapping function.

All possible row mapping f can be illustrated in a trellis shown in Figure 5.3. Two types of element mapping are shown on the right of the figure: 1) one row of face region to one row of template, and 2) two rows to one row. We use the intensity difference square between $\hat{F}$ and F as the matching error (ME) of F_s and F based on the mapping functions f and g.

$$ME_{f,g}(F_s, F) = \sum_{m=1}^{M}\sum_{n=1}^{N}(\hat{F}(m,n) - F(m,n))^2 \tag{5.4}$$

It is easy to see that to find out the minimum matching error between F_s and F, we need to search $2^M \times 2^N$ combination of different f and g. Brute force searching is not feasible even for small size of F. Here we want to utilize dynamic programming to solve this problem. Suppose $F_s^{i,j}$ and $F^{m,n}$ are the top left $i \times j$ and $m \times n$ part of F_s and F respectively. We define the partial error (PE) as the minimum matching error between $F_s^{i,j}$ and $F^{m,n}$,

$$PE(m,n,i,j) = \min_{f',g'} ME_{f',g'}(F_s^{i,j}, F^{m,n}), \tag{5.5}$$

and the best matching value (MV) between F_s and F is

$$MV(F_s, F) = \max_{i=M,\ldots,2M, j=N,\ldots,2N}(1 - \frac{PE(M,N,i,j)}{2\sigma_F^2 MN}), \tag{5.6}$$

where σ_F is the intensity standard deviation of face template. If $MV(F_s, F)$ is higher than a preset threshold, the corresponding portion of F_s is declared as face candidate.

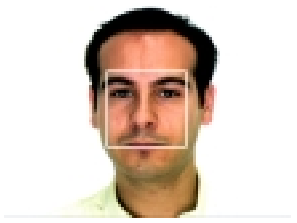
Inspired by the mechanism of dynamic programming, we hope to find the optimal path that start at (0, 0, 0, 0) and end at (M, N, I', J') in a 4-D trellis of PE(m, n, i, j). Based on the constraints of mapping functions f and g, we know PE(m, n, i, j) can be updated from limited number of ancestors, such as PE(m-1, n-1, i-1, j-1). If we can determine PE(m, n, i, j) based on its recent ancestors, and the delta matching error is independent of the trellis path of the corresponding ancestor, dynamic programming algorithm can be used to search the global optimal solution. Unfortunately, we find that these conditions are not met here because the delta matching errors between PE(m, n, i, j) and its ancestors also depend on their trellis paths since the mapping functions of the ancestors are inherited by PE(m, n, i, j) and are used in computing delta matching errors.

Knowing that 2-D dynamic programming is not feasible in our task, we propose an iterative 1-D dynamic programming procedure for row- and column-wise template matching. Suppose that we have the initial column mapping function between F_s and F as g^0. In the i^{th} ($i \geq 1$) iteration, we first set g^{i-1} as the column mapping function and use row-wise dynamic programming to find best f^i that minimizes $ME_{f^i,g^{i-1}}(F_s, F)$. Then we set the f^i constant, and use column-wise DP to find the best g^i that minimizes $ME_{f^i,g^i}(F_s, F)$. This two step DP is iterated until we find a convergent solution. The procedure will converge since in each step of each iteration we get a non-increasing matching error.

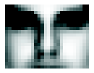
The initial column mapping function g^0 can be chosen randomly or systematically. The method we used is as follows. First, f^0 is set in three configurations so that: 1) each row in the test region maps to one row in the template, 2) alternatively every one row and two rows in the test region map to one row in the template, and 3) every two rows map to one row. Then, we apply column-wise DP to find the best g_0's accordingly. The g_0 that gives minimum matching error is chosen to be the original column mapping function.

Face Detection in Multiple Resolutions

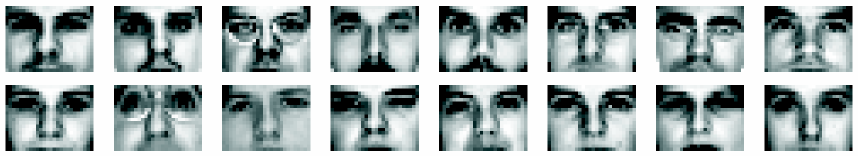
To find out all faces of various sizes in the test image, we apply the basic procedure over multiple resolutions. At each resolution, we process all F_s that start at any pixel of T to find all faces of a certain size. Since the basic procedure can handle faces of the same to twice the size of the template face, two successive resolutions should differ in size by a factor of 2. The coarsest resolution image should have a size equal to or greater than the face template. After the face candidates over all resolutions are detected, we sort them based on their matching values. To finalize face boundaries, we remove all those candidates that overlap with any candidates with higher matching values.



(a) Face template region.



(b) Average face template.



(c) Partial training faces.

Figure 5.4: Generation of average face template.

Generation of Average Face Model

Since essentially the proposed method is a 2-D template matching algorithm, we need to build a face template, which pretty much determines the detection performance. The face template should grasp as much as possible the common features of human face, while at the same time is not vulnerable to the background and individual character, e.g. the style of hair or the shape of beard. With this consideration in mind, we decide to use a rectangle that encloses the eye brows and the upper lips as face template, which is illustrated by the rectangle in Figure 5.4 (a). While larger size template provides more accurate face model, it also requires more computation in face detection. By trial and error, we set the template size 20×26 as a tradeoff. The training data we used to compute the average face template is from the AR face database of Purdue University [80]. All the face images in the database are labeled according to the light condition, facial expression, and etc. We choose 132 faces with neutral expression as training data. The face detection algorithm is applied to all these training images using a face template manually chopped from one image, and the $\hat{F}$'s corresponding to all detected face regions are averaged to produce the trained face template. Figure 5.4 (c) shows partial detected face regions that are used to build the face model. Figure 5.4 (b) shows the final face template.

Improvement of the Performance

We are interested in two issues of performance here: accuracy and speed. To improve the detection accuracy, we impose a preprocessing step before iterated dynamic programming procedure. The intensity values of the three square regions that are used to determine the original column mapping function g^0 are adjusted so that the means and standard deviations of intensity are equal to those of the face template. There are two ways to reduce the computation load of the algorithm. First when there is other information like color available, we can use skin-tone [34] to reduce the search area. Since we only want to use color information to grossly restrict the face location, we use a loose criterion so that no possible face region is lost in the very early stage. Second, we need not to search all possible starting position pixel by pixel, but in a hierarchical way. For example, in the first round, we search at the step $S1 \times S1$, and then at the second round, we only search the neighborhood of the most possible starting positions chosen in first round by step $S2 \times S2$, where $S2 < S1$. This procedure continues until the searching step is 1×1.

5.3.2 Face Tracking in Video

In face tracking, we need to find out different face trajectories along time axis. Instead of tracking faces directly on the entire video, we first segment the video sequence into shots, then track faces in each shot independently, and finally cluster face tracks of all shots such that faces of the same person are grouped together.

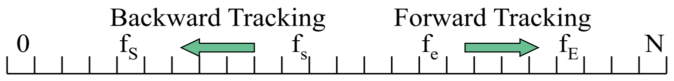
Video Shot Segmentation

Most researchers use the χ^2 distance between color histograms of successive frames to detect shot changes [81]. This method is sensitive to flash light and is apt to miss smooth transition, such as fade in and fade out. Although motion information may help to cope with flash light effect, and accumulated histogram distance can detect smooth transition, these methods introduce more computation load. Here we consider the color histogram distance of frames that are K frames apart. A shot cut typically leads to continuous high values of certain distance that last K frames. On the other hand, a camera flash often yields two single-frame peaks that are K frames apart. When a smooth transition happens, the distance contour usually follows a triangular shape. Based on these observations, smooth transition can be robustly detected and camera flash can be easily filtered out. Through experiments, we find that $K = 6$ gives reliable results for video digitized at 10 frames per second.

