A Study Of Audio-based Sports Video Indexing Techniques

Mark Baillie

UNIVERSITY
of
GLASGOW

Thesis Submitted for the degree of Doctor of Philosophy

Faculty of Information and Mathematical Sciences

University of Glasgow

2004

Abstract

This thesis has focused on the automatic video indexing of sports video, and in particular the sub-domain of football. Televised sporting events are now commonplace especially with the arrival of dedicated digital TV channels, and as a consequence of this, large volumes of such data is generated and stored online. The current process for manually annotating video files is a time consuming and laborious task that is essential for the management of large collections, especially when video is often re-used. Therefore, the development of automatic indexing tools would be advantageous for collection management, as well as the generation of a new wave of applications that are reliant on indexed video.

Three main objectives were addressed successfully for football video indexing, concentrating specifically on audio, a rich and low-dimensional information resource proven through experimentation. The first objective was an investigation into football video domain, analysing how prior knowledge can be utilised for automatic indexing. This was achieved through both inspection, and automatic content analysis, by applying the Hidden Markov Model (HMM) to model the audio track. This study provided a comprehensive resource for algorithm development, as well as the creation of a new test collection.

The final objectives were part of a two phase indexing framework for sports video, addressing the problems of: *segmentation and classification of video structure*, and *event detection*. In the first phase, high level structures such as Studio, Interview, Advert and Game sequences were identified, providing an automatic overview of the video content. In the second phase, key events in the segmented football sequences were recognised automatically, generating a summary of the match. For both problems a number of issues were addressed, such as audio feature set selection, model selection, audio segmentation and classification.

The first phase of the indexing framework developed a new structure segmentation and classification algorithm for football video. This indexing algorithm integrated a new Metric-based segmentation algorithm alongside a set of statistical classifiers, which automatically recognise known content. This approach was compared against widely applied methods to this problem, and was shown to be more precise through experimentation. The advantage with this algorithm is that it is robust and can generalise to other video domains.

The final phase of the framework was an audio-based event detection algorithm, utilising domain knowledge. The advantage with this algorithm, over existing approaches, is that audio patterns not directly correlated to key events were discriminated against, improving precision.

This final indexing framework can then be integrated into video browsing and annotation systems, for the purpose of highlights mapping and generation.

Acknowledgements

I would like to thank the following people.

My supervisor Joemon Jose, for first inviting me start the PhD during the I.T. course, and also for his support, encouragement, and supervision along the way. I am also grateful to Keith van Rijsbergen, my second supervisor, for his guidance and academic support. I never left his office without a new reference, or two, to chase up.

A big thank you is also required for Tassos Tombros for reading the thesis a number of times, especially when he was probably too busy to do so. I don't think ten half pints of Leffe will be enough thanks, but I'm sure it will go part of the way.

I'd also like to mention Robert, Leif and Agathe for reading parts of the thesis, and providing useful tips and advice. Thanks also to Mark for the early disccusions/meetings that helped direct me in the right way.

Vassilis for his constant patience when I had yet 'another' question about how a computer works, and also Jana, for the times when Vassilis wasn't in.

Mr Morrison for discussing the finer points of data patterns and trends - and Thierry Henry.

The Glasgow IR group - past and present, including Marcos, Ian, Craig, Iain, Mirna, Di, Ryen, Iraklis, Sumitha, Claudia, Reede, Ben, Iadh, and anyone else I forgot to mention. Its been a good innings.

Finally, special thanks to my Mum for being very patient and supportive, as well as reading the thesis at the end (even when the Bill was on TV). My sister Heather (and Scott) for also reading and being supportive throughout the PhD. Lastly, Wilmar and Robert for being good enough to let me stay rent, free for large periods of time.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Mark Baillie*)

To the Old Man, Uncle Davie and Ross.

Gone but not forgotten!

Publications

The publications related to this thesis are appended at the end of the thesis. These publications are:

- **Audio-based Event Detection for Sports Video**
  Baillie, M. and Jose, J.M. In the 2nd International Conference of Image and Video Retrieval (CIVR2003). Champaign-Urbana, Il, USA. July 2003. LNCS, Springer.
- **HMM Model Selection Issues for Soccer Video**
  Baillie, M. Jose, J.M. and van Rijsbergen, C.J. In the 3rd International Conference of Image and Video Retrieval (CIVR2004). Dublin, Eire. July 2004. LNCS, Springer.
- **An Audio-based Sports Video Segmentation and Event Detection Algorithm**
  Baillie, M. and Jose, J.M. In the 2nd IEEE Workshop on Event Mining 2004, Detection and Recognition of Events in video in association with IEEE Computer Vision and Pattern Recognition (CVPR2004), Washington DC, USA. July 2004.

Glossary of Acronyms and Abbreviations

AIC	Akiake Information Criterion
ANN	Artificial Neural Network
ANOVA	ANalysis Of VAriance
ASR	Automatic Speech Recognition
BIC	Bayesian Information Criterion
CC	Cepstral coefficients
DCT	Discrete Cosine Transformation
DP	Dynamic Programming
DVD	Digital Versatile Disc
EM	Expectation Maximisation
FFT	Fast Fourier Transform
FSM	Finite State Machine
GAD	General Audio Data
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
i.i.d	identically and independently distributed
JPEG	Joint Photographic Experts Group
kNN	k-Nearest Neighbours
KL	Kullback-Leibler Distance
KL2	Symmetric Kullback-Leibler Distance
LPCC	Linear Prediction Coding Cepstral coefficients
LRT	Likelihood Ratio Test
MCE	Minimum Classification Error
MIR	Music Information Retrieval
MFCC	Mel-frequency Cepstral coefficients
MDL	Minimum Description Length Distance
ML	Maximum Likelihood
MPEG	Potion Pictures Expert Group
PCA	Principle Components Analysis
PDF	probability density function
SVM	Support Vector Machine
ZCR	Zero Crossing Ratio

Common Terms

Class	A content group or category of semantically related data samples.
Frame	A single unit in a parameterised audio sequence.
State	Refers to the hidden state of a Hidden Markov model.

GMM Notation

X	a set of data vectors
x	a sample data vector
d	the dimensionality of the data vectors in X
k	the k^{th} mixture component
M	total number of mixture components in the GMM
μ_k	the mean vector for the k^{th} mixture component
Σ_k	the covariance matrix the k^{th} mixture component
α_k	the weighting coefficient for the k^{th} mixture component
C	number of classes
ω_c	the c^{th} class
θ	the GMM parameter set
Θ	A parameter set of GMM models, $\Theta = \{\theta_1, \ldots, \theta_C\}$

HMM Notation

λ	HMM model
N	number of states
i	i^{th} state
j	j^{th} state
O	an acoustic observation vector sequence
o_t	the observation vector at time t
q_t	the current state at time t
a_{ij}	the transition probability of moving of moving from state i to state j
$b_j(o_t)$	the emission density PDF for state j
M	number of mixture components
k	k^{th} mixture component
α_{jk}	the k^{th} mixture component for state j
μ_{jk}	the mean vector for the k^{th} mixture component for state j
Σ_{jk}	the covariance matrix the k^{th} mixture component for state j
Λ	set of HMMs
λ_c	HMM for the c^{th}

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis is a report of an investigation into the problem of video indexing, focusing specifically on the sub-domain of football video. Each step taken in the development of two audio-based domain specific indexing tools for football video, is described.

1.1 Background

The development of automatic indexing tools for video is important for a variety of reasons. The current process for manually annotating video files, required for efficient storage of video, is a time consuming and laborious task. This task can take a team of trained librarians ten hours to fully index one hour of video (Del Bimbo (1999)). Automating the annotation process will assist the workload for current annotators, as well as provide a relatively cheap means of efficient archiving for all, particularly for those collections that cannot afford to be managed by trained librarians.

Efficient storage is essential for the management of large collections, especially when video is often re-used. For example, searching for specific historical content is often required for the production of current news reports. Automatic video indexing will allow for video summary generation, mapping of content, and efficient browsing[1].

Video indexing is a difficult problem encompassing many different research fields, such as computer vision, pattern recognition and signal processing. In this thesis, I

[1]Appendix C provides a general overview of video indexing and the many problems faced.

reviewed work from a variety of related fields, such as audio parametrisation, feature selection, content analysis, statistical modelling and model selection, audio segmentation and speech recognition. Reviewing these related fields was necessary for gaining an understanding of the tools and techniques, borrowed from such research domains, for solving the problems faced in video indexing.

One particular video indexing challenge is recognising high-level semantic content through the classification of low level features. Attempting to bridge this semantic gap, between high-level content and low level features, for general video is challenging. To minimise this problem, there has been recent focus on specific video domains, such as news, film and sport, with prior knowledge of the genre being utilised for developing domain specific indexing tools. This was a motivating factor, as to why, I have focused on the sports video domain, and in particular the sub-domain of football video.

Another reason why video retrieval is difficult, is the number of modalities that encompass a video stream, such as vision, motion, audio and text. To narrow the video indexing problem further, this thesis concentrated solely on developing audio-based indexing tools for video. Audio is a rich, low dimensional information source that has been shown through experimentation to be an important modality (Alatan et al. (2001), Chaisorn et al. (2003)). Developing a robust audio indexing algorithm first, before further modalities can be integrated in the future.

The aim of this thesis, is to develop techniques that can automate the annotation process for football video. In particular, this thesis focuses on three main problems for domain specific video indexing. The first problem is the gathering of domain knowledge, an important step towards effective indexing algorithms.

Two specific indexing problems are also identified in this thesis: *structure segmentation and classification*, and *event detection*. Structure segmentation and classification is required for identifying the semantic components that constitute a football video sequence, allowing the entire video content to be mapped. The video structure can be presented to the user in a form similar to the table of contents of a book, allowing for instant access to areas of interest, or even whole segments to be skipped, during playback. Mapping video content is important and can enable efficient browsing, retrieval of specific sequences, and support further high-level indexing algorithms.

Event detection is crucial for automating highlights generation, and in general providing detailed video summaries. Logging these key moments can enable future search or summarisation. This technique is also useful for digital TV, where current interactive services allow the viewer to watch highlights during the match. Automating the largely manual highlights generation process is desirable.

As part of this thesis, I rigorously evaluate a number of widely applied techniques, across a large and varied test collection, for the two defined indexing problems. The reasoning behind this, is to facilitate a shortage of comparative research into techniques for audio-based content indexing across a common test collection. Comparing a number of approaches over a large test collection, provides an insight into which techniques are accurate for both indexing problems. The methodologies applied in this thesis, and the outcome, are now summarised below.

1.1.1 Thesis Summary

A number of steps were taken to address the problems identified at the start of the thesis. The first step was to analyse the football video domain, gathering as much knowledge as possible. One of the major weaknesses of current sports indexing algorithms was found to be a lack of understanding, for each sports domain, or the under utilisation of domain knowledge. Many algorithms did not take full advantage of content found in sports video that can be integrated during the development of indexing algorithms. As part of the thesis, a large test collection was generated and manually inspected[2]. A video model for football broadcasts was also defined, which assisted the annotation of the test collection, and algorithm design. This video model reflected the temporal relationship of known structures contained in a typical broadcast.

The next step was to review related research into acoustic modelling and parametrisation. A limitation identified in current audio indexing algorithms was the choice of feature sets. A standard solution was to apply the same approaches used in automatic speech recognition systems, for modelling non-speech sound. To address this limitation, three well-researched feature sets were evaluated across a large test collection: Cepstral Coefficients (CC), Mel-frequency Cepstral coefficients (MFCC) and Linear Predictive Coding Cepstral coefficients (LPCC). The candidate feature sets were cho-

[2]The generation of a large test collection for football video was required because currently there was not a standard test collection for football video freely available.

sen after reviewing the current state of the art in a variety of fields, such as automatic speech recognition and general audio classification. The overall aim was to evaluate the suitability of each, for representing content found in football audio, especially given the many non-speech structures.

To evaluate the three feature sets, a suitable statistical model was required for modelling audio content. From modelling typical acoustic content found in football, the discrimination properties of each feature set could be measured in terms of classification error. The rationale for deciding upon a modelling solution, was that such frameworks were proven to be more successful than the application of ad hoc techniques. Applying heuristic rules were proven not to generalise well, because of the variation found across video, even from the same domain. Thus, two statistical models were selected, and compared, throughout the thesis: the Gaussian Mixture Model (GMM) and the Hidden Markov Model (HMM). Another shortcoming of current research into audio indexing was a scarcity of comparative studies, over a large test collection, and was the reason why two frameworks were selected. Evaluating a number of approaches provided a guide to the final accuracy of each indexing algorithm, given the absence of a baseline system.

The GMM was applied to the evaluation of the three candidate audio feature sets, using a labelled data set of audio content found in football video. From experimentation it was discovered that employing the Mel-frequency Cepstral coefficients minimised classification error, and was significantly better the other two feature sets.

A number of related works in audio indexing also apply the MFCC feature set, using the same coefficients employed in speech recognition systems. From further evaluation of the MFCC, it emerged that using a simpler feature set with coefficients not normally applied to speech recognition systems, resulted in a significant increase in accuracy. In fact, the typical ASR feature set resulted in a significant increase in error, in some cases by 10%. Drawing a conclusion from this experiment, it was highlighted that modelling broad content found in football audio, required different feature set configurations, than for recognising speech alone.

Provided with a method for parameterising the test collection acoustically, the entire audio track was then modelled, using the HMM, for the purpose of discovering the underlying data structures found in football audio. Previously, the HMM had been utilised for problems, such as data mining and clustering temporal data, with a degree of success (Li & Biswas (2000) and Smyth (1997)). Automatic content analysis was planned for two main reasons. The first reason was to verify that the HMM was a suitable representation of the given data. The second reason was to discover structures and sub-structures found in the audio track, gathering further domain knowledge. Those data structures found by the HMM were then compared to the content discovered through manual inspection.

The findings from both investigations reported similar trends. The automatic content analysis also recognised new sub-structures not previously identified through inspection. Two objectives of the experiment were verified, highlighting that the HMM was a robust tool for mining the underlying data structures found in audio.

After identifying the main semantic structures found in football audio, through manual inspection and automatic content analysis, statistical representations were generated for each content class, using both GMM and HMM. However, HMM application for video indexing was often applied, for general audio classification, in an ad hoc manner. In particular, there was little consideration for the underlying data structures found in different content classes (Eickeler & Muller (1999), Huang & Wang (2000), Wang et al. (2000), Xie et al. (2002)). A standard solution was to apply the same HMM settings for modelling various content, even if these groups differed dramatically. In this thesis, the effect of poor HMM model selection was verified by comparing three broad content classes found in football audio. It was highlighted, that HMM models with a differing number of states and mixture components, were optimal for each class, in terms of classification accuracy.

It was believed that HMM model selection was an important step in algorithm development. The data structure for each content was considered separately during HMM generation. It was also important to consider problems, such as overfitting, parameter estimation, and limited training data, during model generation. Hence, three strategies were compared providing an overview of model selection: an exhaustive search approach, the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC). The classification accuracy for the three model selection strategies was measured, to evaluate what method chose the best performing model.

The results of the experiment indicated no significant difference, between the three selection strategies, however, both AIC and BIC were more effective in terms of complexity and computation, when compared to the exhaustive approach. These methods were also shown to generalise to different content classes better than an exhaustive selection strategy. The experiment illustrated the importance of selecting models carefully, for new content.

