CORE
Metadata, citation and similar papers at core.ac.uk
Provided by ScholarBank@NUS

A HIERARCHICAL MULTI-MODAL APPROACH TO STORY SEGMENTATION IN NEWS VIDEO

LEKHA CHAISORN
(M.S., Computer and Information Science, NUS)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE

NATIONAL UNIVERSITY OF SINGAPORE

2004

ACKNOWLEDGMENT

I would like to express my gratitude to my supervisor, Prof. Chua Tat Seng, for his excellent guidance and encouragement. His valuable suggestions and advice helped me tremendously to complete my PhD study. Needless to say, his patient and very high responsibility helped me to overcome a lot of difficulties during the research.

I would like to acknowledge the support of the Agency for Science, Technology and Research (A*STAR) and the Ministry of Education of Singapore for the provision of a research grant RP3960681 under which this research is carried out.

I would like to thank Professors Chin-Hui Lee, Mohan S Kankanhalli, Rudy Setiono and Wee-Kheng Leow for their comments and fruitful suggestions on this research.

I would also like to thank all friends in Multimedia lab especially to Koh Chunkeat, Dr. Zhao Yunlong, Lee Chee Wei, Feng Huamin, Xu Huaxin, Yang Hui, Marchenko Yelizavita and Chandrashekhara Anantharamu for exchanging experiences in research and sharing their programming skill.

I would like to thank Catharine Tan and Ng Li Nah, Stefanie for giving me friendship, and to the staff in the School of Computing who helped me in several ways.

I would like to thank my parents and my family members for their support throughout this research.

Last but not least, I would like to thank Ho Han Tiong who gave me very persistent encouragement and moral support.

TABLE OF CONTENTS

SUMMARY

We propose a framework for story segmentation in news video by comparing two learning-based approaches: (1) Hidden Markov Models (HMM); and (2) Rule induction technique. In both approaches, we divided our framework into 2 levels, shot and story levels. At the shot level, we define three clusters totalling 17 shot categories. The clusters are *heuristic-based* (contains commercial shots); *visual-based* (consists of *Weather* and *Finance* shots, *Anchor* shots, program logo shots etc.) and *Machine-learning-based* clusters (contains *live-reporting* shots, *People* shots, *sport* shots, etc.). We represent each shot using low-level feature (176-Luv colour histogram), temporal features (audio class, shot duration, and motion activity) and high level features (face, shot type, videotexts), and employ a combination of heuristics, specific detectors and decision trees to classify the shots into the respective categories. At the story level, we use the shot category information, scene/location change and cue-phrases as the features, and employ either HMM or rule induction techniques to perform story segmentation. We test our HMM framework on the 120 hours of news video from TRECVID 2003 and the results show that we could achieve an F_1 measure of over *77%* for story segmentation task. Our system achieved the best performance during TRECVID 2003 evaluations [TRECVID 2003]. We also test our rule induction framework on the same TRECVID data and we could achieve an accuracy of over 75%. The results show that our 2-level framework is effective in story segmentation. The framework has the advantage of dividing the complex problem into 2 parts and thus partially alleviates the data sparseness problem in

machine learning. Our further analysis shows that as compared to HMM, the rule induction approach is easier to incorporate new (heuristic) rules and adapt to new corpora.

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction

The rapid advances in computing, multimedia, and networking technologies have resulted in the production and distribution of large amount of multimedia data, in particular digital video. To effectively manage these sources of videos, it is necessary to organize them in a way that facilitates user browsing and retrieval. Much effort has been made by researchers to segment, index and organize digital videos in terms of shots [Gunsel 1996] [Das and Liou 1998] [Ide 1999]. Digital videos, especially news videos such as CNN, ABC, etc that are available on the web are a good source of information. Users normally do not start reading news or viewing news video from the start of news broadcast until the end. Instead, the users often access the news by topics of their interests. Some users give priority to finance or business news while others are interested in world news such as the "war in Iraq", etc. Thus, a news video broadcast needs to be segmented into appropriate units to support this kind of access.

Research on segmenting an input video into shots, and using these shots as the basis for video organization is well established [Zhang 1993][Lin 2000][Anantharamu 2002]. A shot represents a contiguous sequence of visually similar frames. It does not usually convey any coherent semantics to the users. The shot units, however, are

important when the users want to access only some shots of a particular story, such as, a shot of a Prime Minister giving speech on the Iraq war. In order to support such kind of access, it is important to classify the shot units into appropriate categories, such as speech shot, anchor shot, etc.

However, for news video, users usually remember video contents in terms of events or stories but not in terms of changes in visual appearances as in shots. It is thus necessary to organize video contents in terms of small, single-story unit that represents the conceptual chunks in users' memory. Moreover, the stories can be summarized in different scales to support users' query such as "give me a summary on sport news", etc. Thus, the story units serve as the basic units for news video organization. Finally, these story units with their classified shots can be stored in the database to support news retrieval task. A scenario for news video organization and retrieval is illustrated in Figure 1.1.

The problem of segmenting news video into story units is challenging, especially when there is no supplementary text transcript. Story segmentation based on text transcript is easier and less expensive than the segmentation performed on news video using audio-visual based features. There are several techniques to perform text segmentation on news transcript. Most techniques are statistical-based designed to find coherent body of text terms that represents a story or topic. The story boundary therefore occurs at a position where there is least coherent or similarity between adjacent text units. Based on this principle, one successful technique is the tiling technique reported in [Hearst 1994]. However, the maximum accuracy reported for story segmentation based on news transcripts of CNN and ABC news used in

TRECVID 2003 evaluations [TRECVID 2003] was only about 62%. A similar level of performance was reported in [Allan 1998] for text-based topic detection and tracking (TDT) task. The reason for this low-level of performance is because statistics of text alone is insufficient to capture the rich set of semantic clues and presentation features used to signify the end of stories in news video. Thus, there is a need to look into audio-visual features of news video to assist in story segmentation.

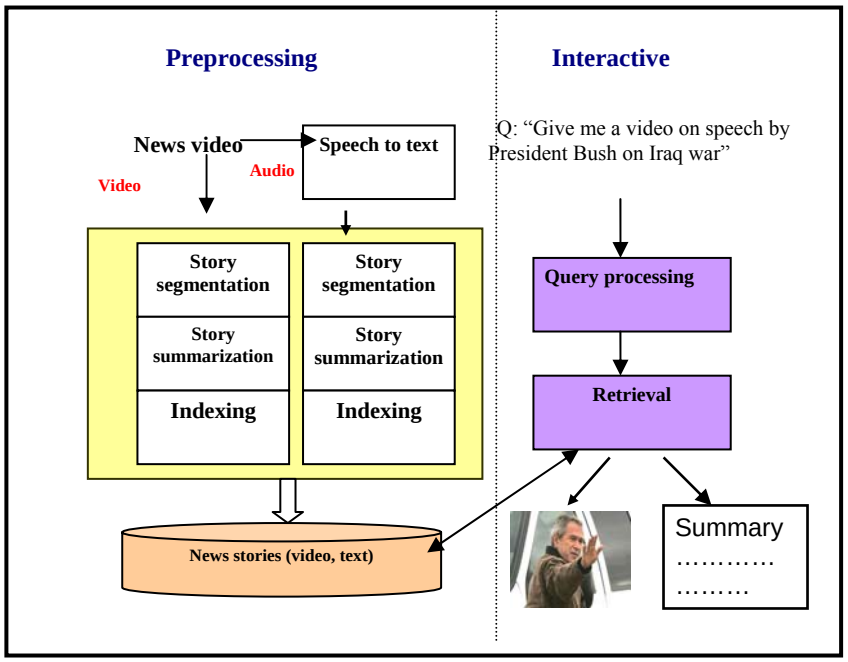


Figure 1.1: A scenario of news video organization

Several reported works [Connor 2001][Wu 2003] focused on capturing anchor shots as the basis to determine the begin/end of stories. The approach works well for news video with simple and little variation in structure in which a new news story always starts with the anchor shot. From the results in TRECVID 2003, such techniques could achieve an accuracy of about 54%. Now, consider the CNN news (Refer to Appendix B for the details of the web site of CNN) , their news reporting structures

are more complex and exhibit great variation in the various programs screened during the news broadcast as shown in Figure 1.2. We can see from the Figure that a news story may begin with: (a) an anchor shot such as types s1, s2, s3, and s7; (b) a program logo shot such as type s5; (c) none of the above at all such as type s4 and s6. As for the stories that begin with anchor shot, the usual type is type s1 in which a story starts with an anchor shot and ends before the next anchor shot. However, it is possible that the reporter is reporting continuous news stories within a studio (type s2) without any other shots or reporting multiple stories with live-reporting or outdoor shots but with no obvious clues for story transition (type s3). Therefore, to tackle the problem efficiently, we need to look more than just at anchor shots but also pay attention to all other program structure within a news broadcast.

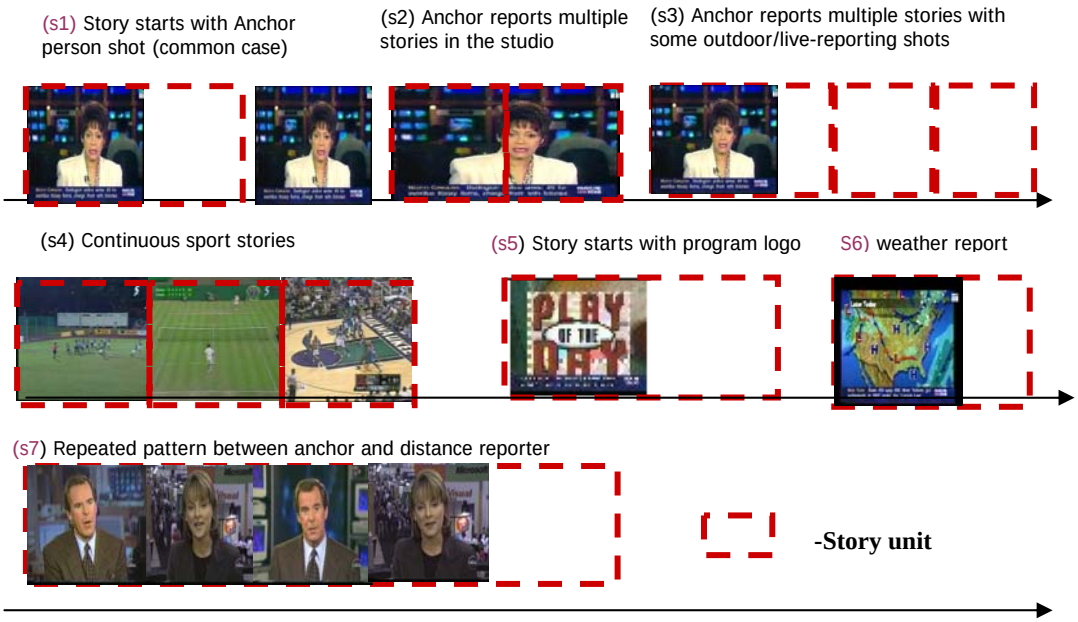


Figure 1.2: News story types found in CNN news broadcast

1.2 Our Approach

This research aims at developing a system that can automatically and effectively segment news video into story units. Our aim is to investigate the choice of features that are important for story segmentation and the selection of statistical approach that best suits the news structures and patterns. For comparison, we propose two learning-based frameworks for news story segmentation based on: a) *Hidden Markov Models* [Rabiner and Juang 1993]; and b) *Rule-induction* approach based on GRID system [Xiao 2003]. It is well known that the learning-based approaches are sensitive to feature selection and often suffers from data sparseness problems due to the difficulties in obtaining sufficient amount of annotated data for training. One approach to tackle the data sparseness problem is to perform the analysis at multiple levels as is done successfully in natural language processing (NLP) research [Dale 2000]. For example, in NLP, it has been found to be effective to perform the part-of-speech tagging at the word level, before the phrase or sentence analysis at the higher level. In this research, the video is analyzed at the shot and story levels using a variety of features.

At the shot level, we use a set of low-level, temporal, and high-level features to model the contents of each shot. Next, we classify the shots into meaningful categories. In our study, there are 13 shot categories that are common to most of the news video. There are: *Intro/Highlight, Anchor, 2Anchor, People, Speech/Interview, Live-reporting, Still-image, Sports, Text-scene, Special, Finance, Weather,* and *Commercials*. In order to cover the data provided by TRECVID [TRECVID 2003],

we also introduce *"LEDS*" (to represent lead-in/out shots), *"TOP"* (top story logo shot), "*PLAY*" (for play of the day logo shot), "*SPORT"* (to capture sport logo shots), and "*HEALTH*" (to represent health logo shots). From these categories, we divided them into three main clusters. They are *visual-based*, *heuristic-based* and *learning-based* clusters. The grouping of each cluster is determined by the characteristics and the method to be used for shot classification. For example, the *visual-based* cluster includes shot categories such as *Weather, Finance, LEDS, TOP*, etc. These categories of shots are visually similar within each broadcast station. Thus, they can best be represented using color histograms of key frames and identified using image similarity matching techniques. The *heuristic-based* cluster contains shots of commercial category. Most countries require the broadcast stations to put some blank frames preceding and/or after the commercials. Also most companies try to pack as much information about their advertising products as possible into short commercial, thus the cut rate of shots within a commercial is much higher than that of other news reports. We thus employed heuristic techniques to identify this shot category. Finally, shots in *learning-based* cluster are those that cannot be described using any structures. Here we use machine learning technique such as the Decision Tree to classify such shot categories. Although, the number of categories may vary slightly when applying to other news corpora, the three clusters of categories can be applied to general news video.

At the story level, we use the shot category information (represented by unique Tag-ID), together with temporal and high-level features within a learning framework to identify news story boundaries.

In order to demonstrate that our 2-level framework is effective, we employ two learning-based approaches at the second level to perform story segmentation. They are the HMM approach and the rule-induction approach based on GRID system [Xiao 2003]. The main idea of the GRID-based rule induction approach is to use global occurrence statistics of each of the features of the current and neighbor shots around the story boundaries to extract rules. We found that, this approach, although simple, gives effective results.

1.3 Motivation

The motivations of this research are:

- To investigate structures of news programs from various TV stations and define a general news structure for further analysis in story segmentation.
- To investigate and select essential features for story segmentation. Our aim is to select key features that can be automatically extracted from MPEG video using the existing tools.
- To define and classify the video shots into meaningful categories. The objectives for doing this are: **a)** to support further browsing and retrieval; and **b)** to facilitate story segmentation process.
- To develop an automated system to segment news video into stories and classify these stories into semantic units while considering the data sparseness problem.

1.4 Main Contributions

The main contributions in this research are:

- ❖ We have designed and developed a two-level multimodal framework for story segmentation in news video.
    - At the first level, we defined shot categories and their characteristic that cover all categories of shot in general news video. We employ a hybrid approach including specific detectors and machine learning techniques to perform shot classification
    - At the second level, we employ different machine learning approaches, including HMM and rule-induction technique to perform story segmentation
- ❖ We demonstrate the effectiveness of our framework on a large scale data provided by TRECVID 2003 using the two machine-learning techniques. The data contains about 120 hours of CNN and ABC news video of year 1998. The evaluations show that we could achieve an accuracy of about 77.5% in F_1 measure when using full set of features in the HMM framework. Our system is one of the best performing systems from TRECVID 2003 evaluations. For rule-induction approach, we achieve an accuracy of about 75% in F_1 measure. Thus, we have demonstrated that our 2-level framework incorporating different machine learning techniques is effective for news story segmentation problem.

1.5 Thesis Organization

The rest of the thesis is organized as follows. Chapter 2 gives background of video segmentation and video structure, news structure, definition of news story, and related work on story segmentation, shot classification and detection of transition boundaries. Chapter 3 presents a design of our multi-modal two-level framework. Chapter 4 discusses details of the selection and extraction of features as well as the selection of shot categories while Chapter 5 describes the classification of shots. Chapter 6 gives details of our Hidden Markov Models (HMM) framework and the evaluation results on small scale test (on local news video) and large scale tests (on TRECVID 2003 data). Chapter 7 discusses details of Global Rule Induction (GRID) technique together with the experimental results on TRECVID 2003 data. Finally, we conclude our work in Chapter 8.

CHAPTER 2

BACKGROUND AND RELATED WORK

2.1 News Story Segmentation

This section describes the background for news story segmentation. We first need to segment an input news video into basic visually contiguous units called shots. Next, we try to structure the shots that comprise a news story. A general news structure and a definition for a news story are also given. Finally, related work on story segmentation and video classification are discussed.

2.1.1 Shot Segmentation and key frame extraction

In order to perform story segmentation in news video, we need to segment the input news video into shots. A shot is a continuous group of frames that the camera takes at a physical location. A semantic scene is defined as a collection of shots that are consistent with respect to a certain semantic theme (for example several shots taken at the beach). Figure 2.1 illustrates the structure of frames, shots, scenes, and video sequence.

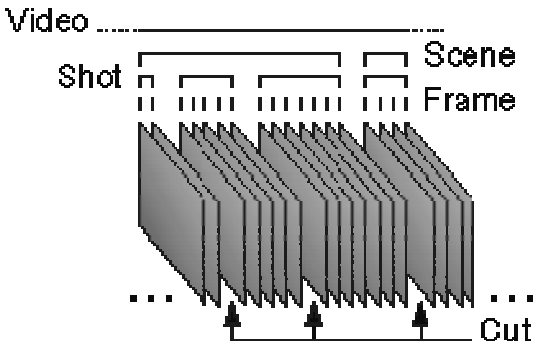


Figure 2.1: The structure of video frames, shots, scenes, and video sequence

Effective techniques for detecting abrupt changes or hard cuts are reported in [TRECVID 2003] and [TRECVID2004]. The best accuracy they could achieve is more than 90%. In CNN and ABC news video used in TRECVID 2003 and TRECVID 2004, more than 60% of the total shots used in shot detection task are hard cuts and more than 20 % are gradual transitions.

Gradual transition is frequently used for editing technique to connect two shots together and can be classified into three common types: fade in/out, dissolve, and wipe. Fade-in is a shot, which begins in total darkness and gradually lightens up to full brightness of a scene; and fade out is the opposite. Dissolve is a gradual change from one scene into another scene, in which one gradually decreasing in intensity (fade out), the other gradually increasing (fade in) at the same time and rate. Lastly, wipe shows the new scene appearing behind the line which moves across the screen. Figure 2.2 presents examples of cut and gradual transition of type dissolve.

Cut

Dissolve

Figure 2.2: Examples of cut and gradual transition.

After the video is decomposed into shots, there are several ways in which the contents of each shot can be modeled. We can model the contents of the shot: (a) using a representative key frame; (b) as feature trajectories; or, (c) using a combination of both. In this research, we adopt the hybrid approach as a compromise to achieve both efficiency and effectiveness. Most visual content features will be extracted from the key frame while motion and audio features will be extracted from the temporal contents of the shots. This is reasonable as we expect the visual contents of shots to be relatively similar so that a key frame is a reasonable representation. Although sophisticated techniques are suitable to select one or more key frames for a shot (see for example [Anantharamanu 2002]), here we simply select the I-frame that is nearest to the center of the shot as the key frame.

2.1.2 News Structure

Most news videos have rather similar and well-defined structures. The news video typically begins with several Intro/Highlight shots that give a brief introduction of the upcoming news to be reported. The main body of news contains a series of stories organized in term of different geographical interests (such as international, regional and local) and in broad categories of social political, business, sports and entertainment. Each news story (though not always true) normally begins with Anchor-person shot. Most news broadcasts end with reports on Sports, Finance, and/or Weather. In a typical half an hour news, there will be at least one period of

commercials, covering both commercial products and self-advertisement by the broadcast station. Figure 2.3 illustrates the structure of a typical news video.

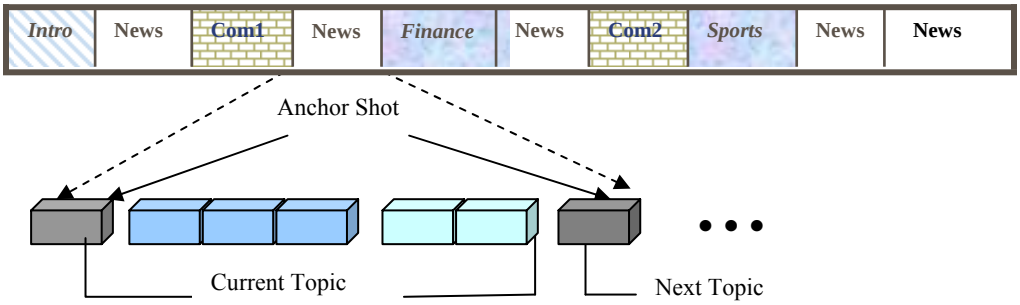


Figure 2.3: The structure of a typical news video.

Although the ordering of news items may differ slightly from broadcast station to station, they all have similar structure and news categories. In order to project the identity of a broadcast station, the visual contents of each news category, like the anchor person shots, finance and weather reporting etc., tends to be highly similar within a station, but differs from that of the other broadcast stations. Hence, it is possible to adopt a learning-based approach to train a system to recognize the contents of each category within each broadcast station.

2.1.3 News Story Definition and the Segmentation problems

2.1.3.a Definition of News Story

In this research, we follow the definition as in the guidelines in TDT-2 (phase 2 of Topic Detection and Tracking (TDT)) project. TDT is a multi-site research project under the Linguistics and Data Consortium (LDC), which was founded in 1992 in the University of Pennsylvania with a grant from the Advanced Research Projects

Agency (ARPA). It is an open consortium of universities, companies and government research laboratories. It creates, collects and distributes speech and text databases, lexicons, and other resources for research and development purposes. More information about LDC can be found on LDC home page [LDC 1992]. The TDT project, now in its third phase, aims to develop core technologies for news understanding systems. Specifically, TDT systems discover the topical structure in un-segmented streams of news reporting as it appears across multiple media and in different languages. For a detailed discussion of the goals of TDT, see [Wayne 1998]. The TDT-2 project addresses multiple sources of information in the form of both text and speech from newswire and radio and television news broadcast programs.

The TDT-2 guidelines were used as the guide for news story segmentation task in TRECVID 2003 evaluations [TRECVID 2003]. In the guidelines, a "news" story is defined as a segment of a news broadcast with a coherent news focus which contains at least two independent, declarative clauses. The rest of coherent segments are labeled as "misc" (miscellaneous). These "misc" stories cover a mixture of footages, including commercials, lead-ins, reporter chit-chats etc. Further details of the guidelines can be found in [TRECVID 2003].

2.1.3.b Problems in News Story Segmentation

As we can see, the definition is defined on text document (news transcript), how can we associate this definition to stories in news video is an important issue. From the structure of news video in Figure 2.3 and the structure of general video in Figure 2.1, a story unit may consist of several scenes. These scenes may not be visually similar to

one another. Thus, the problem of news story segmentation cannot be solved by just looking at the visual contents of video. As a result, story segmentation in news video is a hard problem especially when an input news video comes without transcripts. Because of this, most related works proposed solutions to this problem assuming that news transcripts are available [Merlino 1997]. In this case, we can make the problem simpler by considering the common words or phrase before the news begins, ends, change of topics, switching of person, etc. Each broadcast station has its own pattern word strings to indicate the transitions. For example, in CNN news, there are phrases such as "Good evening/morning, I am <person name> from CNN headlines news" appearing at the beginning before the actual news is being reported. Another example is "weather forecast is next" at the end of the story before the "weather" news report, etc. Thus, locating the transition is the task of locating and matching string patterns. We call such string patterns cue-phrases.

However, not all-individual news has consistent cue-phrases indicating the beginning of the next topic. This is why we cannot achieve high accuracy while only using this feature from the news transcripts. Moreover, from the reported results in [TRECVID 2003], the best performance that we could achieve when using only the features from news transcript is about 62%. This is because the state-of-the-art techniques for topic segmentation seem to under segment the news stories based on text alone [Hearst 1994].

On the other hand, segmenting news stories based only on audio-visual (AV) is an even harder problem. We need a system that can understand story units as semantic

units. The problem then is what types of AV features can identify boundaries of these units.

2.2 Relevant Research

Many works on extracting story units from multimedia documents have been published in recently years. Early work was reported in [Yeung 1996] in which they focused on movies. Others investigated story segmentation for documentary video [Slaney 2001], while some on news video [Merlino 1997] [Hauptmann and Witbrock 1998][Hsu and Chang 2003]. As for news, most of the works performed story segmentation based on news transcripts [Hauptmann 1997] [Merlino 1996] on assumptions that the transcripts were available. However, in actual cases, the transcripts are not always available for all news broadcasts. To give overall view of story segmentation task either when transcript is available or not available (use only video and audio streams), we will discuss related work that performed story segmentation based on text, AV features and both. Furthermore, we will also introduce relevant works that are related to part of our research, namely, video classification and detection of video transition.

2.2.1 Related Work on Story Segmentation

2.2.1.a Text Segmentation Approach

[Hearst 1994] introduced the use of text tiles to segment paragraphs in text documents by topic. Text tiles are adjacent regions that can be separated through automatically detected topic changes. The main concept of using text tile is, for a given window size, each pair of adjacent blocks of text can be compared according to how similar they are lexically. The method assumes that the more similar the two blocks of text are, the more likely that they belong to the same subtopic. Conversely, if two adjacent blocks are dissimilar, this implies a change in topic. The topic boundaries are determined by changes in the sequence of similarity scores. This method is preliminary designed for topic segmentation on news transcript. It works well on the data reported in [Hearst 1994]. However, it tends to under segment the large news transcript data set provided by the TRECVID. Moreover, the story boundaries found by the algorithm tend to be off by a few sentences from the actual boundaries. This is not likely to be acceptable as the boundaries found must be within 5 seconds of actual boundaries allowable by TRECVID.

2.2.1.b Shot Clustering Approach using Color Histogram

[Yeung 1996] introduced scene transition graph (STG) to detect story units in video based on similarity of shot and time-constrained clustering. The partition is found by looking for cut edges in STG. They used color histogram to group shots that are visually similar and temporary closed into clusters. In addition, different clusters should have sufficient difference in characteristics. They defined shot similarity

distance as a number of frames between two shots that are close to each other. The temporal distance is expressed as: $d(S_i, S_j)$ = min (|bj –ei|, |bi – ej|, i ≠ j); and $d(Si, Sj)$ = 0 for i=j. If $d(S_i, S_j)$ is less than a threshold, T (the number of frames), then the two shots are grouped into the same cluster. This work inspires many other works in the area of story segmentation. The system is simple and works well on the data reported. However, a news story unit normally comprises several scenes that might be dissimilar. The system thus may detect two adjacent dissimilar scenes that belong to one story as two separate stories.

2.2.1.c Hybrid approach using Multi-modal Features

In this approach, multiple techniques are used to handle feature extraction and segmentation in each of the available sources such as text (news transcripts), audio, and video stream.

[Merlino 1997] introduced a system called Broadcast News Editor (BNE). BNE captures, analyzes, annotates, segments, summarizes, and stores broadcast news. They used CNN prime news programs from 12/14/96 – 1/13/96 as the test data. Though they used all the data sources, they focused mostly on the use of features from news transcript such as hand-off phrases from anchor to reporter and reporter back to anchor, cue phrases that are likely to occur at the beginning of new stories, speaker change markers (“>>”), topic change markers (“>>>”), and blank lines to determine story boundaries. For video, they performed anchor, black frames, and logo detections. As for audio, they identify sufficiently long silence segments as the beginning and end of commercials. They presented state transitions such as the “start

of story", "advertising" states, etc. using finite state automaton (FSA). Some of the states that they defined are: Start of broadcast; Start of Highlight; End of Highlights; Start of Story; Advertising; and End of Broadcast. Their reported a performance of 74% and 97% for precision and recall respectively. Their system however was not tested on the TRECVID data hence it cannot be directly compared to other recent systems. Further drawback of their system is that it relies heavily on news transcript and draws little cues from video and audio. This is unlikely to be satisfactory when news transcripts are not available.

[Hauptmann and Witbrock 1998] reported research done as part of the Informedia Digital Library Project first introduced by [Wactlar 1996]. The main success of the Informedia project is based on the assumption that they can obtain sufficiently accurate speech recognition outputs from news broadcast for use in information retrieval. [Hauptmann and Witbrock 1998] detected and used black frame to separate commercial blocks from news stories and utilized frame similarity based on color histogram to identify anchor shot based on the assumption that anchor shots would reappear at regular intervals throughout a news program. Each appearance of anchor shot was claimed to denote a segment boundary of some type. In this work, optical flow for motion estimation such as camera motion and object motion was computed and used to examine story boundaries. This is based on their assumption that scenes containing movements may be less likely to occur at story boundaries. For audio track extracted from the news video, they performed speech recognition using their own Sphinx-II system, which extracts text from speech in the audio track. They then used closed-captioned transcripts to correct the possible errors in the output from Sphinx-

II. Also, from close captions, they obtained the marker for changes between advertisements and news story as well as markers for speaker change (using ">>" markers) and topic change (using ">>>" marker). They used anchor shots, motion, and the markers as cues to detect the change in news topic. Their system seems to be very effective. However, the system relies on the corresponding news transcript to correct the audio recognition output, and if the transcript is not available, the system performance will definitely be affected.

[Slaney 2001] introduced a method to detect edges in multimedia documents in which they used two videos, one audio (music) and one text document as the test data. They used color, text, and audio signals as the features and employed singular-value decomposition (SVD) to reduce the dimensionalities of the feature space. They adopted the concept of scale-space technique in the edge finding process. The scale-space contained color space, word (from text documents) space, and acoustic space. By applying scale-space segmentation, they tried to detect the edges in all the signals that correspond to large changes. Their system gives an intuitive and reasonable idea to video segmentation as they looked for large changes in all signals at different scales.

[Hsu and Chang 2003] employed a statistical framework called *exponential* or *maximum entropy* model to select the most significant features of various types. The features they used are *Acoustics* (whether the next shot is dominated by any audio types), *Speaker identification* (to identify anchor speech shot), *Face, Superimposed text captions*, *Motion, A/V combination* (used a combination of face and speech), and *Cue phrase* (whether there is a presence of cue phrase in the segment). They

considered the changes of the features from the previous to the current shots. They used Kullbak Leibler divergence measure in optimization procedure to estimate the model parameters. The exponential model constructs an exponential, log-linear function that fuses multiple features to approximate the posterior probability of an event i.e. story boundary, given the audio-visual data surrounding the point under examination. The construction process contains two main parts: parameter estimation; and feature induction. Finally, they employed dynamic programming approach to estimate possible story transition. They tested on 3.5 hours of Mandarin news in Taiwan. The total data contains 100 news stories and achieved the maximum accuracy of 90% when using the full set of features. When tested their system on the TRECVID data using the full feature set, they could achieve an accuracy (as reported at TRECVID 2003) of about 69%. Their work is the most similar to our work as we first proposed in [Chaisorn 2002]. The main differences are: (a) for the similar subset of selected features, they looked at the changes of feature values, for instance from low motion to high motion, between the previous and current shots, rather than looking only at current shot contents itself; and (b) we divided our framework into two levels (like the approach used in NLP), shot and story levels whereas they employed a single-level framework; and (c) they performed story segmentation using maximum entropy and dynamic programming techniques while we used HMM framework and rule-induction approaches.