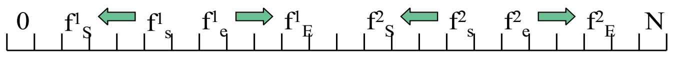
Face Tracking Within Each Shot

Two stages are involved for face tracking in each shot: detecting frontal faces and expanding face tracks in surrounding frames. In the first stage, an average face model is used to detect faces in each frame, where only frontal faces can be effectively detected. In the second stage, we use detected faces as new face templates to search faces in neighboring frames bidirectionally. Since it is reasonable to assume that the location and size of faces within each shot do not change much, we only need to search faces in neighboring regions of the detected faces. The updated face templates are expected to catch up with the smooth transition of faces from frontal view to others. When there is no skin-color occurrence in the first frame of a shot, we may simply skip the whole shot since the frames within each shot share similar color distribution.

Figure 5.5 illustrates two cases for face track expansion. The first case is simple, where one face track is detected in the first stage, and it starts at frame f_s



(a) One face track in a shot.



(b) Two face tracks in a shot.

Figure 5.5: Illustration of face track expansion.

and ends at frame f_e. We use the face detected in frame f_s as template to find face in frame $f_s - 1$. Similar procedure is iterated backward until there is no face detected or the shot boundary is met. The starting frame of the face track is then extended to frame f_S, and following the same method, the ending frame is extended to frame f_E. In the second case, two separated face tracks are detected. We first extend the face track with smaller starting frame number. During forward expansion, once $f_E^1 = f_s^2$, we test whether the two face tracks overlap spatially. If overlap, they are merged into one track, and only f_e^2 need to be extended forward. Otherwise, the two tracks are expanded independently. For cases that more than two separated faces are detected in a shot, similar procedures are applied.

Face Track Clustering

After the faces within each shot are tracked, we want to group the trajectories of the same face in different shots. The similarities among face tracks are measured by the similarity values among their representative faces, which have the maximum matching values corresponding to the average face template. By setting the smaller face as template and the bigger one as test image, the proposed face detection algorithm can determine the similarity between two faces using the matching value. When necessary, we re-sample the smaller face so that both of its height and width are not bigger than the bigger one. After we compute the similarity matrix of the representative faces of the entire video, we use agglomerative hierarchical clustering algorithm [68] to group face tracks.

5.4 Integrated Major Cast Detection

In current study, we only consider detection of major cast appearances that are accompanied by both speech and face. Satoh et al. used visual and text information to associate faces with names [35]. Our approach is basically to associate faces with speech for major casts. Three situations of face occurrences for a specific speaker are 1) speaker's face is not shown in the video, 2) only speaker's face is shown, and 3) more than one face including speaker's face are shown. To determine the face corresponds to a detected speaker, we utilize the temporal correlation between faces and speakers. In this section, we first gives the definition of speaker face correlation matrix. Based on this matrix, we show the integrated speaker segments and face tracks clustering algorithm, and major cast choosing and ordering method.

5.4.1 Speaker Face Correlation Matrix

We utilize the temporal correlations among different speakers and faces to find out the embedded mapping relationship. Suppose there are M speaker segments, $S_1, S_2, ..., S_M$, and N face tracks, $F_1, F_2, ..., F_N$. Different speaker segments or face tracks may be the same person. To make our approach general, we assume that speaker segment S_i has L_i discontinuous sub-segments: $s_1^i, s_2^i, ..., s_{L_i}^i$, each sub-segment has two attributes: starting time(ST) and ending time(ET). Similarly, face track F_i has l_i discontinuous sub-tracks: $f_1^i, f_2^i, ..., f_{l_i}^i$, each sub-track has three attributes: starting time, ending time, and face size(FS). Here we use the middle frame of each face sub-track to determine the face size. Then the speaker face correlation(SFC) matrix is an $N \times M$ matrix, whose item $SFC(i, j)$ is defined as:

$$SFC(i,j) = \sum_{m=1}^{L_i} \sum_{n=1}^{l_j} OL(s_m^i, f_n^j) \times FS(f_n^j), \tag{5.7}$$

where $OL(x,y)$ is the overlapping duration of speaker sub-segment x and face sub-track y, and FS(y) is the face size of face sub-track y. Figure 5.6 illustrates the correlation between speaker segment S_i and face track F_j. Such defined correlation value not only considers the temporal overlapping among speaker segments and face tracks, but also reflects the effect of face size. The second property is very useful when more than one face show up during a speech segment, where the face with bigger size is more likely to be the real speaker.

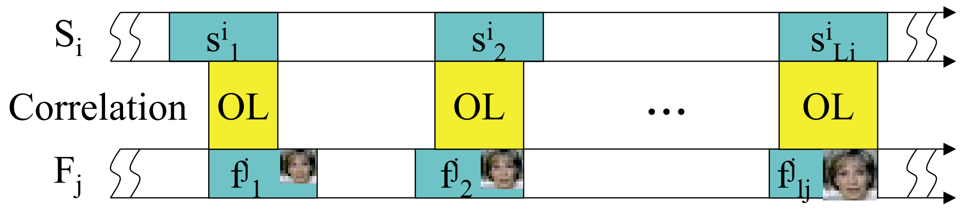


Figure 5.6: Illustration of Speaker Face Correlation.

Based on the speaker face correlation matrix, for each speaker segment, we can easily determine the face track that corresponds to the same person by choosing one with maximum correlation value. Although the speech and face of one person may not be perfectly synchronized, this method is still feasible if only dominant portion of corresponding audio and visual appearances are overlapped.

5.4.2 Integrated Speaker Face Clustering

While speaker segments or face tracks can be clustered independently, it is obvious that they can help each other. For example, when we judge whether two faces are from the same person, if corresponding speakers sound similar, we may tolerate more about the difference of faces. Here we propose a new integrated approach that cluster face tracks and speaker segments simultaneously. Suppose after speaker segmentation and face tracking, we have M speaker segments, N face tracks, denoted in the same way in the last section. The distance matrix among speaker segments is DMS, the distance matrix of face tracks is DMF, and their correlation matrix is SFC. The idea is to define an augmented distance matrix for speaker segments/face tracks based on both distance among speaker segments/face tracks and distance among corresponding face tracks/speaker segments weighted by their correlation values. We use DMS' to denote the augmented distance matrix of speaker segments, and DMF' for face tracks. The item in DMS' and DMF' can be computed as,

$$\begin{aligned} DMS'(i,j) &= \lambda_f \times \min\left(\frac{\sum_{m=1}^{m=N}\sum_{n=1}^{n=N} SFC(i,m)SFC(j,n)DMF(m,n) + T_f\epsilon}{\sum_{m=1}^{m=N}\sum_{n=1}^{n=N} SFC(i,m)SFC(j,n) + \epsilon}, T_f\right) \\ &\quad + DMS(i,j), \quad 1 \le i,j \le M. \qquad (5.8) \\ DMF'(i,j) &= \lambda_s \times \min\left(\frac{\sum_{m=1}^{m=M}\sum_{n=1}^{n=M} SFC(m,i)SFC(n,j)DMS(m,n) + T_s\epsilon}{\sum_{m=1}^{m=M}\sum_{n=1}^{n=M} SFC(m,i)SFC(n,j) + \epsilon}, T_s\right) \\ &\quad + DMF(i,j), \quad 1 \le i,j \le N, \qquad (5.9) \end{aligned}$$

where λ_f and λ_s are ratios that determine the weighting of distance effect from different modality, ϵ is a small constant to prevent division by zero, and T_f and T_s are two thresholds that are used in face tracks/speaker segments independent clustering. The detailed integrated clustering procedure is shown as follows.