After both the best audio feature set representation, and two statistical frameworks (and the surrounding implementation issues involved) were identified, each indexing problem was studied separately. The first problem addressed was *structure segmentation and classification*. A widely applied method for audio segmentation is a Model-based approach. Typically the audio stream is divided into equal sized, overlapping windows that are then classified, given a set of statistical models, with a number of Model-based algorithms were evaluated and compared across a large test collection. Another method was also compared; a robust Metric-based segmentation algorithm called BICseg. This technique was previously applied for the segmentation of speech. In this thesis, this algorithm was adapted for the problem of structure segmentation, improving both the accuracy and efficiency. From experimentation, this approach was also shown to be more effective, than the widely applied Model-based segmentation algorithms to this problem.

From the experiment, another interesting result reported. Findings from related work into audio content classification suggested that the HMM outperformed GMM when directly compared. This finding, however, did not generalise to the broad content classes found in football audio. No significant difference in classification error emerged, when comparing both the GMM and HMM as classifiers. This provided evidence that selecting a complex statistical representation, such as the HMM, before investigating simpler models (such as GMM), would not necessarily equate to improved classification performance.

The second video indexing problem, *event detection*, was then investigated. Assuming that important events are correlated to periods of commentator excitement and crowd cheering, key events were recognised by classifying these patterns. These algorithm addressed a particular shortcoming of audio-based event detection algorithms for sport. Current audio-based event detection algorithms do not consider the audio similarities of unrelated sounds such as crowd singing, which can easily be mistaken for key events. In this thesis, the distinction between these similar sound classes were recognised automatically by applying domain knowledge, gained through inspection and automatic content analysis, improving accuracy.

The event detection algorithm, integrating the robust BICseg algorithm adapted for this problem, was shown to be the most precise strategy, when compared alongside a number of event detection algorithms. Each algorithm was evaluated measuring the precision of detecting key events reported in official match reports of football games.

Again both GMM and HMM were compared as classifiers for the event detection algorithm. It was found through experimentation that the HMM provided more accurate results when compared to the GMM, for this problem. This result differed from the problem of classifying broad content structures, where there was no difference found between GMM and HMM. For modelling very specific content such as crowd cheering and speech, the HMM provided a more robust solution.

1.1.2 Original Contributions

Taken as a whole, the thesis concluded that audio is an important information source for indexing football video. Three main objectives were also addressed successfully as part of the thesis. The problem of sports domain knowledge was investigated, both through inspection, as well as applying an automatic content analysis technique. This study provided a comprehensive resource for future algorithm development, as well as a test collection for a comparative analysis. The generation of a large test collection was required as such a source is not available at present.

A limitation discovered in the sports video indexing literature, was that the problem of *structure segmentation and classification* was not addressed. Hence, in this thesis a new structure segmentation and classification algorithm was developed for football video. The advantage with this algorithm is that it is robust and can generalise to many problems, as well as transfer to other video domains. The algorithm can be integrated into a video browsing or annotation systems for mapping content (see Appendix K for an illustration of such a system).

An audio-based event detection algorithm utilising domain knowledge was also developed. The advantage with this algorithm, over existing systems, is that audio patterns not directly correlated to key events were discriminated against, improving the accuracy of the event detection algorithm. This algorithm can also be integrated into video browsing and annotation systems, for the purpose of highlights mapping or generation.

To summarise, a two layered indexing framework for sports video was developed. In the first phase, high level structures such as Studio, Advert and Game sequences are identified. In the second phase, key events in the Game sequences are recognised automatically, generating a summary of the video.

1.2 Structure of the Thesis

After the introduction, the work in this thesis is divided into ten chapters. An outline of the remaining chapters is given below.

Chapter 2: The problem of sports video indexing is reviewed in more depth in this chapter, with particular attention on football video indexing. After reviewing the current state of the art, potential weaknesses are identified that form the motivation behind the thesis. Three problems are outlined and solutions are proposed. Also, the overall thesis aims are discussed in greater detail.

Chapter 3 The findings of an investigation into football video through manual inspection are recounted. A video model that defines the temporal relationship between content found in a typical football video is introduced.

Chapter 4: The statistical models that are applied throughout this thesis, and the reasoning behind the selection of such frameworks are discussed. The main implementation issues surrounding each statistical model are also reported.

Chapter 5: A number of related fields are reviewed, identifying three candidate feature sets for parameterising the audio stream of football video files. The three chosen feature sets were Cepstral coefficients (CC), Mel-frequency Cepstral coefficients (MFCC) and Linear Prediction Cepstral coefficients (LPCC).

Chapter 6: Three candidate feature sets are evaluated, using a test collection containing typical content found in football video. Once the feature set that minimised classification error was discovered, the configuration is further analysed through experimentation.

Chapter 7: A data mining approach is applied for automatically discovering content found in the audio-track of football video. Both the application of HMM to the problem of automatic content analysis, and the findings are reported.

Chapter 8: The importance of optimal model selection is illustrated when applying the HMM. Three model selection strategies are then formally compared: an exhaustive search approach, the Akaike Information Criterion, and the Bayesian Information Criterion.

Chapter 9: Seven segmentation and classification algorithms are compared, designed specifically for mapping the content of football video. Five Model-based and two Metric-based segmentation and classification algorithms are compared formally over the test collection. These algorithms were integrated into a framework using either HMM or GMM classifiers. The accuracy of both frameworks is also compared.

Chapter 10: An event detection algorithm framework is developed. The specific motivation behind its development is reported, with the approach being compared alongside a number of other existing algorithms.

Chapter 11: The main contributions that this work has made are presented. Possible future research directions are discussed as a result of the investigation in this thesis.

Chapter 2

Sports Video Indexing

2.1 Introduction

Televised sporting events are now commonplace especially with the arrival of dedicated digital TV channels. As a consequence of this, large volumes of such data are generated and stored online. Efficient indexing of each new video is required during the archiving process, as content within these large collections is often reused. However, current archiving methods are manual and labour intensive, where the off-line logging of video can take a team of trained librarians up to ten hours to fully index one hour of video (Del Bimbo (1999)). Automatic indexing is a viable alternative to the labour intensive procedures, currently in practise, although feasible solutions have not been developed so far.

There have also been recent advancement in how sports video is presented to the viewer. While watching a sporting event, the viewer can now browse through a number of interactive services, such as current in-game statistics and match highlight replays. These interactive services can be accessed whilst watching the match, although generation of services such as highlight replays, is again largely dependent on human involvement. Development of tools that can automatically detect key events as they happen, will minimise human intervention and allow for a new generation of interactive services to enrich the viewing experience.

Automatic indexing of video content is beneficial not only for real time broadcast production and interactive services but also advantageous to the consumer. With the advancement of cheaper storage devices and high quality video compression algorithms, digital video can be recorded on set top devices, such as Sky+ (SKY+ (2003)) and TiVo (TiVo Inc. (2003)). Once a video file is captured on disc, the viewer is required to manually bookmark areas of interest. Through the development of more advanced indexing algorithms, viewers will potentially be provided with automatic content summaries, efficient browsing and search functionalities once a video file is recorded to disc, and indexed automatically. This can allow the same interactive services that are currently enjoyed for entertainment DVDs, such as an interactive table of contents.

Automatic sports video indexing is also important for the generation of new applications such as video-based sports coaching systems. There has been an increased development of interactive coaching tools that reuse pre-recorded video sequences. These coaching tools are designed to assist coaching by visually illustrating good sporting technique or for the analysis of team tactics. However, such systems are currently reliant on manual video logging by the user (Dartfish (2004)), which is expensive both in terms of time and annotation expertise. Automating this video annotation process would increase the availability to the general consumer, and minimise the complexity of such applications.

For the above reasons, there has been a concerted effort from the video retrieval community to research methods for the automation of sports video indexing. Such investigation has included the recognition of the pitch markings, player tracking, slow-motion replay detection, the identification of commentator excitement and crowd cheering recognition. This chapter will provide an overview of such techniques, with specific concentration on the sub-domain of football video[1]. From the review of football video indexing, two potential indexing problems are identified, which will be examined in more depth throughout the thesis. These problems are: *segmentation and classification of video structure*, and *event detection*. Segmentation and classification is important for mapping content and providing an instant overview of the structures found in each video. Event detection is beneficial for generating automatic summaries of the important moments during a football match. The importance of both problems will be

[1]For the rest of the thesis, the term football will correspond to the sport also known as soccer in some parts of the world. The terms football and soccer will be interchangeable but will correspond to the same sub-domain of sports video.

discussed in more detail throughout the chapter, as well as the benefits for future applications.

The remainder of this chapter is structured as follows. Section 2.2 provides a general overview of sports video indexing, reviewing the current state of the art. There is specific focus on football video indexing, highlighting the limitations within this field. Following on from this, section 2.3 discusses the motivation behind the thesis, and how the problems of structure segmentation and classification, and event detection are addressed. The chapter is summarised in section 2.4.

2.2 Sports Video Indexing

This section provides an overview of sports based indexing. First the methods for identifying sport sequences are discussed, before focusing upon on specific research into sporting sub-domains, such as American Football (Chang et al. (1996), Intille (1997) Babaguchi et al. (2002)), Baseball (Rui et al. (2000)), Basketball (Nepal et al. (2001)), Motorcar Racing (Petkovic et al. (2002)), Tennis (Sudhir et al. (1998)), Olympic sports events (Assfalg et al. (2002)), and Football (Assfalg et al. (2003), Gong et al. (1995), Xie et al. (2002)). In the following sections, these works are studied in more detail.

2.2.1 General Sport

Kobla et al. (2000) implemented a genre detection algorithm for identifying sports video. A decision tree-based classifier system detected video files that contain sport sequences using features such as superimposed graphics, detection of slow motion and camera motion traits, related to sport. The authors assume slow motion replays are signified by a series of virtually similar sequential frames, that create the effect of slow motion. The algorithm was 93% accurate in labelling sports clips from non-sports clips.

Detection of slow motion is also a useful indicator for the recognition of exciting highlights (Pan et al. (2002)). The content of slow motion replays is normally an exciting event, displayed from a number of camera angles. These replays are shown after an event takes place or as part of an analysis section during breaks in the sport. Replays are detected by modelling the broadcast patterns that usually occur in Sports broadcasts that indicate a highlight replay, which can be used for summarisation. Highlight replays are assumed to contain exciting incidents. These sequences are modelled by a Hidden Markov model, which identifies patterns in the visual stream such as high visual similarities between frames and gradual transition editing effects. Another important clue is the recognition of logos which signify a slow motion replay. This could be a logo for the broadcast channel, show or sporting event. 94% of the slow motion replays were detected across a large data set, but with only an accuracy of 2.1%. Cited as one of the downfalls of the system was the reliance of a broadcast logo appearing during the slow motion replay. Reliance of this logo feature does place doubts on how portable the algorithm would be across various TV shows and channels.

2.2.2 Olympic Sports Categorisation

Assfalg et al. (2002) implemented a genre detection algorithm, classifying sub-genres of sport, specifically various Olympic sporting events. Visual information was extracted from keyframes selected from 10 sports sequences, including swimming, judo and athletics. For each sport, the keyframes were first classified into one of three classes; "*crowd*", "*player*" or "*playfield*" using edge and colour features. These categories are related to the different shot types found in sport sequences. The main camera shot focuses on the playing field (or court) where the sport takes place. The remaining two shot types are player close ups and crowd close ups.

The keyframe colour distribution, the edge distribution and orientation were all used to discriminate between various shot types within a Neural Network classifier. For example, playing field shots were considered to have long lines and edges, with large regions of colour associated with the playing surface. Player shots would have smaller regions of colour that represent the player with more edges, while audience shots would contain a lot of texture and many small edges. The neural network classifier was trained and tested on 400 keyframes taken from a single Olympic competition. The classifier was then evaluated on 200 more keyframes, labelling 80% of the "*crowd*" shots, 84% "*player*" shots and 92% of the "*playfield*" correctly.

Again using a Neural Network, the keyframes were then classified into their appropriate sport category with a varying degree of success. Only 37% of the javelin keyframes were labelled correctly, while 88% of all track keyframes were identified. Using only the *playfield* keyframes, the accuracy increased, with the worst group again being javelin with 58%. Swimming (96%) and high diving (97%) were the most accurate classes.

It would be of interest to evaluate how effective the sub-genre detector would be using other sport categories or from a different Olympics. The currently selected sports each have a very distinctive playing surface. For example, soccer is played on a green grass pitch, track events take place on a red synthetic track shaped in an oval, swimming a blue pool and a judo competition is fought on white to yellow soft mats. How powerful this approach would be in discriminating between sports played on very similar playing fields (e.g. soccer, American football and rugby) would be questionable. A further layer that modelled the distinctive lines found on each court (Sudhir et al. (1998), Gong et al. (1995)), instead of utilising edge and line features in a simplistic way, could improve results.

2.2.3 American Football

American football is a stop and start sport, where each game is divided into individual plays. The attacking team will have 4 attempts to advance 10 yards, each attempt is one play. At the beginning of each attempt, both teams will line up either side of the ball, where lines on the pitch, help identify, how far the team is away from the next ten yards. The overall goal is to reach the opposing End-Zone, (the last ten yards in the opponents half). To identify scoring events, Chang et al. (1996) use a combination of, pitch location, keywords and crowd cheering. Keywords include "*touchdown*", which are identified by utilising a simple template matching procedure. Incoming audio sequences were matched against these stored keyword templates generated for a set of pre-determined keywords, associated with scoring.

To improve event detection, Chang et al. (1996) also detected specific playing surface markings associated with the End-Zone and a simplistic crowd cheering recogniser (described in further detail in Chapter 10). The End-Zone was detected using edge detection and pixel linking techniques, to identify lines attributed to the zone. A small data set was used to evaluate the performance of these features, detecting 4 out of 5 scoring events using key word detection alone, with 3 false alarms. Combining both keyword spotting and crowd cheering detection, the same number of events were detected but with only 1 false detection. With the inclusion of the End-Zone recognition feature, the same number of scoring events were picked up but with no false alarms. It would be difficult to measure the effectiveness of this approach, especially employing simple audio and visual statistics from the video file, given the very limited data set size. Little information was supplied about the data set itself, or if the same data was used for both algorithm training and evaluation.

Babaguchi et al. (2002) also detected scoring events based on keyword spotting. Instead of transcribing speech, keywords were identified in the text stream that accompanies some video files. Video sequences were segmented into shots and then labelled as containing important events, if they were time aligned, with a keyword detected in the closed caption stream[2], such as ‘touchdown’. The disadvantage with this approach is the availability of closed caption text. For many programmes, closed caption text is not available, especially for live programmes that require speech to be manually transcribed as it is spoken, which is an expensive and skilled task. Commentators also frequently use keywords such as “touchdown” in regular conversation or when analysing previous events. However, this information integrated into an event detection system, similar to Chang et al. (1996) would be a potent event detection tool. As automatic speech recognisers improve, this type of information source will also be more readily available.

Intille (1997) implemented a player tracking system for American football sequences in a closed world environment. The goal of the system was to create a play book for coaching and sport management purposes. Player tracking was on a single static camera. This was important so that the pitch markings could be used as a reference point for both player tracking and player location. Due to the stop and start nature of American football, the initial location of players can be identified during the start of each play, by creating a model of the overall playing surface. This model recognised the markings on the pitch such as each 10 yard lines, and the associated yard numbers.