[Greiff 2001] from MITRE Corporation performed story segmentation based on news transcripts. They employed HMM to model the generation of word production during news program. Their investigation was done in two respects: 1) to exploit the

differences in feature patterns that are likely to be observed at different points in the development of news story; and 2) to derive a more detailed modeling of the story-length distribution profile, unique to each new source. They modeled the generation of news stories as a 251-state Hidden Markov Model. From the news transcripts, they extracted 3 features: (a) *coherence* feature of text (based on *N* words immediately prior to the current word. If the current word does not appear in the buffer then its coherence value is 0, otherwise the value is calculated based on some log value, *N* = 50, 100, 150, and 200); (b) the duration of un-transcribed section; and (c) the trigger (cue) words. The system was tested on 15 ABC news video from TDT-2 corpus. The probabilities of false-alarm and missed boundaries were reported to be 0.11 and 0.14 respectively.

Appendix B lists the details of the web sites for ABC news and other broadcasters used in this study and in related work.

2.2.2 Related Work on Video Classification

Another area of research that is related to story segmentation and organization is video classification. It is a hot topic of research for many years and much interesting research has been done. Because of the difficulty and often subjective nature of video classification, most early works examined only certain aspects of video classification in a structured domain such as sports or news. As video classification is not the main emphasize of news video segmentation, we will give a brief review of related work on this topic.

2.2.2.a Statistical Approach

[Wang 1997] employed mainly audio characteristics and motion as the features to classify the TV programs into the categories of news report, weather forecast, commercials and football games. For audio features, they employed mean and standard deviation of volume distribution, silence interval distribution, spectrogram, central frequency and bandwidth. As for the motion, for each frame, they computed histogram of motion vector field, spatial correlation of motion vector field and phase correlation function (CPF). Their analysis is based mainly on the average and standard deviation values of each of the features. This is based on their observations that different TV programs tend to have different audio characteristics and motion levels. For example, weather news and normal news reports have similar audio characteristics. They have smaller standard deviation in volume and silence intervals. On the other hand, in TV commercials, a speech is delivered very quickly. In addition, the silence ratio and mean silence interval is small. Further details on the analysis can be found in [Wang 1997].

2.2.2.b HMM Approach

[Wei 2000] used video text and faces as the features and employed the HMM framework to classify the video clips into the classes of commercials, news, sitcom, and soap. They achieved the accuracy of over 80% on short video clips. In their work, they extracted the trajectories of the video and construct hierarchical information consisting of three layers: 1) *Video layer* that contained the general information such as the number of face trajectories, their average duration, and cut rate, etc.; 2) *Trajectory layer* that was related to each unique face or text trajectory in the video

clip such as their duration, and movement type, etc.; and 3) *Model layer* that explains the face trajectory, which is a series of face models in a sequence of successive frames. One model corresponds to a single face or text detected in the frame. It includes the color, location, and size information. Each frame of the input video is represented by one of the 15 types (Anchor person text, face-text, Wide close up, Close shot, Many-face, Two-face, Medium close face, Many-text-line, Few-text-line, One-text-line, Uniform frame, Shot-start frame, Face-only, No-face-text and Undefined), with one symbol per type. These 15 symbols are then used in their HMM framework. Further details of their work can be found in [Wei 2000].

[Eickeler 1997] considered 6 features, derived from the color histogram and motion variations across the frames, and employed HMM to classify the video sequence into the classes of Studio Speaker, Report, Weather Forecast, Begin, End, and the editing effect classes. They achieved more than 90% for the classes of Studio Speaker and Report, about 40% for editing effects and more than 80% for the rest of the classes.

2.2.2.c Rule-Based Approach

[Chen and Wong 2001] employed a rule-based approach to classify an input video into five classes of news, weather reporting, commercials, basketball, and football. They used the feature set of motion, color, text caption, and cut rate in the analysis. For each individual frame of the input video, the system classified the frame into one of the five classes. They employed CLIPS 6.5 rule-based programming language to extract rules. An example of the rule generated is: *low-motion-magnitude & low-*

colorfulness & *high P-MPC* (Percentage-of-Most Prominent color) => *news*. More details of their work and the generated rules can be found in [Chen and Wong 2001].

2.2.2.d Hybrid Approach

[Ide 1998] tackled the problem of news video classification and used videotext, motion and face as the features. They first segmented the video into shots, and used a hybrid of heuristic approach and clustering technique to classify each shot into one of the five classes of: Speech/report, Anchor, Walking, Gathering, and Computer Graphics categories. These five classes, as reported in their work, covered 57% of the news video used for experiment. Their classification technique seems effective for this restricted class of problems.

2.2.3 Related work on Detection of Transition Boundaries

Another category of techniques incorporated information within and between video segments to determine class transition boundaries using HMM approach. One such work [Alatan 2001] focused on entertainment type videos rather than news video. They aimed to detect dialog and its transitions. They modeled the shots using the features of audio (music/silence/speech), face and location changed, and used HMM to locate the transition boundary between the classes of Establishing, Dialogue, Transition, and Non-dialogue.

Identifying news story boundaries is the task of detecting a change of news topic. The identification can be achieved by detecting a change from some shot category to a particular type of shot category, for example changing from live-reporting shot to

leads-in shot (as found in CNN news video). Thus, it is possible to apply their technique to detect story transition in news video.

2.3 Summary

Related studies on shot classification have demonstrated the effectiveness of their techniques. For example, for the detection of anchor shots in news video, most techniques could achieve quite high accuracy. However, detecting shots such as "Bill Clinton" or "physical violence" is more difficult. The best accuracies as reported in [TRECVID 2003] and [TRECVID 2004] are about 23% for detecting "Bill Clinton" shots and less than 10% for detecting "physical violence" shots. Thus, we can see from the results that there is a need for further research to find a better solution to tackle these problems.

Most related work on story segmentation employed machine-learning based approach in a single-level framework. As we know the machine-learning based approaches tend to suffer from data sparseness problem, thus most systems cannot achieve high accuracy. One way to alleviate the data sparseness problem is to adopt a multi-level learning framework, in which the problem is divided into sub-problems. This is similar to the approach taken in NLP (Natural Language Processing) research in which they perform part-of-speech tagging at the word level, following by other task such as noun-phrase extraction, parsing etc. at the sentence and higher level. Thus, it is reasonable to adopt a similar idea to design a multi-level framework for story segmentation in news video in a learning-based framework.

CHAPTER 3

THE DESIGN OF OUR SYSTEM FRAMEWORK

This Chapter discusses the design of the proposed approach. It presents an overview of the two-level system components includes shot segmentation, feature extraction and the main processes: shot classification (I) and story segmentation (II), as shown in Figure 3.1.

3.1 System Components

Although news video is structured, it presents great challenges in identifying story boundaries. The stories obtained can then be further classified into semantic classes such as “news story” or “miscellaneous story” as defined in TRECVID. In this research, the framework for story segmentation was designed and scaled to cope with large news video corpus such as the test data provided by TRECVID. It is composed of two levels: the shot level that classifies the input video shots into one of the predefined categories using a hybrid of heuristic and learning based approaches; and story level that performs story segmentation using machine learning and statistical methods based on the output of shot level and other temporal features. The framework on story segmentation is similar to the idea of natural language processing

(NLP) research in performing part-of-speech tagging at the word level, and higher-level analysis at the phrase and sentence level.

Before we can design the framework, to tackle the problem effectively, we must address three basic issues. First, we need to identify the suitable units to perform the analysis. Second, we need to extract an appropriate set of features to model and distinguish different categories. Third, we need to adopt an appropriate technique to perform shot classification and identify the boundaries between stories. To achieve this, we adopt the following strategies as shown in Figure 3.1.

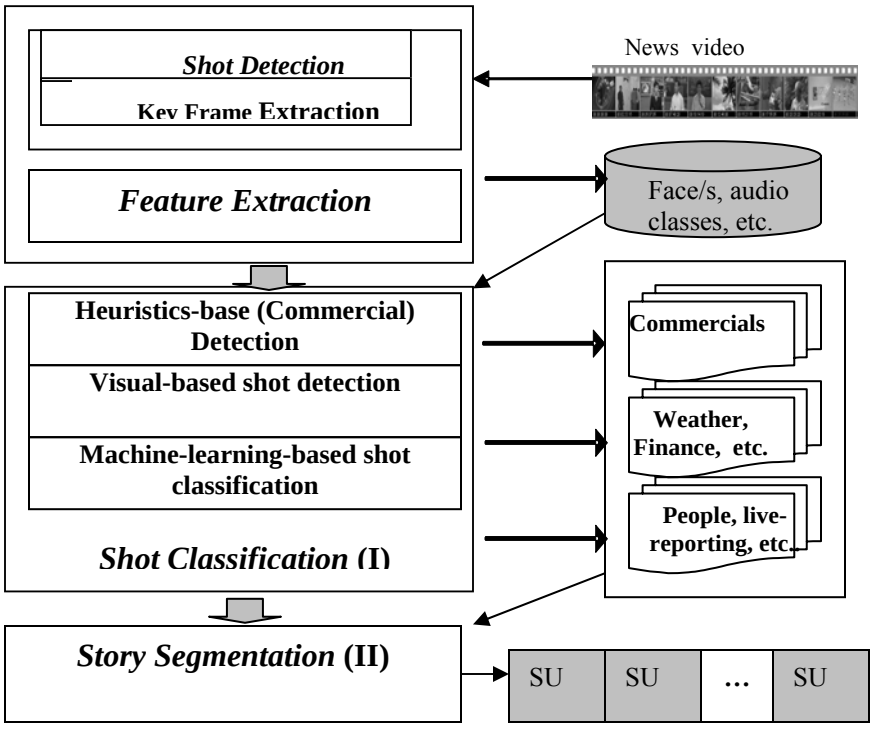


Figure 3.1: Overall system components. Note: SU is story unit.

a) We first segment the input video into shots using a mature technique. This process is called shot detection. In our preliminary experiments, we use the multi-

resolution technique developed in our lab [Lin 2000] and [Anantharamu 2002]. For the data provided by TRECVID, shot boundaries were given together with the videos. Thus we use these common shot boundaries for the test on TRECVID data so that our results can be compared to other participating systems based on these basic units.

b) We extract a suitable set of features to model the contents of shots. The features include low level visual and temporal features, and high-level features. We select only those features that can be automatically extracted in order to automate the entire classification process.

c) We employ a hybrid of specific detectors and a learning-based approach that uses the multi-modal features to classify the shots into the set of well-defined categories. This step is called shot classification (I) which involves three main sub processes: heuristic-based shot detection; visual-based shot detection; and machine-learning-based shot detection. This is based on the characteristics of the defined shot categories.

d) Finally, given a sequence of shots in respective subcategories, we use a combination of shot categories and temporal features to identify story boundaries. The first approach that we proposed for story segmentation is the HMM approach [Chaisorn 2002], which has also been used in the TRECVID evaluations. Although we can compare our system performance with other systems based on the results evaluated by TRECVID, all the systems used different features and frameworks. In order to find alternative technique to compare with our HMM framework based on

the same set of feature and the 2-level framework, we employ rule-induction technique to perform story segmentation. This is based on the observation that the output from HMM corresponds to some rule patterns. For example, the transition from shot category of any type to *Anchor* shot typically indicates story changes. Another example is the transitions from *Commercials* to lead-ins shots also provide indication of story changes. We can therefore see that we can employ technique call rule-induction, which is much simpler than HMM. The details of the two approaches will be discussed in Chapter 5 and 6.

The next chapter discusses the details of shot content analysis, the defined shot categories and the selection and extraction of features.

CHAPTER 4

SHOT CATEGORIES AND FEATURES

The purpose of this Chapter is two-fold. First it presents details of the selection of shot categories and their characteristics. It also briefs on shot segmentation and key frame extraction techniques. Second it presents details of the selection and extraction of features to support shot classification and story segmentation processes. Defining appropriate shot categories and the selection of prominent features is one of the contributions in this thesis.

4.1 The Analysis of Shot Contents

4.1.1 Shot Segmentation and Key Frame Extraction

The first step in news video analysis is to segment the input news video into shots. We employ the multi-resolution analysis technique developed in [Lin 2000] and [Anantharamu 2002] that can effectively locate both abrupt and gradual transition boundaries effectively.

After the video is segmented, there are several ways in which the contents of each shot can be modeled. We can model the contents of the shot: (a) using a representative key frame; (b) as feature trajectories [Chen and Chua 2001]; or, (c)

using a combination of both. In this research, we adopt the hybrid approach as a compromise to achieve both efficiency and effectiveness. Most visual content features will be extracted from the key frame while motion and audio features will be extracted from the temporal contents of the shots. This is reasonable as we expect the visual contents of the shots to be relatively similar so that a key frame is a reasonable representation. We select the I-frame that is nearest to the center of the shot as the key frame.

4.1.2 Shot Categories

4.1.2.a The Selection of Shot Categories

The next step is to determine an appropriate and complete set of categories to cover all shot types. The categories must be meaningful so that the category tag assigned to each shot is reflective of its content and facilitates the subsequent stage of segmenting and classifying news stories. We study the set of categories employed in related works, and the structures of news video. We arrive at the following set of shot categories. They are *Intro/Highlight, Anchor, 2Anchor, People, Speech/Interview, Live-reporting, Still-image, Sports, Text-scene, Special, Finance, Weather, and Commercial*. These 13 categories cover all essential types of shots in news video under study including CNN, ABC, and Singapore news. The coverage of each of the categories in news reports may vary from station to station. For example, each CNN or ABC news broadcast provided by TRECVID 2003 contains about 40% of

commercials, 26% of Live-reporting, 9.5% of Anchor and 2Anchor shots, and the remaining shot categories cover the rest of the shots.

In addition to these categories, in order to cover the data provided by TRECVID [TRECVID 2003], we also introduce a cluster of program logos including: *LEDS* (to represent lead-in/out shots); *TOP* (top story logo shot); *PLAY* (for play of the day logo shot), *SPORT* (to capture sport logo shots); and *HEALTH* (to represent health logo shots). Out of the above categories, some are quite specific such as the *Anchor* or *Speech* categories. Others are more general like the *Sports*, *Special or Live-reporting* categories.

However, some of the categories share common characteristics such as *Weather* and *Finance*, which are visually similar within a broadcast station. Thus, we can group those shots with common properties into the same cluster. In this study, news video shots are grouped into three clusters. The first cluster is the *Heuristic-based* cluster for *Commercial* shots. The detection of commercial shots was performed based on a combination of black frames (detected using color histogram), high cut rate and low confidence value in ASR (Automatic Speech Recognition) output.

The next cluster is the *Visual-similarity-based* cluster which is subdivided into two groups: a) visual-similarity within a broadcast station such as *Weather, Finance* and those program logos shots; and b) visual-similarity across news broadcast station such as*, Anchor* and *2Anchor*. For this cluster, we use 176-Luv-color-histogram of the key frame to represent each shot. Image similarity matching is employed to identify the shots in the first group while clustering technique is used to detect the categories in

the second group. This is based on the fact that *Anchor* and *2Anchor* shots within a news broadcast contain similar back ground throughout the whole sequence of video.

The last cluster is the *Machine-learning-based* (ML) cluster with such categories as *People*, *Live-reporting, Text-scene, etc.* shots. Shots within this cluster are classified using Decision Tree.

Figure 4.1 presents the three clusters of shot categories.

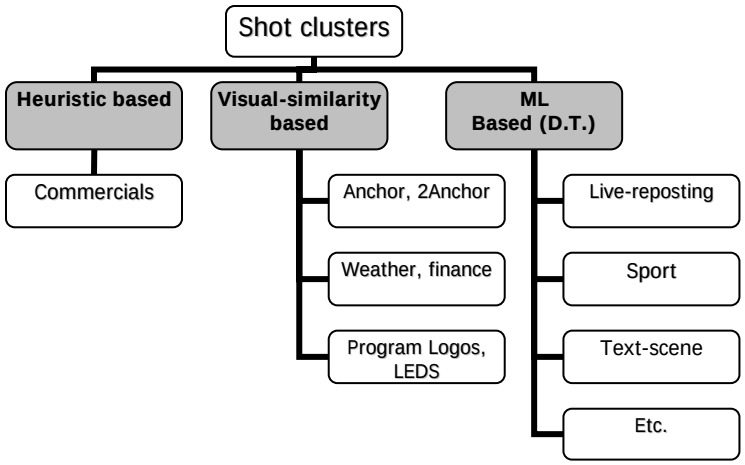


Figure 4.1: Clusters of the shot categories in this framework

It is found from the experiments (details will be discussed later in Chapter 6 on the result of story segmentation) that shot categories of visual-based cluster especially *Anchor* and *2Anchor* are the most significant categories in identifying story change. They contribute to over 70% of the total stories found by our system and the remaining categories of the same cluster contribute more than 10% while ML-based cluster contributes to about 16% of total stories found.

4.1.2.b Shot Categories and their Characteristics

- **Heuristic-based** cluster

- *Commercials.* Commercials are used to present messages or to sell products. Because it is expensive to air commercials especially during prime hours, the commercials tend to be short and packed with product-related information. Thus commercials typically contain fast changing shots, and end with still images showing the company's logos or products. In most countries, it is mandatory to air several black frames preceding or after a block of commercials [Koh and Chua 2000]. However, this is not always the case in many countries, like in Singapore. Our studies show that commercial boundaries can normally be characterized by the presence of black frames, still frames and/or audio silence [Koh and Chua 2000].

- **Visual-similarity-based** cluster

  - ***Finance*.** This type of shots is characterized by image content. Based on our video sample, Finance shots normally appear in the middle of the news. The visual contents of these shots, in terms of colors, position of text, music, etc., are almost the same within a broadcast station. Thus, we can effectively identify such shots simply by using the image similarity between an unseen shot (represented by its key frame) and the representative key frame of Finance shot. Example images of this category are shown in Figure 4.2.

  - ***Weather***. This type of shots is similar to Finance shots in such a way that the visual contents of these shots are very similar. Thus we can again use

image similarity measure to identify these shots. Examples of key frame of *Weather* shot are shown in Figure 4.2.

Finance Finance

Weather Weather

Figure 4.2: Examples of Finance and Weather categories

- ***Program logos*:** Special types of program logos found in CNN news video are ***LEDS, PLAY, HEALTH, SPORT,*** and ***TOP.*** Examples of the categories are shown in Figure 4.3.

LEDS **LEDS** **HEALTH**

TOP **SPORT** **PLAY**

Figure 4.3 Examples of program logos in CNN news video.

- ***Anchor.*** This is the most typical shot type in news reports with one anchor person appearing in fixed background. Thus, the shot is normally of closed up or medium distance shot containing one detected frontal face. Also, at the beginning of the anchor shots, there are usually one to two lines of text at the bottom stating the anchor person's name and title. The shot duration is usually long. Examples of anchor shots in CNN and ABC news videos from the TRECVID data are shown in Figure 4.4

CNN 1 (left) CNN 2 (center) CNN 3 (right)

ABC 1 (left) ABC 2 (center) ABC 3 (right)

Figure 4.4: Examples of anchor shots from CNN and ABC news video

- ***2Anchor.*** Shots in this category containing two anchor persons and are usually shown as part of transition from one topic to the next. They are usually medium shots with two detected (frontal) faces. The shot duration is usually short (switch topics). Examples of *2Anchor* shots are presented in Figure 4.5. From the Figure**,** we can see from the shots of ABC news that, the main anchor person is facing his back to the audiences, thus his face is not seen. In this case, sometimes the face detection system used in

this thesis can only detect only one face. Thus, this shot will likely be misclassified into *Anchor*.

CH5 **CNN** **ABC**

Figure 4.5: Examples of 2Anchor shots from CH5, CNN, and ABC news

- **Machine-learning-based (ML)** cluster

For the remaining categories, their characteristics are not obvious and they can best be discriminated using a machine learning technique. Here Decision Tree is employed to perform the classification. It is found that shots in this cluster are not only useful for story segmentation but also in news video retrieval. The assigned categories are used for further indexing and supporting interactive task such as a query on specific *Speech* or *Text-scene* shots on certain news story. For example, in order to support the query, "give me speech shots by Bush on Iraq war", the system: (a) first locates the story (or stories) of Iraq war; (b) locates Bush's name in the story (stories); and (c) tries to get the *Speech* shots around the region that Bush's name appears. However, this application is beyond the scope of this thesis. The details of each category in this cluster are given below. Figure 4.6 presents some examples of each category in this cluster.

  - *People.* We adopt the notion used in TRECVID 2003 which defined for a shot containing two or more faces as *People* shot. This type of shots usually contains at least two detected faces with similar sizes in a frame.

However, the accuracy is dependent very much on face detection algorithm. If the faces in the frame are not large enough or are not strictly frontal then some of these shots will be misclassified. Shot duration of this type usually varies from medium to long.

Figure 4.6 Examples of categories in the machine-learning based cluster

- *Speech/Interview/Reporting.* For simplicity without lost of generality, we will call this type as "*Speech*" shot**.** When a person is giving a speech, being interviewed, or when a reporter is reporting news from a relay spot, there is usually one person speaking in the middle of a frame. This type of shot is similar to *Anchor* shot, thus additional technique is required to solve the ambiguity. One technique is to perform clustering to identify the background of *Anchor* person shots since anchor person appears much

more frequently in a news broadcast with almost identical background but with different background from the person in the *Speech* shots.

- *Still image*. Usually, when there are one or more faces (or no face) detected with or without text captions and there is no motion in the shot; then it is possible that the shot comprises still images.

- *Intro/Highlight***.** Before the actual news is reported, several Intro/Highlight shots are introduced giving the introduction of the upcoming news to be reported. These shots contain speech with background music, and the shot duration is normally short. In the news under study, the duration of these shots is less than 10 seconds on average.

- *Sports.* In our training and testing samples, there are a few types of sport that are commonly reported such as, soccer (football), basketball, golf, and tennis. These shots normally contain background noise and high motion. In this work, we simply use these characteristics to classify sport shots as a general sport category. Additional techniques are required to further classify the shots into specific types of sport, such as soccer, basketball, etc.

- *Text-scene*. This type of shot is used to present the summary of events such as the results of a sport game, or the latest currency exchange rates etc. They typically contain only multiple lines of normally centralized text. We discover in TRECVID data that *Text-scene* appearing after a sport reporting usually indicates a transition to the next sport news report.

- *Special.* This is a special type of category for the news under study. It is used to present the typically light-hearted events. For example, in our sample, one such shot is about a dolphin giving birth to a baby. This category is hard to detect as its characteristic is similar to that of live reporting, except in some broadcast channel, they will present this report with music in the background. Under this study, this type is found in some CNN news and local news (Mediacorp, channel 5).

- *Live-Reporting.* Live-reporting shots typically involve the reporting of an event, follow by footages of the event. The event can be of any types and in any environment. Thus it is hard to detect such shots. In this research, we simply classify those shots that do not fit into any of the above categories as live-reporting shots.

4.1.2.c Relationship between Shot Categories and Story Units

In general, news video contains a series of stories organized in term of different geographical interests (such as international, regional and local) and in broad categories of social political, business/finance, entertainment, sports and weather. Each story may contain shots of several categories begins with an Anchor shot (~60% of news begins with *Anchor* shot), followed by several *People*, *Live-reporting*, and/or *Speech* shots.

Different news stories may contain similar combination of shot categories. However, the patterns, order, and frequencies of shot categories are different. For example, general news reporting, such as the world news, will contain mostly *Anchor* person

shots with some *People*, *Live-reporting* and a few *Speech* shots. On the other hand, the sports story will contain mostly the actual sports footages, possibly with some *Speech, Live-reporting* and *Text-scene* shots. Figure 4.7 illustrates the relationship between story and shot categories.

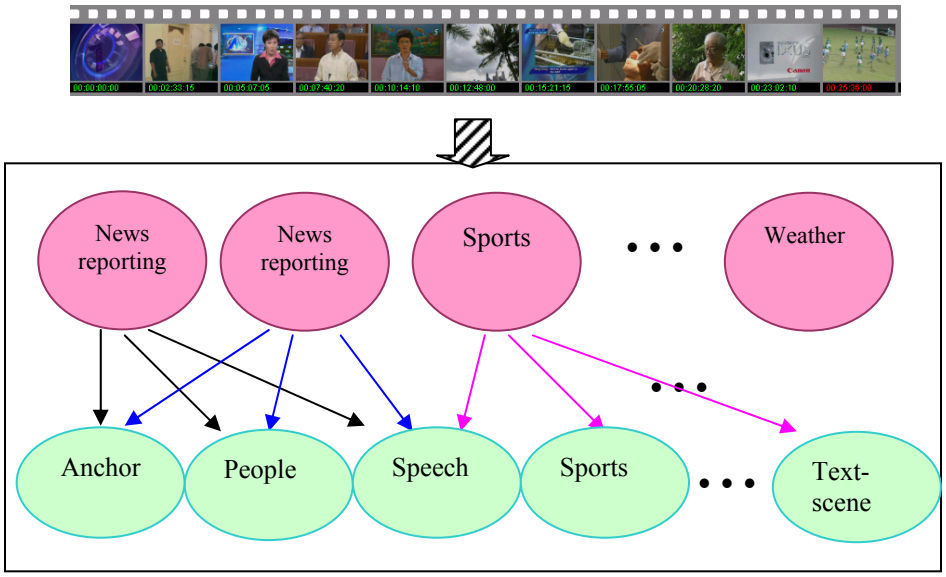


Figure 4.7: A relationship between shot categories and story units

4.2 Choice and Extraction of Features

The choice of suitable features is critical to the success of most learning-based classification systems. Here, we aim to derive a comprehensive set of features that can be automatically extracted from MPEG video to facilitate shot classification. Our focus is on using existing tools to extract the best set of features to solve our problem.

4.2.1 Low-level Visual Content Feature

Color Histogram: Color histogram models the visual composition of the shot. It is particularly useful to resolve two scenarios in shot classification. First, it can be used to identify those shot types with similar visual contents such as the weather and finance reporting. Second, the color histogram can be used to model the changes in background between successive shots, which provides important clues to determining a possible change in shot category or story. Here, we represent the content of key frames using a 176-Luv color histogram as expressed in [Chua and Chu 1998].

4.2.2 Temporal Features

Background scene change: Following the discussions on color histogram, we include the background scene change feature to measure the difference between the color histogram of the current and previous shots. By using clustering method and raising threshold value, we can derive the change feature. It is represented by 'c' if there is a change and 'u' otherwise.

The algorithm to detect background scene changed is applied to key frames extracted from each shot. Euclidean distance was employed to measure the similarity between shots (key frames). After similarities between any two shots of the video have been computed, shots with similarity values higher than the predefined-threshold are passed to clustering algorithm. Shots in the same cluster are considered to be in the same scene, if their distances span within a 50-shot window.

Speaker change: Similar to background scene change feature, this feature measures whether there is a change of speaker between the current and previous shot. It takes the value of 'u' for no change, and 'c' if there is a change. The later condition also applies to shots that do not contain speech but when there is a change from the previous speech to non-speech shot or vice versa. The change from non-speech shot to speech shot can be done on top of the audio classes obtained during audio classification (to be discussed later). However, for those consecutive shots containing audio track of different speakers, we need a speaker identification method to identify the change. Most well known method is based on the Gaussian model of each speaker. Given an unknown speech model, the system compares with the models in the database. Speaker-ID of the model that gives the minimum distance with the unknown speech is then assigned. In general, a person's voice carries unique audio features and different from one another. Thus, it is possible to identify a speaker provided the system has the speaker model in the database. Speaker change feature was used in our early work. However, we did not include this feature in the series of test done on the TRECVID data. The reason is because after we performed the testing on a subset of the training data, we found that this feature degraded our system performance.

***Audio*:** This feature is very important especially for *Sport* and *Intro/Highlight* shots. For *Sport* shots, its audio track includes both commentary and background noise, and for *Intro/Highlight* shots, all the narrative is accompanied by background music. Here, we adopt an algorithm similar to that discussed in [Lu 2001] to classify audio into the broad categories of speech, music, noise, speech and noise, speech and music,

or silence. They employed three sets of Support Vector Machine (SVM) to classify the audio classes. The first SVM is to separate speech from non-speech data. The second SVM is to discriminate pure-speech and non-pure speech on the speech class obtained from the first SVM. Finally, the third SVM is to differentiate background noise from music on non-speech data obtained from the first SVM. Audio features used in their technique include Mel-frequency ceptrum coefficients, Zero-crossing rate, Short time energy, Spectrum Flux, and noise frame ratio. Further details can be found in [Lu 2001]. They reported the accuracies of about 87% and 94% for pure-speech and non-pure speech respectively; and accuracies are about 82 and 92% for music and background sound (noise) respectively. Figure 4.8 shows a binary tree for the SVM classification while Figure 4.9 shows an example of the analysis of some of the audio features used in our work.

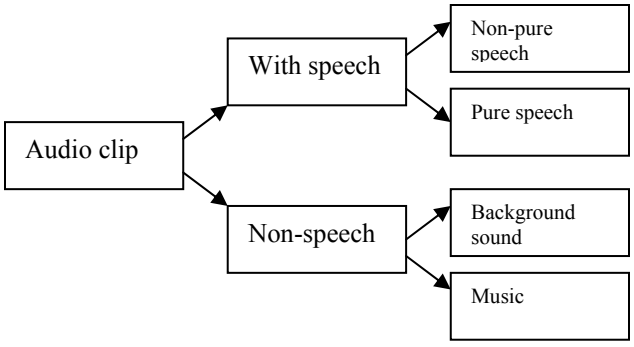


Figure 4.8: Binary tree for multi-class classification

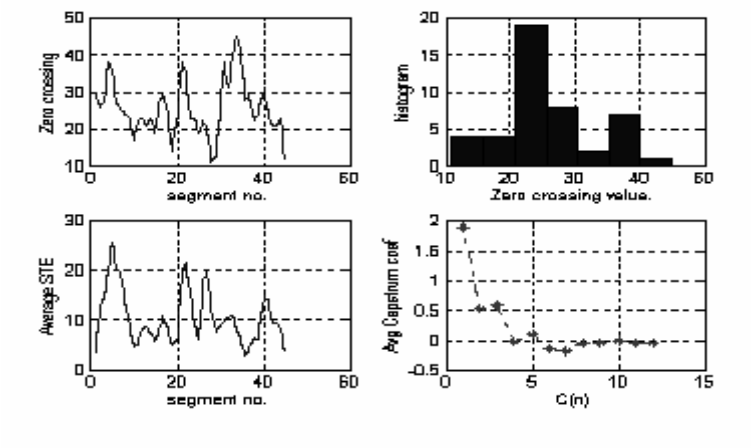


Figure 4.9: Example of the analysis of audio

Motion activity: For MPEG video, there is a direct encoding of motion vectors. Motion vector of current frame and reference frame are used to reconstruct the current frame. This feature can be used to indicate the level of motion activities within the shot. To compute motion activity, for each frame of a shot, the number of macro blocks that contains non-zero motion vectors is counted. This number can be denoted as NMC. Frames with high NMC means there are large number of macro block containing non-zero motion vectors and thus these frames are of high motion whereas frames of low NMC mean they are of low motion. Figure 4.10 illustrates macro block structure and motion vector in MPEG video. Further details of motion vector and MPEG-1 standard can be referred elsewhere or in [MPEG 1993].