1. Starting with $M^{(0)}$ speaker segments, $N^{(0)}$ face tracks, distance matrix $DMS^{(0)}$, $DMF^{(0)}$, and correlation matrix $SFC^{(0)}$. Set $i = 0$.

2. Compute the augmented distance matrix: $DMS'^{(i)}$ and $DMF'^{(i)}$.

3. Merge speaker segment/face track pairs with minimum augment distance if they are less than certain thresholds.

4. Set $i = i + 1$, and update distance matrix $DMS^{(i)}$, $DMF^{(i)}$, and $SFC^{(i)}$.

5. If no merge happens, then stop, otherwise, go to the second step.

The integrated clustering method is more reliable than independent clustering method, since it introduces the correlation information between speaker segments and face tracks. For example, suppose there are two speaker segments of the same person, one with clean speech, one with light background noise, then the speaker alone clustering may fail to merge these two segments. If we know that the two face tracks that shown in these segments are very similar, we are still confident to merge these two segment of speakers.

5.4.3 Major Cast Detection and Ordering

After clustering, each speaker segment corresponds to one speaker, and each face track corresponds to one face. We need to further determine the major casts by linking the faces to corresponding speakers. Then, an importance score is assigned to each major cast, so that a list of sorted major casts is extracted.

Mapping of faces to speakers is entirely dependent on the speaker face correlation matrix. The value of speaker face correlation reflects both the temporal (time span) and the spatial (face size) importance of the major cast. In the following algorithm, we do the speaker-face mapping and major cast ordering at the same time. Suppose after integrated speaker and face clustering, we get M different speakers and N different faces, and an $M \times N$ SFC matrix. The algorithm is,

1. Set $i = 0$.
2. Find an entry in the SFC matrix with maximum SFC value, denote the row and column indices of this entry by s_i and f_i, respectively.
3. Assign the speaker corresponding to row s_i and the face corresponding to column f_i to major cast i.
4. Remove row s_i and column f_i in SFC.
5. Set $i = i + 1$.

6. Go to step 2 unless the maximum value in SFC is smaller than a threshold.

This algorithm produces a list of major cast with corresponding correlation values, which are used as temporal-spatial importance scores.

5.5 Major Cast Presentation System

We develop a java applet for major cast based video presentation system, which is shown in Figure 5.7. The playback of video is controlled by Java Media Framework [82]. The panel on the left side shows the video, and the right panel displays the list of major casts in an intuitive and user friendly way. Speech segments of different major casts are painted in different colors, and the legend of color is plot on the top. Major casts are presented row by row. For each major cast, we present the face image on the left, then a vertical bar representing the importance score, and finally a time streamline identifying the occurrences of speech. By this way, the user may easily get the impression of who are the major casts, and where do they appear in the entire video. The user can directly browse all of certain major cast's video by clicking on the face images, or some specific portion of one major cast's appearance by clicking on the blocks in speech time line, so that only corresponding video clips are played.

5.6 Simulation Results and Discussion

The experimental data consists of 8 half-hour news broadcasts collected from NBC Nightly News off the air in 2000. The audio track is sampled at 16 KHz with resolution 16 bits per sample. The visual track is digitized at 10 frames per second, with size 240×180. We use 4 broadcasts for training and the rest are used for testing, denoted as sequence test1, test2, test3, and test4. The acquired data is manually segmented, tagged as clean speech or non-clean speech. Speaker identification in

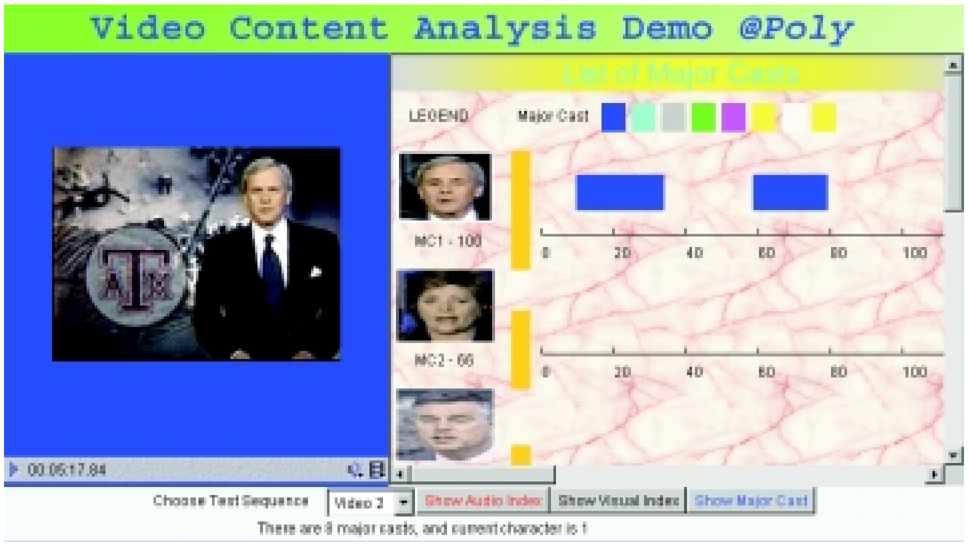


Figure 5.7: Major cast presentation.

clean speech segments and frontal view face identification for each visual shot are also annotated. These labels are used as ground truth to train the required models and measure the performance of proposed algorithms.

For clean speech detection, we use 14 clip based audio features, and benchmark different classification mechanisms: neural network, Gaussian Mixture Model, and Support Vector Machine classifiers, each with various parameter settings. The features are normalized such that they have the same deviation. All the results reported in this section are raw error rate, by which, we refer to the error rate calculated from the initial classification results performed on each clip without any smoothing techniques.

Table 5.1 shows the error rates of the neural network classifiers. From the table, we know that different numbers of hidden neurons give quite consistent results. The best performance is achieved with 6 hidden neurons.

Table 5.2 shows the results of GMM classifiers with different number of mixtures. Note that, during the training, we set the minimum covariance of each feature to 0.001. While GMMs with more mixtures can approximate the feature

Test Data	Number of Hidden Neurons					
	5	6	7	8	9	10
test1	5.3	4.7	6.5	6.5	5.5	6.0
test2	5.4	6.1	6.8	6.3	5.8	6.2
test3	7.1	6.9	6.8	6.4	6.9	7.7
test4	5.7	5.1	6.3	6.2	6.1	7.0
Average	5.9	5.7	6.6	6.4	6.1	6.7

Table 5.1: Error rates of Neural Network classifier. (unit: %)

distribution more accurately, it also require more data to train reliable models. GMM with 2 mixtures gives the best result in our study. The reason of worse performance for more mixtures may due to the limited number of training data.