[2]A textual stream accompanied by TV programmes in America and Japan, used to display subtitles for the hard of hearing. Speech is manually transcribed to text either before broadcast, or manually on the fly for live shows.

2.2.4 Baseball

A baseball match does not have a set time limit and can therefore span a large number of hours. Each baseball team has 9 innings to bat, where the attacking team has three outs per innings. An out occurs when a batter misses three pitches or is caught after striking the ball. Once the batter hits the ball, the aim is to run around the diamond touching the 4 bases, the 4th being the home base. Once this home base is reached, a run is given to the attacking team. The team with the most runs is the winner.

Rui et al. (2000) summarise baseball matches by compressing the video file into a sequence of highlight clips. This is achieved by fusing a number of information sources together. The first information source being the sound of a baseball bat striking a ball, which marks the beginning of an action sequence. This event is detected using template matching (a stored representation of the sound). Periods of excited commentator speech were also recognised (see Chapter 10) using frequency-based audio features modelled by a Support Vector Machine (SVM) classifier, where excited speech can be assumed to be correlated to exciting moments in the game.

Probabilistic fusion was applied to combine the two outputs for baseball hit detection and excited speech, deciding which sequences were to be labelled as a highlight or not. 49 out of 66 clips were correctly identified, however there was no indication of the precision of this algorithm. i.e. whether the algorithm detected a high level of false events. This algorithm is useful for detecting events where the ball is hit, mapping the starting location of each. The further level of commentator emotion can filter what events could be considered exciting. However, a large part of a baseball match is the battle between the pitcher and batter. If a batter is “struck out”, this exciting event would be missed.

2.2.5 Basketball

Basketball is a high tempo, high scoring sport. The turnover from attack to defense is constant, where the aim is to shoot the ball through the opposing teams basket. Two points are awarded if the attacker shoots inside a semi-circle zone starting from underneath the basket. Three points are awarded for scoring attempts outside this zone. Again the aim is to score more points than the opposing team. Nepal et al. (2001) define a set of 5 heuristic rules to detect these scoring events in basketball sequences. This is achieved by crowd cheering recognition, detecting superimposed ‘score’ graphics overlayed on the video sequence and a change in play detector. It can be assumed that the crowd react to scoring events by cheering. resulting in an increase in ‘loudness’ in the audio stream. If the audio stream is louder than a threshold, a crowd cheering sequence is flagged. Thresholding the ‘loudness’ of an audio stream is not a good discriminator between sound events such as crowd cheering, music and speech, as illustrated in later chapters.

Recognising on-screen scoring updates can be achieved by detecting on-screen superimposed graphics. Coefficients found in the MPEG-1 stream which represent ‘sharp edges’ associated with graphic overlays are detected. A change in play recogniser is also used to determine the start and end of each attack, called a turnover. This is detected by recognising a change in camera movement using the MPEG-1 motion vectors. A change in play is assumed to be associated with a switch from the camera panning left to right or vice versa.

These features are combined using a set of rules, such as a goal segment will occur “three seconds surrounding a crowd cheer” and also “10 seconds surrounding a superimposed display of the score graphic”. Two video clips were used to evaluate the approach, where it was found that combing the play change detector with either crowd cheering or score graphics detection, was the most effective algorithm. This achieved 88% recall for both video sequences.

From structure based segmentation and classification algorithms for news video (Browne et al. (2003)), ad hoc rules were found to be too rigid when applied to video indexing. The rules did not generalise well across a more varied test collection. The basketball event detection algorithm by Nepal et al. (2001), was only evaluated across a small test collection. It would be of interest to see how robust these ad hoc rules would be across a larger number of basketball sequences, rather than those sequences where the rules were defined. A further comparison against alternative techniques would also validate the accuracy of the algorithm.

2.2.6 Tennis

Sudhir et al. (1998) detect the court lines and the player locations to segment and label tennis sequences into six categories including “baseline-rally” and “serve and volley”. For tennis, action sequences are filmed by a single fixed camera position, which is centred on the entire playing surface. After the end of each action sequence, the director will switch to different shot types that replay the sequence across various angles as well as displaying player and crowd close up shots. These action sequences are first segmented and classified by detecting the tennis court, which is only in view during this main camera shot. A simple line detection algorithm extracts the lines from each keyframe and is then compared to a predefined geometry model of a tennis court. If the lines extracted from the keyframe match the model, the shot was labelled as an action sequence.

Object tracking was then used to label these action sequences into one of six categories (e.g. “baseline-rally”). During a tennis rally there are two dominant moving objects, that of the players. By locating the player objects in relation to the court lines, the rally type can be determined. For example, if both players are found along the baseline during the entire rally, this would be labelled as “baseline-rally” point. This appears to be a powerful indexing approach for summarising tennis sequences, however, there was no overall evaluation on the system.

2.2.7 Formula One

Petkovic et al. (2002) detect exciting moments in Formula-1 grand prix motorcar races for automatic highlights summarisation. Multi-modal information, such as commentator excitement, keyword spotting in the commentary track, and the annotation of superimposed graphics that contained driver and lap data, were all used to summarise races. Another clue used was the detection of a high proportion of dust which indicated “burn outs”. That is when a car slides off the track, usually resulting in driver retirement from the race. To fuse these information sources together, a Dynamic Bayesian Network (DBN) is used.

To evaluate the summarisation algorithm, a test collection of three races was used comparing DBNs fusing various feature combinations. A DBN with audio alone, correctly found 80% of those sequences containing commentator excitement, with the precision of the DBN being 50%. These commentator excitement events covered 50% of all exciting events. Integrating the remaining information sources, over 80% of all exciting events were detected, although no information was provided about the precision of the algorithm.

2.2.8 Football Video Indexing

Research into indexing techniques for football video has included player tracking, playing surface classification, camera shot classification and crowd cheering recognition. The aim of all these approaches are to locate interesting events that can then be used for summarisation. The following sections provide an overview of relevant research for each technique.

2.2.8.1 Player Tracking

Automatic player tracking would be advantageous for both event detection and the automatic generation of coaching systems (Dartfish (2004)). Team formations and player movement can be interactively displayed allowing for the analysis of team performance for future tactics. Specific passing movements or player skills can also be searched by querying automatically indexed sequences for similar player or team movements. Currently sports management and coaching systems such as Dartfish (2004) manually index video. Intille (1997) have made an attempt to automate this process for American football sequences. However the problem of player tracking particulary on produced video containing multiple shots is hard. And for the problem of sports coaching, other solutions such as placing GPS (Global Positioning System) devices on players may be a more realistic but expensive solution. But tracking player movement is a useful tool for labelling and summarising important events. Locating the player's position and movement can help indicate what type of event has taken place. For example, Yow et al. (1995) generated image mosaics to display action sequences inside one image. The proposed mosaic assists visualisation of scoring events, where the movement of each player and the ball is displayed on the image. How efficient the presentation of a single image mosaic is, when compared to viewing the 10 second sequence instead is debateable. There was no quantitative results presented to indicate the value or accuracy of this technique. However tracking the motion of both the players and the ball is an important step for detecting important events and summarising player movement.

If an opposing player and the ball are closer to the goal posts, then the probability of an important event occurring increases. Seo et al. (1997) developed a simple player tracking scheme by recognising peaks in the colour histogram associated with player uniforms. The location of these peaks were then mapped backed to the original image, identifying the location of each player. A template matching scheme was applied for the actual player tracking, where the location and movement of players was related back to a model of the playing surface. Again, no quantitative results were presented to determine how accurate the algorithm was.

2.2.8.2 Playing Surface Classification

Another important indexing tool is to classify different areas of the pitch. IT is important to determine where the main camera is currently focusing (Gong et al. (1995), Assfalg et al. (2003)). If an algorithm can determine pitch location, it can also determine the probability of scoring opportunities. Gong et al. (1995) divided the pitch into zones containing different pitch markings, such as the corner flag, 18 yard box and centre circle. Keyframes taken from a single football sequence were then labelled into one of eight classes, each a specific location on the playing surface. To classify the pitch location in each keyframe, image processing techniques such as edge and line detection were applied to extract potential playing surface markings. A geometry model of the pitch was used to determine the whereabouts of the camera position using these image features, where lines were matched to the known pitch markings in the model.

A test collection of approximately 120 keyframes was used to evaluate the algorithm (Gong et al. (1995)). Overall there was a high degree of accuracy across all pitch locations (averaging over 90%), although the keyframes were taken from the same football game. It would be difficult to predict how accurate this algorithm would be across other games where there would be variations in pitch quality, camera location, light and the weather. These factors effect the algorithm performance, and should be considered.

Assfalg et al. (2003) also classify the playing surface as part of an event detection and labelling system. The playing surface was first divided into 6 zones for one half of the pitch, 12 zones overall. A play field shape descriptors are then applied to detect the pitch location using features such as the current lines, line orientation, if the corner of the pitch is in view, and the size of the (grass) pitch. To do so, both the ratio of grass pixels and the number of edges and edge orientation were extracted from each keyframe. Edge orientation bins were used to calculate the number of horizontal and vertical lines that could represent pitch markings. A Bayes classifier was then used to classify new keyframes into one of the 12 zones.

Locating the current playfield position was an important part of an overall event detection algorithm (Assfalg et al. (2003)). A Finite State Machine (FSM) then modelled the playfield location information with further features such as camera and player location, for classification of specific highlight sequences found during a match. For example, a change in camera direction, in relation to the current pitch and player location, can be an indicator of a turnover from defense to attack. To measure the algorithm, shot sequences were labelled into several classes, such as an attack on goal, a shot on goal, a turnover or a freekick. The test collection was approximately 100 sequences taken from 15 games. There were six classes overall and each new sequence was labelled using the FSM classifier. The system achieved a high degree of classification accuracy, over 90% for all new sequences, across each category.

Assfalg et al. (2003) also focused on classification of freekick type in order to differentiate between penalty, corner kick or freekick's outside the penalty box, where these freekick types would be considered important events during a football match. Approximately 60 sequences were labelled into one of the three categories, again using a FSM classifier, with over 95% accuracy, but both experiments did not indicate whether the training and test sets were the same. For example, if the test set was the same as the training set for the FSM classifiers then this could result in high classification accuracy that would not be transferable to new matches.

2.2.8.3 Camera Type Classification

As illustrated by Assfalg et al. (2003), classifying the pitch location is an important indicator for event type and importance. Another important clue is labelling the different

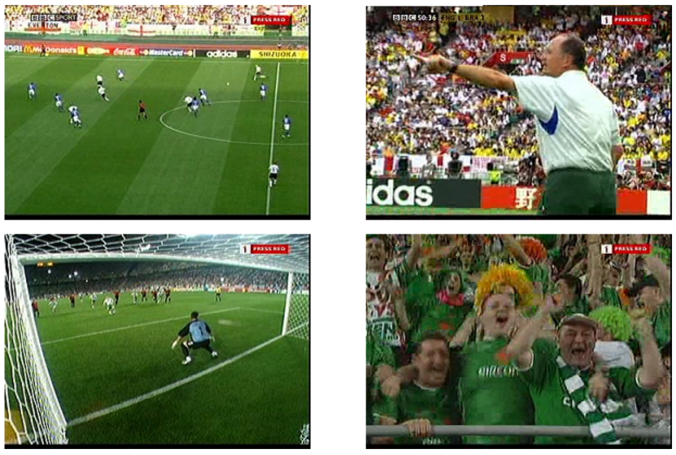


Figure 2.1: Example of shot types. The top left shot is the main or long shot, where this camera angle tracks the movement of the ball. The bottom left camera angle is called a medium shot. This also displays the action but at a closer range. The remaining two shots are examples of close ups.

shots types found during the game sequence. Figure 2.1 is an illustration of the typical camera shots found in this segment. Shot types can be grouped into a few main categories; the main camera angle, medium shots of the game from different angles and close ups of the players, fans or coaches. Detection of the main camera shot is important for extracting information such as pitch location and player tracking. For example, algorithms used by Assfalg et al. (2003), Seo et al. (1997), Gong et al. (1995) and Yow et al. (1995) all rely on the main camera angle being detected. Algorithm accuracy would decrease if there were shot sequences containing medium or close up shots, in the test collections.

Research by Ekin et al. (2003) and Xie et al. (2002) attempt to detect these camera angles. Xie et al. (2002) classify football sequences into two states; ‘Play’ and ‘Break. The ‘Play’ segments correspond to both the main camera and medium shots, where all the action sequences take place. The ‘Break’ segments correspond to player and crowd close up shots, which normally occur when there is a stoppage in the match.

Segmentation and classification of these segments is achieved by using two low level descriptors; shot activity and grass pixel ratio (first outlined in Xu et al. (2001)). These features are modelled by a collection of simple 2 to 3 state HMMs. 80% of both 'Play' and 'Break' segments are classified correctly across four game sequences (this algorithm is described in further detail later).

Xie et al. (2002) quote that their algorithm is distinctive to that of other work because "typically no more than 60% content corresponds to play", in a football sequence. Detecting the 'Play' segments can allow football video files to be compressed by displaying only these 'Play' sequences. This however contradicts the description of a perfect game, as highlighted by Hornby (2000) (see Chapter 3). Hornby indicated that not everything which is interesting occurs during the 'Play' segments. Referee decisions occur during stoppages or 'Breaks'. Therefore compression of the match is not necessarily a good thing without analysing both 'Play' and 'Break' segments together for interesting content.

Classifying shot types can also be used for event detection. In an attempt to index not only goals but other exciting events, such as action in and around the penalty box. Ekin et al. (2003) define a set of heuristic rules based on recognising patterns found in colour vision features associated with events. After shot segmentation, shots are labelled into relevant classes; the main camera shot, player "medium" or "closeup" shots, and "closeups" of the crowd or coach. Both a grass pixel ratio and an object detection sensor, along with heuristics are used to classify these shots.

First keyframes that contain a high ratio of grass pixel decided against a predefined threshold, are automatically labelled as a main shot of the pitch. The grass colour identification is performed in the HSV space. Grass is said to occupy 65-85 degrees HUE (Ekin et al. (2003)). If the shot does not contain a high ratio of grass pixels, it proceeds into the object detection stage. Keyframes with a small number of detected objects and grass pixels are labelled as 'medium' shots of players. The remaining shots are grouped as 'close ups' of the audience or crowd. To evaluate the algorithm, 184 keyframes from each class were taken from a collection of shots from three games, with an overall accuracy of 89% using object and grass pixel features, a 13% improvement over applying the grass ratio measure on its own.

After classifying shot type, goal events and close up sequences of the referee are found

(Ekin et al. (2003)). By identifying shots containing the referee, moments when a yellow or red card is shown can be found. The authors work under the assumption that the referee wears a unique uniform compared to the opposing players. This assumption dose not hold entirely, as both goalkeepers are likely to have unique that is both bright and distinctive, also. Heuristics are again used to find the referee sequences in medium shots based on this unique ‘uniform’ assumption. A medium shot containing colour objects, not associated with the outfield player uniforms or grass pixels, is assumed to be that of the referee. 16 out of 19 appearances of the referee were detected. It was not mentioned whether the test collection contained keyframes of the goalkeepers, who also wear differing uniforms to the outfield players.