We usually see high level of motion in sports and certain live reporting shots such as the rioting scenes. Thus, we classify the motion into *low* (like the *Speech* shots and some *People* shots where only the head regions have some movements), *medium* (such as riot shots), *high* (like in sport shots), or *no* motion (for still frame shots).

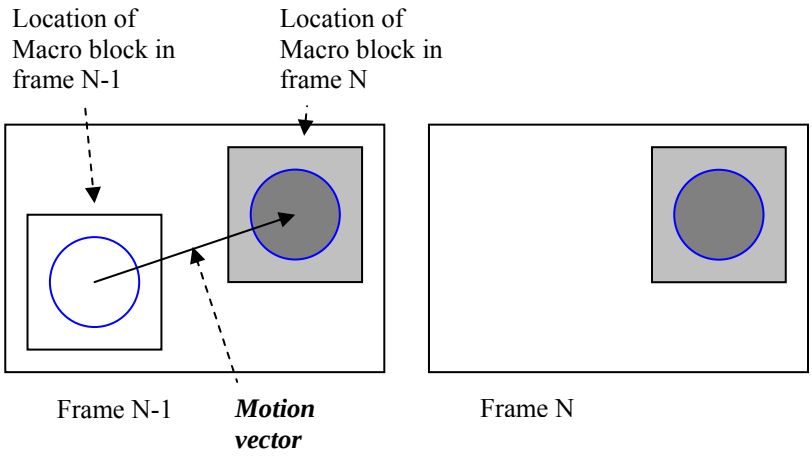


Figure 4.10: Illustrates macro block and motion vector in MPEG video.

Figure 4.11 presents an example of motion activity for a period of a thousand frames taken from two consecutive shots. The first shot is a riot shot (players walking) and the second is a *Speech* shot (one player is being interviewed). We can see that shot 1 is dominant by moderate motion activity while shot 2 is dominant by low motion activity. Here, it is not necessary to differentiate camera movements and object movements. From the observation, shots of high activity (either camera movement or object movement) usually occur in sport news or riot scenes.

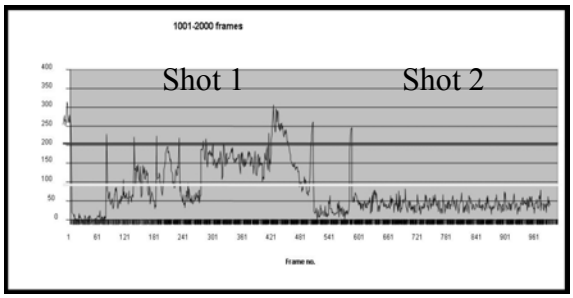


Figure 4.11: A graph of motion activity for a period of a thousand frames taken from sport shots.

In order to classify motion activity into low, medium and high, there are some important factors that need to be considered. Consider the example of the two shots given above. In normal case, we may classify the motion activity into high if its value is higher than the average value plus the standard deviation. However, it is not appropriate in this case. This is because within a shot, in particular, in sport shot, the motion activity may begin with low and then high in the middle and persist through the end of the shot. To obtain the motion activity for this type of shot, we need to find the duration of each type of motions during the shot, and classify the shot based on the motion type with the highest duration, or the most dominant motion activity type. If, we estimate the motion activity of this shot using the average motion value, then this shot will be wrongly classified as medium motion. Therefore, it is reasonable to represent each shot by its dominant motion activity which has the highest occurrence based on all frames of the shot. The formula used for computing motion activities can be expressed as:

$$MA_{avg} = \frac{1}{\sum_{i=1}^{M}\sum_{j=1}^{N_i} S_j} \sum_{i=1}^{M}\sum_{j=1}^{N_i}\sum_{k=1}^{S_j} x(i,j,k)$$

$$\sigma^2 = \frac{\sum_{i=1}^{M}\sum_{j=1}^{Ni} x(i,j,k) - MA_{avg})^2}{(M-1)(N-1)} \; ; \; N = \frac{1}{M}\sum_{i=1}^{M} N_i$$

$$MA_f = \begin{cases} \text{none} & \text{;when } x(i,j,k) = 0 \\ \text{high} & \text{;when } x(i,j,k) > MA_{avg} + \sigma \\ \text{low} & \text{;when } x(i,j,k) < MA_{avg} - \sigma \\ \text{medium; otherwise} \end{cases}$$

where $x(i, j, k)$ is the number of macro block containing motion vector for frame *k,* shot *j* and video clip *i*. S_j is number of frames in shot *j*. N_i is the number of shots in video clip *i*, and M is the total number of video clips to be processed. N is the average number of shots for each video clip. MA_{avg} is the average motion activity (in number of macro block) per frame. MA_f is frame motion activity level. Thus, after the motion activity for all frames in a shot have been classified, we can represent a sequence of the motion activities for the shot as a series of string. For example, a shot contains 10 frames with motion activities as: '*low low high high high high high high high medium*'. To get the motion activity level for this shot, we simply count the highest frequency motion, in this case 'high' has the highest frequency. Therefore, this shot is represented by "high" motion activity.

***Shot duration*:** For Anchor-person or Interview type of shots, the duration tends to range from 20 second to minutes. For other types of shots, such as the Live-reporting or Sports, the duration tends to be much shorter, ranging from a few seconds to about 10 seconds. The duration is thus an important feature to differentiate between these types of shots. We set the shot duration to *short* (if it is less than 10 seconds), *medium* (if it is between 10 to 20 seconds), and *long* (for shot greater than 20 seconds in duration). Shot duration, in symbolic representation, can be derived as follows:

$$\mathrm{S}_{\mathrm{avg}} = \frac{\sum_{i=1}^{M}\sum_{j=1}^{Ni} S_{ij}}{MXN};$$

$$\sigma^2 = \frac{\sum_{i=1}^{M}\sum_{i=1}^{Ni}(S_{ij} - S_{\mathrm{avg}})^2}{N}$$

$$N = \frac{1}{M}\sum_{i=1}^{M} N_i$$

$$D_{ij} = \begin{cases} \mathrm{l} & ;\text{when } \mathrm{S}_{ij} > \mathrm{S}_{\mathrm{avg}} + \sigma \\ \mathrm{s} & ;\text{when } \mathrm{S}_{ij} < \mathrm{S}_{\mathrm{avg}} - \sigma \\ \mathrm{m} & ;\text{otherwise} \end{cases}$$

Where S_{ij} - duration of shot j (in second) in video clip i;
N_i – Number of shots in video clips i;
M – Number of video clips and
N – Average number of shot per clip

4.2.3 High-level Object-based features

***Face*:** Human activities are one of the most important aspects of news videos, and many such activities can be deduced from the presence of faces. Many techniques have been proposed to detect faces in an image or video. In our study, we adopt the algorithm developed in [Chua 2002] to detect mostly frontal faces in the key frame of each shot. The main concept of face detection in this approach can be expressed as follows. First, the system generates contrast representation using average gradient energy. Next, neural network face detection model is trained. The system then looks for faces at multiple locations and scales. Finally, the system applies Gaussian skin color model to verify the faces found. Results of the system tested on a set of video clips containing 285 frontal and slightly slanting faces belonging to 25 subjects yielded an accuracy of over 89%.

The objective for face detection in this research is to extract face as well as its size. Face in static shots such as *Anchor* or most of the *Speech* shots can be detected more accurately than those in the dynamic shots, such as the walking scene or sport shots in which the number of faces in the same shot may vary due to camera movement. Thus, we need to select the appropriate number of faces to reflect the semantic of the shot. The number of face/s for each shot can be derived from:

$$F_i = \arg \max_{j \in K} [\, n_{ij} \,]$$

F_i = No. of faces in shot i;

K = { j: no. of faces detected in shot ***i,*** j = 0, 1, 2, ...}

$\boldsymbol{n_{ij}}$ = no. of consecutive frames in shot ***i*** containing ***j*** faces

Figure 4.12 shows the result of detecting mostly frontal faces and Figure 4.13 presents an example of the number of detected faces in one shot. Our algorithm performs face detection in every I-frame of the shot. Although, processing on a key frame per shot is much cheaper, but we want the accuracy of detecting face in a shot to be highly sufficient for further processing in story segmentation process. Hence, we choose to perform face detection on every I-frame of each shot.

Figure 4.12: Examples of the result of face detection

1	**1**	**2**	**2**	**2**	**2**	**1**	**0**

Figure 4.13: An example of a shot where there are three possible numbers of faces. Number in each cell represents the number of detected face/s

From Figure 4.13, we can see that within this shot we first detected one face, then two faces, followed by one and then ended with zero (camera captured on background scene or other objects). In this case, we consider the number of faces in this shot to be 2. This is because, two faces appears for the longest period in the shot as compared to

that of one face's or zero face's. The resulting shots with one or two detected faces will be used as the candidates for Anchor and 2Anchor shots classification.

***Shot type*:** The type of focal distance of the shot and its background colors help to identify the category of the shots such as the Anchor or 2Anchor shots. As it is difficult to estimate the camera's focal distance, we simply use the size of the detected faces to estimate the shot type, which includes *closed-up* (c), *medium*-distance (m), *long*-distance (l), or *unknown* (u) shot (when no face is detected). The estimation of shot type is based on the following formulas:

$$S_i = WxH$$

$$S_{avg} = \frac{\sum_{i=1}^{N} S_i}{N}$$

$$\sigma^2 = \frac{\sum_{i=1}^{N} (S_i - S_{avg})^2}{N-1}$$

where S_i – The size of the detected face in shot *i*;
W, H - Width and height of the detected face;
N – The number of detected faces from all video clips;
S_{avg} - the average size of a face computed from all detected faces.

We collect a large set of close-up shots from 40 video clips in the training set. These shots are of categories *Anchor* and *Speech.* After we perform face detection on the key frames of these shots, we compute the average size of the detected faces, $S_{avg}(c,)$ and their standard deviation, $\sigma(c)$. The lower bound for the threshold for close-up type, $T(c)$ is set to $S_{avg}(c) - \sigma(c)$. For medium type shot, we collect shots of categories *People* and *2Anchor* from the same 40 video clips. Similar to the estimation of close-up type, we compute the average face size, $S_{avg}(m)$ and its standard deviation, $\sigma(m)$.

We set the upper bound of medium type to be, $T(m)$ and $T_{upper}(m) < S_{avg}(c) - \sigma(c)$ (which is the lower bound of close-up type). As for the lower bound of this type, $T_{lower}(m)$, it is set to $S_{avg}(m) - \sigma(m)$. Therefore, the long distance type shots are those shots with face/s detected but their sizes are smaller than the lower bound of $T_{lower}(m)$. Finally, the unknown type was assigned to those shots with no detected face. We estimate that the accuracy of our shot type estimation technique is above 95% by testing on a different 20 video clips from the training set.

Videotext: Videotext is another type of object that appears frequently in news video and can be used to deduce video semantics. An example of shot category that contains multiple lines of videotext is the Text-scene category of which is frequently used to show sport or score and finance summary. Thus, the detection of multiple lines of centralized videotext is important to differentiate Text-scene shot from other categories. Also in *Anchor* shot or *Speech* shot categories, besides a face appearing in the shot, there will be the person's name appearing mostly at the bottom. For these two categories, videotext is also important. By extracting videotext from the shots containing faces, and performing character recognition using OCR (Optical Character Reader), we can obtain the names that can be used to associate with the persons appearing in the shots. However, the performance of video OCR is not sufficiently high and robust to enable videotext to be recognized accurately, hence in this research we do not rely on OCR text for shot classification. Instead, we simply employ the algorithm developed in [Zhang and Chua 2000] to detect the presence of videotexts. In the key frames of each shot, the system simply tries to detect videotext and

determines the number of lines of text. Examples of detected videotext are shown in Figure 4.14.

Figure 4.14: Examples of the detection of Videotexts from key frames

Centralized Videotext: We often need to differentiate between two types of shots containing videotexts. The normal shot where the videotexts appear at the top or bottom of a shot to indicate its contents; and the Text-scene shot where only a sequence of texts is displayed to summarize an event, such as the results of soccer games or financial summary. A text-scene shot typically contains multiple lines of centralized text, which is different from normal shots that may also contain multiple lines of text but normally un-centralized in a corner or at the bottom. Hence, we include this feature to identify *Text-scene* shots. It takes the value "true" for centralized text and "false" otherwise. Figure 4.15 presents a view of centralized videotext. We develop a simple algorithm for detecting centralized text. From the training data, the algorithm performs videotext detection. From all the frames with detected videotext and belonging to *Text-scene* category, we collect statistics (average, maximum, minimum) of the coordinate of the text box (a text box is where all lines of text in this frame are contained).

The detection is done on a frame by frame basis. For each frame, the algorithm first examines whether there is Videotext and determines the number of lines of text using

the algorithm described previously. If there are text lines, then the algorithm extracts the boundary box containing the text called the textbox. The algorithm then determines whether the box is centralized by considering the 4 main cases as follows (see Figure 4.15):

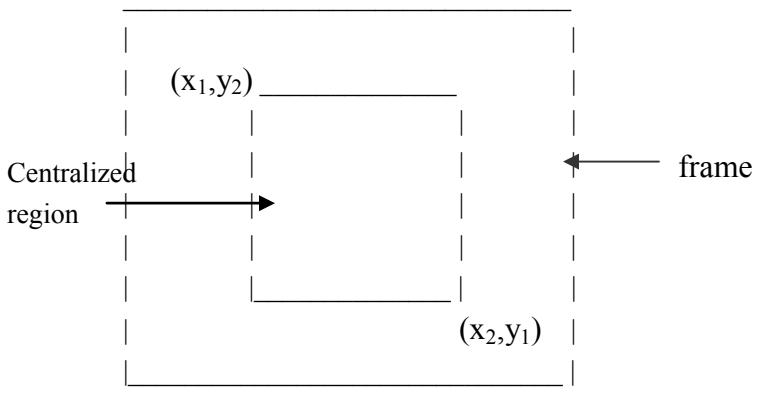


Figure 4.15: Scenario for Centralized Videotext

1) Check if the text box resides in the center box region.
2) Check if the text box is bigger than the center box and overlaps with the center box.
3) Check whether the width of text box exceeds the center box width and overlaps with the center box; and.
4) Check whether the height of the text box exceeds the center box height.

If one of the above conditions is met, we consider the text is to be centralized. Our algorithm could achieve an accuracy of about 85% by testing on a subset of the training data containing ~20 hours of video.

***Cue-phrase**:* From the ASR (automatic speech recognition) results of the speech track in videos; we analyze the statistics of cue-phrases that typically appear at the beginning of news stories (*Begin-Cue*) and miscellaneous stories (*Misc-Cue*). For each shot, we represent *Begin-Cue* as 1 (presence of Begin-Cue) or 0 otherwise. This feature is used during the story segmentation process. *Misc-Cue* is used to align the detected stories.

- *Begin- cue:* We observed that some of the story segments are typically preceded or ended with a set of cue phrases. To extract this list of cue phrases, we first compile a list of unique n-grams from the ASR transcript in all the story segments, and for each n-gram t_i, we calculate the probability that the n-gram indicates the start (p_b) or end of the story (p_e) by:

$$p_b(t_i) = \frac{B(t_i)}{T(t_i)},\ p_e(t_i) = \frac{E(t_i)}{T(t_i)}$$

$$T(t_i) = B(t_i) + M(t_i) + E(t_i)$$

where: $B(t_i)$ - number of stories containing t_i in first b n-gram;
$E(t_i)$ - number of stories containing t_i in last e n-gram;
$M(t_i)$ - number of stories containing t_i in (b+1) to (e-1) n-gram.

The list of $p_b(t_i)$ and $p_e(t_i)$ are ranked, and we select the n-grams with $p(t_i) \geq 0.80$ as the cue phrases. In our experiments, we set b and e to 10. Table 4.1 gives some examples of the selected cue phrases.

Type	Cue phrases
Begin	checking the hour's; good evening I am; brief review of
End	ABC news Washington; john macwethy ABC news; CNN New York

Table 4.1: Examples of begin/end cue phrases.

- *Misc-Cue:* In addition, miscellaneous (*misc*) segments contain similar information such as reporter chit-chat/scoreboard/stock quotes/advertisements as defined in the TDT-2 guidelines. Using a similar method, the list of unique n-grams from the *misc* segments can also be obtained. For each n-gram t_i, we compute the probability:

$$p_{misc}(t_i) = \frac{N_{misc}(t_i)}{N_{misc}(t_i) + N_{news}(t_i)}$$

where: $N_{misc}(t_i)$ is number of *MISC* stories containing n-gram t_i and $N_{news}(t_i)$ is number of *NEWS* stories containing n-gram t_i.

The top ranked n-grams are selected and clustered to generate a list of *Misc-cue* phrases. Some examples of Misc-cue are given in Table 4.2. These *Misc-cue* phrases are used to realign story boundaries (obtained from HMM) to the correct boundaries.

Shot Type	Cue phrases
Anchor, Program logos,	"Weather forecast is next" "top stories are" "Good morning this is" "when we come back"
Finance	"dow industrials" "broader markets" "nasdaq composite"
sports	"On the scoreboard"

Table 4.2: Examples of *Misc-cue* phrases.

The scenario for the realignment of stories boundaries is shown in Figure 4.16. As we can see from the Figure, the detected boundaries are not at the correct positions due to the presence of some Misc-cues that we need to segment out. The detected boundaries are correct if we consider only shot boundaries. However, they are not correct if we consider the beginning of non-Misc text as the actual start of stories as defined in TDT-2.

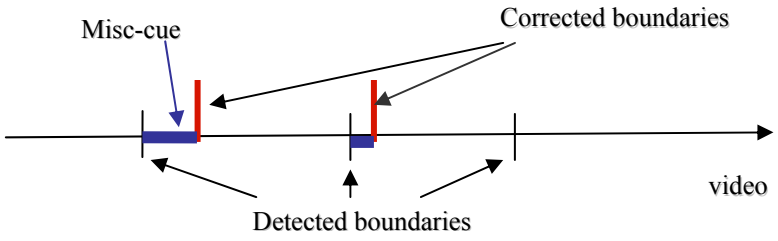


Figure 4.16: Story boundaries before and after the realignments

All the features, discussed above are essential for modeling the content of shots in our approach. Sequence of shots and their contents can be viewed and modeled as shown in Figure 4.17.

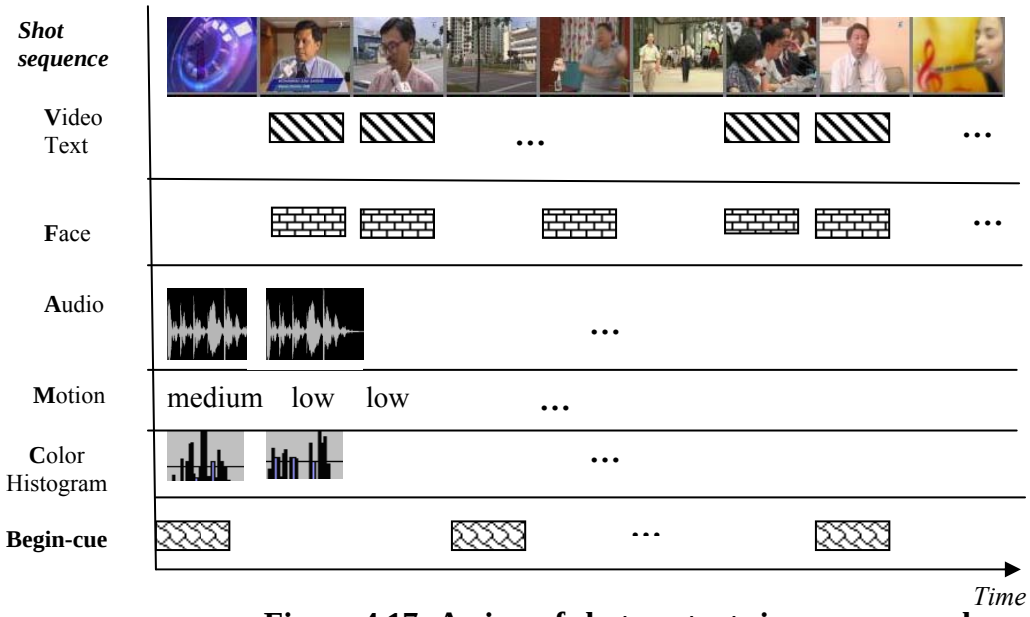


Figure 4.17: A view of shot contents in our approach

If news video were to be given in MPEG-7 format, we would be able to use or extract some of the features directly encoded in MPEG-7 video. These features include color histogram, motion activity, face, and audio type. This will save processing time. More details on MPEG-7 standard can be found in [MPEG -7 2000] or elsewhere.

After all features are extracted, the next task is to perform shot classification. The shot classification produces shot Tag_ID which is unique for each shot category. The shot category information together with scene change and cue-phrase features, are then used in the story segmentation process. We will discuss the shot classification and the result in the next chapter. Details of the story segmentation using HMM approach with the evaluation results are discussed in Chapters 6, while the details of story segmentation using rule-based approach is discussed in Chapter 7.

CHAPTER 5

SHOT CLASSIFICATION

This chapter presents the details of shot classification. It discusses shot representation and the fundamentals of the shot classification methods. It also presents the classification results, which are obtained, from small-scale data set and large-scale data set from different sources. In addition, the analysis of effectiveness of the selected features is also given.

5.1 Shot Representation

After the extraction of all features as discussed in the previous chapter, the contents of each shot are represented in two forms: a color histogram vector; and a feature vector. The histogram vector is used to match the content of a shot with the representative shot of certain categories, while the feature vector is used by the classifier to categorize the shots into one of the remaining categories. The color histogram vector and the feature vector of a shot are of the forms:

$$S_i = (h_1, h_2, \ldots h_{176}) \quad (5.1)$$

$$S_i = (f, s, t, c, a, m\ d) \quad . \quad (5.2)$$

Where - h_i the histogram value of Luv color i, i =1, 2, .., 176

- f the number of faces, $f \in$ N
- s the shot type, $s \in$ {c= closed-up, m=medium, l=long, u=unknown}
- t the number of lines of text in the scene, $t \in$ N
- c the text centralized feature. It is set to "true" if the videotexts present are centralized, c $\in$ {t=true, f=false}
- a the class of audio, $a \in$ {t=speech, m=music, s=silence, n =noise, tn = speech + noise, tm= speech + music, mn=music+noise}
- m the motion activity, $m \in$ {vl = very low, l=low, m=medium, h=high}
- d the shot duration, $d \in$ {s=short, m=medium, l=long}

Note that at this stage we did not include the scene change and speaker change or cue-phrase features in the feature set. These features are not essential for shot classification and will be included in story boundary detection using HMM.

5.2 The Classification of Video Shots

As described in Section 4.1.2, we divided shot categories into three clusters and thus there are three main sub-processes to classify shot categories in each cluster. They are heuristic–based (Commercials) shot detection, visually similar shot identification, and rule-based shot classification. Figure 5.1 shows the process diagram for the classification of shots.

Each cluster has its own characteristics, thus we need three separate techniques to perform the classification of shots. The techniques must be effective and scalable in order to cope with large scale data from different domains. The techniques to be

discussed in the following sections are employed and enhanced to deal with the detection of shot categories in each of the clusters.

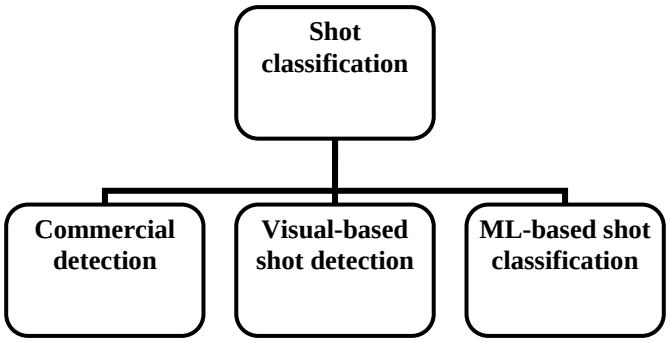


Figure 5.1: Process diagram for shot classification

5.2.1 Heuristic–based (Commercials) shot detection

Commercial boundaries can normally be characterized by the presence of black frames, still frames, audio silence, or any combination of all of them. We therefore employ a heuristic approach to identify the presence of commercials and detect the beginning and ending of the commercial blocks [Koh and Chua 2000]. In addition to news video, TRECVID 2003 also provides the associated ASR (Automatic Speech Recognition) outputs. These ASR outputs (textual data output with their confidence level of each word) can be used as another feature source. Low confidence values in the ASR results indicate potential commercial breaks. This is because commercials tend to have high cut rate with noisy or music background, thus the ASR output tends to have low confidence. From the experiments under this study, in addition to high cut rate and blank frames, the incorporation of confidence level improves the accuracy of commercial detection by about 5%. Figure 5.2 shows the details of our commercial detection process as discussed above.

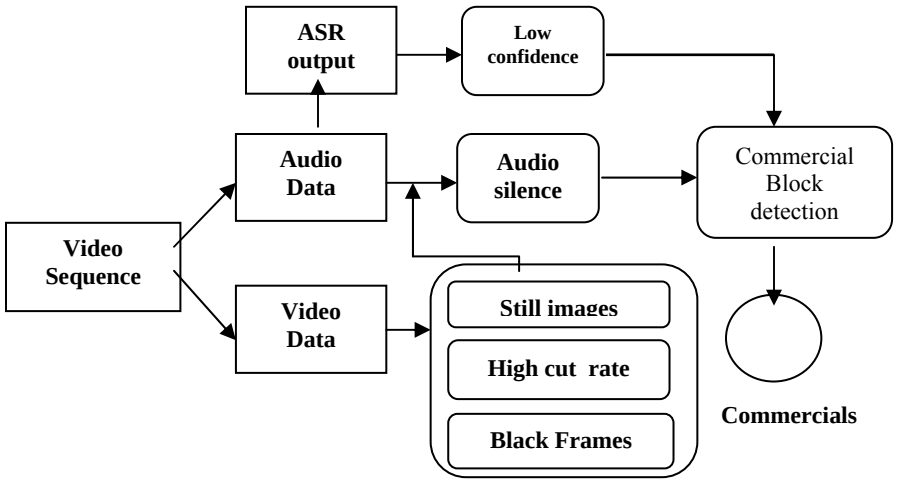


Figure 5.2: Diagram for the steps in commercial detection

5.2.2 Visually Similar Shot Detection

Next we identify the shot types that have high similar visual features. In this cluster, there are two sub-types: visually similar within the broadcast station (VSB); and visually similar across news broadcast (VSN) in general. Examples of VSB sub-type includes Weather and Finance reports, program logos such as "*TOP*" (top stories log), "*PLAY*" (play of the day log), "*SPORT*" (sport logo), "*HEALTH*" (health program logo), and "*LEDS*" (lead-ins/out). Examples of the later sub-type, VSN, include *Ancho*r and *2Anchor* categories.

For the classification of the first sub-type, the representative color histograms from the key frames of the respective categories are extracted. Then, the image-matching algorithm developed in [Chua and Chu 1998] is employed to compute the similarity between the key frames in the database and the test images. The algorithm also takes into account of perceptually similar colors. We adopt a high threshold of 0.8 to

determine whether a given shot belongs to the respective categories. Figure 5.3 presents a scenario to illustrate the technique.

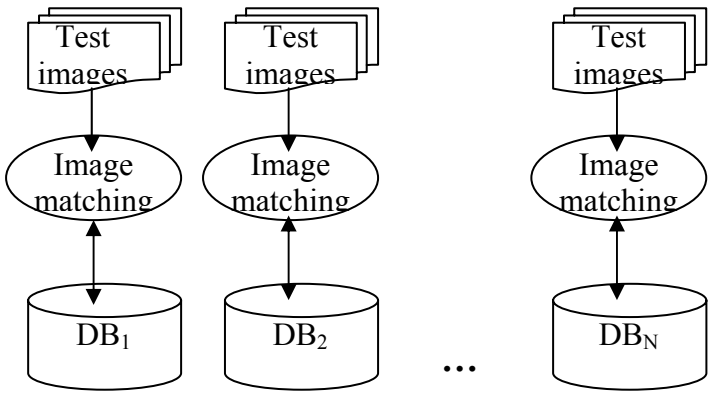


Figure 5.3: A scenario for image matching between the test images and the database images

From Figure 5.3, DB_i represents a database for news video of broadcast station i. For example, DB_{CNN} is the database for CNN news video in which the histograms of key frames (taken from the training set) of the respective categories are stored. The algorithm compares the histogram of the test image with the histograms of images stored in the database of the same broadcast station. The category of the test image is determined based on the major category of the top three database images whose similarity values with the test image are greater than the threshold. The method can be applied to any number of broadcast stations. For best result, domain knowledge (such as the data source, program schedule, etc.) of news video is an advantage. However, if we have no clue as to the source of news video under examination, then this technique needs some adjustments. The formulas used to compute image similarity are given below.

$$EXACT\_SIM_{color}(Q,D) = \frac{1}{N}\sum_{j\in S_N}(1-\frac{|NH(Q,j)-NH(D,j)|}{MAX\{NH(Q,j),NH(D,j)\}}) \quad (5.3)$$

$$PERCEPT\_SIM_{color}(Q,D,i) = \sum_{j\in S_P}[1-\frac{|NH(Q,i)-NH(D,i)|}{MAX\{NH(Q,i),NH(D,i)\}})xS(i,j)] \quad (5.4)$$

$$SIM_{Color}(Q,D,i) = EXACT\_SIM_{Color}(Q,D,i)(1+PERCEPT\_SIM_{Color}(Q,D,i)) \quad (5.5)$$

$$SIM_{color}(Q,D) = \sum_{i\in S_N}[SIM_{color}(Q,D,i)xNH(Q,i)] \quad (5.6)$$

where Q is the test image and D is a database image; $S(i,j)$ stores the similarity value between color i and color j; and $NH(Q, i)$ is the normalized histogram value for color i of image Q. The algorithm first computes $EXACT\_SIM_{color}(Q, D)$ which is the similarity between database image D and test image Q without considering the perceptual similarity between colors. Next, it computes $PERCEPT\_SIM_{color}(Q, D)$, which taking into account of perceptually similar colors. Finally, it adds the two similarity values and normalizes the result to produce $SIM_{color}(Q, D)$, which gives the overall similarity over all the colors between images Q and D. For more details of the algorithm, refer to [Chua and Chu 1998].

For categories *Anchor* and *2Anchor* in the second sub-type, we employ a clustering algorithm to cluster the candidate shots containing one or two detected faces. After clustering; we select the biggest cluster to represent the shots for anchor news reporting. We further divide this cluster into the clusters of *Anchor* and *2Anchor* based on the number of face. There are well-known clustering algorithms such as *Hierarchical clustering*, *K-mean clustering,* etc. Here, we employ the *single-linkage* hierarchical clustering method [Krishnaiah and Kanal 1982] to perform the *Anchor*

and *2Anchor* shot identification. The reason that this algorithm is chosen is because, it performs the best on the test data as compared to other clustering algorithms. For simplicity, the processing steps of the algorithm are expressed as below.

Clustering Algorithm:

Given the set $S = \{x_i\}$, x_i is an instance i, i =1, 2, …N, where N is the total number of instance.