Test Data	Number of mixture			
	2	4	8	16
test1	4.8	4.8	5.9	6.0
test2	6.6	6.3	5.8	7.1
test3	8.0	8.5	8.0	9.0
test4	5.3	5.7	6.3	7.4
Average	6.2	6.3	6.5	7.4

Table 5.2: Error rates of GMM classifier. (unit: %)

Table 5.3 presents the classification results for SVM classifiers. Four kernel functions are tested, they are dot product, 2nd, and 3rd order polynomials, and radial basis function. The γ coefficient used in RBF is set to 0.5. Best performance occurs when dot product is used as the kernel function. This indicates that the two types of audio events are well separable in the raw feature space. Compared to the neural network and GMM classifiers, SVM shows slightly better performance.

Overall the three types of classifiers give comparable performances, all of which are reasonably good. Considering the light computation load of GMM approach compared with the other two, we choose to use GMM classifier with 2 mixtures in this task.

Table 5.4 gives the performance of speaker/visual segmentation. In speaker

Test Data	Kernel Type			
	Dot	2nd Poly	3rd Poly	RBF
test1	4.4	5.1	5.0	5.5
test2	5.1	5.1	5.2	4.8
test3	6.7	6.9	7.1	6.8
test4	4.6	4.1	4.4	4.4
Average	5.2	5.3	5.4	5.4

Table 5.3: Error rates of SVM classifiers. (unit: %)

segmentation, we set the length of window to 3 seconds. If the detected speaker boundary is within 2 seconds away from the real boundary, we count it as a correct one, otherwise, a falsely detected one. In visual shot extraction, we set K to 6 frames, which covers a half second span. We require that the correct visual shot boundary should be within 5 frames away from the true boundary. Visual shot segmentation results are almost perfect, with average false detection rate and missing rate around 1%. The audio segmentation performance is a little bit worse, which may due to the following two reasons. First, we intentionally set the detection thresholds low since we do not want to miss the real speaker boundaries. If the real speaker boundaries are missed, there are multiple speakers in a single segment, and the trained speaker model is meaningless. Second, some of the speaker segments of the same person have different background sound and different speaking styles. For example, the same reporter may speak under different environments to cover the entire story, and the anchor person may abruptly change his/her tone to emphasize one story from the others. We still label them as the same person in ground truth even these segments are acoustically different. Both reasons contribute to a relatively high false detection rate. Note that the falsely separated segments may be grouped together in the clustering step. When real speaker boundaries are missed in one segment, the ground truth is annotated to be the dominant speaker with longest duration.

Table 5.5 shows the results of speaker clustering based on speaker segment distance matrix. The second row shows the total number of speaker segments, and

Test Data	test1	test2	test3	test4
Correctly Detected Speaker Segments	75	75	73	89
Falsely Detected Speaker Segments	12	11	8	11
Missed Speaker Segments	3	2	2	1
Correctly Detected Visual Shots	476	467	393	476
Falsely Detected Visual Shots	4	5	3	4
Missed Visual Shots	7	5	4	6

Table 5.4: Audio/visual segmentation results.

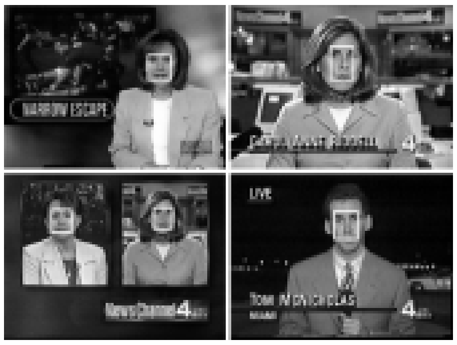
the third row gives the number of different speakers manually labeled. Normally, in one test sequence, anchor person has more than 10 segments, reporters have about 5 segments, and most of the persons in live report only have single segment. The forth row counts how many speakers are split into different clusters. If one speaker is distributed into N clusters, then this speaker contributes $N - 1$ in the final count. The last row measures how many different speakers are mixed in one cluster, where, for each cluster, the number of different speakers minus one is counted. During the clustering, we intentionally tune the thresholds such that fewer different speakers are mixed in the same clusters, which leads to higher speaker split rate. Other reasons for more speaker splits include the accumulated influence from previous stages: the error in clean speech and non clean speech classification, as well as the speaker segmentation error. The first kind of error may not filter out speech with noisy or music background, and the second one may mix different speakers in one segment, as well as over-segment speakers which produces many short segments. All these situations deteriorate the reliability of speaker models, and further influence the final clustering results. By observing the clustering results in detail, we find out that the effect of speaker splits is not serious, since many of split speaker segments are short, about 3 to 5 seconds. Considering that the average duration of anchor person or reporter segments is more than 20 seconds, the influence of split segments is tolerable.

Figure 5.8 shows some face detection results on still images based on the 20×26 average face template shown in Figure 5.4 (b). The detected faces are marked

Test Data	test1	test2	test3	test4
Total speaker segments	87	86	81	100
True different speakers	34	41	36	42
Speaker splits	29	23	23	29
Speaker mixes	5	6	2	2

Table 5.5: Speaker clustering results based on audio information.

by white rectangles. It takes a Pentium II-333 MHz machine about 2 seconds to detect faces in an image of size 180 × 240, which is much faster than the neural network based approach.


(*a*) 4 *color images* (180*x*240).

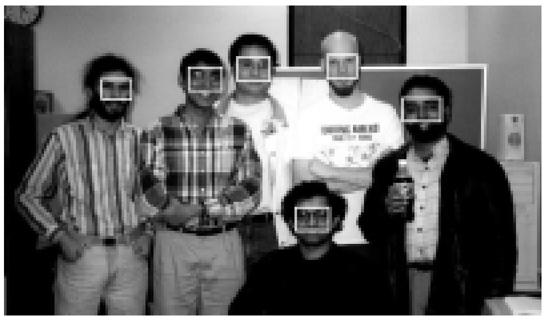

(*b*) 1 *grayscale image* (358*x*600).

Figure 5.8: Face detection results of still images.

Figure 5.9 shows some face tracking results from sequence test1. In the first stage, faces are successfully detected using average face template from frame 90 to 107, and from frame 118 to 126. the faces between frames 108 and 117 are detected by face tracking algorithm. In the figure, we show some of the detected faces in the forward tracking procedure. From the figure, we know the reason they fail in the first stage is that the faces are slant. Since in frame 118, the two face tracks overlap spatially, they are merged.

Table 5.6 gives the results of face tracking and clustering based on face distance matrix. The second row is the number of correctly detected face tracks, the

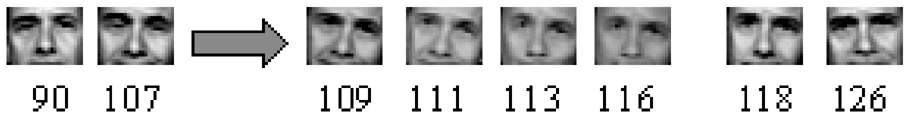


Figure 5.9: Face tracking within a shot.

Test data	test1	test2	test3	test4
Correctly detected face tracks	30	18	21	23
Falsely detected face tracks	4	6	5	5
Missed face tracks	3	4	2	3
Face splits	2	3	2	5
Face mixes	0	0	1	4

Table 5.6: Face tracking and clustering results based on visual information.

third row lists the number of falsely detected face tracks, and the forth row shows the missed ones. The reason for both false detection and missing is due to the face detection errors because of lighting, glass reflection, and etc. The last two rows are performance measurements for face track clustering, similarly defined as those in Table 5.5. Falsely detected face tracks are manually labeled as distinct faces while computing the numbers in the last two rows. Normally, anchor person shows up about 10 times, reporters show up various times from once to three times. From the table, we can see that most of different faces are separated into different clusters, and the number of face mixes is relatively low.