Goal events were found again using heuristic rules (Ekin et al. (2003)). First the location of the main camera is determined using a penalty box detector. The hough transformation was applied to detect the three parallel lines of the penalty box. Then rules based on football video production are used to determine if a scoring event has occurred. For example, after a goal event, it is common for both slow motion replays, that were manually detected, and “closeup” shots of players and the crowd celebrating to directly follow the event. These sequences will then be followed by a main shot of the game commencing again. Goals are flagged if this sequence is followed, where the algorithm detected 27 out of 30 goals from 15 test sequences, with 45.8% precision.

2.2.8.4 Crowd Cheering

At many sporting events, and in particular football matches, the crowd react to exciting events by vocally cheering. Baillie & Jose (2003) developed a set of statistical classifiers that identify segments of audio that correspond to crowd cheering, for the purpose of event detection. The audio track was first divided into overlapping sequential segments, and then each segment was classified into one of seven pattern classes, using a maximum likelihood decision process. An “event window” filter then identified the location of key events.

Xiong et al. (2003*a*) followed a similar method, although the audio track was manually segmented instead. Audio sequences were classified into crowd cheering, speech and music groups, using a set of statistical models. Events were again assumed to be correlated to crowd cheering. However, Xiong et al. (2003*a*) ignored the audio similarities of other sounds such as crowd singing, not so directly correlated to key incidents, resulting in a number of falsely detected events. Baillie & Jose (2003) identified that the accuracy of event detection can be affected by poor discrimination, between crowd cheering and other sounds, however, this algorithm also suffered from the same problem.

For both Baillie & Jose (2003) and Xiong et al. (2003*a*), an in depth investigation into the application of the statistical models applied, was minimal. A particular shortcoming of the models applied by Xiong et al. (2003*a*), where that each model shared similar parameters even when the content differed in structure. As part of this thesis, this issue is investigated in more detail (Chapter 8).

Both algorithms were similar to event detection algorithms by Chang et al. (1996) for American football and Nepal et al. (2001) for basketball, although more complex audio feature sets were employed. Instead of applying heuristic rules, both Baillie & Jose (2003) and Xiong et al. (2003*b*) applied statistical representations of known crowd cheering to recognise key event locations. There has been no direct comparison between ad hoc and statistical modelling approaches, for audio-based event detection. However, for other video indexing problems, such as news story segmentation, statistical models have been found to be more robust to the varying nature of video (see Appendix C for more details).

2.2.9 Discussion

Research into sports video indexing in general, has concentrated on a small number of problems, such as the detection of sports video, sub-genre detection[3], and event detection. Genre detection is advantageous for labelling video, while event detection is beneficial for instant summarisation and generation of automatic replays. From reviewing the current state of the art, there appeared to be minimal investigation into the problem of structure segmentation and classification. This process is important for mapping video content, providing an instant overview in a similar fashion to the table of contents for a book.

[3]Both these problems come under the banner of genre detection, see Appendix C for more details about this problem for general video.

For the many sports video indexing algorithms, it was apparent that domain knowledge can assist accuracy. For example, knowledge of both the rules and production values of a sport can assist the development of indexing algorithms. An illustration, is the recognition of pitch markings for detecting key events. The closer the action is to scoring areas on the playing surface, the higher the possibility of a score event occurring. However, from reviewing the related work, a limitation of current research showed that domain knowledge was under used (Ekin et al. (2003), Xie et al. (2002)). A more in depth analysis of both the production of sports video, and the rules for each sport would be advantageous for indexing development.

Audio, visual and motion features are prominently used as information sources for sports video indexing. The application of audio, especially for the problem of event detection is simplistic (Chang et al. (1996), Nepal et al. (2001)). In other domains such as news, audio has been identified as a highly descriptive information source (see Appendix C). For sport, audio is an important clue for event detection, where the crowd and commentators react excitedly to important events. Many applications apply simplistic audio feature sets in an ad hoc manner (Chang et al. (1996), Nepal et al. (2001)), instead of maximising audio as in other video domains or research fields, such as automatic speech recognition (Rabiner & Juang (1993)). Given that the application of audio has been successful for other video domains, a more in depth investigation in suitable audio feature set, is planned in Chapter 5.

Ad hoc rules and statistical models are applied for detecting key events in sport. For other video domains such as news and film, statistical models have been shown to be more effective than applying heuristics, particulary given the varying nature of video. Heuristic rules do not generalise well to these variations (Appendix C provides a more in depth review). For the problem of event detection a number of statistical models have been applied, for representing audio patterns. These frameworks include the Support Vector Machine, the Gaussian Mixture Model (both Rui et al. (2000)) and the Hidden Markov Model (Baillie & Jose (2003), Xiong et al. (2003*b*)). There has been little investigation into the suitability of such frameworks applied to event detection, analysis of the important issues applying statistical models, and a comparison of such frameworks across a large test collection.

2.3 Thesis Motivation

From reviewing the related research into sports video indexing, three problems were identified: a lack of domain understanding, structure segmentation and classification, and event detection. These problems are the motivation behind this thesis and are discussed in the following sections.

2.3.1 Domain Knowledge

The utilisation of domain knowledge is important for domain specific indexing algorithms. However, current research into sports video indexing did not maximise the advantages of domain knowledge. There is a current limitation and in-depth investigation into specific sports domains, resulting in a lack of understanding for domains such as football video. This has resulted in the generation of sub-optimal algorithms (Ekin et al. (2003), Yow et al. (1995), Xie et al. (2002), Xiong et al. (2003*b*)). Effective indexing tools will only be generated once there is an in-depth understanding about both the rules and the production of sporting events. Clues gathered from both the rules and video production are important for automatic annotation. News video is one particular example, where an understanding of this domain has assisted the development of effective story segmentation algorithms (Eickeler & Muller (1999), Chaisorn et al. (2003)).

2.3.1.1 Proposed Solution

To address this problem, in this thesis the content structures found in football video are investigated both manually (Chapter 3) and with data mining techniques (Chapter 7), before the development of indexing tools. A specific goal is to develop a video model for football video that can be used as a guide for algorithm development. By gathering as much domain knowledge as possible, the relationship between content structures can be modelled statistically. Hence, for new football videos, these structures can be automatically recognised.

For the problem of event detection, I also investigated what key events are required for accurate summaries for football video, in Chapter 3. This also provided an insight into the motivation behind why crowd cheering is an important indicator for event detection.

2.3.2 Structure Segmentation and Classification

At present, there has been no investigation into the problem of structural segmentation and classification for sports video. Segmentation and classification is the mapping of each semantic component found in a sports broadcast, such as player interviews, studio analysis segments and the actual sporting event. This mapping is important for a variety of reasons, such as providing an instant overview of content, video browsing and retrieval, as well as the recognition of high level content using domain knowledge. A general overview of the problem for all video domains is provided in Appendix C.

Mapping a video provides an instant overview of the content to the viewer, similar to the table of contents found in most books. This mapping of content can allow for specific segments of interest to be retrieved rather than the entire video, whilst querying content. Segmentation and classification also enables efficient browsing, where a viewer can skip to areas of interest rather than linearly searching through the video. Provided with an automatically generated table of contents, the viewer can jump to points of interest or even skip commercial breaks. Also, the automatic segmentation and classification of low level semantic structures can reduce both the time and workload for manual annotators. Given an automatic overview of the video content, the logger is provided with reference points to begin annotation (see Appendix K for details on such a system).

Current research into sports video indexing has focused on identifying patterns in the specific sport sequences only. These sequences have been manually segmented from the remaining sections of the video file, where an entire sports broadcast contains other segmentation structures, other than those, representing the actual sporting event. It would be advantageous for indexing algorithms focused solely on summarising the sport, if the segmentation of such sports sequences was automatic, in terms of time and human involvement. In other words, a preprocessing step before event detection.

The problem of segmentation and classification of content is well researched (but still an open problem) in other video domains, such as the news and film (see Appendix C for further details). A reason for the popularity of this problem for news and film, are the benefits of mapping video content automatically. Mapping video structure automatically, is also advantageous for developing tools that index higher level concepts. Classification of each semantic structure can enable domain specific indexing tools to be enhanced using prior knowledge of this content. For example, algorithms that classify camera shots (Assfalg et al. (2003), Ekin et al. (2003)) are an important step for event detection and player tracking. However, these algorithms are only effective when applied to sequences containing the actual sports sequence. Deploying the algorithm on adverts or sequences filmed inside a studio would be meaningless, and would generate false errors. Mapping the complete structure in football video files can then allow for such algorithms to be automatically run on the appropriate sequences. So automating this process is desirable rather than being dependent on human intervention through manual segmentation and labelling.

Another important reason for segmenting and classifying each content structure is that many of these semantic components, other than the Game sequences, will also contain content important information, which can be indexed. For example, before and after football game sequences, segments are filmed inside a studio to both introduce the match and provide a link between advert and game sequences. These studio components will also contain match highlights and critical analysis of important events. Both the replays and match analysis can be a rich source of information for summarisation, and extracting game or player information.

2.3.2.1 Proposed Solution

The problem of structure segmentation and classification is a challenging research problem (see Appendix C for further details). Video is a collection of many modalities such as visual, motion, textual and audio information. Recognising known content, and the transition between one semantic structure and another within these modalities, is difficult. To minimise this problem, the thesis concentrates on one modality optimising the algorithm for audio. Breaking the problem down into a number of achievable goals will help simplify the problem. Once the algorithm has been optimised for one modality, the integration of further modalities can be achieved with a higher degree of accuracy.

One reason for concentrating on audio as an information source, is that audio is a very under utilised modality for sports video indexing, in particular football video. Audio has been found to be a rich information source important for both film and news video. Changes in audio from one semantic structure to another can be modelled statistically.

Applying simple visual features for structure segmentation and classification, can also generate more problems than solutions. A simple illustration would be using colour distributions (Xie et al. (2002)). The green grass pitch that appears both during the match will also be displayed in other sequences. Figure 2.2 is an example of visually similar shots from both Game, Studio and Advert sequences.

Another reason for selecting audio to begin with, is low-level visual features such as grass pixel ratio detectors (Ekin et al. (2003), Xie et al. (2002)) or edge orientation (Assfalg et al. (2003)) will not always be portable across new football sequences. Features such as grass pixel detectors will be affected by changes in colour due to factors such as light (e.g shadows, day to night, floodlights), weather elements including rain or snow, or even deterioration of the pitch over the course of a game. It is also common for the grass pitches to be cut so that complex patterns are visible. Figure 2.2 is a simple example of a playing surface containing different grass segments. These patterns could be a problem for line detection or edge algorithms (Gong et al. (1995), Assfalg et al. (2003)).

There has been little investigation into how these factors could affect accuracy (Assfalg et al. (2003), Ekin et al. (2003), Xie et al. (2002)). For this reason, attention is focused on extracting information from the audio stream, with the intension of integrating visual information in future work. Audio provides a rich, low dimensional alternative to visual features, already proven to be effective for other video domains (Chaisorn et al. (2003)).

For this thesis, audio is investigated in depth for segmentation and classification of structure, found in football video. Suitable feature sets are first identified, for parameterising audio in football video. Audio parametrisation is a well researched problem for speech recognition, there has ,however, been little analysis of suitable representations for sports video. It is essential to review the state of the art for audio parametrisation in a variety of related research fields, identifying suitable audio feature sets (see Chapter 5). Given a candidate set of audio features, these feature sets can be evaluated

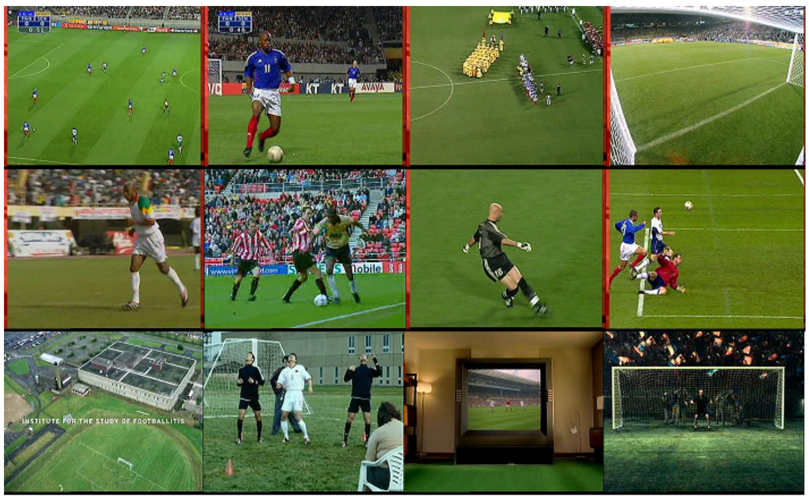


Figure 2.2: This is an illustration of the visually similar information contained in unrelated semantic units found in football video. These keyframes are taken from the same video file. Frames from the top row were taken from Game sequences, the middle row from Studio sequences, and the bottom were taken from Adverts.

formally across content found in football video sequences (Chapter 6).

Audio patterns related to known content can then be statistically modelled allowing for future video sequences to be labelled automatically. Within this thesis, statistical models that are suitable for modelling content found in the football audio are analysed. The main reason for investigating statistical frameworks, is that for related video domains, those segmentation and classification algorithms that applied statistical models performed better than the more heuristic based techniques (Appendix C). Algorithms based on heuristic rules are found not to generalise well across different video sequences. For example, rules set for one video sequence will not always be appropriate for another. A degree of flexibility is required in the decision process to account for expected variability. Statistical modelling is assumed to be a possible solution for representing this variation.

As part of this investigation, the important issues surrounding the application of such models for football video content are also addressed, such as model selection and audio segmentation. These issues are discussed in depth for the problem of structure segmentation and classification (Chapter 8).

A final limitation of indexing algorithms for sports video, is the lack of algorithm comparison across a large test collection. For news video, a large scale test collection by TREC has been introduced recently to compare story segmentation algorithms (Smeaton & Over (2003*b*)). This has provided an insight into which frameworks perform best for this domain. To address this issue for football video, a number of statistical models (and implementations), are compared across the same test collection. The accuracy of each algorithm is measured identifying the best performing strategy for segmentation and classification.

2.3.3 Event Detection

The majority of sports video indexing algorithms concentrate on detecting important events. There are many methods proposed for solving this problem, from player tracking (Seo et al. (1997)) to playing surface classification (Assfalg et al. (2003)), to crowd cheering recognition (Baillie & Jose (2003), Xiong et al. (2003*a*)). Event detection is an important step in providing automatic highlight summaries of the key moments in a football match. Automatic event detection will enable the indexing of these key moments. This is desirable for automating interactive services for digital TV such as in game highlights. Logging of key events in these highlight sequences is currently a manual process. Automating this process will enable generation of highlight summaries, minimising human involvement.

By detecting events, video that is recorded to disc can be indexed automatically allowing for more efficient content browsing, and content querying. Users interested solely on important events would be able to locate goal or scoring events. This is beneficial for a variety of applications such as home set top devices (SKY+ (2003), TiVo Inc. (2003)), and sports coaching software (Dartfish (2004)).

2.3.3.1 Proposed Solution

By first concentrating on audio as an information source, the problem of event detection for football video is addressed, within this thesis. This is similar to the problem of segmentation and classification. The event detection algorithm is optimised for audio first, before concentrating on the integration of other modalities. Again audio is believed to be a rich, low dimensional information source, which has been utilised in a variety of problem areas, such as speech recognition and audio content classification (see Chapter 5). The same reasons outlined for selecting audio first for segmentation and classification, also apply for the problem of event detection.