1. *Place each instance of S in its own cluster (singleton), creating the list of clusters L (initially, the leaves of T):*

   $$L = S_1, S_2, S_3, ..., S_{N-1}, S_N.$$

2. *Compute a* ***merging cost function*** *between every pair of elements in L to find the two closest clusters* $\{S_i, S_j\}$ *which will be the cheapest couple to merge.*
3. *Remove* S_i *and* S_j *from L.*
4. *Merge* S_i *and* S_j *to create a new internal node* S_{ij} *in T which will be the parent of* S_i *and* S_j *in the result tree.*
5. *Go to (2) until there is only one set remaining.*

(Source: http://genome.imim.es/~eblanco/seminars/docs/clustering/index_types.html)

The most common ***merging cost functions*** to measure the distance between any two clusters for the *Hierarchical clustering* are defined as follows.

Single-linkage:

$$\min_{x_i \in S_i, x_j \in S_j} d(x_i, x_j) \tag{5.7}$$

Average-linkage:

$$\frac{1}{|S_i||S_j|} \sum_{x_i \in S_i} \sum_{x_j \in S_j} d(x_i, x_j) \tag{5.8}$$

Complete-linkage:

$$\max_{x_i \in S_i, x_j \in S_j} d(x_i, x_j) \tag{5.9}$$

and the Euclidean distance; $$d(x_i, x_j) = \sqrt{(x_i - x_j)^2} \tag{5.10}$$

Further details of the algorithm can be found in [Krishnaiah and Kanal 1982]. Figure 5.4 illustrates the clustering of 8 sample data into clusters using the hierarchical clustering technique. Here, there are 8 samples to be clustered, namely, a, b, c, d, e, f, g and h. The result of clustering yields 3 clusters. The first cluster contains a, b and c; the second cluster contains d, e, f, and g; and the last cluster consists of only one member h.

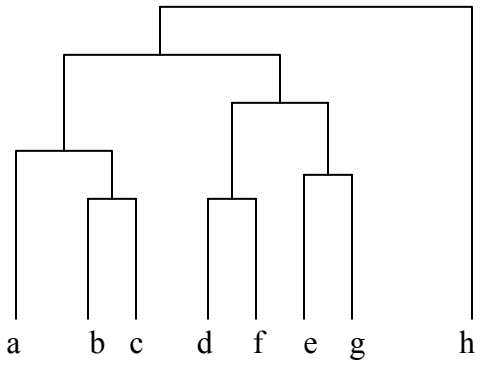


Figure 5.4: Illustrates clustering algorithm.

The identification of *Anchor* shots is based on the observations that within news broadcast, *Anchor* shots occur more frequent than any other type of shots containing faces and with the same background. Thus, the cluster that contains the largest number of shots belongs to anchor category which can be *Anchor* or *2Anchor* depending on the number of faces detected. For the example described in Figure 5.4,

assuming that the 8 samples are all the shots to be clustered based on color histogram distance measurement; the second cluster is selected as *Anchor* and *2Anchor* cluster.

5.2.3 Classification Using Decision Trees

5.2.3.a Overview of Decision Trees

After visual-based shots are filtered out, for the rest of the shots, we employ a Decision Tree algorithm [Quinlan 1986] and [Quinlan 1997] to perform the classification in a learning-based approach. Decision tree (DT) is one of the most widely use methods in machine learning. We employ the decision tree because it is robust to noisy data, capable of learning disjunctive expression, and the training data may contain missing or unknown values [Quinlan 1986]. Decision tree or classification and regression tree (CART) [Breiman 1993] has been successfully employed in many multi-class classification problems [Dietterich and Bakiri 1995] [Zhou 2000]. Thus, decision tree is selected for the shot classification problem in machine-learning based cluster. Before we discuss the classification using the decision tree, we will first discuss its foundation and an overview.

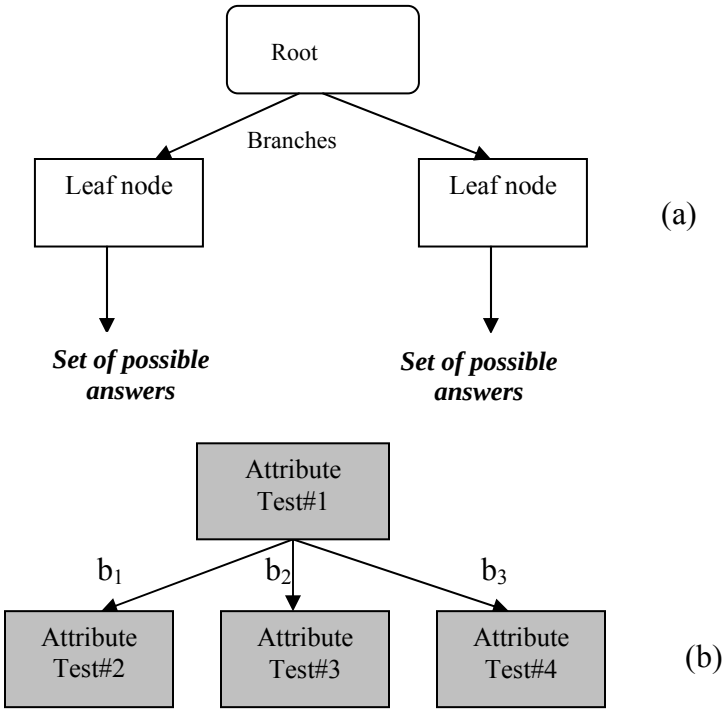
A decision tree partitions the input space (also known as the attribute space) of a dataset into mutually exclusive regions. Each of which is assigned a label, a value, or an action to characterize its data points. A decision tree is a tree structure consisting of internal and external nodes connected by branches. An **internal node** is a decision-making unit that evaluates a decision function to determine which child node to visit

next. In contrast, an **external node**, also known as leaf or a **leaf** or **terminal node** has no child nodes and is associated with a label or value that characterizes the given data.

In general, a decision tree is employed as follows. A vector of data (usually composed of several attributes or elements) is presented to the starting node of the decision tree. Depending on the result of the decision function used by the internal node, the tree will branch to one of the node's children. This is repeated until a terminal node is reached whose label or value is then assigned to the given input data. The decision mechanism is transparent so we can follow a tree structure easily to explain how a decision is made.

Decision trees used for classification problems are often called classification trees, and each terminal node contains a label that indicates the predicted class of a given feature vector. However, the term decision tree is more commonly used than classification tree, thus we will use the term decision tree in this thesis. General diagram of a decision tree is presented in Figures 5.5 (a) –5.5 (b).

Each node in the tree specifies the test of some attribute of the instance (as shown in Figure 5.5 b), and each branch descending from that node corresponds to one of the possible values for this attribute. Given ***m*** attributes, a decision tree may have a maximum height of ***m.*** Let n_b be the total number of instances in branch b; n_{bc} be the total number of *positive instance* on branch b of class c; and n_t, be the total number of instances in all branches. At each iteration, the feature with the minimum average entropy is selected for processing.



Note: b_1, b_2, and b_3 are branches of the tree.

(Source: http://www2.cs.uregina.ca/~hamilton/courses/831/notes/ml/dtrees/4_dtrees1.html)

Figure 5.5: Decision tree diagram

The formulas for computing the entropy and average entropy are expressed as follow.

$$Entropy = \sum_{c} -(\frac{n_{bc}}{n_b}) \log 2(\frac{n_{bc}}{n_b}) \qquad (5.11)$$

$$AverageEntropy = \sum_{b} \frac{n_b}{n_t} (\sum_{c} -(\frac{n_{bc}}{n_b}) \log 2(\frac{n_b c}{n_b})) \qquad (5.12)$$

where $\frac{n_{bc}}{n_b}$ is the probability of positive instance on branch ***b***

At each stage of the learning process, the decision tree will calculate the average entropy of each attribute, and select the attribute with the *lowest entropy* as the branching attribute. The process stops when all the leaf nodes have only one individual class.

5.2.3.b Presentation of result and measurements

Normally, the results from the Decision Tree are presented using a confusion matrix [Kohavi and Provost, 1998], which contains information about actual and predicted classifications done by a classification system. Performance of such systems is commonly evaluated using the data in the matrix. The following Table shows the confusion matrix for a two class (positive and negative) classifier. The entries in the confusion matrix as shown in Table 5.1 have the following meaning in the context of our study:

- *a* is the number of **correct** predictions of class negative;
- *b* is the number of instances of class negative that are misclassified as positive;
- *c* is the number of instances of class positive that are misclassified as negative; and
- *d* is the number of **correct** predictions of positive instances.

		Predicted	
		Negative	Positive
Actual	Negative	**a**	**b**
	Positive	**c**	**d**

Table 5.1: Confusion matrix

Several standard measures have been defined for the 2 class matrix:

- The *accuracy* (*AC*) is the proportion of the total number of predictions that were correct. It is determined using the equation:

$$AC = \frac{a+d}{a+b+c+d} \tag{5.13}$$

- The *recall* or *true positive rat*e (*TP*) is the proportion of positive cases that were correctly identified, as calculated using the equation:

$$TP = \frac{d}{c+d} \tag{5.14}$$

- The *false positive rate* (*FP*) is the proportion of negative cases that were incorrectly classified as positive, as calculated using the equation:

$$FP = \frac{b}{a+b} \tag{5.15}$$

- The *true negative rate* (*TN*) is defined as the proportion of negative cases that were classified correctly, as calculated using the equation:

$$TN = \frac{a}{a+b} \tag{5.16}$$

- The *false negative rate* (*FN*) is the proportion of positive cases that were incorrectly classified as negative, as calculated using the equation:

$$FN = \frac{c}{c+d} \tag{5.17}$$

- Finally, *precision* (*P*) is the proportion of the predicted positive cases that were correct, as calculated using the equation:

$$P = \frac{d}{b+d} \tag{5.18}$$

Out of the above, the most commonly used measures are the accuracy (AC), recall or true positive rate (TP) and precision (P).

Further details on decision trees can be found in [Quinlan 1986], [Quinlan 1986] and in [WWW2].

5.3 Trial Test on Small Data Set

5.3.1 Training and Test Data

We want to test our framework on a small data set before applying to the large scale data set. This is to investigate how well the system performs and to analyze the incurred errors during this test in order to improve the performance on the large scale test. In our preliminary test, we use two days of news video (one from May 2001, the other from June 2001) obtained from the MediaCorp of Singapore. Each day of news video is half an hour in duration. One day is used for training, and the other for testing. In order to eliminate indexing errors, we manually index all the features of the shots segmented using the multi-resolution analysis algorithm [Lin 2000, and Anantharamu 2002]. The training data set contains 440 shots and testing data set contains 403 shots.

5.3.2 Results of the Shot Classification

Table 5.2 presents the results of shot classification which includes the results from the decision tree, commercial detection and image matching for *Weather* and *Finance* shots. Table 5.3 gives the detailed result from the decision tree.

process	F_1-value (%)
1.Commercial detection	97.00
2.Weather and Finance filtering	100.00
3.classification using the decision tree	95.1
Average	**97.4**

Table 5.2: Summary of shot classification results

Classified as->	a	b	c	d	e	f	g	j	k	l
a) intro/highlight	26					1				
b) anchor		16					4			
c) 2anchor			2							
d) People				13						
e) still image					1					
f) Live-reporting						82		1		
g) Speech		1					11			
j) sport						1		8		
k) text-scene									6	
l) special										5

Table 5.3: The classification result from the decision tree

Note that this test was done in the earlier stage of research and the visual-based clusters contain only *Weather* and *Finance* shot categories. Also, *Anchor* shots were classified using the decision tree. From this result, we subsequently improve the algorithm by integrating clustering technique to identify *Anchor* shots (details of the result using clustering technique are given later in Section 5.4 on the evaluation of TRECVID data). From Table 5.3, the diagonal entries show the number of shots correctly classified into the respective category, while the off-diagonal entries show those wrongly classified. It can be seen that the largest classification error occurs in

the *Anchor* category where a large number of shots is misclassified as *Speech*. This is because their contents are quite similar, and thus we need additional features like background and the context of neighboring shots to differentiate them. Overall, our initial results indicate that we could achieve a classification accuracy of over 97%. As mentioned above, we did not introduce additional technique to detect *Anchor* shots. Thus, we can see from the result in Table 5.3 that high errors occur in differentiating *Anchor* shots and *Speech* shots. We will present the results of employing the clustering algorithm (as discussed in Section 5.2.2) to resolve the ambiguity of these categories, including the *2Anchor* category in Section 5.4.

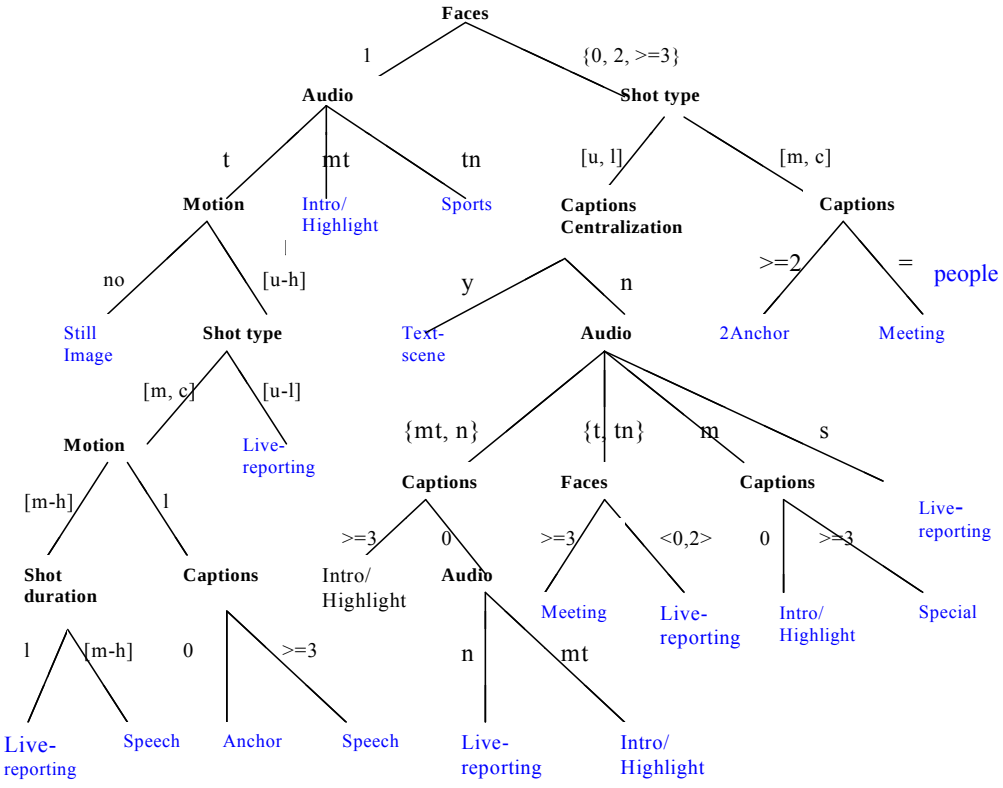


Figure 5.6: The learnt tree created from the training data

Figure 5.6 presents the learnt decision tree. As we can see from the figure, complex tree is not easy to read. One useful by-product of performing the decision tree analysis is the set of rules generated by the decision tree. These rules may be used to provide better insight into how certain decisions were made. Table 5.4 presents the set of 14 rules extracted from the learnt tree.

Rule 1	shot type in [u-l], audio = speech, class-> Live-reporting ** (0.684)
Rule 2	face =1, shot type in [m-c], captions <= 1, audio = speech, motion = l, Class-> Anchor (0.56)
Rule 3	audio = speech and noise, Class ->Sport (0.686)
Rule 4	audio in {music and speech, noise), Class -> Intro/highlight 0.796)
Rule 5	face =1, captions > 1, motion = l, Class -> Speech/Interview (0.59)
Rule 6	captions <= 1, audio = m, Class -> Intro/Highlight (0.894)
Rule 7	face >=3, Class-> People (0.807)
Rule 8	face = 2, shot type in {m-c}, Captions > 1, Class -> 2Anchor (0.581)
Rule 9	captions Position Center = yes, Class ->Text-only (0.832)
Rule 10	captions > 1, centralized captions = no, audio = music, Class -> Special (0.773)
Rule 11	face = 1, motion in [l-m], shot duration in [m-h], Class -> Speech/Interview (0.771)
Rule 12	face = 1, motion in [?-n], Class -> Still image (0.763)
Rule 13:	face in {>=2}, shot type in [m-c], Captions <= 1, Class -> People (0.593)
Rule 14	face =1, audio = speech, motion in [m-h], shot duration = l, Class ->Live-reporting (0.708)

Table 5.4: Rules extracted from the learnt tree

5.3.3 Effectiveness of the Selected Features

In order to ascertain the effectiveness of the set of features selected, we perform separate experiments by using different number of features. As face is found to be the most important feature, we use the face as the first feature to be given to the system. With the face feature alone, the system returns an accuracy of only 59.6%. If we include the audio feature, the accuracy increases rapidly to 78.2%. However, this

accuracy is still far below the accuracy that we could achieve by using all the features. When we successively add in the rest of features in the order of shot type, motion, videotext, text centralization, and shot duration, the performance of the system improves steadily and eventually reaches the accuracy of 95.10%. The analysis indicates that all the features are essential in shot classification. Figure 5.7 shows summary of the analysis.

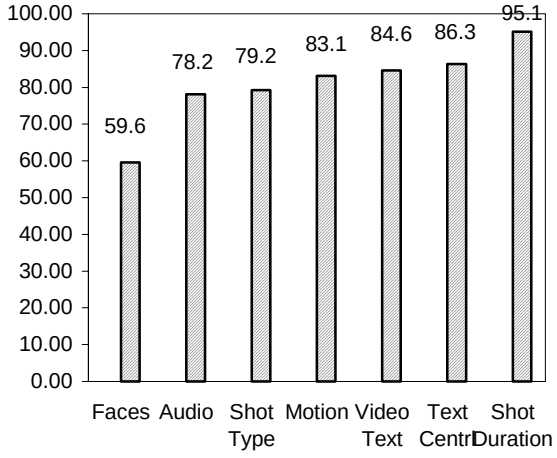


Figure 5.7: Summary of the feature analysis

5.4 Evaluation on TRECVID 2003 Data

In order to test our system on large scale data set, we participated in the story segmentation task in TREC Video Retrieval 2003 evaluations (TRECVID 2003). Here, we present only the shot classification results while the results of story segmentation evaluated by TRECVID will be given in the next Chapter.

5.4.1 Training and Test Data

The training and test data provided by TRECVID 2003 are CNN and ABC news video for the year 1998. Altogether, there are about 120 hours of video (or ~ 240 video broadcasts). Each half an hour broadcast of CNN and ABC news video contains about 300-400 shots. From the training set, we use 20 video clips for training and a different 20 video clips for testing. The results are presented in the following Section.

5.4.2 Shot Classification Result

The result of the shot classification grouped into the three clusters is presented in Table 5.5.

Cluster	Precision (%)	Recall (%)
Commercials	99.10	95.80
Visual-similarity-based	89.71	90.30
ML-based	91.0	90.0
Average	**93.27**	**92.03**

Table 5.5: Summary of shot classification results

From Table 5.5, we can see that, the best result is obtained for commercial detection which we could achieve over 99% in precision and over 95% in recall. For visual similarity based shots, we could achieve reasonable accuracy over 89% and 90% for precision and recall respectively. Shot categories in this cluster are the most important categories. In general, the appearance of the category potentially indicates story change. Therefore, it is essential to have an effective algorithm to detect the correct

shot categories in this cluster. Our algorithm is considered reasonable and acceptable for the detection of the wide range of such categories over the large set of data. Finally, for machine-learning based cluster, we could achieve a performance of over 91% and 90% for precision and recall respectively. Overall, the accuracy of shot classification is over 90%. The detailed results of each category of Visual-based cluster and ML-based cluster are provided in Table 5.6 and Table 5.7 respectively.

Category	Precision (%)	Recall (%)
Anchor/2Anchor	84.84	87.6
Weather	100.00	100.00
Finance	100.00	100.00
Program logos	74.00	73.62
(*TOP, HEALTH, SPORT, LEDS and PLAY*)		
Average	**89.71**	**90.30**

Table 5.6: Result of each category of Visual-based cluster

a	b	c	d	e	f	g	<-Classified as
38			5				a sports
	38		10				b Intro/highlight
	4	58		3			c People
2	13		571			6	d Live-reporting
			4	292			e Speech/Face
					16		f Text-scene
	1		2			5	g Special

Note: diagonal entries are the number of correct prediction.

Table 5.7: Result of each category of ML-based cluster using the Decision Tree

We can see from Table 5.6 that most of the errors are from the detection of those temporal-visual based shot types, for example "*LEDS*", "*TOP*", etc. These types of

shots typically appear in very short durations, and the duration may be varying even for the same broadcast station but for news of different days. Thus, our algorithm which is designed to handle longer video shots fails to detect them effectively. However, we have improved the shot classification algorithm by solving the ambiguity between *Speech* and *Anchor* shots (as can be seen from the results shown in Table 5.8 that the detection rate for *Anchor* category is more than 87% and most of the *Speech* shots, as shown in Table 5.9, are correctly classified). In ML-based cluster, most errors are from Live-reporting and Intro/Highlight categories. Intro/Highlight shots usually contain background music while Live-reporting shots sometimes consist of background noise. The problem occurs when our audio classification algorithm misclassified music as noise or vice versa. However, overall accuracy of our shot classification algorithm is about 90% (see Table 5.6), hence our shot classification algorithm has been demonstrated to be effective. The learnt tree from this corpus is similar to that was discussed in Section 5.3, except it doesn't include *Anchor and 2Anchor* shot categories in the classification.

The next chapter discusses details of HMM framework for story segmentation level, the second stage of our framework. It also presents the result evaluated by the TRECVID on the large-scale data set.

CHAPTER 6

HIDDEN MARKOV MODEL APPROACH TO STORY SEGMENTATION

This Chapter discusses the details of our HMM approach. It also presents the results on both the small scale data set and large scale data set evaluated by TRECVID 2003. In addition, it also discusses the analysis of errors on both data sets.

Our two-level approach is similar to the idea of part-of-speech (POS) tagging problem in NLP [Dale 2000] that uses a combination of POS tags and lexical information to perform the analysis. Here we use the tagged categories obtained from shot classification (the first-level) plus the scene change and cue-phrase to identify story boundaries. Before presenting details of story segmentation, we will discuss the foundation and important issues of HMM. Our discussions are based on the materials derived from [Rabiner and Juang 1993].

6.1 Hidden Markov Models (HMM) [Rabiner and Juang 1993]

HMM is a powerful statistical tool first successfully utilized in speech recognition research. In recent years, HMM has also been applied successfully to applications such as pattern recognition, computer vision, and computational molecular biology, etc. We employ HMM to model our data for story segmentation because news video

is a temporal and sequencing media, composing of audio and video signals. Thus, HMM which was first designed for time sequencing data such as speech can be readily applied here. The followings discuss some important issues that are needed when employing HMM for data analysis.

Notation

Suppose HMM contains a finite set of *states*, each of which is associated with a probability distribution. Transitions among the states are governed by a set of probabilities called *transition probabilities.* In a particular state, an outcome or *observation* can be generated according to the associated probability distribution. In general, HMM is modeled with $\lambda = \{\pi, A, B\}$, where

T = the length of the observation sequence

N = the number of states in the model

M = the number of observation symbols

$Q = \{q_1, q_2, .., q_N\}$ = the states of Markov process

$V = \{1, 2 \ldots M\}$ = set of possible observation symbols

$A = \{a_{ij}\}$ = the state transition probability matrix, where

$$a_{ij} = P(q_{t+1} = j \mid q_t = i) \qquad 1 \le i \le N \text{ and } 1 \le j \le N, \tag{6.1}$$

defines the transition probability of moving from state i to state j

$\pi = \{\pi_i\}$ is the set initial state probability;

$$\pi_i = P(q_1 = i) \qquad 1 \le i \le N \tag{6.2}$$

$B = \{b_j(k)\}$ is the observation probability distribution matrix, where

$$b_j(k) = P(O_t = v_k \mid q_t = j) \qquad 1 \le k \le M; \tag{6.3}$$

defines the symbol distribution in state j, $1 \le j \le N$

$O = (O_1, O_2, \ldots O_T)$ = observation sequence, $O_i \in V$ for i = 1, 2, 3, … T

Model Assumptions

There are some assumptions that need to be made when using HMM. The assumptions are:

- Model uses finite number of discrete hidden states
- First order Markov property, that is $P(q_t \mid q_{t-1}, q_{t-2}, \ldots) = P(q_t \mid q_{t-1})$
- Emission probability depends only on current state
- For homogenous HMM, all probabilities are time-invariant

Types of HMM

There are two types or topologies of HMM; ergodic and left-to-right. One way to classify the types of HMM is by the structure of the transition matrix, A, of the Markov chain. Consider the special case of ergodic or fully connected HMM in which every state of the model could be reached (in a single step) from every other state of the model. As shown in Figure 6.1 (a), for N =4 state model, this type of model has the property that every a_{ij} coefficient is positive. Hence, for example of Figure 6.1 (a), we have,

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

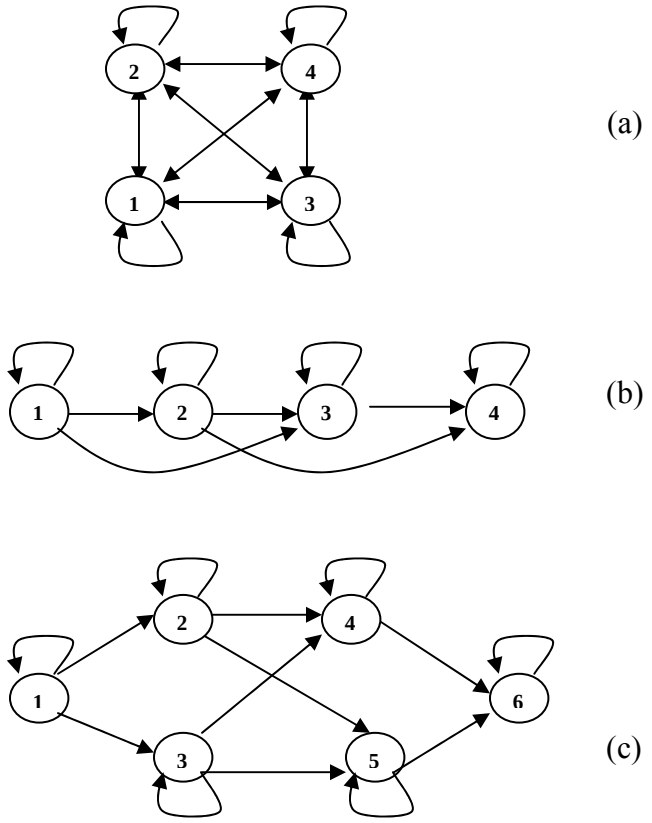


Figure 6.1: Illustrates three distinct HMMs. (a) A 4-state ergodic model. (b) A 4-state left-to-right model. (c) A 6-state parallel path left-to-right model

Ergodic model is one type of models that best fit to our data. As discussed earlier that one way to classify the types of HMM is by the structure of the transition matrix, A [Rabiner and Juang 1993]. From our data analysis, a_{ij} coefficients are positive and thus they satisfy the properties of the Ergodic model

The left-to-right model (as shown in Figure 6.1 (b)) has the property that as time increases, the state index increases (or stays the same) – that is, the system states proceed from left to right.

The fundamental property of the left-to-right HMM is that the state-transition coefficients have the property

$$a_{ij} = 0, \qquad j < i$$

That is, no transitions are allowed to states whose indices are lower than that of the current state. Furthermore, the initial state probabilities have the property

$$\pi_i = \begin{cases} 0, & i \neq 1 \\ 1, & i = 1 \end{cases}$$

This means that the initial state must be state 1 with the probability of 1.0. Often with the left-to-right models, additional constraint are placed on the state-transition coefficients to make sure that large changes in state indices do not occur; hence a constraint of the following form is often used.

$$a_{ij} = 0, \quad j > i+\Delta i$$

In particular, for the example in Figure 6.1 (b), the value of Δi is *2;* that is, no jump of more than two states are allowed. The form of the state-transition matrix for the example is thus,

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22} & a_{23} & a_{24} \\ 0 & 0 & a_{33} & a_{34} \\ 0 & 0 & 0 & a_{44} \end{pmatrix}$$

We can see from above A matrix that the coefficients a_{21}, a_{31}, a_{32}, and so on are of zero values. This means that no allow of transitions to state index that lower than the current state index.

Besides the above fully connected (ergodic) and left-to-right models; there are many other possible variations and combinations. Figure 6.1 (c) shows a cross-coupled connection of two parallel left-to-right HMMs.

From the above descriptions, we can see that the left-to-right HMM is designed to model signals whose properties change over time in a successive manner, such as the speech; whereas the ergodic model is designed to deal with signals that may transit between different states in a more flexible manner. For story segmentation problem, we want to be able to flexibly transit a state to any state at any point of time; hence the left-to-right HMM is not suitable. To model this flexibility using the left-to-right model would require a lot of training data in order to accurately estimate the model parameters (refer to implementation issues for the left-to-right model). This is not possible because of data-sparseness problem. Hence, we choose ergodic HMM as the basic model for our analysis.

Three basic problems

There are three basic issues that need to be solved when employing HMM. Given a HMM λ, and a sequence of observation O, $O = (O_1, O_2, \ldots O_T)$, the problems can be described as follows.

i) ***Evaluation:*** Find $P(O|\lambda)$ which is the probability that HMM λ produces the observation sequence O.

ii) ***Decoding:*** Given the observation sequence O, and the model λ, how to estimate state sequence $Q = q_1, q_2, \ldots q_T$. that is optimum

iii) ***Parameter Estimation (Learning):*** How to train the model to maximize $P(O|\lambda)$

Solutions to the three basic problems

Here, we discuss the solutions to the above problems when employing the HMM framework.

The Forward-Backward algorithm

We need this algorithm to solve the first problem, i.e. given O, $O = (O_1, O_2, \ldots O_T)$, find $P(O|\lambda)$. Figure 6.2 illustrates the Markov process for the forwarding algorithm.

The algorithm can be derived as follows:

For t = 1, 2, …T, and i=1, 2, …N, define

$$\alpha_t(i) = P(O_1, O_2, \ldots O_t, q_t = S_i/\lambda) \quad (6.4)$$

The $\alpha_t(i)$ can be computed recursively as:

1. *Initialization:* Let $\alpha_1(i) = \pi_i b_i(O_1)$; (6.5)
   for i = 1,2, …,N
2. *Iteration:* For t = 1, 2, …T-1 and j =1, 2, …,N; compute

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) \quad (6.6)$$

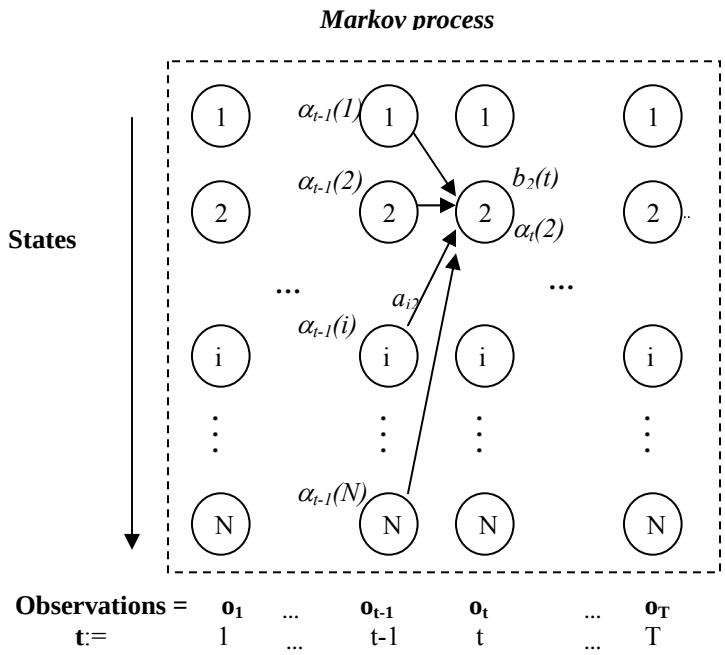


Figure 6.2: Illustrate Markov process of the forward algorithm. In the figure, forward probability for state 2 at time *t* is $\alpha_t(2)$ calculated from summing all possible $\alpha_{t-1}(i).a_{ij}.b_2(t)$

3. Termination,

$$P(O/\lambda) = \sum_{i=1}^{N} \alpha_T(i) \qquad (6.7)$$

Similarly, we can define the backward algorithm, which is analogous to the forward algorithm, except that it starts at the end and works towards the beginning. Figure 6.3 shows the diagram of Markov process for the backward algorithm.