Using integrated speaker face clustering method provides slightly better results. Table 5.7 shows the results of integrated clustering approach. Compared with Table 5.5 and Table 5.6, we can see that the speaker segment clustering results of test1, test2, and test3 are improved, and the face track clustering results of test1 and test4 are improved. Although the improvement is not large, the results are quite encouraging since they show that the integrated approach is feasible and helpful. The integrated clustering mechanism largely depends on the synchronization of person's

Test data	test1	test2	test3	test4
Speaker splits	29	23	23	30
Speaker mixes	2	5	1	2
Face splits	1	1	3	2
Face mixes	0	2	1	5

Table 5.7: Integrated speaker face clustering results.

speech and visual occurrence. In news broadcast, when reporters deliver the story, very often, they do not show up on the screen since the customer can hear the speech and want to see more about the live scene. On the other hand, anchor person's audio and visual appearances are well synchronized, and the improvement of integrated clustering approach is mainly from those speaker segments and face tracks of anchor person.

For the four test sequences, we detect 8, 9, 6, and 8 major casts respectively. Among all these characters, the most important ones are consistently the anchor persons, followed by different reporters and interviewers. Figure 5.10 shows the face images of the eight major casts detected in test1 in the order of their importance values. The top major cast is the anchor person: Tom Brokaw. The third, the forth and the last major casts are news reporters, and the rest are interviewers. The reason why some reporters earn low importance scores is that they just occasionally show up during the reporting. It is obvious that this list gives a meaningful guidance for the content understanding.

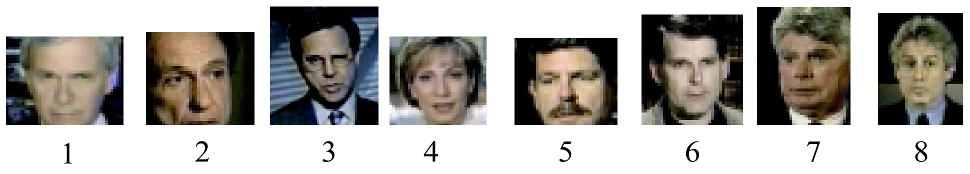


Figure 5.10: Faces of major casts detected in test1.

5.7 Summary

This chapter proposes a new approach to detect major casts in video based on both audio and visual information and develops a major cast based video browsing system. Several techniques are addressed to detect the major casts. 1) Video segmentation: The clean speech chunk is segmented based on speaker changes, and the visual track is segmented into smaller shots, within which the content is homogeneous. Audio and visual segmentation approaches are designed based on corresponding feature sets and criteria. 2) Face detection and tracking: A new template matching based face detection algorithm is proposed, which is basically an iterative dynamic programming procedure. Face detection algorithm is designed to find out front view faces, and the face tracking algorithm can update the face templates adaptively to follow the change of faces within each visual shot. 3) Speaker and face clustering: Speaker segments and face tracks are clustered such that the same person's audio/visual appearances can be grouped together. Besides clustering speaker segments and face tracks based on audio/visual information independently, we propose a new integrated approach to simultaneously cluster both speaker segments and face tracks, where the temporal correlations among them are also utilized. 4) Major cast extraction: The major casts are chosen based on face and speaker correlation values, and they are sorted based on the importance scores determined by their temporal/spatial volumes. A major cast based video browsing system is developed, which provides an intuitive interface for the user to digest the theme of the video and skim the content efficiently.

Chapter 6

Conclusions and Possible Future Works

This thesis has conducted various research related to multimedia content analysis. In this chapter, we first summarize our contributions, and then suggest some possible future works.

6.1 Summary of Major Contributions

Inspired by the two observations described in chapter 1: integration of multiple modalities and content analysis at multiple levels, the major contributions of this thesis are investigating them and proposing feasible and efficient solutions for multimedia content analysis.

At perceptual level, we solve the problem of detecting different video events in broadcast news. A set of audio and visual features are exploited, and their discrimination capabilities are studied. Several classification schemes are tested, and all of them give reasonably good results. While the proposed solution is appliable for detecting various categories of multimedia events, different combinations of audio/visual feature sets and classifiers may be suitable for specific tasks.

At conceptual level, there is more diversity in the detailed approaches, including the choice of target semantic objects, processing algorithm within each modality, and integration method for multiple modalities. Individually, we propose feasible algorithms for three applications. In news broadcast browsing system, we propose an integrated approach to adaptively detect unknown anchor person. The novelty is that it not only combines visual and audio information but also integrates model based and unsupervised approaches. Semantically meaningful news storys are extracted based on both audio and text information, whose boundaries are more accurate than those generated by text alone approach. The system provides a table-of-content for headline news stories and multimedia summary for each news program and each story, which help the user browse the data more efficiently. Part of the algorithms developed in this system are integrated in the Digital Video Library system at AT&T Labs - Research.

We propose a new approach for content based audio indexing and query. The audio signal is segmented into homogeneous events whose features are characterized using GMMs. To measure the dissimilarity between two audio events, a new distance metric framework is proposed to compute the difference between corresponding GMMs. Both audio clustering and audio querying are based on this new metric. Due to the succinct representation of audio events and efficiency of event comparison mechanism, the proposed method is especially useful for middle and large size database.

Major cast detection system is another conceptual level multimedia content analysis application that integrates both audio (speaker) and visual (face) information. Within this system, a new template matching based face detection algorithm is proposed, which finds the best warping functions between the test region and the template using an iterative dynamic programming procedure. Although the face detection algorithm is designed to find front view faces, the face tracking algorithm can update the face templates adaptively to follow the change of faces within each visual shot. We also propose an integrated approach to simultaneously cluster both speaker segments and face tracks, where the temporal correlations among them are also utilized. Based on face and speaker correlation values, major casts are chosen and they are sorted by their temporal/spatial presence. The user may digest the theme of the video and skim the content efficiently by the help of the major cast list.

Our experimental results from the previous chapters show that (1) integrating data from different media can achieve what a single medium approach can not, thereby attaining better performance, (2) abstracting multimedia data into semantically meaningful units can significantly improves the effectiveness in browsing, (3) deriving well defined semantics from data makes it possible to form a table of content as well as content based representations for different semantics that further enhance the quality of data retrieval, allowing users to go through large amounts of multimedia data with convenience, efficiency, and confidence.

6.2 Possible Future Work

There are a number of possible extensions and applications for the work presented in this thesis. Following are some suggestions.

First, it is possible to extract other effective audio and visual features for perceptual level content indexing. Some of the developed audio features including the Energy Ratio in Subbands are designed for broad bandwidth audio with sampling rate no less than 16K Hz. If the same features are applied in narrow band audio, they need to be modified accordingly. Feature space analysis is also a very interesting task. By studying the distribution of features from different events, we can measure the discrimination capability of each feature, and come up with the optimal feature set of lower dimension. Besides Karhunen Loeve Transform and Multiple Discriminant Analysis, we can also measure the capability of each feature using information theory. One example is to test the joint entropy and mutual information among different events based on certain set of features. The feature set that gives higher joint entropy and lower mutual information among events should be good one.

Second, in news story extraction, we provide a group of keywords for each story, such that stories can be indexed and retrieved. It is more helpful to archive and retrieve the stories if categories are assigned to them, e.g. politics, sports, domestic or international news. The categories may not be mutually exclusive, which means that one story can be tagged with different categories. How to accurately classify story based on statistics of text streams has been a hot research area for quite long, and it is also a possible future work of this thesis.