The audio stream of football video also contains a very important information source for event detection is human reaction to an exciting event. The crowd and the commentator react to exciting events, with the crowd cheering, and the commentator(s) becoming vocally more excitable. For other sports such as American football (Chang et al. (1996)), baseball (Rui et al. (2000)) and basketball (Nepal et al. (2001)), audio has been employed to detect events. Recently this has been applied to football video (Baillie & Jose (2003), Xiong et al. (2003*a*)). However, there has been little investigation into the various similarity in sound classes not related to commentator excitement and crowd cheering. This limitation is addressed, using the results from both the manual and automatic analysis of football content (see Chapter 10).

Important patterns for event detection are also statistically modelled. Related research for audio-based event detection such as Chang et al. (1996) and Nepal et al. (2001) rely on simple heuristic rules on simple audio features. These algorithms are not robust to other environmental sounds unrelated to crowd cheering. Applying heuristic rules have also been shown through experimentation, not to generalise well for other video domains when directly compared to systems employing statistical frameworks (Appendix C.4.3).

Recently audio-based event detection algorithms for sport (Baillie & Jose (2003), Rui et al. (2000), Xiong et al. (2003*a*)) have utilised statistical models successfully. However, these models are not necessarily implemented in a optimal way. One particular problem is how the audio track is segmented. There has also been little analysis for segmenting audio in sports sequences effectively. A number of strategies are investigated for segmenting the audio stream in difficult game sequence (Chapter 10).

It would also appear that statistical model frameworks in the sports domain, and in Video Retrieval in general, are very much applied in an ad hoc manner (Rui et al. (2000) and Xiong et al. (2003*a*)). There has been little investigation into what type of modelling framework would be appropriate for each information source, as well as any large scale comparison between modelling frameworks. There has also been minimal research into model and parameter selection. Therefore, in Chapter 10 statistical models for event detection are analysed in further detail, and in particular the implementation issues surrounding such algorithms.

2.4 Chapter Summary

A summary of the main thesis objectives are:

- to develop a robust segmentation and classification that can automatically map the content contained in football video,
- to automatically detect key events in football sequences that can enable match summarisation.

In order to achieve these aims, a number of steps were outlined:

- to generate a large test collection for football video.
- to first investigate, through manual inspection and automatic content analysis, the main structures and content found in football video. Part of this task will be to identify domain knowledge that can be utilised to assist indexing algorithms. There will be specific concentration on important information contained in the audio track of football video.
- to identify and then formally evaluate what feature set representations are suitable for parameterising football audio.
- to investigate which statistical models would represent known content identified in the audio track, for future classification of new video files.
- to identify the important issues surrounding the application of such statistical frameworks and the relevant steps that should be taken. These issues include model design, selection and application.
- to formally compare a number of algorithms over the test collection (for both problems). This will provide an insight into what techniques are the most accurate.

Chapter 3

Football Domain Knowledge

3.1 Introduction

Domain knowledge is important for the generation of effective video indexing tools, where prior knowledge of content can be utilised for indexing video sequences. The accuracy of indexing algorithms can be improved by applying clues identified from both the video production process, and the rules of the sport. An in depth understanding of the domain can also aid the development of more robust algorithms, as highlighted in related research from other genre, such as News and Film (see Appendix C for more details).

The following sections report a manual investigation into the sub-domain of football video. A brief guide to the rules of football are first described, in section 3.2, presenting an insight into the sport. A limitation in the field of sports video indexing is the availability a standardised test collection for comparing algorithms. At present such a collection does not exist. As a result of this limitation, a large collection of football video was generated for algorithm development, allowing for algorithm evaluation. This test collection is described in further detail in section 3.3. The test collection is then investigated, recording the main structures and production clues found in football video. The findings from this study, are outlined in this section. Provided with the overall findings of the study into football video, a topology model of a typical video sequence is then defined, in section 3.4. Particular attention is given to which highlights are required for a complete summary of a football match, in section 3.5. This

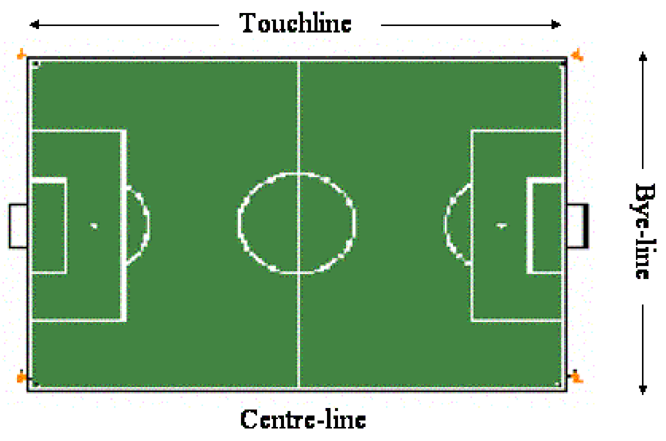


Figure 3.1: The markings on a football pitch.

served as a guide for developing an effective event detection algorithm. The chapter is summarised in section 3.6

3.2 The Rules of Football

The rules and regulations for a football game have a major influence on how a live match is broadcasted (FIFA (2003)). These rules also provide useful clues that can assist automatic annotation and indexing algorithms. For example, Figure 3.1 is a typical example of a football pitch where it can be seen that the rectangular grass pitch contains distinct markings. A common visual indexing tool is to detect these markings for recognising player or camera location, an important indicator to the whereabouts of the action, at any one time, during a football sequence (Assfalg et al. (2003), Gong et al. (1995), Ekin et al. (2003)).

These markings include the centre line and centre spot, that are surrounded by a circle. In either half of the pitch there is an 18 yard box, also called the penalty box. Inside the penalty box is a 6 yard box, originally used as a guideline for the goalkeeper during penalty kicks, or for taking a goal kick. A penalty is taken from the penalty spot, 12 yards from the centre of the goalposts. The goal posts are found on the ‘touch line’ in the centre of the 6 yard box. The height and distance of the goalposts are again specific.

For a football match, the 2 competing teams are each made up of 11 players, 1 goalkeeper and 10 outfield players. The goalkeeper wears a different colour uniform from the outfield players in his or her team. The uniforms of both teams are not allowed to be similar; hence a team has both a home and away uniform. The visiting team normally wears their away uniform if there is a clash of colours. Detecting players and uniforms is an important and very much active research area (Yow et al. (1995), Seo et al. (1997), Assfalg et al. (2003) Ekin et al. (2003)).

A game is umpired by a referee and two assistant linesmen who also wear distinctive uniforms. Traditionally the uniform is all black but recently uniforms include bright yellow or green, dependent on the opposing team's colours. The referee has the authority to caution (yellow card) or dismiss (red card) players. A player is allowed one caution during the game, a second would result in a dismissal from the rest of the match. A caution can be a result of a foul such as an illegal tackle, a handball, etc. Detection of the referee is an important method of identifying cautions (Ekin et al. (2003)).

A foul outside the penalty box results in a freekick for the opposition. The closer the foul is to the penalty box, the better the scoring opportunity. A foul inside the penalty box results in a penalty kick, a free kick at goal from the penalty spot. If the ball goes out of play from the bye-line, the result is either a corner kick for the attacking team or a bye-kick. A bye-kick is when the defending team have a free kick from their six yard box. A corner kick is taken at the corner of the bye-line and touchline. If the ball crosses the touchline, a 'throw in' is taken by a member of the opposing team. Freekicks are an important event in football matches and classifying specific freekick types is an active research area (Assfalg et al. (2003)).

The aim of a football game is to score goals, and the winning team is obviously the team that scores the most goals. That is when the ball crosses the bye-line (see Figure 3.1), in between the goal posts (often called the goal-line). If the score is level after 90 minutes, the game is drawn, unless the match is during a 'cup' competition. The duration of a match is approximately two 45 minute halves, give or take a few minutes, added on for stoppages. Normally, for 'cup' matches, the game proceeds into a further two 15 minute halves, known as 'extra time' in the case of a draw. If the game is still tied after extra time, the game proceeds to a penalty shoot out, where each team is allowed 5 penalties each, to decide the match. For a more concise version of the rules please refer to FIFA (2003).

3.3 Football Domain

In this section, the details and findings of a small experiment are provided for manually investigating a collection of football videos. As of yet, there are no standardised test collections for sports video, the only existing large scale test collection is for the news video domain (Smeaton & Over (2003*b*)). Therefore, the first important step for evaluating assumptions, and gathering domain knowledge, is to generate a large collection of football videos. This collection is manually inspected, noting the common structures and production clues that could be used to solve the two indexing problems this thesis addresses: structure segmentation and classification, and event detection. Using these findings, the test collection is then annotated manually.

3.3.1 Test Collection

In order to develop and evaluate indexing tools, a test collection of over 48 games from the 2002 FIFA World Cup Football competition were captured. Each match was recorded and then digitally stored on disc using the MPEG-1 compression format (see Appendix B for details on MPEG-1). A schedular, similar to a home video recorder was used to capture the games. The files were taken from two British Digital Broadcast channels, one containing commercial breaks (ITV1) and the other not (BBC1). The overall length for each video file ranged from 2 to 3 hours in length. As a result, each video file contained programmes that were not part of the broadcasted football match. Also, the start or end of the broadcast was sometimes missed. The official match report for each match was also stored. These contained game facts and statistics taken from the official FIFA site (FIFA (2002)).

Using this test collection, the files were then manually inspected recording any useful findings. The overall aim of this task was to identify the various structures found in football video, and identify any important production clues that could be used in the development of automated indexing tools.

3.3.2 Structure of Live Football Videos

Each video file in the test collection was inspected locating any useful information, and noting similar trends that could be employed for indexing. The overall aim is to gather as much domain knowledge as possible about the test collection. Firstly discovered was that the rules that govern a football match dictate to a large extent the overall structure of a live televised football match. For example, a typical match consisted of approximately two ‘45’ minute game sections, separated by a ‘15’ minute studio unit. These studio units were further broken up by commercial breaks, dependent on schedule constraints or the importance of the match. However, for the matches recorded on the BBC channel(s) there were no advertisement breaks. The game units were usually sandwiched between two further studio segments on either side, to introduce and then wrap up the football match.

In the various semantic units of a broadcast, the audio and visual environment differed in terms of production, sound quality, background noise, location and other peculiarities including music and sound effects. An example would be the difference in sound quality of an interview held in a silent studio environment, compared to an interview recorded inside a noisy football stadium. These clues in production were an important guide for developing indexing algorithms. The main findings are now presented.

3.3.2.1 Game Sequences

The Game sequences contained the actual match. Such sequences were a mixture of both speech and vocal crowd reactions, alongside other environmental sounds including whistles, drums, clapping, etc. Microphones are strategically placed at pitch level to recreate this stadium atmosphere for the viewing public. The stadium atmosphere is then mixed with the commentary track to provide an enriched depiction of the action unfolding. A typical Game sequence contains one or two commentators, usually based in the press box inside the stadium. One speaker leads the commentary, describing the game, while the other summarises. Each commentator is assigned a band-limited dynamic microphone (see Appendix A for an explanation on dynamic microphones), specially designed to limit interference from noise and crowd sounds, producing the best possible audio quality. These microphones are designed to limit the effect of environmental noise within the stadium interfering with capturing the commentary. Special dynamic microphones also record the stadium atmosphere.

Typically, Game sequences contain many exciting events that the crowd inside the stadium react to by cheering and clapping. Game sequences contain other audio content such as music played inside the stadium. For international matches, the national anthem of each team is normally played at the beginning of the match. Music is also played as part of the goal celebrations for many games. Sections of the crowd may also play musical instruments, in particular drums, which are used to generate an atmosphere inside the stadium.

3.3.2.2 Studio Sequences

The main function of a Studio sequence for a football broadcast is to introduce and wrap up the match, as well as host an organised discussion and analysis. These units are normally set inside a controlled soundproof environment to minimise unnecessary noise that can affect sound quality. For example, each speaker is assigned an isolated condenser microphone (see Appendix A), tested before the show for 'optimal' mixing levels. Each Studio sequence has a host (or anchor-person) who introduces the programme and chairs discussions with expert panellists. These discussions can contain game analysis or other debates, about various aspects of the match.

The order of each Studio unit dictates content as well. For example, the first Studio segment introduces the team lineups and any other important information with respect to the current match. Studio sequences that follow a Game sequence, in particular the halftime and fulltime Studio sequences also contain highlights and analysis of the game. This match analysis involves the examination of highlight replays plus debate on all major incidents.

Studio units also contain many sub-structures such as musical segues, post match interviews and outside reports. These components could be considered independent of the actual studio sequences, particularly those sequences filmed outside the confines of the studio. These, are briefly outlined in following sections.

3.3.2.3 Interview Sequences

There are many types of interview found in a football broadcast. These include an impromptu ‘pitch side’ interview with a player or coach, a pre game press conference with a coach, or a player interview staged in a controlled environment such as a studio set. These interviews can be recognised by changes in sound quality and other acoustic characteristics. For example, a ‘pitch side’ interview consisted of an interviewer and interviewee sharing a microphone. In order to limit the background noise the microphone was usually band-limited. As a result of the interviews being held in busy stadiums, the sound quality would normally be low, containing noise and crowd sounds.

A press conference interview is again not set in a completely controlled environment. The recording microphone would normally be placed as close to the interviewee as possible to reduce the effect of environmental sounds. However, as press conferences contain many journalists, broadcasters and photographers contesting for attention and asking questions, the resulting sequence would often contain a lot of background environmental noise, such as hushed speech, the sound of cameras clicking, etc.

Studio or prearranged interviews on the other hand contained minimal background noise. These interviews appeared to be planned and normally set in a controlled environment. Each speaker would be given an isolated microphone, where the optimal sound levels for each microphone were probably checked. These steps are taken to minimise unwanted noise if the interview is pre-recorded. Any eventual unwanted interference could then be smoothed out using postproduction techniques that can enhance the sound quality.

These three main types of interview were grouped into three broad categories; ‘Stadium’, ‘Press Conference’ and ‘Studio’ based from reviewing the test collection.

3.3.2.4 Report Sequences

Report sequences were used to provide details on teams, provide player profiles, or present current sporting news. These reports were similar to news reports, where one main reporter would provide a running commentary throughout the piece. These reports were also interspersed with interviews. Reports were normally pre-recorded;

therefore the sound quality was enhanced. Another feature of reports were that they often included music or other sound effects to add to the appeal.

3.3.2.5 Musical segues

A musical piece usually accompanied the start and end to a show. These musical pieces are also known as the opening and closing titles. Music can also be an entertaining device directors use to grab the audience's attention. For example, segues that replay previous game or player highlights were set against a musical soundtrack for greater impact. These 'Musical segues' sometimes contain edited commentary and crowd atmosphere to add to the piece.

3.3.2.6 Advert Sequences

A '45' minute half in a football match is continuous; hence an advertisement break can only occur before and after a Game segment and during the half time interval. An Advert break can be identified by the almost chaotic mixture of highly produced music, voice and other sound effects that grab the consumer's attention. A key identifier of Advert sequences were a series of black frames accompanied by complete silence that indicated the beginning and end of an advert[1].

3.4 Football Video Model

From the in depth study of the test collection, it became apparent that the production of live football provided a number of clues for automatic video indexing. In particular, the audio track itself contained a wealth of information that could be utilised. For example, the transitions from one semantic unit to another often resulted in a change in recording quality, and audio environment, with the method for capturing speech varying between content structures such as Game and Studio. Also, the occurrence of specific background or environmental sound would be beneficial for distinguishing

[1]Note: From experimentation in Chapter 7, the silence found separating commercial breaks differed from the silence or pauses between speech in the Studio segments. These pauses still contained low levels of noise while the silence between commercial breaks did not.