The backward algorithm can be expressed as the following.

For t =1,2, …, T and i=1,2,…N, define

$$\beta_t(i) = P(o_{t+1}, o_{t+2,..} o_T | q_t = i, \lambda)$$

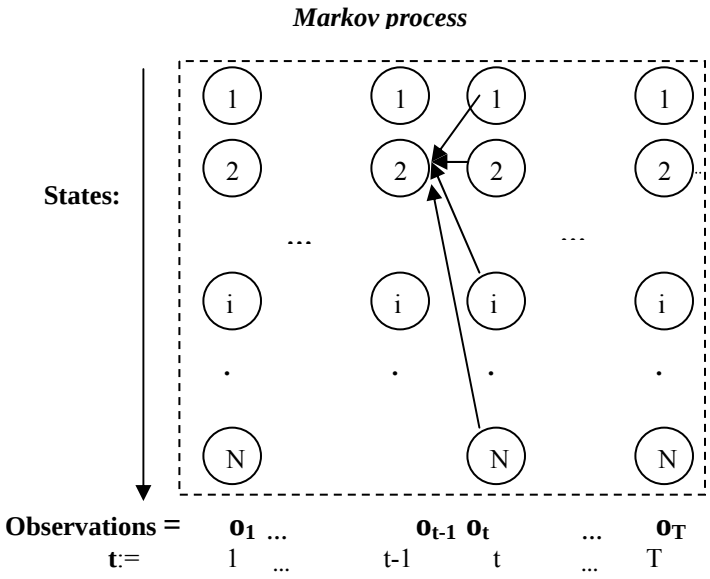


Figure 6.3: Illustrate Markov process of the backward algorithm.

That is the probability of the partial observation sequence from time t+1 to the end, given state *i* at time *t* and the model λ. We can solve for $\beta_t(i)$ inductively as follows:

1. *Initialization:* $\beta_T(i) = 1$, (6.8)

   for $i = 1, 2, \ldots, N$

2. *Iteration:* For $t = T-1, T-2, \ldots 1$ and $i = 1, 2, \ldots N$, compute

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j) \quad (6.9)$$

We will see later how the forward and backward algorithms help to solve the fundamental problems 2 and 3 of HMMs.

The Viterbi Algorithm

We need this algorithm to tackle problem 2, i.e. to decode the optimum state sequence for our test data. The algorithm can be summarized as follows:

1. *Define optimal partial path score*

$$\delta_t(i) = \max_{q1,q2,...qt-1} P(q_1 q_2 .. q_{t-1}, q_t = i, o_1 o_2 ... o_t \mid \lambda) \qquad (6.10)$$

2. *Initialization*

$$\delta_1(i) = \pi_i b_i(o_1) \qquad (6.11a)$$

$$\psi_1(i) = 0 \ ; \ 1 \le i \le \mathrm{N} \qquad (6.11b)$$

3. *Dynamic programming and book keeping*

$$\delta_t(j) = \max_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}] b_j(O_t) \qquad (6.12)$$

$$2 \le t \le T; \qquad 1 \le j \le N;$$

$$\psi_t(j) = \arg\max_{1 \le i \le N} [\delta_{t-1}(i) a_{ij}] \qquad (6.13)$$

$$2 \le t \le T; \qquad 1 \le j \le N;$$

4. *Termination*

$$P_{\max} = \max_{1 \le i \le N} \delta_T(i) \qquad (6.14)$$

and $$\hat{q}_T = \arg\max_{1 \le i \le N} \delta_T(i) \qquad (6.15)$$

5. *Trace back to get the optimal state sequence (back tracking)*

$$\hat{q}_t = \psi_{t+1}(\hat{q}_{t+1}) \qquad (6.16)$$

$$t = T\text{-}1, T\text{-}2, \ldots 1.$$

$$\hat{q} = (\hat{q}_1, \ldots \hat{q}_T) \qquad (6.17)$$

There is alternative way to performing Viterbi decoding using log probability. By using log, multiplication will be replaced by summation which is easier. The algorithm can be derived as:

1. *Define log probabilities*

$$\tilde{\pi}_i = \log \pi_i \quad \tilde{a}_{ij} = \log a_{ij}; \quad \tilde{b}_i(o_t) = \log b_i(o_t) \tag{6.18}$$

2. *Initialization*

$$\tilde{\delta}_1(i) = \tilde{\pi}_i + \tilde{b}_i(o_1) \tag{6.19}$$

3. *Dynamic programming, recursion*

$$\tilde{\delta}_t(j) = \max_{1 \le i \le N} [\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}] + \tilde{b}_j(O_t) \tag{6.20}$$

$$1 \le t \le T; 1 \le j \le N;$$

$$\psi_t(j) = \arg\max_{1 \le i \le N} [\tilde{\delta}_{t-1}(i) + \tilde{a}_{ij}] \tag{6.21}$$

$$2 \le t \le T; \ 1 \le j \le N;$$

4. *Termination*

$$\tilde{P}_{\max} = \log P_{\max} = \max_{1 \le i \le N} \tilde{\delta}_T(i) \tag{6.22}$$

$$\hat{q}_T = \arg\max_{1 \le i \le N} [\tilde{\delta}_T(i) \tag{6.23}$$

5. *Trace back to get the optimal state sequence*,

$$\hat{q}_t = \psi_{t+1}(\hat{q}_{t+1}) \tag{6.24}$$

$$t = T, T-1, \ldots 1$$

$$\hat{q} = (\hat{q}_1, \ldots \hat{q}_T) \tag{6.25}$$

Parameter Estimation

Estimating a model which maximizes $P(\lambda/O)$ can be solved as a maximum likelihood problem. The algorithm called **Baum-Welch** can be used to maximize $P(\lambda/O)$ locally.

a) Define new variable,

$$\xi_t(i,j) = P(q_t = i, q_{t+1} = j / O_1, \ldots O_t, \ldots O_T, \lambda) \tag{6.26}$$

$$= P(q_t = i, q_{t+1} = j, O_1^T / \lambda) \Big/ \sum_{i=1}^{N}\sum_{j=1}^{N} P(q_t = i, q_{t+1} = j, O_1^T / \lambda) \qquad (6.27)$$

$$= \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j) \Big/ \sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j) \qquad (6.28)$$

b) Counting state occupancy and transitions, we get

$$\gamma_t(i) = P(q_t = i \mid O_1^T) = \sum_{j=1}^{N} \xi_t(i,j) \qquad (6.29)$$

c) Using maximum likelihood (ML) for re-estimation of HMM parameters,

$$\hat{\lambda}_{ML} = \arg\max_{\lambda \in \Theta} P(O_1^T \mid \lambda) \qquad (6.30)$$

Then the model λ can be estimated as,

1. For i =1, 2, …N, let

$$\hat{\pi}_j = \gamma_1(i) \qquad (6.31)$$

= expected frequency (number of times) in state i at time t =1

2. For i =1, 2, …N, and j =1,2,…N, compute

$$\hat{a}_{ij} = \sum_{t=1}^{T-1} \xi_t(i,j) \Big/ \sum_{t=1}^{T-1} \gamma_t(i) \qquad (6.32)$$

$$= \frac{\text{Expected number of transitions from state i to state j}}{\text{Expected number of transitions from state i}}$$

3. For j=*1,2,...N* and k =*1,2,...M*, compute

$$\hat{b}_j(k) = \sum_{t=1}^{T} \gamma_t(j \mid o_t = v_k) \Big/ \sum_{t=1}^{T} \gamma_t(j) \qquad (6.33)$$

$$= \frac{\text{expected number of times in state j and observing symbols } v_k}{\text{expected number of times in state j}}$$

Re-estimation is an iterative process. The process can be summarized as:

1. Initialize, λ = (A, B, π)

2. Compute $\alpha_t(i)$, $\beta_t(i)$, $\gamma_t(i,j)$ and $\gamma_t(i)$

3. Re-estimate the model $\lambda = (A, B, \pi)$

4. If $P(O|\lambda)$ increases or reaches some desired threshold, go to 2, else stop.

The above solutions to the three problems are necessary when employing HMM to perform data analysis. Other than the above solutions, we have to deal with various issues related to the implementation of HMM to solve our problem. The issues include scaling, initial parameters estimations and effect of insufficient training data.

6.2 HMM Implementation Issues

The discussion above has dealt primarily with the theory of HMM and several variations on the form of the model. There are several important implementation issues that vary across applications. This section discusses the variation when employing the model to tackle news video segmentation problem.

- **Scaling.** In the Viterbi algorithm (not the log likelihood algorithm)**,** we need to perform scaling if the number of our observations is greater than 100 which is considered to be sufficiently large. Each video used in our experiments contains a large number of shots ranging from ~300-400 shots. Here, the content of each shot is represented as a feature vector, and is viewed as one observation. Thus, each of the input video comprises about ~300-400 observations to be used in estimating the model's parameters. We can see from the definition of $\alpha_t(i)$ of Eq (6.4) that $\alpha_t(i)$ consists of the sum of a large number of terms, each of the form

$$\left(\prod_{s=1}^{t-1} a_{q_s q_{s+1}} \prod_{s=1}^{t} b_{q_s}(o_s) \right)$$

With $q_t = i$ and b is discrete probability as defined by Eq (6.3). Since both a and b are less than 1, it can be seen that as t increases (e.g. to 10 or more) the dynamic range of $\alpha_t(i)$ computation will exceed the precision range of any machine. Thus, the only reasonable way to perform the computation is to incorporate a scaling procedure.

The basic idea of scaling is to keep the scaled $\alpha_t(i)$ within the dynamic range of the computer $1 \le t \le T$. The procedure multiplies $\alpha_t(i)$ by a scaling confident that is independent of i but depends only on t. A similar scaling is also done to $\beta_t(i)$ coefficients. Consider the computation of $\alpha_t(i)$. We use the notation $\alpha_t(i)$ to denote the unscaled αs; $\hat{\alpha}_t(i)$ to denote the scaled and iterated αs; and $\bar{\alpha}_t(i)$ to denote the local version of α being scaled. Initially, for t =1, we compute $\alpha_1(i)$ according to *Eq* (6.5) and set

$$\bar{\alpha}_1(i) = \alpha_1(i), \tag{6.34}$$

$$c_1 = \frac{1}{\sum_{i=1}^{N} \alpha_1(i)} \tag{6.35}$$

$$\hat{\alpha}_1(i) = c_1 \alpha_1(i). \tag{6.36}$$

For each t, $2 \le t \le T$, we first compute $\bar{\alpha}_t(i)$ and this $\bar{\alpha}_t(i)$ will be used to replace $\alpha_t(i)$ elsewhere in the relevant formulas where the term $\alpha_t(i)$ appears. Similarly, we

use the same scaling factors for each time t for βs as was used in αs. At the end of the process we have

$$\text{Log } [P(O|\lambda)] = \sum_{t=1}^{T} \log c_t \,. \qquad (6.37)$$

We can see that to avoid additional computation, we can use the log probability (alternate Viterbi implementation) to obtain the maximum likelihood state sequence. Thus, in this research, we chose the log likelihood algorithm to determine the maximum likelihood state sequence.

- **Initial Estimates of HMM Parameters.** From experience, it has been shown that either random (subject to stochastic and nonzero value constraints) or uniform initial estimates of π and A parameters are adequate for giving useful re-estimates of these parameters in almost all cases. However, for B parameters, experience has shown that good initial estimates are helpful in the discrete symbol case, and are essential (when dealing with multi mixtures) in the continuous-distribution case. Such estimates can be obtained in several ways; (a) manual segmentation of the observation sequence into states and averaging of observations within states; (b) maximum likelihood segmentation of observations and averaging; and (c) segment k-means segmentation with clustering, etc. The details of each method can be found in [Rabiner and Juang 1993].

Here, we estimate π and A by using uniform distribution. Thus for each trial experiment with different number of states (as we do not know which model best suit our observation data) , we assign:

$$\pi_i = 1/N ; \qquad a_{ij} = \quad 1/N$$

where N is the number of states. For example in the preliminary test on local news, we perform several experiments with states ranging from 2 to 9. For each selected number of states, we assign the initial values for the model parameters uniformly. Thus, if the number of states equals to 4, then we have

$$\pi_i = 0.25 ; \qquad a_{ij} = \quad 0.25, \quad i = 1, 2, \ldots, 4; j = 1, 2, .. 4$$

As for B, we adopt method (a) to manual segment the observation sequence into states and average the observations within states. The elaboration of computing the values of b_{jk} will be given later in the Section on the preliminary tests.

In order to have a right model to be used for the testing sequences, we estimate multiple models from multiple training sequences. We then test these models on the subset of the training set, the model that gives the best result or maximum likelihood probability in obtaining the given observation sequence is then selected.

Effect of Insufficient Training Data. Another problem associated with training HMM parameters via re-estimating methods is that the observation sequence used for training is finite. Thus, there is always an inadequate number of occurrences of low-probability events (e.g. symbol occurrences within states) to give good estimates of the model parameters. Here we consider the case of a discrete observation HMM. Recall that the re-estimation transformation of $\hat{b}_j(k)$ using Eq (6.33), it requires a count of expected number of times in state j and observing symbol v_k simultaneously. If the training sequence is so small that it does not have any occurrence of this event

(i.e., $q_t = j$ and $O_t = v_k$), $b_j(k) = 0$ and will remain zero after re-estimation. The resultant model would produce a zero probability result for any observation sequence that actually includes ($O_t = v_k$ and $q_t = j$). Such outcome is obviously a consequence of the unreliable estimate that $b_j(k) = 0$ due to insufficient data.

One solution to this problem is to increase the size of the training observations. However, this is not often practical. A second solution is to reduce the size of the model (e.g. number of states, number of symbols per state). Although this is possible, there are physical reasons why the model is used and thus its size cannot be changed. A third solution is to find statistical methods that can enhance the reliability of the parameters estimates even based on limited training data. Parameters threshold constraint is one of the alternatives. This is to ensure that no model parameter estimate falls below a specified level. For example, we may specify the floor for a discrete symbol model (such as our model), that

$$b_j(k) = \begin{cases} b_j(k), & \text{if } b_j(k) > \delta_b \\ \delta_b, & \text{Otherwise} \end{cases} \qquad (6.40)$$

For TRECVID data, after we obtain the semantic shot categories, there are 68 possible observation symbols. Although we have 20 training observation sequences each containing about 300-400 observations (symbols), it is found that some symbols do not occur in the training sequences. This causes some of the values of $b_j(k)$ to be zero. Thus, we need to set a floor value for $b_j(k)$. Follow Eq (6.41), we set δ_b to 0.001.

From observations, this is the minimum value of the values in *B* matrix independent of the number of states used. Therefore, Eq (6.41) becomes,

$$b_j(k) = \begin{cases} b_j(k), & \text{if } b_j(k) > .001 \\ .001 & \text{Otherwise} \end{cases} \quad (6.41)$$

Besides setting the threshold for model parameters, our two-level framework also significantly helps to minimize the data sparseness problem. As we can see in the shot classification stage, we have several features such as faces, shot type, audio classes, etc. If we use all these features into a vector to represent a shot for the HMM model, it will result in a larger space for possible observations, it will aggregate the above data sparseness problem. Therefore, by dividing the framework into two stages, the number of possible observation symbols required has been significantly reduced. As a result, we only have 3 features (shot categories, scene change, and cue-phrase) in the observation sequences, and thus greatly minimize the case of no occurrence of certain symbols in the training observations. We will carry out further experiments using a different method to verify that our two-level framework is superior to one level using the same set of features (see Chapter 7).

6.3 The Proposed HMM Data Model

6.3.1 Preliminary Tests

We first perform a small scale trial test to observe how well our system performs. This is to fine-tune our algorithm and try to reduce the errors that may occur during the large scale test. We carry out trial test for shot classification as discussed in the previous Chapter. The shot tag_IDs obtained from shot classification will be used here for story segmentation. In our trial test for HMM analysis, each shot is represented by: (a) its tagged category (a unique number is given to each category), t_i; (b) scene/location change (*Sc*), s_i (c = change, u = unchanged); and, (c) speaker changed (*Sp*), p_i (c = speaker changed, u = unchanged). The feature vector is expressed as:

$$S_i = (t_i \ s_i \ p_i) \tag{6.42}$$

In the preliminary experiments, the system was tested on news video data from Mediacorp Singapore. The data consists of two days of news, each containing about 400 shots. The video shots were first classified into 13 categories as explained in details in Section 4.1.2 (except the five additional program logos). Thus, each output symbol is represented by 1 of the 13 possible categories of shots (refer to shot category ID in Appendix A), 1 out of 2 possible scene change feature values, and 1 out of 2 possible speaker changed feature values (note that in our earlier work, cue-phrase was not included in the feature set). This gives a total of 13x2x2 = 52 distinct vectors for modeling using the HMM framework. In most applications, it is hard to have prior knowledge for the number of hidden states. Thus, we perform the experiments by varying the number of states from 2 to 9 to evaluate the results. The initial experiments indicate that the number of state equals to 4 gives the best result. Figure 6.4 illustrates an ergodic HMM with 4 states used in our approach.

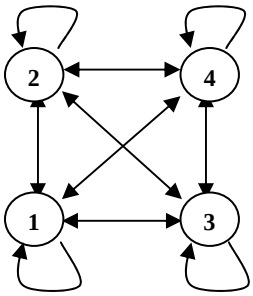


Figure 6.4: The ergodic HMM with 4 hidden states

When 4 states are used, we need to estimate b_{jk} for 4x52 = 208 probabilities. Since it is ergodic model, we can use uniform distribution as discussed above to give initial values to the model parameters. Thus, for the 4-states HMM, we have

$$\pi_i = 0.25;\ i = 1, 2, 3, 4$$

$$A = \begin{pmatrix} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \end{pmatrix}$$

As for *B*, we manually segmented the training sequence evenly according to the number of states, each segment has a duration *T/N* (*T* is the length of the sequence and *N* is the number of states). Consider a 4-state model with *T* equals to 12. State 1 corresponds to the first 3 observations, state 2 to the second 3 observations, and so on. For the specific example in our case in which 4-state is used, the observation sequence (part of the whole sequence from local news video data) of length 12 is: *1c0 1u0 1u0 1u0 2c1 4c0 4u0 6c0 6u0 2c1 10c0 10u0.* Here, we have symbols *1c0 1u0 1u0* in state 1, symbols *1u0 2c1 4c0* in state 2, *4u0 6c0 6u0* in state 3 and *2c1 10c0*

10u0 in state 4. We simply compute the probability of each symbol in each state, we get *P*(state =1, symbol = *1c0*) = 0.33 and *P*(state =1, symbol = *1u0*) = 0.67, etc.. The rest of the probabilities are computed in the same manner. A summary of the probabilities computed is listed in Table 6.1.

Symbols / state	*1c0*	*1u0*	*2c1*	*4c0*	*4u0*	*6c0*	*6u0*	*10c0*	*10u0*
1	0.33	0.67	0	0	0	0	0	0	0
2	0	0	0.33	0.33	0.33	0	0	0	0
3	0	0	0	0	0.33	0.33	0.33	0	0
4	0	0	0.33	0	0	0	0	0.33	0.33

Table 6.1: B matrix associated with the observation sequence. Value in each cell is the probability the symbol comes from the corresponding state.

6.3.1.a Results of the Preliminary Test

We conducted three experiments (Ex I, II, and III) for story boundary detection. As explained in Section 4.3.2 a, our initial experiments indicate that the number of states equals to 4 gives the best results. Thus, we set the number of states to 4 in these three HMM tests. For **Ex I**, we assume that all the shots are correctly tagged. We perform the HMM to locate the story boundaries and we could achieve a F_1 value of 93.7%. This experiment demonstrates that HMM is effective in news story boundary detection. **Ex II** is similar to **Ex I** except that we perform the HMM analysis on the set of shots tagged using the earlier shot classification stage with about 5% tagging error. The test shows that we are able to achieve an F_1 measure of 89.7%. The results of both tests are presented in Table 6.2.

Ex	NB	NC	FN	FP	R (%)	P (%)	F_1 (%)
I	40	37	3	2	94.9	92.5	93.7
II	40	35	5	3	***87.5***	***92.1***	***89.7***

Note: NB = total number of correct boundaries, NC – number of corrected detected boundaries, FN – number of false positive and FP – number of false positive.

Table 6.2: Results of HMM analysis of tests Ex I & II

In **Ex III,** we want to verify whether it is necessary to perform the two-level analysis in order to achieve the desired level of performance. We perform HMM analysis on the set of shots with their original feature set but without the category information. We vary the number of features used from the full feature set to only a few essential features. The best result we could achieve is only 37.6% in F_1 value. This test shows that although in theory a single stage analysis should perform the best, in practice, because of data sparseness, the 2-level analysis is superior.

6.3.1.b Effectiveness of the Features Selected for HMM Analysis

In order to evaluate the importance of each feature used in **Ex II**, we perform another set of experiments using only the individual feature one at a time, and by adding the second and third feature to the Tag-ID feature. Table 6.3 presents the results of this experiment.

Feature/s	NC	FN	FP	R(%)	P(%)	F_1 (%)
Tag_ID	35	5	6	87.5	85.4	86.4
Sp	35	5	93	87.5	27.3	41.7
Sc	26	14	90	65	22.4	33.3
Tag_ID+Sp	37	3	7	92.5	84.1	88.9
Tag_ID +Sp+Sc	35	5	3	87.5	92.1	89.7

Note: NC – number of correct boundaries, FN – number of false negative,
FP – number of false positive, R – Recall, and P – Precision.

Table 6.3: Results of the analysis of Features Selected for HMM

Figure 6.5 shows the precision and recall values of the results when using different combinations of features.

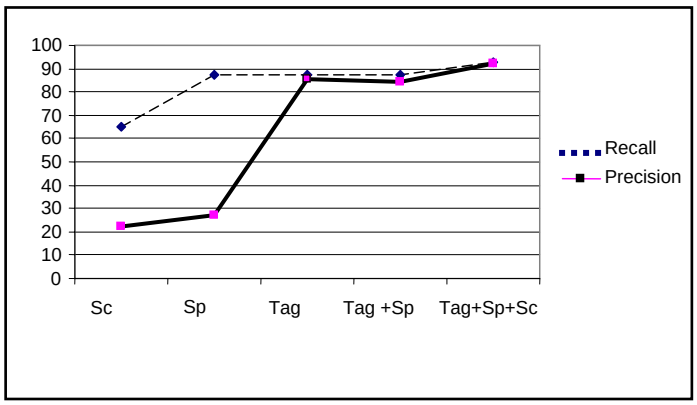


Figure 6.5: Precision and recall values of the result from EX II.

From Table 6.3, it can be seen that when using only the Tag-ID feature, the system could achieve an F_1 measure of 86.4%. On the other hand, the use of the second and third feature alone returns low F_1 measures of 41.7 and 33.3 respectively. However, by combining the last two features with the Tag-ID feature, the system's F_1 performance improves gradually from 86.4% (with Tag-ID as the only feature) to 88.9% (Tag-ID +Sp), and reaches 89.7% when all the three features are included (Tag-ID +Sp +Sc). The analysis indicates that the first feature (Tag-ID) is the most important feature for story boundary detection. It further confirms that shot classification facilitates the detection of news boundaries, and therefore our two-level approach is effective.

6.3.1.c Analysis of HMM results

The HMM parameters obtained from training the HMM is given below.

The HMM model parameters:

A:
0.927110 0.001050 0.001062 0.073777
0.001048 0.923241 0.001081 0.077629
0.001110 0.001020 0.852544 0.148326
0.475093 0.144626 0.361021 0.022260

B:
0.001000 0.001000 0.001000 0.001000 0.009450 ……
0.001000 0.001000 0.001000 0.001000 0.023350 ……
0.141763 0.329531 0.024386 0.165216 0.023980 ……
0.001176 0.001199 0.001215 0.001225 0.728444 …….

π = 0.001000 0.001000 0.999996 0.001004

We can see that the begin state is State 3 (π = 0.99999). The transition probabilities (coefficients), $a_{ij,}$ are positive for all *i* and *j*. Thus, these properties satisfy the ergodic model.

***I_1*:**	**1cc**	**1uu**	**1cu**	**….**	**1uu**	**2cc**	**4cc**	**4uu**	**6uu**	**6uu …**
O_1:	3	3	3	…	3	*4*	1	1	1	1 …
I_2:	***6uc***	***6uu***	***6uu …***		***6uu***	***10cu***	***10uu***	***10uu***	***10uu …***	
O_2:	1	1	1	….	1	*4*	3	3	3 …	

Note: I_i means observation sequence I, O_i means output state sequence i

Figure 6.6: Two examples of the observation sequences and their output state sequences

From the result, it is found that, for a sequence of shots within the same story, the HMM outputs the same state number. For instance, the observation and output state sequences in the example shown in Figure 6.6 is State 3, which is the output state for all shots of *Intro/highlight* category even though the observation symbols are different (refer to the example of the observation sequence above). State 4 is the output state for *Anchor* shot. However, when the framework is applied to the

TRECVID data, the HMM output states within a story have a great change such that there are several output states within a story.

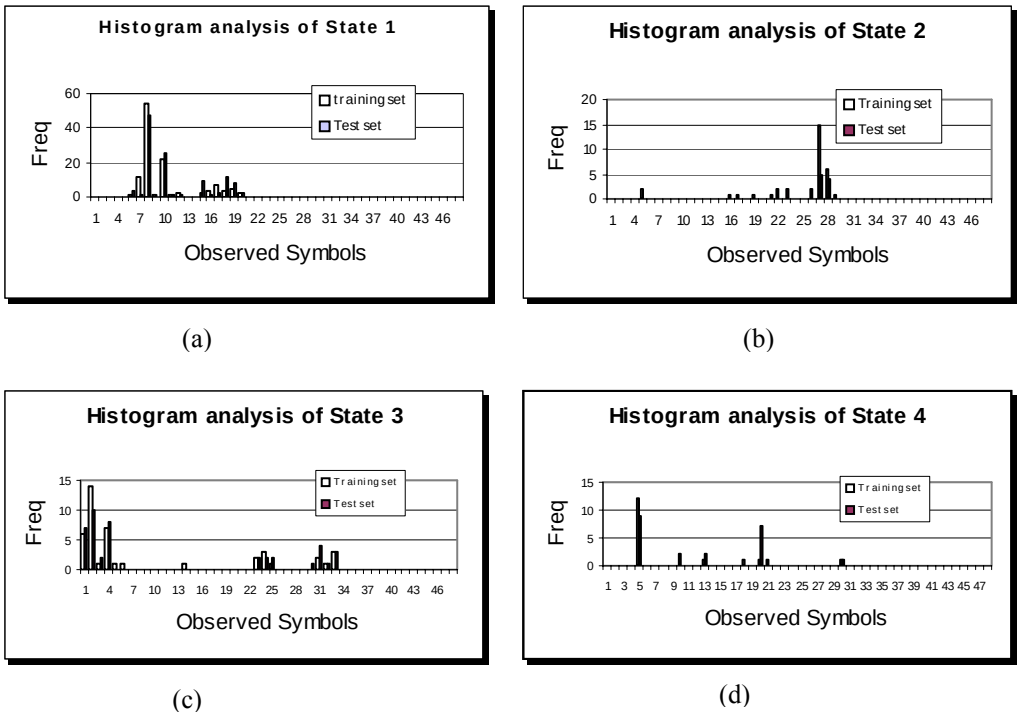


Figure 6.7: Present the distributions of the observed symbols of the 4 states

In order to compare the distribution of the observation symbols for the training and test data sets, we further analyze the distribution of the observation symbols in each of the 4 states as shown in Figures 6.7(a) – 6.7(d). From the Figures, we can see that for each state, the distribution is dominated by a few (sometime only one) symbols, and that the frequency distribution of the same symbol is similar in both the training and test data sets. For example, in Figure 6.7(a), which shows the distribution of *State 1*, the distribution is dominated by symbols 8 with the frequency of about 50 out of 183 shots for both the training and test data sets. The corresponding observation vector for this symbol is ***6uu*** which represents a Live-reporting shot. Figure 6.7 (b)

shows that the distribution of symbols in *State 2* is dominated by symbol 27. It corresponds to feature ***8uu*** that represents a sport shot. Figure 6.7 (c) shows that the distribution of *State 3* is dominated by symbols 2 and 4, which correspond to feature vectors ***1uc*** and ***1uu*.** Both vectors represent Intro/highlight shots. Finally, the distribution of symbols in *State 4* as shown in Figure 6.7(d) is dominated by symbol 5. It corresponds to feature vector ***2cc*** that represents an Anchor person shot.

6.3.2 HMM framework on TRECVID 2003 data

General news video has a similar structure as discussed in Chapter 2. However, in CNN news, the structure is more complex and contains a greater variety of programs. Therefore, in order to cope with the data provided by TRECVID which contains about 120-hours of ABC and CNN news video, some new features and shot categories have to be added or adjusted. This process is necessary when dealing with a new data source. However, in general, the 13 categories as used in the above test work well for most of the news video.

Based on the characteristic of the training set, we found that five additional categories need to be included. They are, *"LEDS",* "*HEALTH, "TOP"*, *"SPORT",* and *"PLAY*", as detailed in Chapter 4. Still-image category is dropped for this test as there are always moving texts at the bottom of the videos even for still images. Speaker changed feature is also dropped for this test as it is found to have no contribution to the system performance. In addition, cue-phrase feature is incorporated for story

segmentation process. The system is then re-trained to cope with the new categories and features.

As our framework is designed for general news video, we train one model each for the data from CNN and ABC news video. Figure 6.8 illustrates the training (using the **Baum-Welch** algorithm) and testing processes (using the **Viterbi** algorithm) of the HMM framework. The training process is to estimate the optimum models for both CNN and ABC data. The parameter initialization is done in similar way as in the preliminary test. From the output states, we collect all states that indicate story transitions.