Third, the content based audio query system is currently tested on news broadcast data only, and the size of database is relatively small. Since the proposed approach is a general one, we can try on other types of data, e.g. music and songs. With the size of database growing, the proposed query processing engine may not work efficiently since we need to compare the query with all events in the database. One possible work is to introduce structure in the database, for example, a binary tree structure, and code the audio events by the tree nodes. In such a way, the computation load of query processing engine is not in the linear order of the database size, but logarithmic.

Forth, while the distance metric for mixture type PDFs is proposed to measure the difference between audio events, it can also be used in many other applications, e.g. speech and speaker recognition. In speaker recognition, models of target speakers are built off-line. It is obvious that off-line model may not be accurate for on-line data, since the speaking conditions are different. If we use GMM to model the speaker, the proposed metric can be used to measure difference between off-line model and on-line model trained with new data, and further update the off-line model accordingly. In speech recognition, continuous Hidden Markov Model is widely used, where the observation PDFs of different states are normally modeled by GMM. The proposed method can measure the difference of states within the HMM, and optimize HMM structure by merging similar states. Considering the huge market of speech/speaker recognition applications, this work will be a worthy topic to study.

Fifth, although the major cast detection algorithm is tested on news broadcast data in our simulation, it can be extended to other kinds of data, e.g. video conference and movies. In video conference, the audio and video are well synchronized, which means the person showing in the screen is also the person who is talking. This will make the integrated speaker and face clustering algorithm work well. The structure of the content is also relatively simple. The challenge is that the quality of both audio and visual signals may not be perfect due to the performance of digitization equipment and signal compression mechanism as well as the capacity of network. Noisy audio source will deteriorate speaker segmentation and modeling accuracy, and low quality visual frames will challenge the face detection and tracking algorithm. On the other hand, to analyze movies, we can always find high quality data, but the difficulty is how to cope with the complex and diverse structures. Also the audio and visual channels of movies are not always well synchronized, speech is spontaneous and short, characters are shot at various views, and etc. There is no doubt that the extension of major cast detection is meaningful. It is also a hard problem.

Finally, in current study, the integration of multiple modalities are pretty much application dependent. How to utilize the information from different media such that they help or compliment each other at the maximum extent is a very interesting topic. A general framework for combining multiple cues for different applications is very necessary and useful. This also requires future research.

Bibliography

[1] D. H. Ballard and C. M. Brown, *Computer Visoin*, Prentice-Hall, 1982.

[2] R. Chellappa, C. L. Wilson, and S. Sirohey, “Human and machine recognition of faces: A survey,” *Proc. IEEE*, vol. 83, no. 5, pp. 705–741, May 1995.

[3] H. J. Zhang, A. Kankanhalli, and S. W. Smoliar, “Automatic partioning of video,” *Multimedia Systems*, vol. 1, no. 1, pp. 10–28, 1993.

[4] S. Smoliar and H. Zhang, “Content-based video indexing and retrieval,” *IEEE Multimedia Magazine*, pp. 62–72, Summer 1994.

[5] S. Smoliar H. Zhang, A. Kankanhalli, “Automatic partitioning of full-motion video,” *A Guided Tour of Multimedia Systems and Applications, IEEE Computer Society Press*, 1995.

[6] S.-F. Chang, W. Chen, H.J. Meng, H. Sundaram, and D. Zhong, “VideoQ — an automatic content-based video search system using visual cues,” in *ACM Multimedia Conference*, Seattle, WA, Nov. 1997.

[7] Y. Deng and B. S. Manjunath, “Netra-V: Toward an object-based video representation,” *IEEE Trans. Circuit and System on Visual Technology*, vol. 8, no. 5, pp. 616–627, Sept. 1998.

[8] M. Flickner et la, “Query by image and video content: The QBIC system,” *IEEE Computer*, vol. 28, no. 9, pp. 23–32, Sept. 1995.

[9] M. M. Yeung, B. L. Yeo, and B. Liu, "Time-constrained clustering for segmentation of video into story units," in *Proc. Int. Conf. Pattern Recognition*, Vienna, August 1996, pp. 375–380.

[10] M. M. Yeung and B. L. Yeo, "Video visualization for compact presentation and fast browsing of pictorial content," *IEEE Trans. Circuits and System for Video Technology*, vol. 7, no. 5, pp. 771–785, Oct. 1997.

[11] Y. Rui, T. S. Huang, and S.-F. Chang, "Image retrieval: current technologies, promising directions, and open issues," *Journal of Visual Communication and Image Representation*, vol. 10, no. 1, pp. 39–62, Mar. 1999.

[12] A. Ferman and A. Tekalp, "Efficient filtering and clustering methods for temporal video segmentation and visual summarization," *J. Vis. Comm. and Image Rep.*, vol. 9, no. 4, pp. 336–351, December 1998.

[13] A. Samal and P. A. Iyengar, "Automatic recognition and analysis of human faces and facial expressions: A survey," *Pattern Recognition*, vol. 25, no. 1, pp. 65–77, 1992.

[14] J. Zhang, Y. Yan, and M. Lades, "Face recognition: Eigenface, elastic matching, and neural nets," *Proc. IEEE*, vol. 85, no. 9, pp. 1423–1435, Sept. 1997.

[15] H. A. Rowley, S. Baluja, and T. Kanade, "Neural network-baesd face detection," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 22–38, Jan. 1998.

[16] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, 1993.

[17] JR. J. P. Campbell, "Speaker recognition: A tutorial," *Proc. IEEE*, vol. 85, no. 9, Sept. 1997.

[18] D. A. Reynolds and R. C. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Trans. Speech and Audio Processing*, vol. 3, no. 1, pp. 72–83, 1995.

[19] J. Saunders, "Real-time discrimination of broadcast speech/music," in *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Atlanta, GA, May 7-10 1996, vol. 2, pp. 993–996.

[20] C. Saraceno and R. Leonardi, "Audio as a support to scene change detection and characterization of video sequences," in *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Munich, Germany, Apr. 21-24 1997, vol. 4.

[21] E. Wold, T. Blum, D. Keislar, and J. Wheaton, "Content-based classification, search, and retrieval of audio," *IEEE Multimedia Magazine*, vol. 3, no. 3, pp. 27–36, Fall 1996.

[22] T. Zhang and C.-C. Jay Kuo, "Hierarchical classification of audio data for archiving and retrieving," in *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Phoenix, AZ, Mar. 15-19 1999, vol. 6, pp. 3001–3004.

[23] Y.L.Chang, W. Zeng, I. Kamel, and R. Alonso, "Integrated image and speech analysis for content-based video indexing," *Proc. of Multimedia*, pp. 306–313, September 1996.

[24] Q. Huang, Z. Liu, A. Rosenberg, D. Gibbon, and B. Shahraray, "Automated generation of news content hierarchy by integrating audio, video, and text information," in *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Phoenix, Arizona, March 1999.

[25] S. Eickeler and S. Mueller, "Content based video indexing of TV broadcast news using hidden Markov models," in *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Phoenix, USA, March 1999, vol. 6, pp. 2997–3000.

[26] Q. Huang, A. Puri, and Z. Liu, "Multimedia search and retrieval: New concepts, system implementation, and application," *IEEE Trans. Circuit and System on Visual Technology*, vol. 10, no. 5, pp. 679–692, August 2000.