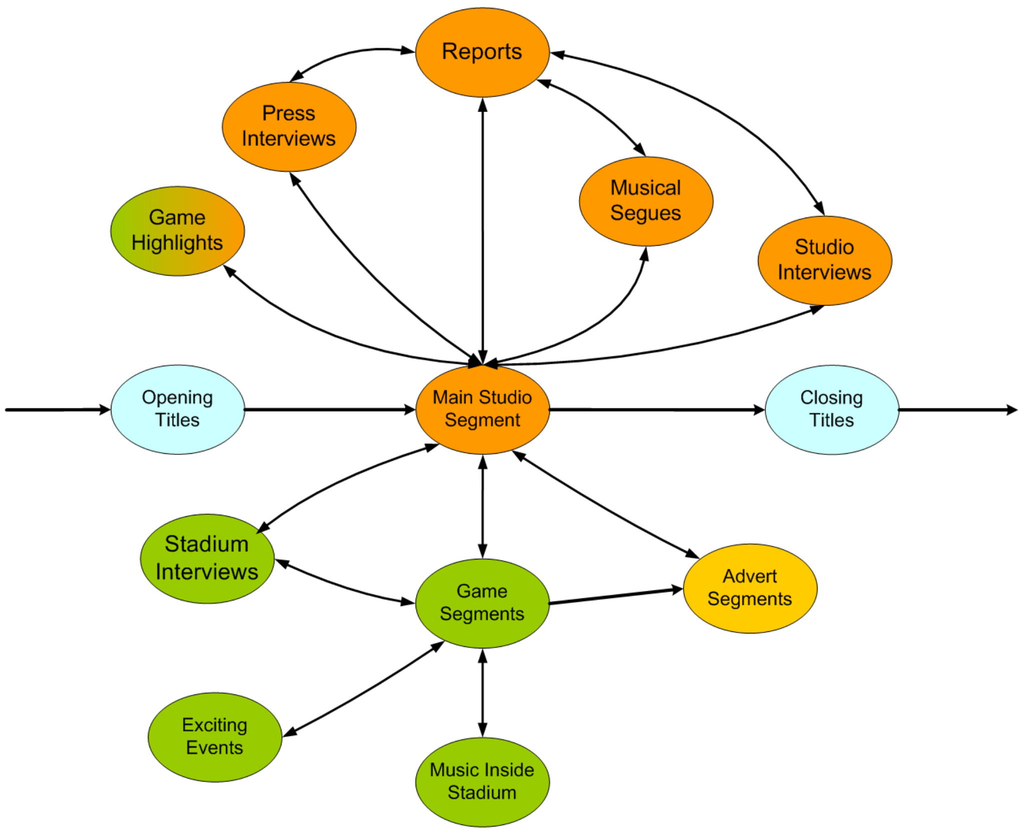


Figure 3.2: A video model for a live football broadcast displaying the temporal flow through the video, and the relationship between known content structures.

between content structures. It was believed that the differences between each identified unit could be modelled for the purpose of automatic segmentation and classification of structure.

To assist automatic structure segmentation and classification, a model containing the typical structures found in football video file was defined (see Figure 3.2). The model encapsulating the temporal flow between each major unit was a summary of the main components identified from the test collection investigation. The video model contained three main structures: the Game (the green units), Studio (the orange units) and Advert sequences (yellow). Within these sequences were a number of different sub-structures, such as Reports, Interviews and Events.

A typical football programme would begin with an opening title sequence, before moving onto the main studio section, where, the anchorperson would introduce the match. From the Studio section, the video would alternate between a number of segments, such as Reports, Interviews or proceeding directly to Game sequences. Once in the Game sequence, the match would usually commence and finish after 45 minutes. However, for extra time sequences this would be reduced to 15-30 minutes, dependent on the actual match. From Game sequences, the video file would then switch to Adverts or back to the Studio segment. This entire process would be repeated until the final closing titles, which signify the end of a broadcast.

As part of the Studio sequence, match highlights from other games were also shown. This was the reasoning behind defining a Game Highlights sequence (and why these sequences are partially coloured both green and orange). These sequences are not related to the current game, but display a previously played match(es). The audio track from such sequences would often differ from the actual Game sequences.

The temporal flow of the video model was selected over other representations, such as a linear or left to right model. The reasoning behind this, was that it was assumed that the variation found within football video could be modelled more efficiently. This representation was believed to encapsulate the temporal flow between the important structures, and also model a range of possibilities or variation. For example, some matches contain extra time (further Game sequences) if the match is drawn. Also, in the test collection, some video files did not contain Advert sequences that were recorded from the BBC TV channel.

In short, the video model represented the main structures found in football video as well as representing the variation and temporal flow between these structures. The model also accommodated the possible variations found across the domain. Provided with this model, a structure segmentation and classification indexing algorithm was developed.

3.5 Summarising Football Video

In this section, a number of reference points are introduced that can be used for event detection in football video. These reference points were identified during the observation of the test collection. It was noted that football differs from other sports that have been researched in video retrieval, such as American football, tennis, baseball and basketball. In these sports, there is often a quick play, such as a point in tennis, a play in American football, a pitch in baseball or an attack in basketball. These plays are clues for locating the start and end point of each logical unit. For example, after each play the game either stops in the case of American football, tennis and baseball, or in basketball matches the defending team then attacks the opposing basket. Identifying the transition from one play to another allows for the entire game to be mapped. These plays can be used as a reference point for mapping each attack, serve or pitch, by locating the start and end point of every action sequence, thus providing an overview of the content to the viewer.

Football does not contain such immediate reference points. One reason for this is that football is not a high scoring or "end to end" sport, especially in comparison to other comparable sports such as basketball. It is very common for close games to end with no scoring. It is also characteristic in tight or competitive games, for the two opposing teams to battle for possession, of the ball, in the middle of the pitch for long periods. In these cases the distinction between attack and defence is difficult. This does not suggest that such a match does not comprise of exciting events. As a result, is important to recognise other reference points that can be utilised for football video summarisation.

3.5.1 Example of a Football Summary

A popular journalist and writer once described what events would occur in his perfect football match; "Seven Goals and a Punch-up - Arsenal v Norwich, 4.11.89" (Hornby (2000)). The author stated that "for a match to be really, truly memorable, the kind of a game that sends you home buzzing inside with the fulfilment of it all, you require as many of the following features as possible:". Events include plenty of goals, poor refereeing decisions, a missed penalty, a red card and a "disgraceful" incident such as a brawl. To illustrate, he reflected on the atmosphere of a previous game involving his favourite team. His team, Arsenal, won 4-3 but only after a last gasp fight back that resulted in an ordering off of an opposition player. His team was typically involved in a mass brawl as well, which added to the drama.

If he had not attended the match, he would have been interested in watching the highlights. These highlights would include all the major incidents that took place. The important events would include the goals, but also the key moments that affected the outcome of the match. In this specific case, the mass brawl and the dismissal of an opposing player. For a perfect video summary, the highlights would include all these incidents. This memory is an individual example of what events a supporter would use to summarise a game. Highlights include not only goals but also key incidents that determined the outcome. This can be an attractive passing movement, an attempt at goal or a poor refereeing decision.

A typical football match contains key events that determine the result, alongside the other possible talking points that are of interest to expert panellists, journalists and fans. An appropriate automated match summary should include all of these important events, not just the goals.

3.5.2 Reference Points for Event Detection

From inspecting the test collection, one possible reference point for the detection of a significant event is the reaction of the crowd within the stadium. For example, the supporters of a scoring team celebrate a goal by cheering, singing and clapping. However, the reaction to a ‘dismissal’ can vary from a chorus of cheering or booing, depending on the team each set of fans follow. In contrast, a round of applause often occurs to acknowledge an exciting skill, possibly from both sets of supporters.

Another recognisable reference point was found in the commentary track. During an exciting event, the emotion of the commentator(s) voice can be an important indicator. For example, during a goal, it is typical for the commentators voice to become excited, raising the pitch of the voice. This was another indicator for the occurrence of a key event.

When designing automatic indexing tools for football such as an event detection algorithm, it would be important to utilise information sources such as crowd cheering and commentator excitement. Both of these reference points are correlated to key events.

3.6 Chapter Summary

The aim of this chapter was to investigate the domain of football video, recording particular domain knowledge that can be utilised for the problems of segmentation and classification of structure, and event detection. First, rules of the sport were discussed. A test collection was then generated and manually inspected, providing a detailed description of the production of football video. Clues in the soundtrack were also recorded for future algorithm development. The final outcome of the chapter was a video model describing the main content found in football video (see Figure 3.2). Those key events that are required for accurate match summaries were also noted. Examining the test collection provided two possible reference points for the identification of such events.

Chapter 4

Modelling Audio Data

4.1 Introduction

In this chapter, two statistical frameworks suitable for modelling audio data generated from football videos are outlined: the Gaussian Mixture Model (GMM) and the Hidden Markov Model (HMM). From applying these models to efficiently represent specific content found in football audio sequences, the same structures can be automatically detected in future sequences. Identifying appropriate statistical model frameworks is an important step for the two indexing problems: structure segmentation and classification, and event detection.

Related research into statistical modelling for audio content is reviewed, before introducing the two modelling frameworks, which have been widely applied in audio content classification and speech recognition (GMM and HMM). These two modelling frameworks dominate current audio classification and speech recognition research, for a number of reasons. Both have been applied to audio classification problems accurately, where speech (and other sound classes) are modelled by multivariate Gaussian probability density functions, even if the underlying data structure is non-Gaussian. Investigation into the underlying data processes is required for efficient model representation, particularly for non-speech sound. Therefore, the surrounding application issues of both the GMM and HMM are discussed, including model design, model selection and classification.

The remainder of the chapter is structured as follows. In section 4.2, I will introduce the motivation behind the selection of both the GMM and HMM. The GMM and HMM are discussed in sections 4.3 and 4.4 respectively. The important issues that have to be considered when applying both frameworks will subsequently be examined, finally summarising the chapter in section 4.5.

4.2 Background

There have been numerous strategies for modelling audio content. These include:

- Template matching and Dynamic Time Warping (DTW),
- k-Nearest Neighbours (kNN),
- the Support Vector Machine (SVM),
- the Gaussian Mixture Model (GMM),
- the Hidden Markov Model (HMM).

Template matching has largely been superseded in terms of performance by algorithms such as the kNN, SVM, GMM and HMM for audio classification (Theodoridis & Koutroumbas (1999), Rabiner & Juang (1993)). One of the main difficulties in automatically recognising broad sound classes, such as those found in football video (as described in section 3.3), is that many sub-structures are contained within each class. Although different video sequences, from the same content class, will include the same sub-structures, and often found in the same order, the precise timing and duration of each sub-unit will not necessarily match. As a result, efforts to recognise broad content by matching them to templates, will provide inaccurate results if there is no temporal alignment.

A Dynamic Programming technique called Dynamic Time Warping (DTW) can accommodate differences in timing between new sequences, and the corresponding templates (Theodoridis & Koutroumbas (1999)). This algorithm is used to determine how similar new sequences match a predefined template. DTW is applied to select the best-matching template, when given a set of possible templates. Again this technique, which has been used for word recognition (Chang et al. (1996)) and baseball hit detection (Rui et al. (2000)), has been superseded in terms of accuracy and performance by other pattern recognition tools for audio indexing (Theodoridis & Koutroumbas (1999), Rabiner & Juang (1993)).

The kNN algorithm can be thought of as a non-parametric equivalent of the GMM, often used for the classification of sound (Duda et al. (2000)). GMM is a parametric mixture model that is used as a statistical representation of a content class, where new data are grouped into the class that most likely generated the sequence. The kNN on the other hand, determines the similarity of new audio sequences with estimates from a labelled training set, with the distribution of each class not pre-supposed by a statistical model. Instead, a new audio sequence is labelled into the class with the ‘k’ closest examples in the training sample. The advantage of the kNN is that no prior knowledge is required about the data distribution for each class. During a training phase, the regions and boundaries in the feature space, between each sound class, are learned. New data sequences are then labelled as belonging to the closest class in the feature space to that sequence.

There have been numerous papers comparing GMM and kNN for problems such as speech/music recognition (Scheirer & Slaney (1997)), speech/non-speech recognition (Rui et al. (2000)) and music genre recognition (Tzanetakis & Cook (2002)). A disadvantage with kNN over GMM, is the complexity of classifying new sequences especially given high dimensional data sequences (Theodoridis & Koutroumbas (1999)). Also, performance wise the GMM has been proven to be as effective for audio classification, if not more so (Tzanetakis & Cook (2002)).

Recently, the non-parametric Support Vector Machine (SVM) has been applied to general audio classification problems, such as excited speech detection (Rui et al. (2000)), musical instrument recognition (Marques & Moreno (1999)) and sound-effect classification (Guo & Li (2003)). The SVM defines a hyperplane in the feature space that separates two data classes (Burges (1998)), estimated during a training phase with labelled data samples. The SVM has been shown to outperform both the GMM and kNN classifiers (Rui et al. (2000), Marques & Moreno (1999), Guo & Li (2003)). The SVM is essentially a binary classifier and requires a complex decision process for multi-class problems. To address this problem, Guo & Li (2003) implemented a binary tree decision process for multi-class audio classification. Marques & Moreno (1999) also discriminated between classes, by examining one classifier against all remaining classifiers (one versus all), or by comparing all pairwise combination of classifiers for each new sample.

The SVM is a promising application for audio classification, however, research into the accuracy of this framework is still at an early phase. At present, there has been no large-scale comparison between SVM and more established acoustic modelling frameworks, such as the HMM. For the purpose of this thesis, I focused upon the more established models for this reason, with the GMM and HMM being investigated instead of the SVM.

A reason for the popularity of HMM (and GMM), is that audio content such as speech can be efficiently represented by multivariate Gaussian distributions (Rabiner & Juang (1993)). Complex sound classes can be represented by a mixture of simpler components, even when the underlying data assumption is highly non-Gaussian. Research from a number of related fields have demonstrated that audio content, when parameterised by perceptual feature sets, such as the Cepstral coefficients (CC), the Mel-frequency Cepstral coefficients (MFCC), and the Linear Prediction Cepstral coefficients (LPCC), are well modelled by multivariate Gaussian distributions (Reyes-Gomez & Ellis (2003), Roach & Mason (2001), Adams et al. (2003)).

The HMM models the time varying nature of audio data, which is another assumed advantage over other statistical frameworks, such as GMM and SVM. In comparison, both these frameworks require the audio signal to be pre-segmented first, in order to account for time varying nature of audio. To manage temporal data, the audio signal is divided into equal sized units, which are then classified. The HMM can be applied in this manner, and also as a segmentation tool itself, using a Dynamic Programming algorithm called Viterbi to both segment and classify audio sequences in one pass (Rabiner & Juang (1993)).