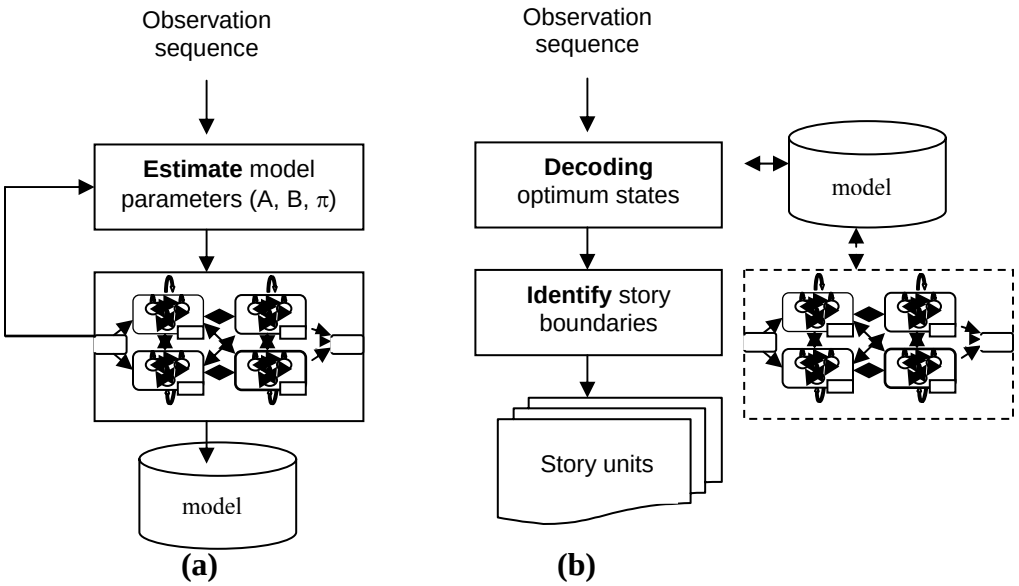


Figure 6.8: (a) A Training steps of the HMM framework and (b) Decoding (testing) steps of the HMM framework

Given the test observed sequence, the algorithm applies the trained model to the test data. It decodes the observed sequence and estimates the optimum output state sequence. After we have obtained the output state sequence for each of the observed

sequences (we have about 120 test sequences to decode), the algorithm then identifies story boundaries by detecting transition states. From the training data, we collect the ground truth for story boundaries in which we could match the boundaries with the states. Those states that indicate story transitions are then used for identifying story boundaries. The test process is illustrated in Figure 6.8(b).

The following shows an example of an observation sequence from the CNN news video data when using the full feature set. The sequence is similar to that of the preliminary work except that there are more possible observation symbols (as there are more shot categories incorporated).

Observation sequence:

1c0 1u0 … 2c1 4u0 6u0 … 15c0 6c0 5u0 13c0 13u0 …

The evaluation results and the analysis of the errors are discussed in the following Sections

6.3.2.a Evaluation results

For the test data, there are about 60 hours of news videos comprising of 2929 story boundaries. As part of the requirements from TRECVID 2003, we need to perform story segmentation using different combination of features. This is to test the effectiveness of using different features for this task, in particular, to illustrate the contrast between text-based ASR features, the audio-visual (AV) features and their combination. The feature sets used are: (a) only AV features; (b) only ASR feature; and (c) a combination of AV and ASR features (AVT). Under the three requirements, we conduct five experiments as follows: Runs 1 and 3: using only AV features; Runs

2 and 4: using AVT features ; and Run 5: using only ASR features. For the first four runs, we employ the HMM framework to locate story boundaries. For each feature set AV and AVT, we train the system using different numbers of hidden states. From the experiments, the number of hidden states equals to 8 gives the best results for AV feature set and the number of hidden states equals to 9 gives the best results for AVT feature set. For each feature set, we train a HMM model to be used in our tests. Thus, we have one model for AV feature set and another model for AVT feature set. We need to estimate b_{jk} for 8x17x2 = 272 probabilities, when 8-state is used for the AV feature set, and need to estimate b_{jk} for 9x17x2x2 = 612 probabilities when 9-state is used for the AVT features set. Parameter initialization and estimation are done in the similar way as in the preliminary test.

Figure 6.9 shows an example of an observation sequence and the associated output state sequence from the HMM when using the AVT feature set. The detection of story boundary is the same as is done in the preliminary test, except that transition states that indicate story changes are different. For Run 5, we perform text segmentation [Li 2001] based on the sequence of text from the ASR.

I_3	*1c0*	*1u0*	*1u0*	*2c1*	*4c0*	*4u0*	*6u0*	*13c0*	*13u0*
O_3	**5**	**3**	**3**	**4**	**1**	**1**	**1**	**7**	**9**

Figure 6.9: Example of observed symbols and output state sequences when using the AVT feature set.

From Figure 6.9, State 4 corresponds to the symbol *2c1* which is of type *Anchor* shot with scene changed and presence of cue-phrase. State 7 corresponds to symbol, *13c0,*

which is a commercial shot. We can see that the next commercial shot (also symbol *13c0*) corresponds to State 9. However, this State 9 is an internal state within a commercial block, and thus it does not indicate story changes. Our algorithm therefore ignores this state during the detection of story boundaries. The evaluation results of story segmentation for the five runs on the test data containing 2864 story boundaries are presented in Table 6.4.

From Table 6.4, we can see that when using the AV feature set, we could achieve an F_1 value of 75% (Run3) and when using the AVT feature set (Run 4), an F_1 value of 77.5%. This demonstrates that the ASR based feature helps to improve the result by 2.5%. The last row presents the result of story segmentation using only text feature which could achieve an F_1 measure of only 53% (Details of text segmentation are not covered and are beyond the scope of this thesis). As compare to the results when using multi-modality features such as in Runs 1 – 4, it is obvious that the use of text feature alone is insufficient for story segmentation task.

Run	Feature	DeTBD	DeTCoRR	FoundinTruth	Recall	Precision	F_1
1	AV	2913	2174	2123	0.741	0.746	**0.743**
2	AVT	2818	2218	2177	0.76	0.787	**0.773**
3	AV	2806	2150	2102	0.734	0.766	**0.750**
4	AVT	2724	2185	2146	0.749	0.802	***0.775***
5	ASR	2426	1417	1398	0.488	0.584	**0.532**

DeTBD – no. of detected boundaries; DeTCoRR – no. correct detected boundaries; FoundinTruth – no. detected boundaries found in the truth data; Recall = FoundinTruth/Total boundaries), Precision = DeTCoRR / DeTBD)

Table 6.4: Results of story segmentation on this corpus

In order to compare our system with other state-of-the-art systems taking part in TRECVID 2003 story segmentation task [TRECVID 2003], Figure 6.10 presents the best results based on different set of features. We can see from Figure 6.10 that our system achieves the best performance in cases when using only audio-visual based features (AV) or when using the full set of features (AV+Text). These results demonstrate that our HMM framework is effective and approaches a level of performance towards practical deployment of story segmentation system. Also, we can see from the results of the two top performing systems (IBM_CU and NUS) that by using text feature alone is insufficient to achieve high accuracy as compared to using the full set of features.

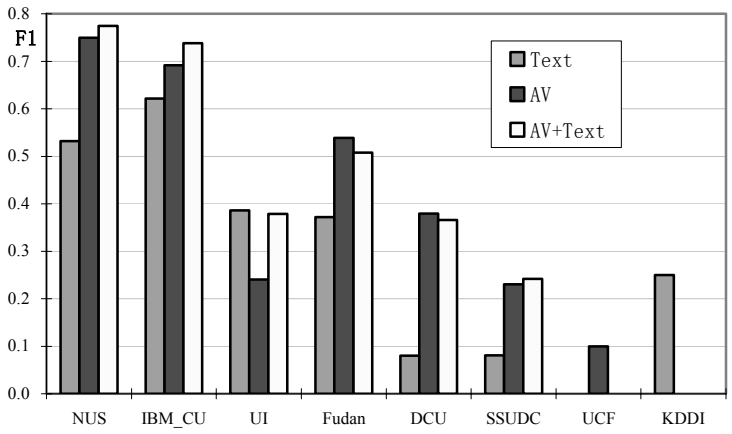


Figure 6.10: Presents the best results achieved by each group.

6.3.2.b Analysis of results from the HMM

Here we analyze the distribution of the types of stories detected, the errors incurred in identifying story boundaries, and the effect of the number of states *N*.

♦ **Error Analysis of the story segmentation result**

From observations, stories found in CNN corpus can be grouped into 7 types as shown in Figure 6.11.

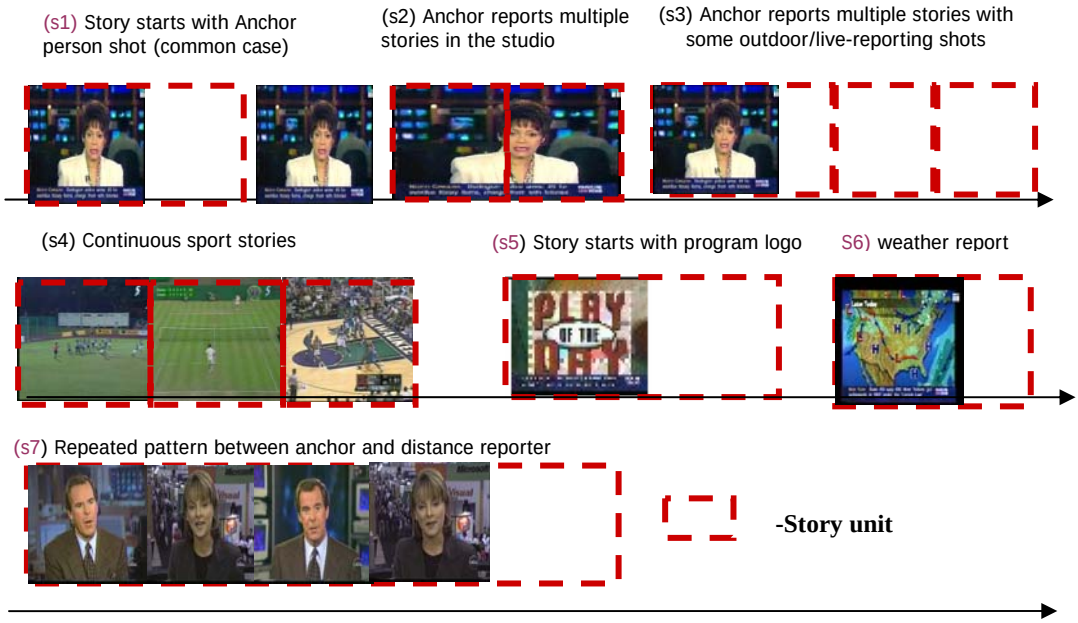


Figure 6.11: General stories found in CNN corpus

We present only the analysis of CNN data as the structure for ABC data is simpler and causes fewer problems to our system. The most common type of news story is of type s1, which starts with the anchor shot and ends before the next anchor shot. This type of stories covers about 60% of the total stories. The types of stories that are easy to handle are s5 and s6, which can be detected by using visual matching techniques. The types of stories that also start with anchor person include *s2* and *s3* (which cover multiple stories) and s7 (which is a repeated pattern between anchor and distance reporter and it covers only one story). The structures of these types are more

complex; additional techniques are needed to handle anchor person reporting multiple stories. Another type with multiple stories is *s4*: the sports. It causes the most detection problem since we do not have features to differentiate different sport types.

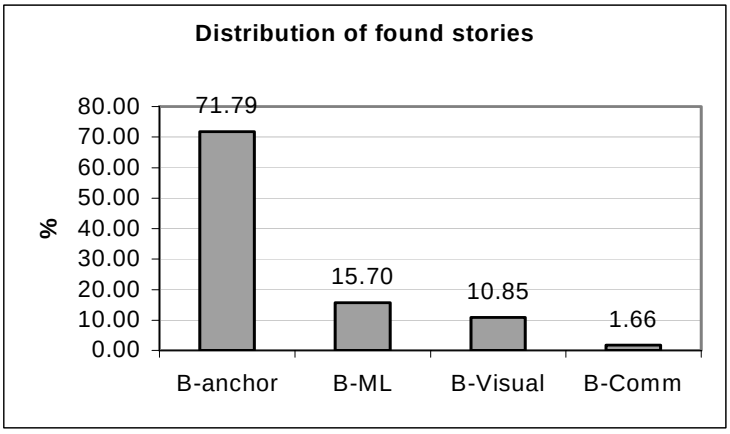
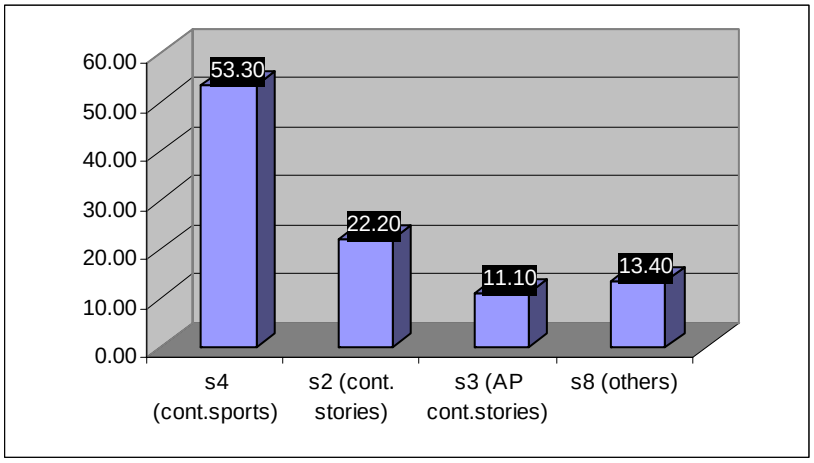


Figure 6.12: Presents histogram of the distribution of found stories

Figure 6.12 presents the distribution of the types of stories found by our system. We compute the distribution of the stories in terms of shot category appearing at the beginning of each story. The most common type of stories begins with anchor person shot (represented by B-anchor in the Figure) which accounts of 72% of all stories. For the rest of 28% of stories, we divide them into 3 clusters based on our techniques used to detect shot categories. They are: (1) B-Visual: for the stories that begin with shot category of visual similarity type; (2) B-Comm: for stories that begin with Commercial shot type; and (3) B-ML: for stories that begin with shot of machine-learning type. B-Visual are stories that begin with shots of categories *LEDS, Weather,* and *HEALTH*; and B-ML are stories that begin with shots of categories

Intro/Highlight and *Sport.* As can be seen, B-anchor is the most common type, follow by B-ML and B-Visual.

From the error analysis of our HMM framework, there are four main types of incurred errors as shown in Figure 6.13. The details of each type are given below.



Note: AP – Anchor person shot

Figure 6.13: The error analysis result of the total error rate 22.5%

(a) Error in detecting multiple sport stories. This error accounts for more than 50% of total errors (as can be seen n in the Figure). This is because; our system did not incorporate the right method to tackle the problem of individual sport reports. The HMM can detect a transition of news story (entering into sport news). But it detects the whole chunk of several sport reports as one news report.

(b) The second source of errors is from detecting multiple live-reporting stories without any clues of story break (s2). This occurred in CNN news video such as session called “TOP stories”. The reporter reported multiple stories containing mostly

shots of live-reporting categories in which story transitions could not be found by the HMM. This error accounts for about 22% of total errors.

(c) It is often that the reporters (in *Anchor* shots) report several news stories in the studio without any outdoor/indoor or live-reporting shots. Thus there is no change of shot between stories, and our system failed to find such story breaks. This error accounts for about 11% of total errors.

(d) Another type of error is called un-seen pattern error (s8). During the test phase of story segmentation, many story patterns that were not discovered by HMM during training. This is because such "unexpected" patterns did not occur sufficiently frequently in training data for HMM to learn the patterns, which is a case of data sparseness problem. One typical story pattern learned by HMM is: *Anchor* shot (Tag-ID 2) followed by a remote reporter which is under *Speech/Face* category (Tag-ID 5), and *2Anchor* (Tag-ID 3), or the Tag-ID sequence; 2 5 3, which is considered as one story. However, there are patterns of Tag-ID sequence such as: *2 5 3 2 5 3 2 5 3* found during testing that should belong to one story. However, for this kind of output from the HMM, our algorithm which is a straight forward algorithm in detecting the state transitions, detected that as 3 stories. This leads to over segmentation of stories in such cases. We estimate that this accounts for about 13% of total errors. One approach to overcome this problem is to introduce heuristics. But a better solution is to investigate the use of higher order statistics.

♦ Effects on number of states N

To illustrate the effect of varying the number of states in our story segmentation model, Figure 6.14 shows a plot of story boundary error rate versus *N* (x-axis), the number of states in the HMM model. The experiment is based on the full set of feature AVT on the test data. As we can see, although we obtained local minimum at *N* = 3, 6 and 9, however, we found that the system is more stable when *N* = 9 at which the system gives the minimum error. Therefore, states =9 is adopted in our model.

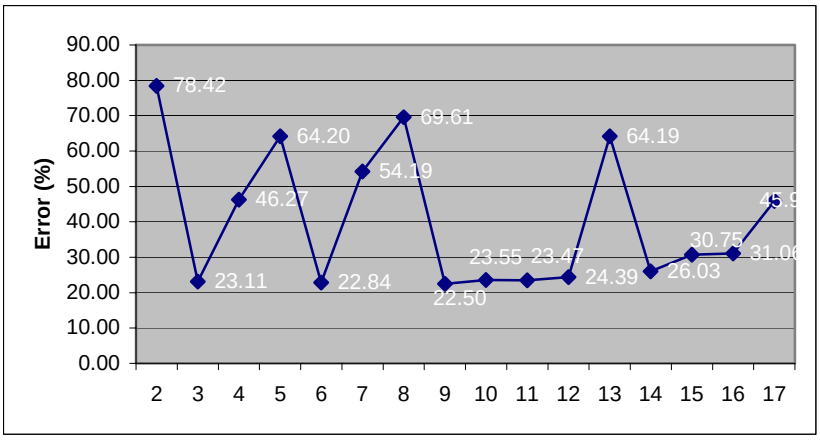


Figure 6.14: Average story boundary error rate versus the number of states N in the HMM

♦ Association of news video sequence with the HMM output states

From the observations on the news video sequences and on the HMM output states, a graphical figure to represent news sequence model and HMM output states for story boundary identification is shown in Figure 6.15.

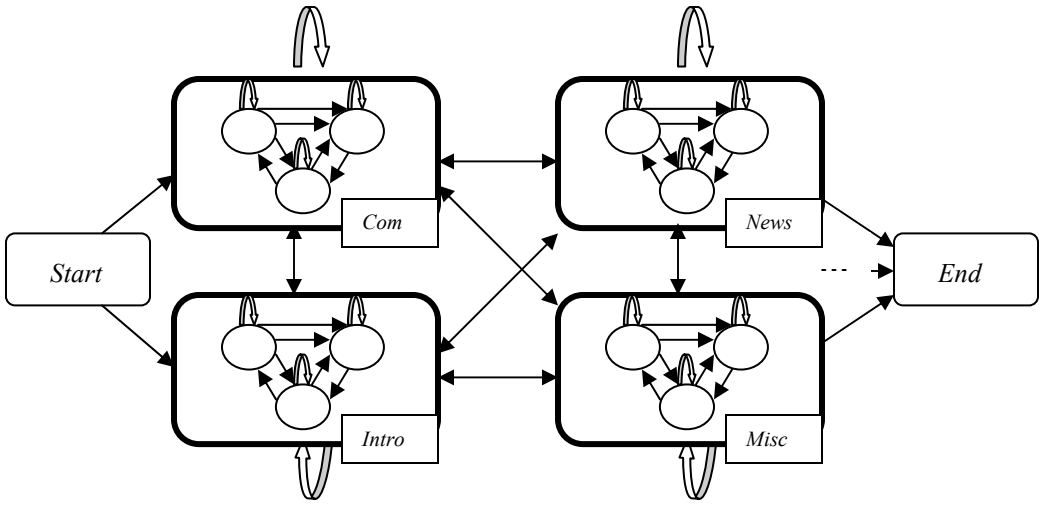


Figure 6.15: HMM architecture of news story segmentation.

There are six main stages (note that it is not "state") that the HMM may enter. There are: *Start_news* stage, *commercial* stage (*Com*), *intro/highlight* stage (*Intro*), news (*News*) stage, miscellaneous (*Misc stage*) and *End_news* stage. For each stage, there is a series of HMM output states beginning with "start" and ending with "end" states. The "start" and "end" states of each of the stages may not be the same. The stages of the HMM can be summarized as follows.

1. ***Start_news.*** This stage occurs only once in each of the input news sequence. Following this stage, the system may either enter the Intro/highlight (*Intro* in the figure) or commercial (*Com*) stage. If it enters *Intro*, it then moves to *News* stage directly. But if it starts with *Com* it may move to *Intro* before entering the *News* stage.

2. ***Intro.*** In general news broadcast, there are 3-4 blocks of intro/highlight, at the beginning of news broadcast and before the commercial breaks. This intro/highlight block contains several shots of *Intro/highlight* category. The system may exit from *Com, Misc*, or *News* stage before it comes to this stage (for the 2nd time).

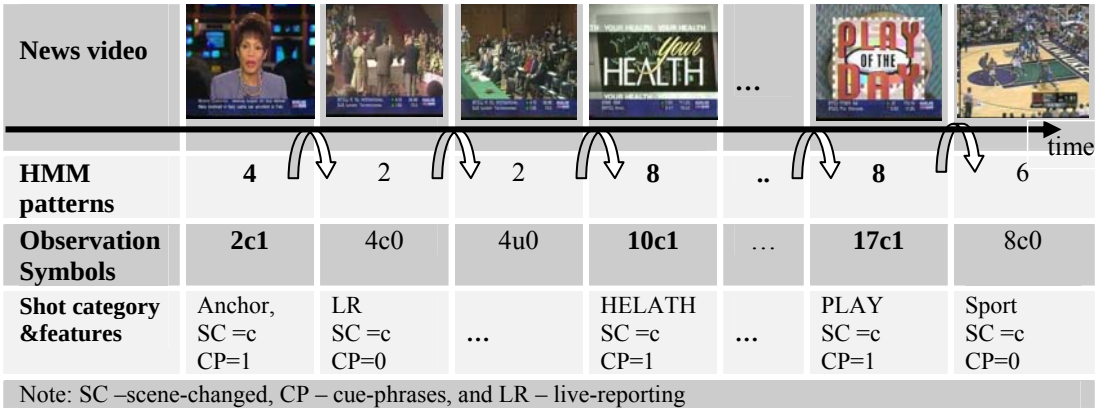


Figure 6.16: The relationship between HMM output states and the observation symbols of the test data.

3. ***News.*** The system may stay in the *News* stage for some repeated iterations before entering one of the other three stages.

4. ***Com.*** There are several commercial blocks within news broadcast especially for CNN news. Except for ***Com*** that appears at the beginning of news broadcast, it may be entered from *News*, *Intro* or *Misc* stage.

5. ***Misc.*** This is the non-news story segments such as the reporters' chitchats (mostly from *2Anchor* shots when switching topic) or several *Live-reporting* shots (before commercial break) but with no coherent contents. The system stays here for a short duration and goes into other stages again.

6. ***End_news.*** The news reporting sequence repeats between 3, 4, and 5 until it reaches the end of the news program.

The stages discussed above are the main stages to represent a sequence of news reporting for news video covered in this research. Figure 6.16 shows the relationships

between the HMM output states and the corresponding observation symbols of the test data.

6.3.3 Classification of News Stories

Another task that is defined in TRECVID 2003 is story classification. After story boundaries are identified, we need to classify the detected stories into the classes of "news" or "misc". For this task, we use the assigned tag of the first shot of each story, time constraint, and heuristic rules to perform the classification.

Common rules for ABC & CNN news

rule 1. if (Curr = COMMERCIAL), then the story is "misc"

rule 2. if (Curr = LEDS), then the story is "misc";

rule 3: if (Curr = Intro/Highlight), then the story is "misc";

rule 4. if (Curr = ANCHOR) and (Next = LEDS) and story duration <=TOLERANCE, then the story is "misc";

rule 5. if (Curr = ANCHOR) and (Next = COMMERCIAL) then the story is "misc";

rule 6. if (Curr = ANCHOR) and if story_dur <=TOLERANCE), then the story is "misc" , else the story is "news";

rule 7. if (Curr = 2ANCHOR) and (story duration <= TOLERANCE), then the story is "misc";

rule 8. if (Curr = OTHERS), then the story is "news";

(a)

Specific rules for CNN

rule 1: if (Curr = ANCHOR) and ((Next = WEATHER) or (Next = HEALTH) or (Next = 2ANCHOR) or (Next = Intro/Highlight)), then the story is "misc";

rule 2: if (Curr = SPORT), then the story is "news";

rule 3: if (Curr = WEATHER), then the story is "news";

rule 4: if (Curr = HEALTH) and (Next = HEALTH) then the story is "news";

rule 5: if (Curr = TEXT-SCENE) and (Prev = sport) then the story is "misc";

(b)

Figure 6.17 presents the simple rules for classifying the detected stories into the desired class. (a) Common rules for both CNN and ABC news and (b) specific rules for CNN news.

Figure 6.17 presents simple rules for classifying the detected stories into the desired class. For each story of the training data (either they are under "misc" or "news" stories), we collect average duration of a story, category of the first shot of the current

story, category of the first shot of the next story and category of the first shot of the previous story. We then derive simple heuristic rules as shown in Figure 6.17 to perform news classification. Note that, the rules are formed based on the definitions of "news" and "misc" as observed in the news videos provided by TRECVID 2003. Most of the rules can be applied to news from other sources too. For example, the rule: 'if the first shot of the detected story is commercial shot, then this story is "misc" ', is always true for all known news broadcasts. However, in order to achieve similar level of the performance, other broadcast stations should have similar sequences of news categories, otherwise, further study on the new data set and learning of new rule set are required.

Results of our news classification for Runs 1 – 5 are presented in Table 6.5.

Run	**Feature**	**Story Classification**		
		Recall	**Precision**	**F_1**
1	F1	0.937	0.939	**0.938**
2	F3	0.925	0.963	***0.944***
3	F1	0.918	0.953	**0.935**
4	F3	0.916	0.965	**0.940**
5	F2	0.921	0.773	**0.841**

Table 6.5: Result of news classification on this corpus

Figure 6.18 shows our results as compared to other participating groups under this task. Note that, some groups may submit the result only based on one or two of the required feature sets. We can see from Figure 6.18 that, our classification algorithm performs the best under all required feature sets. The best result of our system is

achieved when using the complete set of features, which is the combination of audio, video and ASR features; which we could obtain an F_1-value of over 94%.

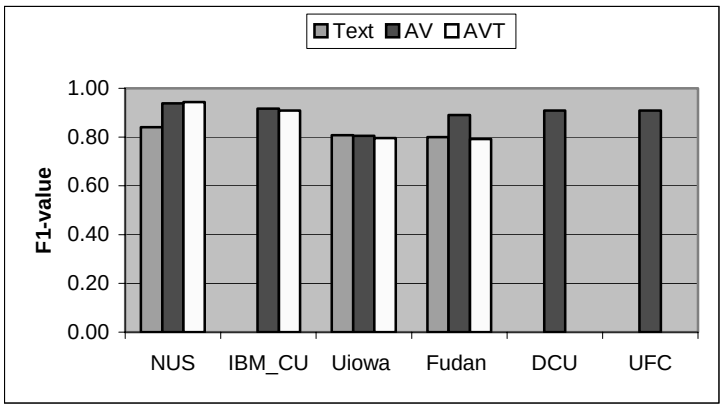


Figure 6.18: The results comparing to other participating groups

The next chapter discusses another approach to detect story boundaries based on rule induction technique. The framework for rule induction is based on our first level output shot categories, together with temporal feature set as that used in the HMM framework. The purpose of introducing this approach is to compare its performance to that of HMM approach on news segmentation on TRECVID data. Another reason is that we found the HMM approach to be more complex to understand and maintain, and we want to explore the rule induction method which is simpler and easier.

CHAPTER 7

GLOBAL RULE INDUCTION APPROACH

This chapter provides an overview of a global rule induction model called GRID (Global Rule Induction for text Document) and discusses the application of GRID to news story segmentation. It presents the results of story segmentation using GRID and compares the result to that of the HMM approach. Finally, it presents the error analysis of the results.

7.1 **Overview of GRID [**Xiao 2003]

GRID is designed to extract rules to perform information extraction (IE) from text documents. It emphasizes the use of global feature distribution in all of the training examples in order to make better decision on rule induction. It examines all the training instances at the representational levels of lexical syntactic and semantic simultaneously, and selects a global optimal feature to start the rule induction process. In addition, it adopts information chunks as units to determine the context of rules. It is of higher syntactical level than word or token (in text document), and thus it provides a more appropriate unit to model the context. The features used by GRID are general and applicable to a wide variety of domains, ranging from semi-structured corpus to free-text corpus.

7.1.1 GRID on Text Documents

GRID is a supervised rule induction algorithm that learns from training documents in which sentences have been tagged for specific slot types such as the name of speaker, venue in a seminar announcement etc. The tagged sentences (referred to as training instances) of each type are regarded as positive examples, while the remaining sentences in the documents are regarded as negative examples. GRID uses information chunk derived from shallow parsing as the basic granularity of context information. This is to avoid the difficulties in deciding slot boundaries if words were to be used as units for context. Both training and testing documents are pre-processed by the same basic NLP modules such as the sentence splitter, tokenizer and morphological analyzer etc. This step aims to perform syntactic analysis to generate information on Part-of-Speech (PoS), noun group and verb group chunking. The algorithm then derives the semantic classes of some noun groups, such as person, organization, location, and time etc. This is done by using a rule-based named entity recognition module, which employs global information such as acronyms, sequence of initial capitals, initial capitals of other occurrences, organization suffixes and person prefixes of other occurrences from the whole document to resolve the ambiguous words that could be person name, organization or others.

Further details on global information can be found in [Xiao 2003]. The following Sections discuss details of data representation and details of how GRID performs rules extraction in text documents.

7.1.2 The Context Feature Vector

For every tagged training instance, GRID generates a context feature vector centered around the tagged slot. The context feature vector is of the general form:

$$<c_{-k}> \ldots <c_{-2}> <c_{-1}> \textit{tagged\_slot} <c_{+1}> <c_{+2}> \ldots <c_{+k}> \quad (7.1)$$

Here $<c_i>$ $\{i=-k \text{ to } +k\}$ represents the context units of the tagged slot, and k is the number of context units considered. $<c_i>$ can be a token, a noun or verb phrase or even a syntactic unit such as subject or object and it can be of various feature types, including: words, PoS (if it's a single token); various types of verbs and noun chunks, and semantic classes. The context feature vector for a single tagged instance can therefore be represented as follows:

$$<(-k,f_{-k}^{1}), .., (-k,f_{-k}^{m}),.., (-1, f_{-1}^{1}),.. , (-1, f_{-1}^{m}), (0,f_{0}^{1}),.., (0,f_{0}^{m}), (1,f_{1}^{1}),.. , (1,f_{1}^{m}), .., (k,f_{k}^{1}),.., (k,f_{k}^{m})> \quad (7.2)$$

where m is the number of linguistic features for each element.

As shown in Equation (7.2), each element is represented as a tuple (g, f_g^i). The first part of the tuple, i.e. g, indicates the position of the element within the tagged instance. $g=0$ gives the position of tagged slot, and positive g (or negative g) gives the g^{th} right (or left) hand context element from the tagged slot. If there are m features and k context elements, then we have a context vector of size $(2k+1)\times m$.