[27] Jincheng Huang, Zhu Liu, and Yao Wang, "Joint video segmentation and classification based on HMM," in *Proc. IEEE Int. Conf. Multimedia & Expo*, New York, NY, July 30 - Aug. 2 2000.

[28] J. Huang, Z. Liu, Y. Wang, and E. K. Wong, "Video understanding by using audio and visual information, part I - audio and visual features and feature analysis," 1999, Submitted to IEEE Trans. Multimedia.

[29] J. Huang, Z. Liu, Y. Wang, and E. K. Wong, "Video understanding by using audio and visual information, part II - automatic video segmentation and classification," 1999, Submitted to IEEE Trans. Multimedia.

[30] J. S. Boreczky and L. D. Wilcox, "A hidden Markov model framework for video segmentation using audio and image features," in *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Seattle, WA, May 12-15 1998, vol. 6, pp. 3741–3744.

[31] C. Saraceno and R. Leonardi, "Identification of story units in audio-visual sequencies by joint audio and video processing," in *Proc. Int. Conf. Image Processing*, Chicago, IL, Oct. 4-7 1998, vol. 1, pp. 363–367.

[32] J. Huang, Z. Liu, and Y. Wang, "Integration of audio and visual information for content-based video segmentation," in *Proc. Int. Conf. Image Processing*, Chicago, IL, Oct. 4-7 1998, vol. 3, pp. 526–530.

[33] R. Lienhart, S. Pfeiffer, and W. Effelsberg, "Scene determination based on video and audio features," in *IEEE Int. Conf. on Multimedia Computing and Systems*, Florence, Italy, June 7-11 1999, vol. 1, pp. 685–690.

[34] Z. Liu and Q. Huang, "Adaptive anchor detection using on-line trained audio/visual model," in *Proc. SPIE: Storage and Retrieval for Media Database*, San Jose, Jan. 2000, pp. 156–167.

[35] S. Satoh, Y. Nakamura, and T. Kanade, "Name-it: Naming and detecting faces in news videos," *IEEE Multimedia Magazine*, vol. 6, no. 1, pp. 22–35, Jan-Mar 1999.

[36] J. Nam and A. H. Twefik, "Combined audio and visual stream analysis for video sequence seqmentation," in *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Munich, Germany, Apr. 21-24 1997, vol. 4.

[37] Y. Wang, J. Huang, Z. Liu, and T. Chen, "Multimedia content classification using motion and audio information," in *Proc. IEEE Int. Symposium on Circuit and Systems*, Hong Kong, June 1997, vol. 2, pp. 1488–1491.

[38] S. Pfeiffer, S. Fischer, and W. Effelsberg, "Automatic audio content analysis," in *ACM Multimedia 1996*, Boston, MA, Nov. 18-22 1996.

[39] Z. Liu, J. Huang, Y. Wang, and T. Chen, "Audio feature extraction & analysis for scene classification," in *Proc. IEEE 1st Workshop on Multimedia Signal Processing*, Princeton, NJ, June 1997, pp. 343–348.

[40] Z. Liu and Q. Huang, "Classification for audio events in broadcast news," in *Proc. IEEE 2nd Workshop on Multimedia Signal Processing*, Log Angeles, DA, Dec. 7-9 1998, pp. 27–32.

[41] Z. Liu, Y. Wang, and T. Chen, "Audio feature extraction and analysis for scene segmentation and classification," *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, vol. 20, no. 1/2, pp. 61–79, Oct. 1998.

[42] Y. Wang, Z. Liu, and J. Huang, "Multimedia content analysis using audio and

visual information," Nov. 2000, to appear IEEE Signal Processing Magazine (invited paper).

[43] E. Scheirer and M. Slaney, "Construction and evaluation of a robust multifeatures speech/music discriminator," in *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Munich, Germany, April 21-24 1997, vol. 2, pp. 1331–1334.

[44] W. Hess, *Pitch Determination of Speech Signals*, Springer-Verlag, 1983.

[45] N. Jayant, J. Johnston, and R. Safranek, "Signal compression based on models of human perception," *Proc. IEEE*, vol. 81, no. 10, pp. 1385–1422, Oct. 1993.

[46] F. Idris and S. Panchanathan, "Review of image and video indexing techniques," *Journal of Visual Communication and Image Representation*, vol. 8, no. 2, pp. 146–166, June 1997.

[47] X. Wan and C.-C. J. Kuo, "A new approach to image retrieval with hierarchical color clustering," *IEEE Trans. Circuits and Systems for Video Technology*, vol. 8, no. 5, pp. 628–643, Sept. 1998.

[48] A. N. Netravali and B. G. Haskell, *Digital Pictures Representation, Compression and Standards (Second Edition)*, Plenum Press, 1995.

[49] K. Fukunaga, *Introduction to statistical pattern recognition*, Academic Press, 1972.

[50] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, John Wiley & Sons, 1972.

[51] Carl G. Looney, *Pattern Recognition Using Neural Networks*, Oxford University Press, 1997.

[52] R. P. Lippman, "An introduction to computing with neural nets," *IEEE ASSP Magazine*, pp. 4–22, April 1987.

[53] S. H. Lin, S. Y. Kung, and L. J. Lin, “Face recognition / detection by probabilistic decision-based neural network,” *IEEE Trans. Neural Networks*, vol. 8, no. 1, pp. 114–132, Jan. 1997.

[54] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, 1998.

[55] Walter Rudin, *Functional Analysis*, McGraw-Hill, 1991.

[56] C. J.C. Burges, “Tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, no. 2, 1998.

[57] J. A. K. Suykens and J. Vandewalle, *Nonlinear Modeling*, Kluwer Academic Publishers, 1998.

[58] Z. Liu and Q. Huang, “Detecting news reporting using audio/visual information,” in *Proc. Int. Conf. Image Processing*, Kobe, Japan, Oct. 1999, vol. 3, pp. 324–328.

[59] Z. Liu, J. Huang, and Y. Wang, “Classification of TV programs based on audio information using hidden Markov model,” in *Proc. IEEE 2nd Workshop on Multimedia Signal Processing*, Log Angeles, DA, Dec. 7-9 1998, pp. 27–32.

[60] J. Huang, Z. Liu, Y. Wang, Y. Chen, and E. K. Wong, “Integration of multimodal features for video classification based on HMM,” in *Proc. IEEE 3rd Workshop on Multimedia Signal Processing*, Copenhagen, Denmark, Sept 13-15 1999, pp. 53–58.

[61] L. Chen and P. Faudemay, “Multi-criteria video segmentation for TV news,” in *Proc. IEEE 1st Multimedia Signal Processing Workshop*, Princeton, NJ, June 1997, pp. 319–324.

[62] Chien Yong Low, Qi Tian, and Hongjiang Zhang, “An automatic news video parsing, indexing, and browsing system,” in *Proc. of The Fourth ACM International Multimedia Conference*, Boston, November 1996, pp. 425–426.

[63] Q. Huang, Z. Liu, and A. Rosenberg, "Automated semantic structure reconstruction and representation generation for broadcast news," in *Proc. SPIE: Storage and Retrieval for Image and Video Database VII*, San Jose, CA, Jan. 1999, pp. 50–62.

[64] A. Hanjalic, R. L. Lagendijk, and J. Biemond, "Semi-automatic news analysis, indexing, and classification system based on topics preselection," in *Proc. SPIE: Storage and Retrieval of Image and Video Databases VII*, San Jose, CA, January 1999, pp. 86–97.