However, for accurate segmentation and classification, when applying statistical models, the number of content classes must be known. This is an important reason why both domain knowledge, through manual inspection (as described in section 3.3) and automatic content analysis (introduced later in Chapter 7), are vital for accurate segmentation and classification. With detailed domain knowledge, both the GMM and HMM can be applied within a segmentation and classification framework for complex multi-class problems with a high degree of accuracy (as illustrated in Chapter 9), including audio classification (Spina (2000)), and automatic speech recognition (ASR) (Rabiner & Juang (1993)). In fact, for the majority of ASR systems, the HMM is the most widely applied modelling framework (Ney & Ortmanns (2000)). GMM and HMM have also been shown to be effective for classifying speech and non-speech audio across a variety of problems. These include the recognition of broad audio classes, such as video genre [Roach & Mason (2001), Wang et al. (2000), Huang & Wang (2000)], sound-effects [Cowling & Sitte (2003), Reyes-Gomez & Ellis (2003)], general audio, such as speech and music [Li et al. (2001), Scheirer & Slaney (1997)], excited commentator speech (Rui et al. (2000)), crowd cheering (Xiong et al. (2003*a*)), and even space rockets blasting off (Adams et al. (2003)).

The main reasons for concentrating on both GMM and HMM in this thesis are:

- Complex audio content can be well modelled by a mixture of multivariate Gaussian probability density functions (Rabiner & Juang (1993)),
- GMM and HMM are among the most statistically mature methods for multi-class audio classification problems, in comparison to the SVM,
- Both GMM and HMM have indicated good performance and shown to be robust when applied to the classification of broad audio content,
- HMM is believed to be the current state of the art for ASR (Ney & Ortmanns (2000)).

In the remainder of this chapter, GMM and HMM will be discussed in further detail.

4.3 Gaussian Mixture Model

In following sections, the basic theory of the GMM is introduced, describing both the model generation and training steps, and how GMM can be applied for classification.

4.3.1 GMM Formulation

The GMM is formally denoted as follows. Given a set of independently and identically distributed feature data vectors $X = \{x_1, \ldots, x_N\}$ for $x \in \mathbb{R}^d$, where d is the dimensionality of the feature space. A single multivariate Gaussian PDF can be defined as,

$$\mathcal{N}(X;\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left[\frac{1}{2}(X-\mu)^T\Sigma^{-1}(X-\mu)\right] \tag{4.1}$$

where μ is the mean vector and Σ is the covariance matrix. A GMM contains two or more Gaussian PDF components to represent one data class e.g. a GMM is a mixture of several multivariate Gaussian distributions. A mixture of PDFs can represent the many sub-units in one content class more efficiently than a single PDF, for multi-modal data. For example, both speech and non-speech periods in a Studio sequence (see Chapter 3.3). The GMM of a data vector, X, is defined as the weighted sum of Gaussian PDFs, defined as:

$$p(X;\theta) = \sum_{c=1}^{C} \alpha_c \, \mathcal{N}(X;\mu_c,\Sigma_c) \tag{4.2}$$

where C is the total number of mixture components in GMM, α_c is the mixture weight of the component, where

$$\sum_{c=1,\ldots,C}^{C} \alpha_c = 1, (0 < \alpha_c < 1)$$

and θ is the parameter set of the GMM,

$$\theta = \{\alpha_1, \mu_1, \Sigma_1, \ldots, \alpha_C, \mu_C, \Sigma_C\} \tag{4.3}$$

Figure 4.1 is an example of a two class data set plotted in a two-dimensional feature space. Both data classes are also displayed separately in the top plots in Figures 4.2 and 4.3. Analysing both figures, it can be seen that both data classes would not be efficiently represented by a single Gaussian PDF. In fact, both classes seem to contain at least two sub-structures. To argue this point, it can be shown that class 2, (the red +) contains a separation in the feature space between the two sub-structures, in that

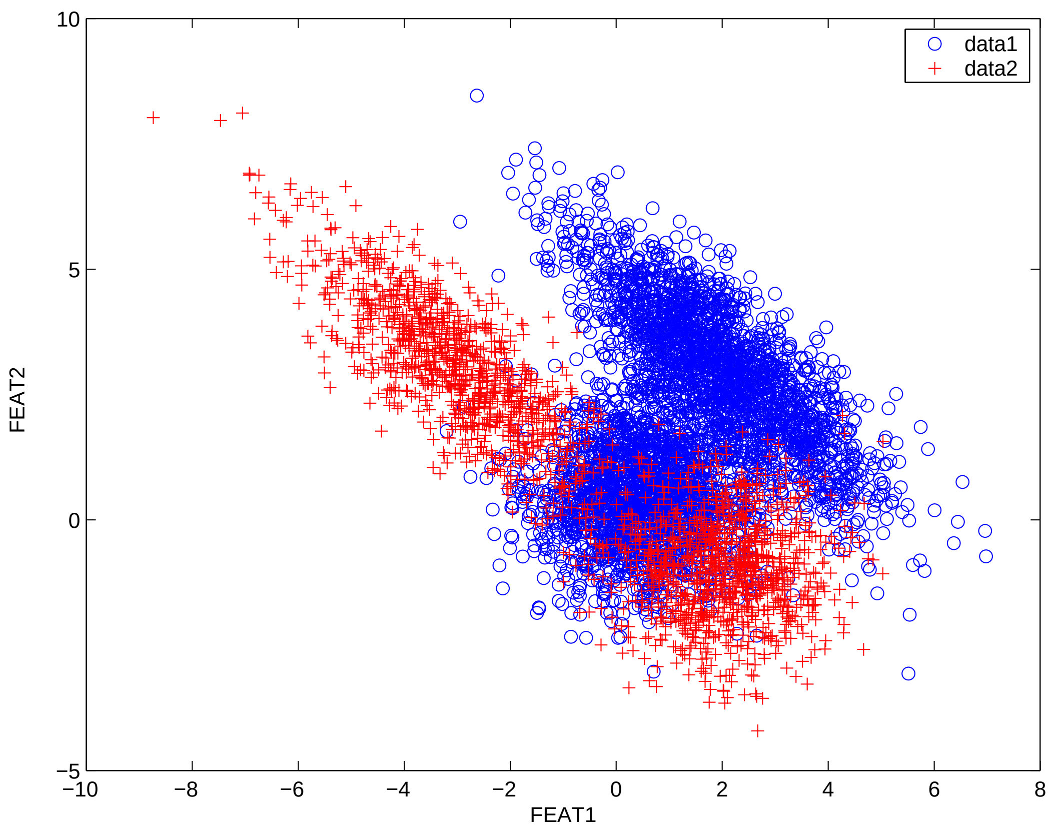


Figure 4.1: Example of a two class data set. The data points belonging to class 1 are coloured blue, and data corresponding to class 2, are red.

class (Figure 4.1). This can be shown more clearly in Figure 4.3. It can be argued that a single multivariate Gaussian PDF would not be a good representation due to the variation found within this class.

Therefore, a 2 mixture GMM was generated for each data class, illustrated in the bottom plots in Figures 4.2 and 4.3. A mixture of Gaussian PDFs can effectively model these non-Gaussian distributions, where each ellipse is one mixture component, with mean (μ_c) and covariance (Σ_c). A training phase is required to estimate these parameters for each GMM. This process is described in the following section, focusing upon how the mode number is estimated (how many mixture components).

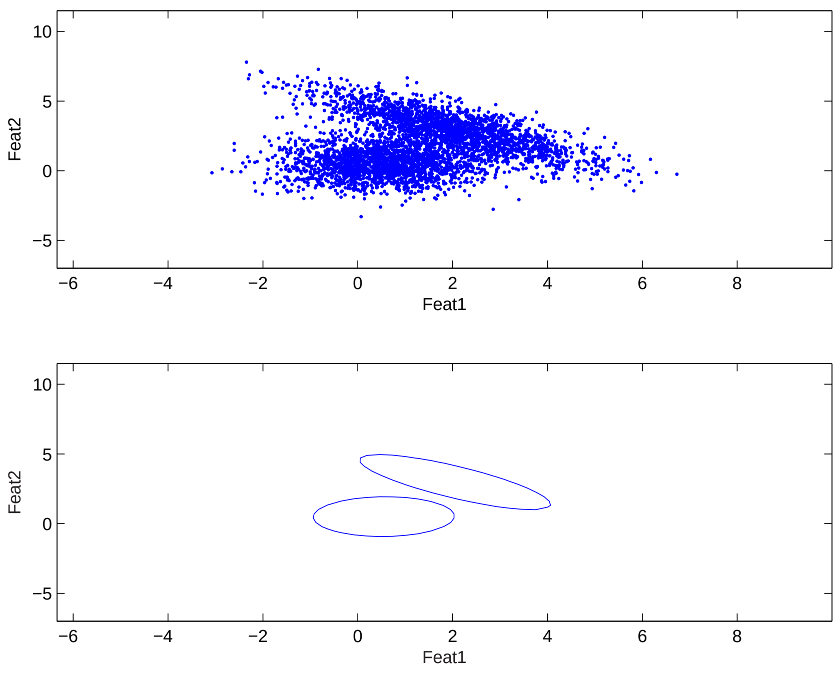


Figure 4.2: An example of a two mixture GMM representation for data class 1. The top plot is the data plotted in the two dimensional feature space. The bottom plot is the GMM fitted for class 1. Each mixture is an ellipsoid in the feature space with a mean vector and covariance matrix.

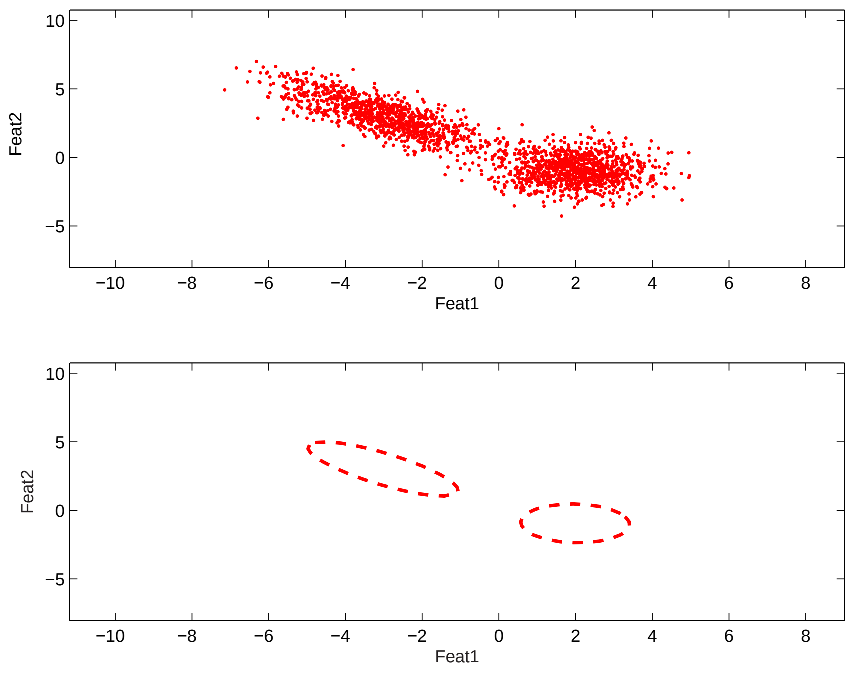


Figure 4.3: An example of a two mixture GMM representation for data class 2.

4.3.2 Training and Parameter Estimation

The process of estimating the number of mixtures and parameter estimation are now discussed.

4.3.2.1 Mixture Estimation

A particular problem with applying GMM is identifying the correct number of modes. A typical solution for video or audio indexing, is to apply a fixed number of modes for each data class. For example, Marques & Moreno (1999) classified music clips into instrument categories using GMM classifiers, where each instrument class was represented by a GMM with the same number of fixed mixture components. This procedure is also followed for other research problems, such as video genre detection (Roach & Mason (2001)), speech/music classification (Rui et al. (2000), Scheirer & Slaney (1997)), and music genre detection (Tzanetakis & Cook (2002)).

However, forcing all GMM models to share the same number of modes can generate unforeseen problems, especially when modelling very broad content classes, such as video genre. It would be unreasonable to expect that each class will be efficiently represented by the same number of mixture components. In fact, this approach, although easy to implement, ignores the differences found in content found across these broad content classes. A fixed number of modes for one class will not necessarily generalise to all content classes. For example, estimating an ‘optimal’ fixed number of modes that all classes share is difficult, especially when a large number of classes contained in the data.

There are a variety of methods for estimating the number of mixtures for each content class. Examples include the exhaustive search, a brute force method of finding the optimal GMM over an exhaustive combination of parameters. Exhaustive search and cross validation involve implementing a range of models with a varying number of modes. This pool of GMMs is then evaluated using a criterion, such as predictive likelihood, to determine the optimal model settings.

Alternative methods include top down or bottom up approaches (Baggenstoss (2002)). For the top down method, a large number of modes are placed on the training data, with weak modes being merged or removed until the GMM converges. To determine a weak mode, the mixing weight for each mixture component α_c is compared against a threshold, after each training iteration. If α_c falls below this threshold, the mode is pruned or merged. This process is repeated until the GMM converges, again based on a constraint such as likelihood, or the number of iterations.

In this thesis, unless stated otherwise, a bottom up method is applied. The bottom up approach initialises the GMM with one mode, which is then split. This process is repeated until the GMM converges, based on a pre-specified criterion. Splitting each mode is determined by calculating the weighted kurtosis for each mixture, a method proposed by Vlassis et al. (1999)[1].

It was believed that this method could be more efficient, in terms of time, computation and complexity, than cross validation or a brute force approach, without effecting classification performance. This assumption was based on both the number of GMM models that are required to be implemented, and tested, before the optimal model is found, and also the performance of the GMM selected by this algorithm.

To summarise the main advantage with the bottom up approach, is that the number of modes can be estimated for large collections, without implementing a large pool of models. Starting with one mode was also a contributing factor, when compared to the top down approach, where an initial number of modes must be estimated. It was believed that estimation of an appropriate starting number of modes would be difficult, given the differences between each content class. Therefore, unless stated otherwise, the bottom up approach to GMM generation was implemented.

4.3.2.2 GMM Training

Parameters in the GMM (θ), such as the mean vector and covariance matrix for each mixture component, are estimated using labelled training data. For this thesis, to limit the number of parameters to be estimated in the GMM, the covariance matrices were constrained to be diagonal. Parameters are estimated using the Expectation-Maximisation (EM) algorithm (Duda et al. (2000)). First, the initial parameters esti-

[1]A Matlab toolkit was used to implement the training and parameter estimation routines for the GMM models. The toolkit contains implementations for a standard Expectation-Maximisation training, and both the top down and bottom up mode estimation strategies. The toolkit also contains training scripts for managing large quantities of data. For further details about the toolkit please refer to Baggenstoss (2002).

mates are found using k-means clustering algorithm (Duda et al. (2000)), and then the iterative EM training algorithm is applied. After each iteration, the GMM parameters are re-estimated, which is repeated until the GMM converges, based on a stopping criterion. In this thesis, the log-likelihood is used as a stopping criterion, with a record stored after each training cycle. If there was no significant increase log-likelihood over a number of steps, training is terminated (Baggenstoss (2002))[2].

4.3.3 Classification

Assuming that the total number of content classes found in a test collection is known, a set of GMMs can be generated, with each GMM being a statistical representation of one specific content class. Given this set of GMMs, new data can then be classified. Figure 4.4 is an illustration of how a new data sequence, generated by an unknown class. Previous knowledge of the data indicates that there are two content classes. For each content class a single GMM has been generated. If the new data sequence is displayed in the feature space alongside the two GMM models, the class this sequence most likely belongs to, can be guessed.