The second part of the tuple, f_g^i, gives the possible appropriate linguistic representation for each element. There are 12 features (m=12) to be used. They are lexical, syntactic and semantic features. The details are given in Table 7.1. These

representations are stored as string type. For NP and VP, the head noun and root verb are also stored.

Feature	Description	Feature	Description
f_g^1	Lex. String	f_g^2	PoS
f_g^3	NP_Person	f_g^4	NP_Org.
f_g^5	NP_Loc	f_g^6	NP_Date
f_g^7	NP_Time	f_g^8	NP_Perc.
f_g^9	NP_Mon.	f_g^{10}	NP_Num.
f_g^{11}	VP_Pass.	f_g^{12}	VP_Act.

Table 7.1: Features that GRID employed

7.1.3 Global Representation of Training Examples

Given a cluster of training instances of a specific slot type, GRID generates a context feature vector for each instance using Eq.(7.2). By arranging all the instances using Eq.(7.2) and feature set in Table 7.1, we obtain a global context feature representation for the training documents as shown in Figure 7.1. The occurrences of the common element features at a specific position g are cumulated as e_{gi}. We can see from the Figure that we can easily obtain the global distribution frequency of any element feature and at any position. Thereafter, a set of the instances covered by any feature set $\underline{f}$ can be derived. As different features play different importance in various domains, for example, in the free text terrorist attacks corpus, verb features play crucial roles, thus for each feature, a weight coefficient β_{gi} is assigned.

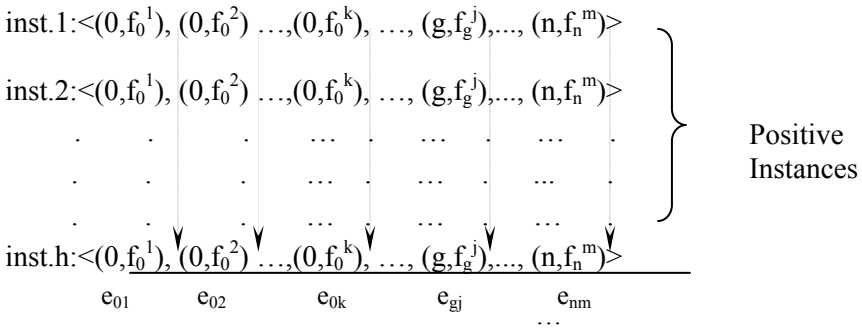


Figure 7.1: Global distribution of instances & representations

To start the rule induction algorithm, GRID selects element feature that has the highest $\beta_{gi} \times e_{gi}$ value in the active positive training set. By adding this element feature f_{gi} into an active feature set $\underline{f}$, a pattern $r_k(f)$ that covers a large number of active training instances which have the most prominent feature f_{gi} can be generated. The quality of $r_k(f)$ is determined by using the *Laplacian* measure. Let n_k denotes the number of both positive and negative examples covered by rule $r_k(f)$, and m_k be the number of negative examples or errors covered by the rule. The *Laplacian* expected error is defined as:

$$Laplacian(r_k(\underline{f})) = \frac{m_k + 1}{n_k + 1} \qquad (7.3)$$

GRID evaluates the *Laplacian* measure for the top w element features with high $\beta_{gi} \times e_{gi}$ values in the active positive training set. The aim is to select the rule that has prominent feature f_{gi} with high $\beta_{gi} \times e_{gi}$ value while its $Laplacian(r_k(f))$ satisfies the pre-defined error tolerance.

7.1.4 An Example of GRID Learning

In this section, we present a simple example taken from [Xiao 2003] to illustrate how GRID learns pattern extraction rules. In this example, only a subset of feature representations and context elements is presented. The example aims to extract the semantic slot <stime>, which indicates the "starting time" of a seminar announcement.

context position	**-2**	**-1**	**0**
instance 1	Time	:	<stime> 3:30 PM </stime>
instance 2	Time	:	<stime> 2 p.m. </stime>
instance 3	Time	:	<stime> 4 p.m </stime>
instance 4	start	at	<stime> 10 am </stime>
instance 5	begin	from	<stime> 11:30 AM </stime>

Table 7.2: An example for extracting slot <stime>

Table 7.2 shows 5 positive instances where the desired slots are tagged. The example uses w=1 (i.e. start with the most frequent feature). By examining the feature frequency for the context elements at every position around the tagged slot for the 5 positive instances in Table 7.2, we can see that the tagged slot "NP_Time" appears most frequently (it occurs 5 times). Thus, it has the highest coverage in the training example pool. This feature is then selected, and the generated rule is:

"NP_Time → NP_Time is starting time"

Next, the rule is examined whether it is satisfied the *Laplacian* measure. This rule will cover all the "negative" instances with the presence of NP_Time but are not

<stime>. Thus the *Laplacian* measure will not be satisfied. In order to improve the quality of the rule, more contextual information must be included to constraint the rule. This is done by examining the context information beside the tagged slot. We can see from the table that the token ":" at the 1st left context position appears 3 times for all positive instances with <stime> in slot 0, it is therefore selected next, and the rule is now constrained as:

": NP_Time → NP_Time is starting time"

For the CMU seminar announcement corpus, this rule is sufficient to satisfy the Laplacian measure, and thus we obtained the 1st rule (see below). Once this rule is obtained, the 3 instances are removed from the positive training example pool. The above process is iterated on the remaining two positive examples and finally obtains another two rules (2nd rule & 3rd rule). Thus, all the generated rules that satisfied the *Laplacian* measure, are:

1st rule	": NP_Time → NP_Time is starting time"
2nd rule	"start at NP_Time → NP_Time is starting time"
3rd rule	"begin from NP_Time → NP_Time is starting time"

During testing, if any of these rules applies, the tags <stime> and </stime> will be inserted beside the NP_Time's boundaries. Further details of GRID algorithm can be found in [Xiao 2003].

7.2 Extension of GRID to News Story Segmentation

As discussed in Chapter 4, we try to define the shot categories that are meaningful and reflect the semantic of the shots. Some specific categories are useful clues to detect the transitions of story. Such categories are *Anchor*, program logos (such as "*LEDS*", "*PLAY*", etc.), Text-scene, etc. From the observations of output from the HMM, we found that there are rules embedded in the patterns. For example, the output when using a complete set of features with state equals to 9, a transition from any states to State 4 (which corresponds to *Anchor* shot with the presence of cue-phrase) indicates a story change. Another example is a transition from State 9 (which corresponds to *Commercial* shot) to State 8 (that corresponds to *LEDS* shot) also indicates a change of story. Given a sequence of shots typed by the category ids and the essential features, it seems reasonable that some form of rules can be extracted to identify the story boundaries. Here, we investigate the use of GRID system to perform rule induction for story segmentation task.

7.2.1 Context Feature Vector

Given the set of training data (the same as that used at the story segmentation level in the HMM approach) with information of story boundaries, we collect positive training instances by extracting *k* shots to the left and *k* shots to the right of each story boundary and use these *2k* surrounding shots to create a context feature vector. The remaining non-boundary shots in the training data are created in the same way and are placed in the negative instance pool. The term tagged slot in GRID system is referred

to as story boundary (BD) in our positive training instances. The context feature vector is similar to that of Eq (7.1), the difference is only in replacing *tagged_slot* with *BD*. Thus, we have

$$<c_{-k}> \ldots <c_{-2}> <c_{-1}> BD <c_{+1}> <c_{+2}> \ldots <c_{+k}> \quad (7.4)$$

where $<c_i>$ is a token that represents shot content features. Here, its content features are shot category, scene change feature and cue-phrase features as used in our HMM framework for TRECVID. A general context feature vector for a single tagged instance is the same as in Eq (7.2).

The overall feature set and its possible values that we used in our experiments are given in Table 7.3 (refer to Chapter 4 on the details of each feature).

Feature	Description	Possible values
f_g^1	Shot category (*SC*)	1, 2, ..,17
f_g^2	Scene change (*SN*)	"c" or "u"
f_g^3	Cue-phrases (*CU)*	"1" or "0"

Table 7.3: Features used in our experiments

7.2.2 An Example of GRID Learning

In this section, we present a simple example to illustrate how GRID learns pattern extraction rules from news video data. Table 7.4 shows 5 positive instances where the

desired slots are tagged. The example instances are selected from our training data. For simplicity, we give an example when using only one feature per context unit. The feature is shot categories. Note that, <BD> is inserted as a dummy tagged slot to indicate story boundary.

Instances	**Context position**				
	***LH*$_{-2}$**	***LH*$_{-1}$**	***C*$_0$**	***RH*$_{+1}$**	*RH*$_{+2}$
1	Speech	LR	**<BD>**	LEDS	LR
2	sport	LR	**<BD>**	LEDS	sport
3	LR	LR	**<BD>**	LEDS	sport
4	sport	sport	**<BD>**	LEDS	LR
5	sport	Text-scene	**<BD>**	LEDS	Speech

Note: LR – Live-reporting category, LEDS – Lead-ins category, LH – left hand context, RH – right hand context, and C_0 --tagged slot (or story boundary);

Table 7.4: An example for extracting slot <BD>

How GRID extracts rules here is similar to the example on text document described earlier in Section 7.1.3. From Table 7.4, with w =1, we can see that at $RH_{+1,}$ "LEDS" appears most frequently and thus has the highest coverage in the training example pool. This feature is then selected, and the generated rule is:

"RH_{+1} = LEDS → C_0 is story boundary"

This rule, however, does not satisfy the *Laplacian* measure as we found that there are many LEDS shots in the corpus that are not the successor of story breaks. For example, there are many continuous LEDS shots that belong to one chunk of miscellaneous story. So the rule has to be further constrained. Next we examine the

contextual information beside the tagged slot. We see that at LH_{-1} position, the token "LR" appears 3 times, and is therefore selected next. The rule is now constrained as:

"LH_{-1}= LR, RH_{+1} = LEDS → C_0 is story boundary"

In our corpus this rule is sufficient to meet our *Laplacian* measure, and thus the first rule found is:

Rule 1: "LH_{-1}= LR, RH_{+1} = LEDS → C_0 is story boundary"

Once we obtain this rule, we remove the 3 instances from the positive training example pool. The algorithm then iterates the above process on the remaining two positive examples and finally obtains another two rules as follows:

Rule 2: "LH_{-1} = sport, RH_{+1} = LEDS → C_0 is story boundary"

Rule 3: "LH_{-2}= sport, LH_{-1} = Text-scene, RH_{+1}= LEDS → C_0 is story boundary"

7.2.3 The Overall Rule Induction Algorithm

General algorithm for GRID is given as follows:

a) Group tagged instances of the same slot type into one cluster.

b) Generate context feature vectors for all positive instances in every cluster. The resulting k^{th} cluster is $\underline{C}_k$, with the positive instance set $\underline{P}_k$ and negative instance set $\underline{N}_k$. Let $\underline{r}_k$ be the set of rules extracted so far to cover $\underline{P}_k$; and set $\underline{r}_k = null$.

c) For every cluster $\underline{C}_k$, perform the followings:

(c_1) Loop-1: // to generate new rules

Let $\underline{f}_c$=*null* be the current feature set; $r_c(\underline{f}_c)$ be the current rule; and

$\underline{P_c}$, $\underline{N_c}$ be the set of instances covered by $r_c(\underline{f_c})$. Initially, set: $\underline{P_c} = \underline{P_k}$, $\underline{N_c} = \underline{N_k}$

RuleAttempt = 0;

(c_2) Loop-2: // to refine current rule $r_c(\underline{f_c})$

- Find top w element features $\{f_g^j\}$ (based on $\beta_{gi} \times e_{gi}$ values) that covers at least one instance in $\underline{P_c}$;
- Select the f_i^j that minimizes the *Laplacian* measure of the current rule $r_c(\underline{f_c} \cup f_i^j)$; Add f_i^j to $\underline{f_c}$, i.e. $\underline{f_c} = \underline{f_c} \cup f_i^j$
- RuleAttempt++;

(c_3) IF *Laplacian*$(r_c(\underline{f_c})) < \sigma$ (error tolerance)

THEN // the quality of resulting rule is good

Add rule r_c to rule set $\underline{r_k}$; or $\underline{r_k} = \underline{r_k} \cup r_c$;

Update $\underline{P_k} = \underline{P_k} - \{$all instances covered by rule $r_c\}$;

Go to Loop-1 to generate another rule.

ELSE // more work is needed to constraint rule r_c

Update $\underline{P_c}$ by removing those instances that are not covered by r_c;

IF RuleAttempt $\geq \lambda$ (max. rule attempt for constraining rules)

THEN // relaxing error tolerance;

Increase σ;

Go to Loop-1 to generate new rule:

(with bigger error tolerance)

ELSE Go to Loop-2 to find new feature f'^j_i to refine rule r_c.

Repeat until $\underline{P_k}$ is empty.

The "RuleAttempt" is related to the length of the generated rule, which the user could pre-specify. For example, if we specify the rule length is "4", then the "RuleAttempt" could be 8. That is to say, we constrain the rule to the maximum size of 4 contextual units (4 for left and right side around the tagged slot respectively). Based on the above algorithm, GRID will generate rules that incorporate the most prominent features. If using a single feature cannot satisfy the error tolerance for quality, then more features will be added to tighten the constraints until the quality of the resulting rule is good enough. GRID is a covering algorithm and each instance in the positive training pool is involved to induce one rule. We can also see that GRID is a local search algorithm. It performs a form of hill climbing and once the rule with current features satisfies the error tolerance it will be output even though adding more features would result in lower *Laplacian* value. In case there are noises in the positive training examples, we can apply some "post-pruning" strategy to control the whole quality of the learned rules. After the entire rule set has been generated, some of the rules may have low coverage on the training set. A post-pruning step that discards all rules with Laplacian expected error greater than a threshold has the effect of removing the least reliable rules.

7.3 Evaluation on the TRECVID 2003 data

This section discusses the tests performed on the TRECVID data and the evaluation results using GRID algorithm. This approach was employed to compared the result to that of the HMM approach. For the rule-based approach, we conduct four experiments using different set of features as follows: (i) shot category (SC); (ii) shot

category and cue-phrases (SC + CU); (iii) shot category and scene change (SC +SN); and (iv) shot category, scene change and cue-phrases (SC+SN+CU). Before we present the results, we will discuss the setting up of the testing instances and the algorithm to detect story boundaries after the qualified rules are generated.

7.3.1 Creating Testing Instances

Testing instances are constructed in a similar way to that of training instances. Consider the case when k = 2, then we need to create 4-shot instances using the window of 4 shots (2 left and 2 right shots) move along the shot sequence. The main difference is that for training data, we construct the instances surrounding story boundaries. But for test data, we construct the instances from all shots. During testing, if any of the generated rules can apply, the tags <BD> will be inserted at the position C_0. Figure 7.2 presents the construction of N instances when size k =2.

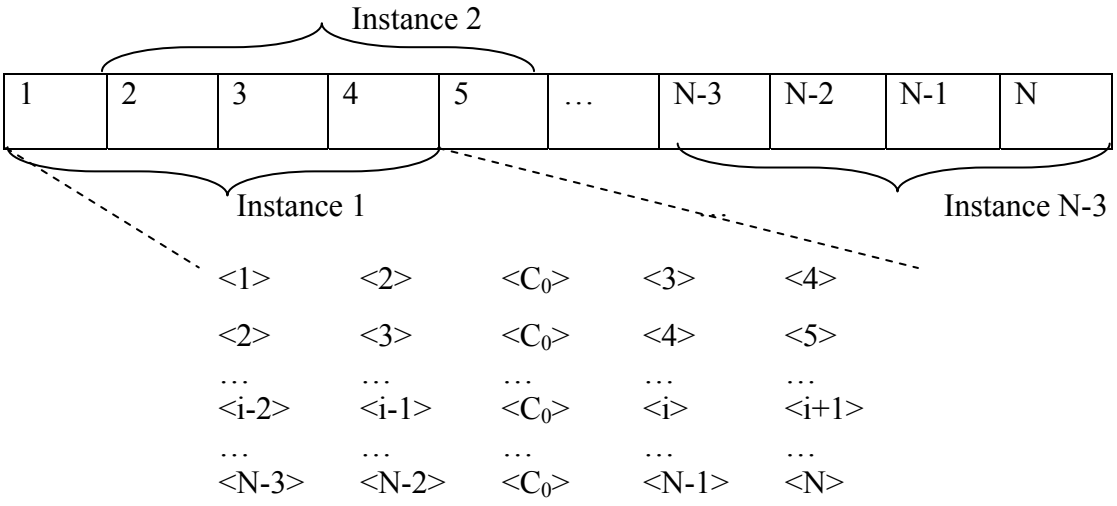


Figure 7.2: Illustrates the construction of the instances when size k =2

In the Figure, C_0 is inserted to facilitate the examination of the current instance as discussed in Section 7.2.2.

After rules are extracted from the training data, we can then apply these rules to the test data. Form our experiments, the number of context units k = 2 gives the best result. Below describes the algorithm to apply rules to the test instances to detect story boundaries.

Algorithm

Given X = { $X_1, X_2, .. = X_M$), M – total number of test instances

1. For *i* = 1,..M
   - 1.1 read an instance X_i from the test instance pool
     - for *j* = 1, .. N (when N = total generated rules)
       1. check whether rule [*j*] is matched with a pattern in the test instance X_i
       2. if yes, instance X_i contain story boundary at position <C_0>
       3. else apply the next rule, rule [*j*+1] until no more rule to apply
   - 1.2 If there are more test instances, then go to step 1.1 until there are no more test instances
2. End of algorithm.

The evaluation results and errors analysis are presented in the following sections.

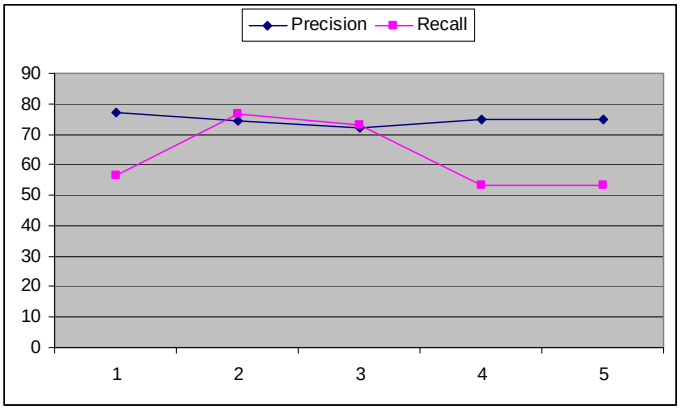
7.3.2 Evaluation Results

This section presents the effect of: (1) different context length; and (2) different set of features. It also gives the best results achieved by this approach and the set of rules generated. Finally, it presents the results as compared to that of the HMM approach and the based-line which based on the anchor shot.

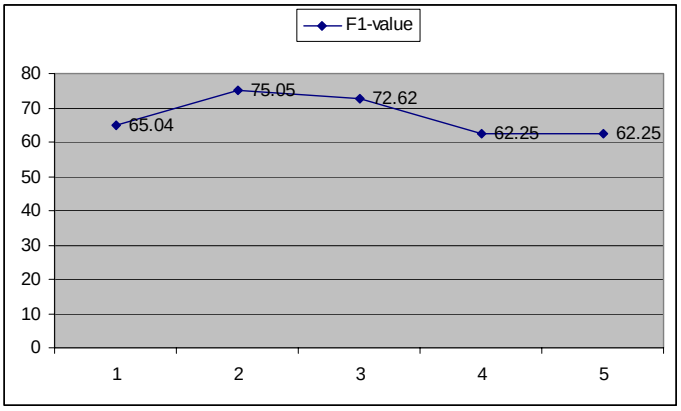
7.3.2.a Effects of different k

In order to investigate the effect of employing different context length on the performance of story segmentation, we conduct a series of experiments by using different context length based on the number of shots. Figure 7.3 (a) presents the performance of GRID in terms of precision and recall and Figure 7.3 (b) shows it corresponding F_1 value.

We can see from the Figures that, the best result is obtained when the contextual length *k is* equal to 2. After k increases to 3 and above, the performance starts to drop. When k equals to 4, the recall drops rapidly. This is because when using more context units, the learnt rules become more specific (or over learnt). As a result, there are fewer patterns in the test data that match the rules, thus resulting in decreasing performance.



(a)



(b)

Figure 7.3: Effect of number of context units (x-axis) on performance of GRID in: (a) precision-recall value; and (b) F_1-value

7.3.2.b Effect of different set of features

Figure 7.4 compares the result from the four experiments. The results are presented using F_1 measurement. From the Figure we can see that, when using only shot category as the feature, GRID performs better than when using more features, which may introduces more errors rather than improving the result. It is demonstrated that, shot category is the prominent feature for story segmentation with our rule induction algorithm. This is also confirmed from the result of the HMM framework that shot categories contribute significantly to the overall accuracy.

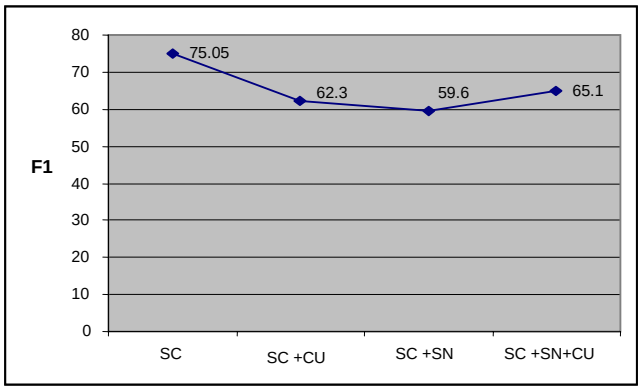


Figure 7.4: A comparison of results when using different features for rules induction

7.3.2.c The best results and the set of rules generated

Table 7.5 presents the best result from the selected features (only shot category) when the number of contextual unit k = 2. We can see from the Table that GRID performs reasonable well on this large corpus and achieves an F_1 value of over 75%. It performs better on ABC corpus rather than CNN corpus. This is because the structure of CNN news is more dynamic and complex than that of ABC, where its structure is much simpler and only varies slightly within the same corpus.

Data	Precision	Recall	F_1
ABC	71.95	84.51	77.72
CNN	76.76	68.47	72.38
average	74.36	76.49	75.05

Table 7.5: Result when using shot category as the feature

Rule1: RH_{+1} = Anchor

Rule 2: LH_{-2} = *sport,* RH_{+1} =*LEDS*

Rule 3: LH_{-1} = *ADV,* LH_2 = *LEDS*, RH_{+1} =*sport*

Rule 4: RH_{+1} = *LEDS,* RH_{+2} = *LR*

Rule 5: RH_{+1} = *Text-scene,* RH_{+2} = *sport*

Rule 6: RH_{+1} = *2Anchor*

Rule 7: LH_1 = *sport,* LH_2 = *Text-scene*

Rule 8: LH_{-2} =*People*, RH_{+1} =*sport*

Rule 9: LH_{-2} =LR, RH_{+1} =*LEDS*

Rule 10: RH_1 =*Text-scene,* RH_{+2} = *Anchor*

Rule 11: LH_{-2} = LR, RH_{+1} = *Weather*

Rule 12: RH_{+1} = *HEALTH*

Rule 13: LH_{-2} = *HEALTH*

Rule 14: LH_{-2} = Anchor, RH_{+1} =*Weather*

Rule 15: LH_{-2} = *LR*, RH_{+1} =*Intro/highlight*

Figure 7.5: Presents the rules extracted from the training set when GRID gives the best result

Figure 7.5 presents the rules extracted from the training set when GRID produces the best result for story segmentation. We can see from the Figure that there are only 15 rules that GRID leant from the training set of this corpus. The most common rule, which is generated even when using different number of k, is Rule 1. The implication of this rule is any transition to *Anchor* shot indicates a story change. This is independent of the number of contextual unit k as the rule satisfies the *Laplacian* measure for all k. Refer to Table 7.4 and Figure 7.2 for the format of the test instances and the interpretation of the rules. During testing, for each instance, if one of these rules can be applied, then story boundary is declared at the position C_0.

7.3.2.d Comparison of Performances between GRID and HMM approaches

In order to compare the results, we also peform the experiments that use the detection of *Anchor* shots as a base line. The method of detecting story boundaries by detecting just anchor shots is used in various works as reported in the proceedings of TRECVID 2003. Table 7.6 presents the best results of the three methods.

Methods	Precision	Recall	F_1
HMM	80.2	74.9	**77.50**
Rules	74.36	76.49	75.05
Based-line (Anchor)	77.02	52.4	62.37

Table 7.6: Comparing the results of the two approaches and the based-line

We can see from the Table that, by detecting only anchor person shots, we could achieve F1-value of about 62% which is considered high. Among the three methods, the HMM framework is the best for story segmentation. However, rule induction approach also gives a promising result closely to that of HMM. Thus, it may be an alternative method that can still be further improved to achieve better performance.

Next Chapter presents the conclusion of our 2-level framework for story segmentation of the two approaches.

CHAPTER 8

CONCLUSION AND FUTURE WORK

This Chapter concludes our system framework. It first summarizes the framework, the results and the errors incurred during story segmentation. It then provides possible solutions to tackle the major problem when adopting HMM approach. Finally it discusses trends of future work.

8.1 Conclusion

We have discussed two approaches to story segmentation in news video: Hidden Markov Models (HMM) and rule induction based approaches. In both approaches, we divided our framework into 2 levels, shot and story levels. In the shot level, we defined three clusters of total 17 shot categories. The clusters are *heuristic-based* (contains commercial shots); *visual-based* (consists of *Weather* and *Finance* shots, *Anchor* shots, program logo shots etc.) and *rule-based* clusters (contains *live-reporting* shots, *People* shots, *sport* shots, etc.). For each shot, we used low level feature (176-Luv colour histogram), temporal features (audio class, shot duration, and motion activity), and high level features (face, shot type, videotexts) and employed a combination of heuristics, specific detectors and the decision trees to classify the shots into the respective categories. At the story level, we used the shot category

information, scene/location change and cue-phrases as the features, and employed either HMM or rule induction techniques to perform story segmentation. Besides the errors that occur during the experiments, we found that HMM is not easy to be enhanced to cope with new corpuses or incorporate any known heuristic patterns. We need to re-train the system which involves parameter estimations, re-estimation, finding the number of hidden states that give the best results, etc. In comparison, with the rule induction approach, it is easy to incorporate new rules and adapt to new corpuses.

The results and errors from each approach can be summarized as follows:

8.1.1 HMM Approach

The tests on 120-hour of news video from TRECVID showed that our approach is effective, and we could achieve the F1 measure of over 77% for story segmentation. From the result, we applied simple rules to classify the detected stories into the classes of "misc" and "news" and we could achieve over 94% for news classification. Our system achieved the best performance during TRECVID 2003 evaluations [TRECVID 2003]. We found that there are about 72% of the stories found begin with Anchor shot, about 16% with shot of categories of ML-based cluster, and about 11 % with shot of categories of Visual-based cluster. From the error analysis (see Section 6.3.2 b), we found that they are four sources of errors when using this approach. The errors are: (1) detection of individual sport news reports which accounted for more than 53% of total errors; (2) detection of multiple live-reporting stories accounted for

about 22% of total errors; (3) detection of multiple stories in *Anchor* shot accounted for about 11% of total errors; and (4) detection of unexpected patterns accounted for about 13% of total errors.

As discussed above, the main source of error resulted from the detection of individual sport stories. In order to improve our system performance, additional technique is required to resolve this problem. One possible technique is by adopting multi-scale image similarity (MSIS) method. We did a test on a small set of data from the training set and found that it helped to improve story segmentation result by about 3% compared to when it was not used. The algorithm is based on the observation that same sport news has similar colors (for example, soccer news video contains mostly green color). Thus, once the background color changes, it is likely that it indicates the change to the next sport news. Two scenarios of sport news detection using the multi-scale image similarity algorithm are presented in Figure 8.1.

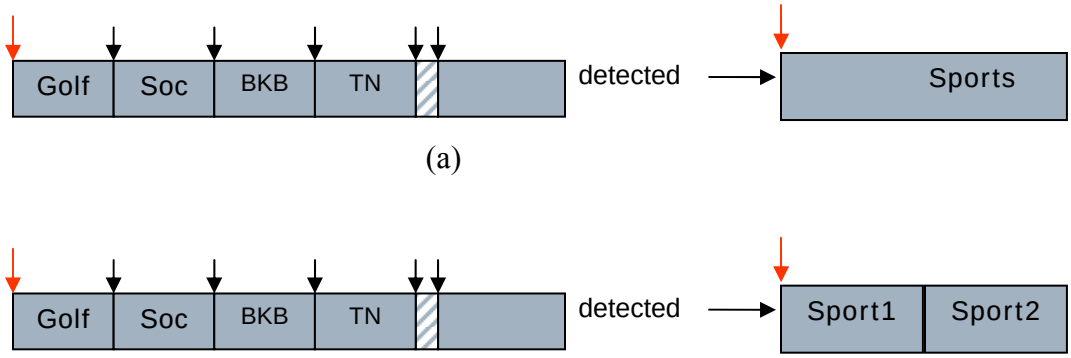


Figure 8.1: Two scenarios for sport news detection in our work, (a) without MSIS, and (b) with MSIS. (Note: Soc –soccer; BKB – Basketball; and TN – Tennis)

Figure 8.1 (a) presents the result of the detection by the HMM method; and (b) presents the result after applying the multi-scale image similarity method. We can see that at least one more boundary was detected.

Another probable solution is by analyzing the associated transcript to spot the specific words that are likely to indicate the type of sport. Examples of such words are "NBA", "basketball", "goal", and etc. However, by spotting the words, there is no guarantee that we can obtain the exact story boundaries as the words may appear in the middle or close to the end of the shot or even a few shots after the boundary of a new sports story. Thus, if using this algorithm, the boundaries found would not be of sufficient accuracy, i.e. it will tend to exceed the 5-second allowance [TRECVID 2003].

Beside the above problem, we are also concerned about the cost when dealing with large corpora. From the analysis of the effectiveness of the features extracted from Singapore news as presented in Figure 5.7, we found that face and audio are two of the most important features. The use of only these two features contributes about 78% to shot classification accuracy. By including the rest of the features to the system, we could then achieve a shot classification accuracy of about 95% (~ 90% for TRECVID 2003 data set) and about 90% for story segmentation (over 77% for TRECVID 2003 data set). This demonstrated that the selected features are necessary to achieve such performance.

As for story segmentation using rule induction method, we found that shot categories such as *Live-reporting* and *People* are less important, and can be dropped from the analysis.

8.1.2 Rule-induction Approach

In this approach, we used the same set of features as used in the HMM framework and employed rule induction system called GRID to extract rules from the training data. The learnt rules were then applied to detect story boundaries in the test data. We conducted several experiments based on the same training and test data as used in the HMM approach. The results showed that we could achieve the accuracy of about 75%. Although the accuracy of rule induction approach is slightly lower than that of HMM, it has the advantage that it is less complex and easier to adapt to new corpora.

The errors that incurred during story segmentation using this approach are similar to that of the HMM except in sport reports, the rule induction algorithm performs slightly better. This is because it could capture the rules like Rules 5, 7 and 8 (see Section 7.3.2 c) which indicate transitions of story within sport report. HMM, on the other hand, would output the whole chunk of sport report as the same state, resulting in only one story.