[65] A. Rosenberg, I.Magrin, S. Partha, and Q. Huang, "Speaker detection in broadcast speech databases," in *Proc. of Int. Conf. on Spoken Language Processing*, Sydney, Austrilia, November 1998.

[66] Q. Huang, Y. Cui, and S. Samarasekera, "Content based active video data acquisition via automated cameramen," in *Proc. Int. Conf. Image Processing*, Chicago, IL, October 1998, vol. 2, pp. 808–812.

[67] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison-Wesley, 1992.

[68] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice-Hall, 1998.

[69] R. C. Rose, E. M. Hofstetter, and D. A. Reynolds, "Integrated models of signal and background with application to speaker identification in noise," *IEEE Trans. Speech and Audio Processing*, vol. 2, no. 2, pp. 245–257, 1994.

[70] M. A. Hearst, "Multi-paragrah segmentation of expository text," in *The 32nd Annual Meeting of the Association For Computational Linguistics*, New Mexico, USA, June 1994, pp. 9–16.

[71] B. Shahraray and D. Gibbon, “Automatic generation of pictorial transcripts,” in *Proc. SPIE: Multimedia Coomputing and Networking*, San Jose, CA, Mar. 1995, vol. 2417, pp. 512–518.

[72] Z. Liu and Q. Huang, “A new distance measure for probability distribution function of mixture type,” in *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Istanbul, Turkey, June 2000.

[73] Z. Liu and Q. Huang, “Content-based indexing and retrieval-by-example in audio,” in *Proc. IEEE Int. Conf. Multimedia & Expo*, New York, NY, July 30 - Aug. 2 2000.

[74] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, John Wiley & Sons, 1991.

[75] Eberhard Zeidler, *Applied Functional Analysis: Applications to Mathematical Physics*, Springer-Verlag, 1995.

[76] D. A. Pierre, *Optimization Theory With Application*, Dover, 1986.

[77] H. L. Van Trees, *Detection, Estimation, and Modulation Theory*, John Wiley & Sons, 1967.

[78] Z. Liu, Y. Wang, and J. Huang, “Major cast detection in video using both speaker and face information,” 2000, Prepared to submit to IEEE Transactions on Multimedia.

[79] Z. Liu and Y. Wang, “Face detection and tracking in video using dynamic programming,” in *Proc. Int. Conf. Image Processing*, Vancouver, Canada, Sept. 10 - 13 2000.

[80] A.M. Martinez and R. Benavente, “The AR face database,” in *CVC Technical Report #24*, http://rvl1.ecn.purdue.edu/∼aleix/aleix_face_DB.html, June 1998.

[81] G. Lupatini, C. Saraceno, and R. Leonardi, “Scene break detection: A comparison,” in *Eighth International Workshop on Reasearch Issues In Data Engineering*, Feb. 1998, pp. 34–41.

[82] Sun Microsystem, “Java media framework,” http://www.javasoft.com.

List of Publication

1. Z. Liu, Y. Wang, and J. Huang, "Major cast detection in video using both speaker and face information," Prepared for *IEEE Trans. Multimedia*

2. Y. Wang, Z. Liu, and J. Huang, "Multimedia content analysis using audio and visual information," to appear *IEEE Signal Processing Magazine (invited paper)*, Nov. 2000.

3. Q. Huang, A. Puri, and Z. Liu, "Multimedia search and retrieval: New concepts, system implementation, and application," *IEEE Trans. Circuit and System on Visual Technology*, Vol. 10, No. 5, pp. 679-692, August 2000

4. Z. Liu and Y. Wang, "Face detection and tracking in video using dynamic programming," *Proc. Int. Conf. Image Processing*, Vancouver, Canada, Sept. 10 - 13, 2000.

5. J. Huang, Z. Liu, and Y. Wang, "Joint video segmentation and classification based on HMM," *Proc. IEEE Int. Conf. Multimedia & Expo*, New York, NY, July 30 - Aug. 2, 2000.

6. Z. Liu and Q. Huang, "Content-based indexing and retrieval-by-example in audio," *Proc. IEEE Int. Conf. Multimedia & Expo*, New York, NY, July 30 - Aug. 2, 2000.

7. Z. Liu and Q. Huang, "A new distance measure for probability distribution function of mixture type," *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Istanbul, Turkey, June 5-9, 2000.

8. Z. Liu and Q. Huang, "Adaptive anchor detection using on-line trained audio/visual model," *Proc. SPIE: Storage and Retrieval for Media Database*, San Jose, CA, Jan. 2000.

9. J. Huang, Z. Liu, Y. Wang, and E. K. Wong, “Video understanding by using audio and visual information, part I — audio and visual features and feature analysis,” *submitted to IEEE Trans. on Multimedia.*

10. J. Huang, Z. Liu, Y. Wang, and E. K. Wong, “Video understanding by using audio and visual information, part II — automatic video segmentation and classification,” *submitted to IEEE Trans. on Multimedia.*

11. J. Huang, Z. Liu, Y. Wang, Y. Chen and E. K. Wong, “Integration of multimedia features for video classification based on HMM,” *Proc. IEEE 3rd Workshop on Multimedia Signal Processing*, pp. 53-58, Copenhagen, Denmark, Sept. 1999.

12. Z. Liu and Q. Huang, “Detecting news reporting using audio/visaul information,” *ICIP-1999*, Vol. 3, pp. 324-328, Kobe, Japan, Oct. 1999.

13. Q. Huang, Z. Liu, A. Rosenberg, D. Gibbon, and B. Shahraray, “Automated generation of news content hierarchy by integrating audio, video, and text information,” *Proc. Int. Conf. Acoustic, Speech, and Signal Processing*, Vol. 6, pp. 3025-3028, Phoenix, Arizona, March 1999.

14. Q. Huang, Z. Liu, and A. Rosenberg, “Automated semantic structure recognition and representation generation for broadcast news,” *Proc. SPIE: Storage and Retrieval for Image and Video Database VII*, Vol. 3656, pp. 50-62, San Jose, Jan. 1999.

15. Z. Liu, J. Huang, and Y. Wang, “Classification of TV programs based on audio information using hidden Markov model,” *Proc. IEEE 2nd Workshop on Multimedia Signal Processing*, pp. 27-32, Los Angeles, CA, Dec. 1998.

16. Z. Liu and Q. Huang, “Classification of audio events in broadcast news,” *Proc. IEEE 2nd Workshop on Multimedia Signal Processing*, pp. 364-369, Los Angeles, CA, Dec. 1998.

17. J. Huang, Z. Liu, and Y. Wang, "Integration of audio and visual information for content-based video segmentation," *Proc. Int. Conf. Image Processing*, pp. 526-530, Chicago, IL, Oct. 1998.

18. Z. Liu, Y. Wang, and T. Chen, "Audio feature extraction and analysis for scene segmentation and classification, " *Journal of VLSI Signal Processing Systems for Signal, Image, and Video Technology*, Vol. 20, No. 1/2, pp. 61-79, Oct. 1998.

19. Z. Liu, J. Huang, Y. Wang, and T. Chen, "Audio feature extraction and analysis for scene classification," *Proc. IEEE 1st Workshop on Multimedia Signal Processing*, pp. 343-348, Princeton, NJ, June 1997.

20. Y. Wang, J. Huang, Z. Liu, and T. Chen, "Multimedia content classification using motion and audio information," *Proc. IEEE Int. Symposium on Circuit and Systems*, Vol. 2, pp. 1488-1491, Hong Kong, June 1997.