8.2 Trends and Future Work

First we can see from the results of TRECVID evaluations that techniques based mainly on statistical machine learning methods work well for story segmentation problem. However, from the discussion of problems that cause high errors in the segmentation, it suggests that the news segmentation is a hierarchical classification problem. Although it can be tackled using heuristic approaches as discussed earlier

for sports, a principled approach to tackle this problem in a unified framework is to employ higher order statistical methods such as the graphical model [Jordan 1998].

Second, one example of the applications of news stories is in news retrieval and summarization. Given a query, the system should be able to retrieve the appropriate news stories (news video and the associated text stories) according to the user's specified time constraint. To support this task, adaptive news summarization is required. A technique to support question answering on news video was proposed in [Yang 2003]. Given a query, the algorithm first locates the relevant news stories, summarizes it and then packed the news video into the story within the desired time constraint which can be, for example 30 second, to 1 minute etc. Finally, the system presents this video summary to the user. Figure 8.2 illustrates a view of news summary proposed in this work.

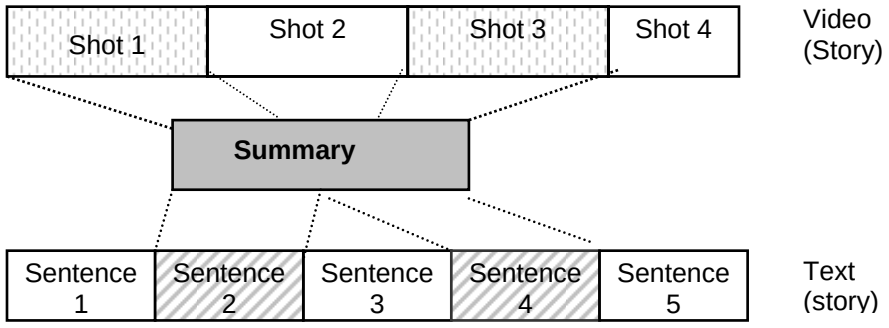
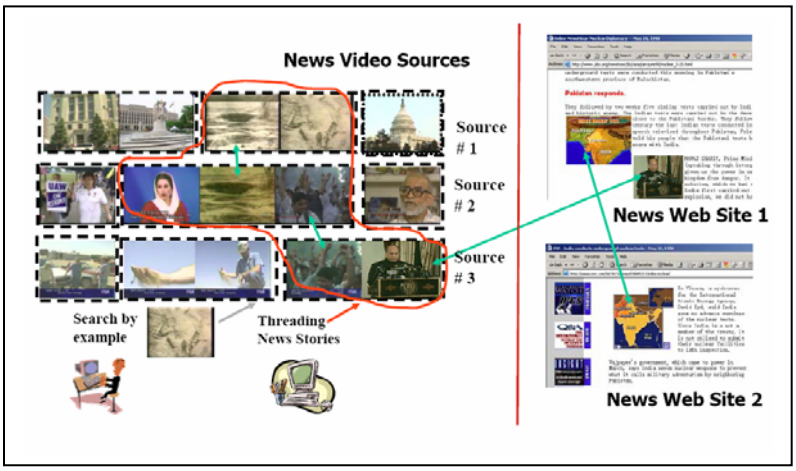


Figure 8.2: A view of a summary of news story

Third, beyond news segmentation, there is a need to create threading of same news stories over a period of time as the work proposed in [Chang 2004]. One variation of this work is to detect news events of incoming news streams, similar to the work done in TDT (Topic detection and tracking) task in text [TDT 1997]. There is a need to

extract semantic threads through multi-level video content analysis, to establish automatic concept detection for video semantics understanding and user interactions [Chang 2004]. The reconstruction of semantic threads is to be done across multiple video broadcast news sources using multi-level concept modeling. Given a query such as "*give me all video news related to suicide bomb in Indonesia*", this work is able to retrieve news of the same topic from multiple news sources through the pre-linking method. Figure 8.3 shows a scenario of news linking from multiple sources of video news broadcast.



(source: S.F. Chang 's talk, July 5 2004, Inforcomm research, Singapore)

Figure 8.3: A scenario of news linking from multiple sources of video news broadcast

Last but not least, the framework of HMM and Rule induction can be applied to segmenting stories (episode) in movie and other genre of video that is composed of different coherent units appearing along the time line. Shot categories can be enhanced (or dropped) to cope with more (or less) categories in the new data set. However, techniques used in our 2-level framework should be applicable. In addition, the defined shot categories are useful for video retrieval which is one of the tasks in

TRECVID evaluations. Given a query in text and/or video clips, the system must search and return the relevant shots. Having well defined shot categories, it helps the system to search more efficiently. For example, given a query about weather, the system can directly access those shots that are classified into the category of weather and perform similarity matching to return the relevant shots. Another example is finding "Bill Clinton" shots from the videos in the database. This is a hard problem. It involves: searching for shots with detected face(s); and recognizing "Bill Clinton" face in the candidate shots. However, by defining shot categories *"Speech/Interview*" and "*People*" shot, it helps the system to reduce searching time and false alarm rate as the system can directly retrieve shots in these two categories and further perform face recognition for "Bill Clinton".

To conclude, our 2-level multi-modal approach is successful for story segmentation in news video and it can be applied to segmenting coherent units in other videos such as movies and documentaries. In addition it can also be applied to multimedia content analysis in general videos. The analysis is necessary for further video browsing and retrieval.

BIBLIOGRAPHY

[Allan 1998] J. Allan, J. Carbonell, G. Doddington, J. Yamron & Y. Yang (1998). *Topic detection and tracking pilot study final report*. Proceedings of DARPA Broadcast News Transcription and Understanding Workshop, pp. 194-218.

[Alatan 2001] A. A. Alatan, A. N. Akansu, and W. Wolf. *Multi-modal dialog Scene Detection using Hidden Markov Models for Content-based Multi-media Indexing. Multimedia Tools and applications*. Vol.14, pp. 137-151, 2001.

[Anantharamu 2002] C. Anantharamu, H. Feng, T.-S. Chua. *Temporal multi-resolution Framework for Shot Boundary Detection and Key Frame Extraction*. Proc. of Int'l. Text Retrieval Conference (TREC'02). NIST, Gaithersburg USA, Nov 2002, pp. 500-504.

[Breiman 1993] L. Breiman, J. H. Friedman, R. Olshen, and C. Stone (1993). *Classification and Regression Trees*. Chapman & Hall, New York, 1993.

[Chaisorn 2002] Lekha Chaisorn, Tat-Seng Chua, and Chin-Hui Lee (2002). *The Segmentation of News Video into Story Units*. Proc. of IEEE Int'l Conference on Multimedia and Expo 2002, Lausanne, Switzerland, Aug 26-29, 2002.

[Chang 2004] Shih-Fu Chang. *Threading and Linking News Video from Multiple Sources using Multi-modal cues.* A Talk at Institute for Infocomm Research, Singapore, July 5, 2004.

[Chua and Chu 1998] T.-S. Chua and C. Chu. *Color-based Pseudo-object for image retrieval with relevance feedback*. Proc. of Int'l. Conference Advanced Multimedia Content Processing '98. Osaka, Japan, Nov 1998, pp. 148-162.

[Chen and Chua 2001] L. Chen and T.-S. Chua. *A Match and Tiling Approach to Content-based Image Retrieval*. Proc. of IEEE Int'l Conference on Multimedia and Expo (ICME), Tokyo Japan, Aug 2001, pp. 417-420.

[Chen and Wong 2001] Y. Chen and E. K. Wong. *A knowledge-based Approach to Video Content Classification.* Proc. of Int'l. Conference SPIE , 2001, pp.292-300.

[Chua 2000] T.-S. Chua, Y. Zhao and M.S. Kankanhalli. *An Automated Compressed-Domain Face Detection Method for Video Stratification.* Proc. of Int'l. Conference on Multimedia Modeling (MMM'2000), Nagoya, Japan, Nov 2000, pp. 333-348.

[Dale 2000] R. Dale, H. Moisl, and H. Somers. *Handbook of Natural Language Processing.* Marcel Dekker, New York USA, 2000.

[Das and Liou 1998] Madirakshi Das and Shih-Ping Liou. *A New Hybrid Approach to Video Organization for Content-Based Indexing*. Porc. of IEEE Int'l Conference on Multimedia Computing and Systems, June 28 - July 01, 1998, Austin, Texas, pp. 372

[Dietterich and Bakiri 1995] T. G. Dietterich, and G. Bakiri. *Solving Multi-class Learning Problems via Error-Correcting Output Codes***.** Journal of Artificial Intelligence Research, 1995, pp. 263-286.

[Eickeler 1997] Eickeler., A. Kosmala, G. Rigoll. *A New Approach To Content-based Video Indexing Using Hidden Markov Models*. IEEE workshop on Image Analysis for Multimedia Interactive Service (WIAMIS), Louvain la Neuve Belgium, June 1997, pp. 149-154.

[Greiff 2001] Warren Greiff, Alex Morgan, Randall Fish, Marc Richards and Amlan Kundu. *Fine-Grained Hidden Markov Modeling for Broadcast-News Story Segmentation. Proc. of Int'l. Conference on Human Language Technology (HLT), California, USA. March 2001*

[Gunsel 1996] Bilge Günsel, A. Müfit Ferman, A. Murat Tekalp. *Video Indexing Through Integration of Syntactic and Semantic Features***.** *Proc. of the 3rd IEEE Workshop on Applications of Computer Vision (WACV '96), 1996, pp 90*

[Hauptmann and Witbrock, 1998] A.G. Hauptmann and Michael J. Witbrock. *Story Segmentation and Detection of Commercials in Broadcast News Video.* Proc. of Int'l Conference on Advances in Digital Libraries (ADL), California, USA, 1998. pp.168-179.

[Hearst 1994] M.A. Hearst (1994). *Multi-paragraph segmentation of expository text*. Proc. of the 32nd Annual Meeting of the Association for Computational Linguistics, Las Cruces, New Mexico, June, 9-16.

[Hsu and Chang 2003] W. H.-M. Hsu and S.-F. Chang (2003). *A Statistical Framework for Fusing Mid-level Perceptual Features in News Video.* Proc. of IEEE Int'l Conference on Multimedia and Expo (ICME), Baltimore, USA, July 6-9, 2003.

[Ide 1998] I. Ide, K. Yamamoto, and H. Tanaka. *Automatic Video Indexing Based on Shot Classification.* Proc. of Conference on Advanced Multimedia Content Processing (AMCP), Osaka, Japan, 1998.

[Jordan 1998] M-I. Jordan. *Learning in Graphical Models*. Cambridge MA, MIT press, 1998.

[Koh and Chua 2000] C.-K. Koh and T.-S. Chua. *Detection and Segmentation of Commercials in News Video.* Technical report, The School of computing, National University of Singapore, 2000.

[Krishnaiah and Kanal 1982] P.R. Krishnaiah and L.N. Kanal. *Classification Pattern Recognition and Reduction of Dimensionality*. North-Holland Publishing Company, 1982. Chapter 12, pp. 267 – 284

[Kohavi and Provost 1998] R. Kohavi and F. Provost. Glossary of Terms, Editorial for the Special Issue on *Applications of Machine Learning and the Knowledge Discovery Process*, Vol 30, No 2/3, Feb/March 1998.

[LDC 1992] LDC web site, http://www.ldc.upenn.edu/

[Li 2001] Yang Li. *Multi-Resolution Analysis on Text Segmentation*. Master degree thesis, School of Computing, National University of Singapore.

[Lin 2000] Y. Lin, M. S. Kankanhalli, and T.-S. Chua. *Temporal Multi-resolution Analysis for Video Segmentation*. Proc. of Int'l Conference on Storage and Retrieval for Media Databases (SPIE), San Jose, USA, Vol. 3972, Jan 2000, pp. 494-505.

[Liu 1998] Z. Liu , J. Huang, and Y. Wang. *Classification of TV Programs Based on Audio Information using Hidden Markov Models.* IEEE Signal Processing Society, Workshop on Multimedia Signal Processing, Los Angeles, California USA, 1998, pp. 27-31

[Lu 2001] L. Lu, S. Z. Li and H.-J. Zhang (2001). *Content-based Audio Segmentation using Support Vector Machine*. Proc. of IEEE Int'l Conference on Multimedia and Expo (ICME), Japan, 2001, pp. 956-959.

[Merlino 1997] Andrew Merlino, Daryl Morey and Mark Maybury. *Broadcast news navigation using story segmentation*. Proc. of the fifth ACM international conference on Multimedia**,** Seattle, Washington, United States, pp.381 - 391 .

[MPEG 1993] Implementation of ISO/IEC 11172-2:1993. *Information Technology – Coding of Moving Pictures and associated audio for digital storage media at up to about 1.5 Mbits.*

[MPEG-7 2000] *http://xml.coverpages.org/mpeg7.html*

[O'Connor 2001] O'Connor N, Czirjek C, Deasy S, Marlow S, Murphy N and Alan Smeaton. *News Story Segmentation in the Fischlar Video Indexing System.* Proc. Of Int'l Image Processing (ICIP) 2001 - International Conference on Image Processing. Thessaloniki, Greece, 7-10 October 2001.

Quinlan 1986] J. R. *Quinlan. Induction of Decision Trees:* Machine Learning. Vol. 1, 1986, pp. 81-106.

[Quinlan 1997] J. R. *Quinlan*. C 5.0: Programs for Machine Learning. Morgan Kaufmann Publisher, San Mateo, California, 1997. See also http://www.rulequest.com/

[Rabiner and Juang 1993] L. Rabiner and B. Juang. *Fundamentals of Speech Recognition.* Prentice-Hall, Englewood Cliffs New Jersey, 1993.

[Slaney 2001] Malcolm Slaney, Dulce Ponceleon and James Kaufman. *Multimedia edges: finding hierarchy in all dimensions*. Proc. of the ninth ACM international conference on Multimedia**,** Ottawa, Canada, Sep 30 –Oct 5, 2001. pp. 29 - 40

[TDT 1997] *Topic Detection and Tracking (TDT) Pilot Study Evaluation Plan.* version 2.8, 22 Oct 1997, from Linguistic Data Consortium.

[TRECVID 2003] *TREC video retrieval evaluation home page*. http://www-nlpir.nist.gov/projects/trecvid/.

[Wactlar 1996] Wactlar, H.D., Kanade, T., Smith, M.A. and Stenven, S.M. *Intelligent Access to Digital Video: Informedia Project*. IEEE Computer. May 1996, 29 (5) pp. 46-52. See also http: www.informedia.cs.cmu.edu/.

[Wang 1997] Yao Wang, Jincheng Huang, Zhu Liu and Tsuhan Chen. *Multimedia Content Classification using Motion and Audio Information.* Proc. of Int'l. Symposium on Circuits and Systems (ISCAS), Hongkong, 1997. Vol. 2, p 1488 – 1491.

[Wayne 2000] Charles L. Wayne. *Multilingual Topic Detection and Tracking: Successful Research Enabled by Corpora and Evaluation.* In Proceedings of Second Internation Conference on Language Resources and Evaluation. LREC-2000 Athens, Greece, 31 May - 2 June 2000.

[Wei 2000] G. Wei, L. Agnihotri, N. Dimitrova. *TV Program Classification Based on Face and Text Processing*, Proc. of IEEE Int'l Conference on Multimedia and Expo (ICME), New York, USA, July 2000.

[Wu 2003] L. Wu, Y. Guo, X. Qiu, Z. Feng, J. Rong, W. Jin, D. Zhou, R. Wang & M. Jing (2003). *Fudan University at TRECVID 2003*. Proc. Of TRECVID 2003 workshops.

[WWW2] http://www2.cs.uregina.ca/~hamilton/courses/831/notes/ml/dtrees/4_dtrees1.html

[Xiao 2003] J. Xiao, T. -S. Chua and J. Liu. *A Global Rule Induction Approach to Information Extraction.* Proc. of the 15th IEEE International Conference on Tools with Artificial Intelligence (ICTAI-03), 2003, pp. 530-536.

[Yang 2003] Hui Yang, Lekha Chaisorn, Yunlong Zhao, Shi-Yong Neo, Tat-Seng Chua. *VideoQA: Question Answering on News Video*. Proc. of Int'l ACM Multimedia Conference, California, USA, Nov. 2-9, 2003.

[Yeung 1996] Minera M. Yeung, Boon-Lock Yeo and Bede Liu. *Extracting Story Units from Long Programs for Video Browsing and Navigation.* Int'l. Conference on Multimedia Computing and Systems (ICMCS), Hiroshima, Japan. June 17 – 23, 1996, pp. 296-305.

[Zhang 1993] H.-J. Zhang, A. Kankanhalli and S.W. Smoliar. *Automatic Partitioning of Full-motion Video*. Multimedia Systems, Vol.1(1), 1993, pp. 10-28.

[Zhang and Chua, 2000] Y. Zhang and T.-S. Chua. *Detection of Text Captions in Compressed domain Video.* Proc. of ACM Multimedia'2000 Workshops (Multimedia Information Retrieval), California, USA, Nov 2000, pp. 201-204.

[Zhou 2000] W. Zhou, A. Vellaikal, and C–C Jay Kuo. *Rule-based Classification System for basketball video indexing*. Proc. of ACM Multimedia'2000 Workshops (Multimedia Information Retrieval), California, USA, Nov 2000, pp. 213-216.

List of Publications

1. Chua Tat-Seng, Shih-Fu Chang, Lekha Chaisorn and Winston Hsu. *Story Boundary Detection in Large Broadcast News Video Archives – Techniques, Experience and Trends*. ACM Multimedia Conference, 10-16 October 2004, New York, USA.

2. Lekha Chaisorn, Chua Tat-Seng, Chin-Hui Lee and Qi Tian. *A Hierarchical Approach to Story Segmentation of Large Broadcast News Video Corpus*. Proceedings of IEEE Itnl Conf. on Multimedia and Expo (ICME), 26-30 June 2004, Taiwan.

3. Lekha Chaisorn, Tat-Seng Chua, and Chunkeat Koh. *Experience in News Story Segmentation of Large Video Corpus*. Proceedings of IWAIT 2004, 12-13 January 2004, Singapore.

4. Hui Yang, Lekha Chaisorn, Yunlong Zhao, Shi-Yong Neo, Tat-Seng Chua. *VideoQA: Question Answering on News Video*. Proceedings of ACM Multimedia conference, 2-7 November 2003, CA, USA.

5. Lekha Chaisorn, Tat-Seng Chua, Chun-Keat Koh, Yunlong Zhao, Huaxin Xu, Huamin Feng. *Story Segmentation and Classification for News Video*. Proceedings of TRECVID 2003, 17-18 November, Washington D.C., USA.

6. Lekha Chaisorn, Tat-Seng Chua, and Chin-Hui Lee. News *Video Segmentation*. Handbook of Video Database: Design and Applications, 2003. Chapter 47, CRC Publishers.

7. Lekha Chaisorn, Tat-Seng Chua, and Chin-Hui Lee. *A Multimodal Framework to Story Segmentation for News Video*. Journal of World Wide Web (JWWW) 2003, Kluwer Academic Publishers.

8. Lekha Chaisorn, Tat-Seng Chua, and Chin-Hui Lee. *Extracting Story Units in News Video*. Proceedings of IWAIT, 21-22 January 2003, Nagasaki, Japan.

9. Lekha Chaisorn, Tat-Seng Chua, and Chin-Hui Lee. *The Segmentation of News Video into Story Units*. Proceedings of IEEE Itnl Conf. on Multimedia and Expo (ICME), 26-29 August 2002, Lausanne, Switzerland

10. Lekha Chaisorn and Tat-Seng Chua. *The Segmentation and Classification of Story Boundaries in News Video*. Proceeding of 6th IFIP working conference on Visual Database Systems- VDB6 2002, Australia .

News Broadcaster Web sites

Websites of the broadcasters of the news video under study and in related work, together with the researchers who have worked with these broadcasters, are given in the table below.

Broadcasters	**Website**	**Researcher(s)**
CNN	http://www.cnn.com/	TRECVID2003 & 2004
ABC	http://abcnews.go.com/	TRECVID2003 & 2004
Mandarin Taiwan	http://www.tvbs.com.tw/ http://www.ettoday.com http://www.ttv.com.tw http://www.ftv.com.tw/	Hsu & Chang 2003
Channel 5, MediaCorp Singapore	http://ch5.mediacorptv.com/	Chaisorn et al 2002

An Overview of TRECVID

The details of TRECVID are given below. The information is taken mainly from TRECVID 2003 web site (http://www-nlpir.nist.gov/projects/tv2003/tv2003.html).

TREC Video Retrieval Evaluation (TRECVID)

The TREC conference series is sponsored by the National Institute of Standards and Technology (NIST) with additional support from other U.S. government agencies. The goal of the conference series is to encourage research in information retrieval by providing a large test collection, uniform scoring procedures, and a forum for organizations interested in comparing their results. In 2001 and 2002 the TREC series sponsored a video "track" devoted to research in automatic segmentation, indexing, and content-based retrieval of digital video. Beginning in 2003, this track became an independent evaluation (**TRECVID**) with a 2-day workshop taking place just before TREC.

TRECVID is coordinated by Alan Smeaton (Dublin City University) and Wessel Kraaij (TNO-TPD). Paul Over and Joaquim Arlandis provide support at NIST.

The following experts serve as an advisory committee: John Eakins (University of Northumbria at Newcastle), Peter Enser University of Brighton), Alex Hauptmann (CMU), Annemieke de Jong (Netherlands Institute for Sound and Vision), Michael Lew (Leiden Insitute of Advanced Computer Science), Georges Quenot (CLIPS-IMAG Laboratory), John Smith (IBM), and Richard Wright (BBC).

The followings are the guidelines of TREVID 2003 workshop

1. Goal:

The main goal of the TREC Video Retrieval Evaluation (TRECVID) is to promote progress in content-based retrieval from digital video via open, metrics-based evaluation.

2. Tasks:

TRECVID is a laboratory-style evaluation that attempts to model real world situations or significant component tasks involved in such situations.

There are four main tasks with tests associated and participants must complete at least one of these in order to attend the workshop.

- shot boundary determination
- story segmentation
- high-level feature extraction
- search

Details of each of the above tasks can be found on TRECVID 2003 web site at: http://www-nlpir.nist.gov/projects/tv2003/tv2003.html. Here we present the details of story segmentation task.

Story segmentation:

The task is as follows: given the story boundary test collection, identify the story boundaries with their location (time) and type (miscellaneous or news) in the given video clip(s). This is a new task for 2003.

A story can be composed of multiple shots, e.g. an anchorperson introduces a reporter and the story is finished back in the studio-setting. On the other hand, a single shot can contain story boundaries, e.g. an anchorperson switching to the next news topic.

The task is based on manual story boundary annotations made by LDC for the TDT-2 project. Therefore, LDC's definition of a story will be used in the task: A **news** story is defined as a segment of a news broadcast with a coherent news focus which contains at least two independent, declarative clauses. Other coherent segments are labeled as **miscellaneous**. These non-news stories cover a mixture of footage: commercials, lead-ins and reporter chit-chat. Guidelines that were used for annotating the TDT-2 dataset are available at http://www.ldc.upenn.edu/Projects/TDT2/Guide/manual.front.html. Other useful documents are the guidelines document for the annotation of the TDT4 corpus and a similar document on TDT3, which discuss the annotation guidelines for the different corpora. Section 2 in the TDT4 document is of particular interest for the story segmentation task.
Note: adjacent non-news stories are merged together and annotated as one single story classified as "miscellaneous".

Differences with the TDT-2 story segmentation task:

1. TRECVID 2003 uses a subset of TDT2 dataset: only video sources.
2. Video stream is available to enhance story segmentation.
3. The task is modeled as a retrospective action, so it is allowed to use global data.
4. TRECVID 2003 has a story classification task (which is optional).

There are several required and recommended runs:

1. Required: Video + Audio (no ASR/CC)
2. Required: Video + Audio + ASR
3. Required: ASR (no Video + Audio)
4. The ASR in the required and recommended runs is the ASR provided by LIMSI. **We have dropped the use of the CC data on the hard drive and adopted use the LIMSI ASR rather than that provided on the hard drive because the LIMSI ASR is based on the MPEG-1 version of the video and requires no alignment.** Additional runs can use other ASR systems.
5. It is recommended that story segmentation runs are complemented with story classification.

With TRECVID 2003's story segmentation task, we hope to show how video information can enhance story segmentation algorithms.

3. Video data:

Unless indicated, the 2003 test and development data is fully available only to TRECVID participants. This includes the basic MPEG-1 files, and derived files such as ASR, story segmentation, and transcript files. LDC may make some of the data generally available.

Sources

The total identified collection comprises

- ~120 hours (241 30-minute programs) of ABC World News Tonight and CNN Headline News recorded by the Linguistic Data Consortium from late January through June 1998
- ~13 hours of C-SPAN programming (~ 30 mostly 10- or 20-minute programs) about two thirds 2001, others from 1999, one or two from 1998 and 2000. The C-SPAN programming includes various government committee meetings, discussions of public affairs, some lectures, news conferences, forums of various sorts, public hearings, etc.

Additional ASR output from LIMSI-CNRS:

Jean-Luc Gauvain of the Spoken Language Processing Group at LIMSI has graciously donated ASR output for the entire collection Be sure to credit them for this contribution by a non-participant.

```
J.L. Gauvain, L. Lamel, and G. Adda.
The LIMSI Broadcast News Transcription System.
Speech Communication, 37(1-2):89-108, 2002.
ftp://tlp.limsi.fr/public/spcH4_limsi.ps.Z
```

Development versus test data

About 6 hours of data were selected from the total collection to be used solely as the shot boundary test collection.

The remainder was sorted more or less chronologically (C-SPAN covers a slightly different period than the ABC/CNN data). The first half was designated the feature / search / story segmentation development collection. The second is the feature / search / story segmentation test collection. Note that the story segmentation task will not use the C-SPAN files for development or test.

All of the development and test data with the exception of the shot boundary test data will be shipped by the Linguistic Data Consortium (LDC) on an IDE hard disk to each participating site at no cost to the participants. Each such site will need to offload the data onto local storage and pay to return the disk to LDC. The size of data on the hardrive will be a little over 100 gigabytes. The shot boundary test data (~ 5 gigabytes) will be shipped by NIST to participants on DVDs (DVD+R).

Restrictions on use of development and test data

Each participating group is responsible for adhering to the letter and spirit of these rules, the intent of which is to make the TRECVID evaluation realistic, fair and maximally informative about system effectiveness as opposed to other confounding effects on performance. Submissions, which in the judgment of the coordinators and NIST do not comply, will not be accepted.

Test data

The test data shipped by LDC cannot be used for system development and system developers should have no knowledge of it until after they have submitted their results for evaluation to NIST. Depending on the size of the team and tasks undertaken, this may mean isolating certain team members from certain information or operations, freezing system development early, etc.

Participants may use donated feature extraction output from the test collection but incorporation of such features should be automatic so that system development is not affected by knowledge of the extracted features. Anyone doing searches must be isolated from knowledge of that output.

Participants cannot use the knowledge that the test collection comes from news video recorded during the first half of 1998 in the development of their systems. This would be unrealistic.

Development data

The development data shipped by LDC is intended for the participants' use in developing their systems. It is up to the participants how the development data is used, e.g., divided into training and validation data, etc.

Other data sets created by LDC for earlier evaluations and derived from the same original videos as the test data cannot be used in developing systems for TRECVID 2003.

If participants use the output of an ASR system, they must submit at least one run using that provided on the loaner drive from LDC. They are free to use the output of other ASR systems in additional runs.

If participants use a closed-captions-based transcript, they must use only that provided on the loaner drive from LDC.

Participants may use other development resources not excluded in these guidelines. Such resources should be reported at the workshop. Note that use of other resources will change the submission's status with respect to system development type, which is described next.

There is a group of participants creating and sharing annotation of the development data. See the Video Collaborative Annotation Forum webpage for details. Here is the set of collaborative annotations created for TRECVID 2003.

In order to help isolate system development as a factor in system performance each feature extraction task submission, search task submission, or donation of extracted features must declare its type:

- A - system trained only on common development collection and the common annotation of it
- B - system trained only on common development collection but not on (just) common annotation of it
- C - system is not of type A or B

3.1 Common shot boundary reference and key frames:

A common shot boundary reference has again kindly been provided by Georges Quenot at CLIPS-IMAG. Key frames have also been selected for use in the search and feature extraction tasks. NIST can provide the key frames on DVD+R with some delay to participating groups unable to extract the key frames themselves.

4. Submissions and Evaluation:

Here, we present only the submission of story segmentation. The evaluations of other tasks can be found on the TRECVID web site.

The results of the evaluation will be made available to attendees at the TRECVID 2003 workshop and will be published in the final proceedings and/or on the TRECVID website within six months after the workshop. All submissions will likewise be available to interested researchers via the TRECVID website within six months of the workshop.

Story segmentation

Submissions

- Participating groups may submit up to 10 runs. All runs will be evaluated.
- The task is defined on the search dataset, which is partitioned in a development and test collection.
- The reference data is defined such that there are no gaps between stories and stories do not overlap.

- The evaluation of the story segmentation task will be defined on the video segment defined by its clipping points (the overlap between the mpeg file and the ground truth data). A table of clipping points is available.
    - For the segmentation task, a boundary <= the first clipping point will be ignored (truth and submission); a boundary >= the last clipping point will be ignored (truth and submission).
    - For the classification task - only and ALL of the time interval between the two clipping points for a file will be considered in scoring even parts of stories split by a clipping point.
- Each group is allowed to submit up to 10 runs by sending the submission in an email to Cedric.Coulon@nist.gov.

Evaluation

- Since story boundaries are rather abrupt changes of focus, story boundary evaluation is modeled on the evaluation of shot boundaries (the cuts, not the gradual boundaries). A story boundary is expressed as a time offset with respect to the start of the video file in seconds, accurate to nearest hundredth of a second. Each reference boundary is expanded with a fuzziness factor of five seconds in each direction, resulting in an evaluation interval of 10 seconds.
- A reference boundary is detected when one or more computed story boundaries lies within its evaluation interval.
- If a computed boundary does not fall in the evaluation interval of a reference boundary, it is considered a false alarm.

- Story boundary recall= number of reference boundaries detected/ total number of reference boundaries
- Story boundary precision= (total number of submitted boundaries minus the total amount of false alarms)/ total number of submitted boundaries
- The evaluation of story classification is defined as follows: for each reference news segment, we check in the submission file how many seconds of this timespan are marked as news. This yields the total amount of correctly identified news subsegments in seconds.
- News segment precision = total time of correctly identified news subsegments/ total time of news segments in submission
- News segment recall = total time of correctly identified news subsegments / total time of reference news segments

Comparability with TDT-2 Results

Results of the TRECVID 2003 story segmentation task cannot be directly compared to TDT-2 results because the evaluation datasets differ and different evaluation measures are used. TRECVID 2003 participants have shown a preference for a precision/recall oriented evaluation, whereas TDT used (and is still using) normalized detection cost. Finally, TDT was modeled as an on-line task, whereas TRECVID examines story segmentation in an archival setting, permitting the use of global information. However, the TRECVID 2003 story segmentation task provides an interesting testbed for cross-resource experiments. In principle, a TDT system could be used to produce an ASR+CC or ASR+CC+Audio run.