From the plot, it can be concluded informally that class 1 (the blue solid ellipses) would fit the new data sequence better than GMM for class 2 (the red dashed ellipses). Suggesting the data is most likely to belong to class 1, since class 1 is a better fit, visually, of the new sequence.

This is a graphical example of a popular method for classifying new sequences when applying a GMM framework. By calculating the likelihood of the data sequence, generated by each GMM, new data sequences are labelled into the class returning the highest model likelihood score. This is often referred to as maximum likelihood classification.

This can be calculated formally as follows. If the data set contains K content classes $\omega_k = \{\omega_1, \dots, \omega_K\}$, where each content class is represented by a single GMM, $\Theta = \{\theta_1, \dots, \theta_K\}$. Then to classify a new data sequence, $X = \{x_1, \dots, x_N\}$ for $x \in \mathbb{R}^d$, the likelihood of X being generated by each GMM is calculated.

[2]When applying the bottom up approach for GMM generation, after each iteration, modes in the GMM were also evaluated using kurtosis, and weak modes were split accordingly (Vlassis et al. (1999)).

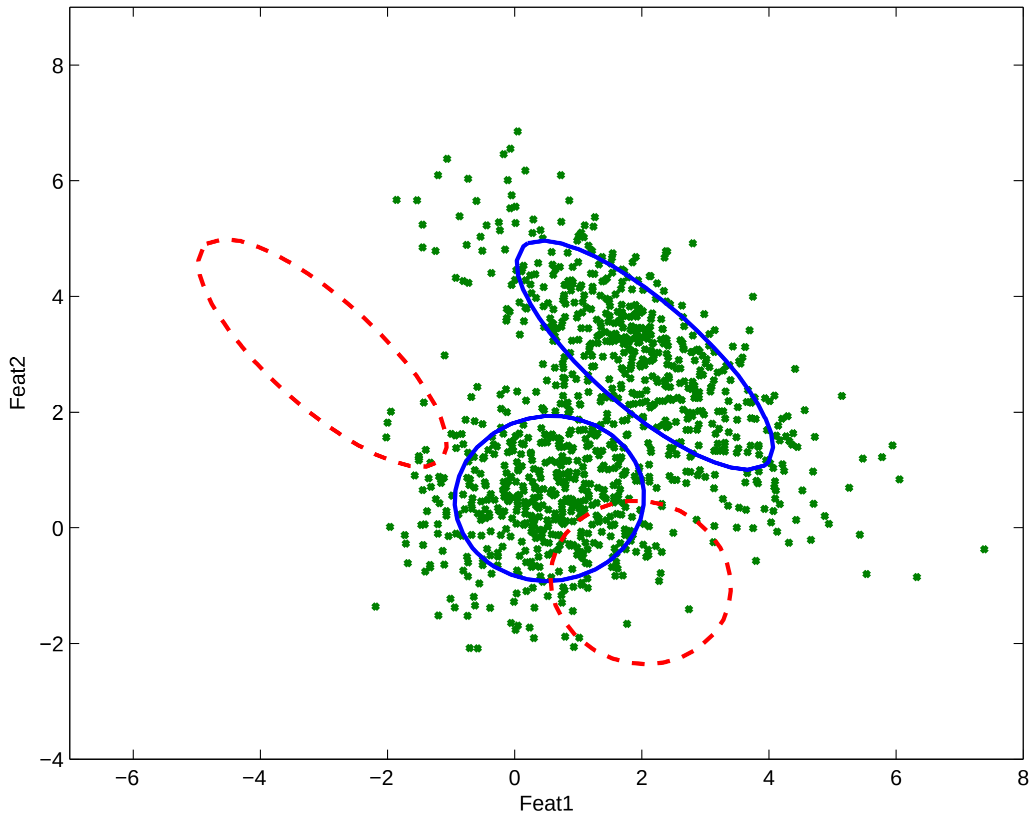


Figure 4.4: Classifying new data.

The likelihood that a set of data vectors X was generated from a content class ω_k is,

$$L(X = \{x_1, \ldots, x_N\} | \omega_k) = \prod_{i=1,\ldots,N} p(x_i | \omega_k) \tag{4.4}$$

To classify the data vector X, the maximum likelihood score is found,

$$X = \arg \max_{k=1,\ldots,K} L(X|\Theta) \tag{4.5}$$

This is a very simplistic, but effective decision process. There are, however, other methods that can be applied, such as Hypothesis testing, Log odds ratio (Theodoridis & Koutroumbas (1999)) and Dynamic Programming (Rabiner & Juang (1993)). In Chapter 9, Dynamic Programming is analysed further, for the classification of temporal data.

4.4 The Hidden Markov Model

The Hidden Markov Model (HMM) is a powerful statistical model that can closely approximate many time varying processes such as human speech. For this reason the HMM has been widely used in the field of automatic speech recognition (ASR) (Rabiner & Juang (1993)). HMM has also been applied recently to video indexing problems such as shot boundary detection (Boreczky & Wilcox (1998)), news video segmentation and classification (Eickeler & Muller (1999), Chaisorn et al. (2003)), TV genre classification (Huang & Wang (2000), Wang et al. (2000)), event detection (Xiong et al. (2003*a*)), Play/Break segmentation for soccer video (Xie et al. (2002)) and general audio classification (Adams et al. (2003), Cowling & Sitte (2003)). The HMM has also been applied to data mining and clustering, discovering the underlying structure found in temporal data (Li & Biswas (2000), Rabiner & Juang (1993), Smyth (1997)).

The HMM is able to capture the dynamic properties of audio data such as the variation of an audio signal over a period of time. Similar static representations, such as the GMM, are considered to be limited for audio data because they ignore the temporal properties of such data. This is one of the main reasons why the HMM is the current model of choice for the majority of ASR systems (Ney & Ortmanns (2000)). In this section, the Hidden Markov Model is introduced, as well as the main issues concerning the application of such a framework for audio-based segmentation and classification.

4.4.1 HMM Anatomy

The HMM is now defined following the notation used by Rabiner & Juang (1993). The fundamental assumption of a HMM is, that the process to be modelled is governed by a finite number of states, $S = \{s_1, \ldots, s_N\}$. These states change once per time step t in a random but statistically predictable way, where $1 \leqslant t \leqslant T$. The actual state at time t is denoted as q_t. The output generated by a PDF dependent on the current state q_t can be observed, where the PDF that generates o_t, under state j, can be denoted as $b_j(o_t)$.

At each time step, one observation vector o_t is emitted from the current state, where $O = \{o_t : t = 1, \ldots, T\}$ is one complete observation vector generated by a HMM, λ. It is assumed that the current state q_t in a HMM at any one point in time, is dependent only on the previous state q_{t-1}. This is called the Markovian assumption.

The basic structure of a HMM λ is

$$\lambda = (A, B, \pi) \tag{4.6}$$

where A is the state transition matrix, B is the set of emission probabilities and π is the initial state probabilities. A is the collection of probabilities that define the transition behaviour between states, as well as the topology of the HMM. For example, the transition probability for moving from state i to state j is defined as a_{ij}. States are connected by these transition probabilities:

$$a_{ij} = P(q_t = j | q_{t-1} = i), 1 \leqslant i, j \leqslant N \tag{4.7}$$

According to the Markovian property, given that state q_{t-1} is known, the conditional probability of the next state q_t is independent of the states prior to the current state. Under this assumption, a change in state will occur once per time step, in a random but predictable way. Each state has a PDF, b_j, defining the probability of generating an observation vector o_t at time step t. For this thesis, the PDF for each state is assumed to be a singular or mixture of Gaussian components, $b_j(O)$, defined as

$$b_j(O) = \sum_{k=1}^{M} c_{jk} \, \mathcal{N}(O; \mu_{jk}, \Sigma_{jk}), 1 \leqslant j \leqslant N \tag{4.8}$$

where μ_{jk} is the mean of mixture component k for the PDF of state j, and Σ_{jk} is the covariance matrix[3]. c_{jk} is the weighting term for each mixture where

$$\sum_{k=1}^{M} c_{jk} = 1, (0 \leqslant c_{jk} \leqslant 1), (1 \leqslant j \leqslant N), (1 \leqslant k \leqslant M) \tag{4.9}$$

Finally, the initial state probability vector $\pi = \{\pi_j : j = 1, \ldots, N\}$, contains the probabilities of the HMM commencing at any state, given the observation sequence O.

[3]For large data sets such as those found in large vocabulary speech recognition systems, 1000s of hidden states are required. This can cause parameter estimation problems when using a mixture architecture for each state in the HMM. Therefore state covariance matrices are often constrained in practice to be diagonal to limit the number of parameters to be estimated (Rabiner & Juang (1993)).

Figure 4.5 is an example of a fully connected two state HMM, displaying the transition probabilities between both states. The relationship between the two states are defined by the transition probabilities a_{ij}. The output of each state is defined by the PDF b, which at each time step, the current state outputs an observation vector o_t.

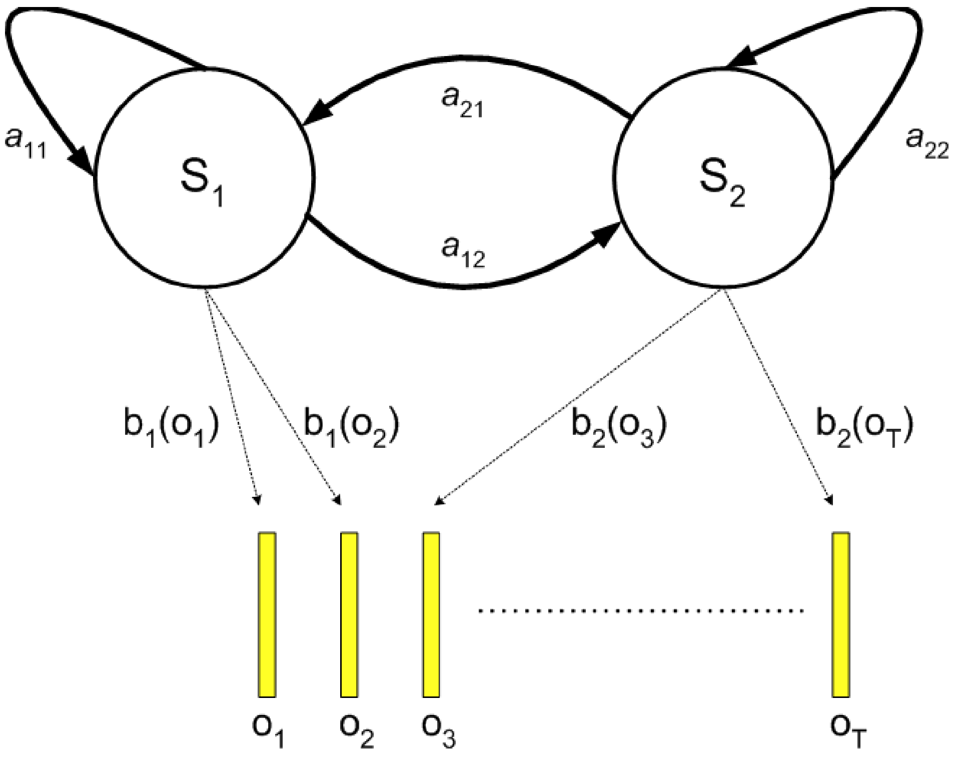


Figure 4.5: An example of an audio sequence O being generated from a simple two state HMM. At each time step, the HMM emits an observation vector o_t represented by the yellow rectangles.

4.4.2 HMM Design

The design of a HMM is an important factor in performance, where model topology and state architecture can both have an impact on the accuracy. Design issues include, the HMM behaviour (A), and whether to implement a discrete or continuous HMM ($b_j(O)$). Some of the important issues concerning HMM application are now addressed, which are believed important for robust model generation. These issues include restrictions in A, which defines the topology and behaviour of the HMM, and a suitable architecture for $b_j(O)$.

4.4.2.1 HMM Topology

Restrictions in the transition matrix A, can define the topology and behaviour of the HMM, where allowing movement from one to all states in the HMM, is called a fully connected or ergodic HMM. That is, every state can be reached from every other state in one single time step, t. Ergodic HMMs are normally implemented if no underlying assumptions are made on the audio data. Ergodic HMMs are also suitable for modelling data that is cyclic or repetitive. The transition matrix A for a 3 state HMM would contain the following probabilities:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

Graphically this HMM would follow Figure 4.6.

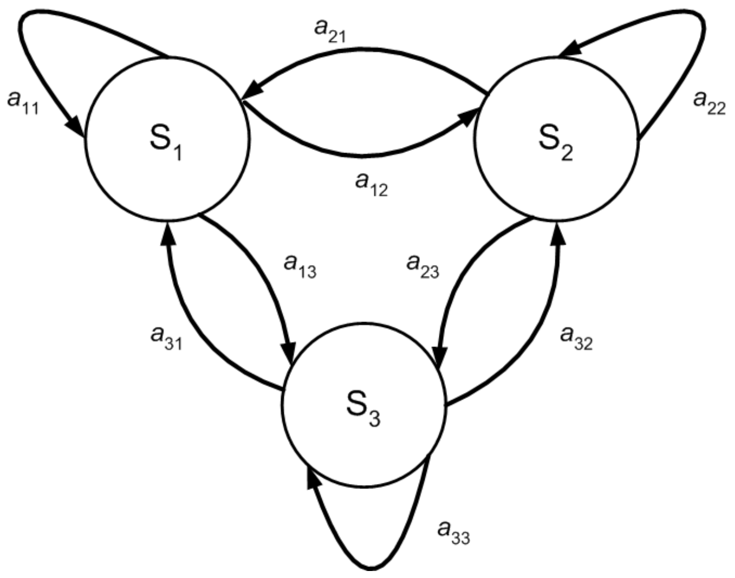


Figure 4.6: An example of a 3 state ergodic HMM

Restricting movement from the left states to right states only, are called Bakis HMM. These HMMs are considered suitable for modelling phones in ASR systems, because speech is a decaying process. Restricting movement from left to right, can represent this decaying process better than an ergodic HMM. The transition matrix A for a 3 state Bakis HMM would contain the following probabilities:

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{bmatrix}$$

Graphically the HMM for the 3 state Bakis HMM can be displayed as in Figure 4.7.

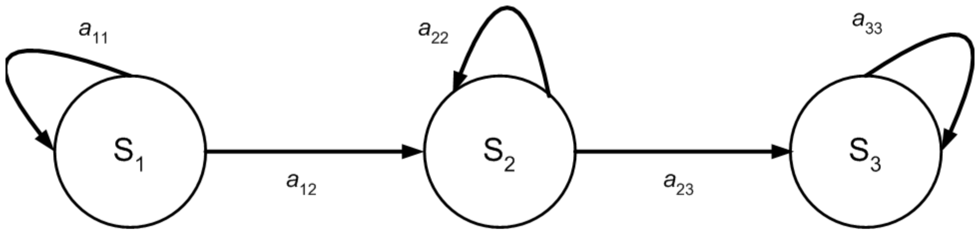


Figure 4.7: An example of a 3 state Bakis HMM

An objective of this thesis, is to efficiently model broad content classes such as Studio, Game and Advert sequences. Little is known about the underlying data structure found in each of these classes, as of yet. Therefore, restricting movement between these states, without prior investigation into these content classes, is justified. The HMM behaviour is not restricted for all experiments (unless stated), where fully connected ergodic HMM models are applied[4].

4.4.2.2 Continuous or Discrete HMM

Various applications of HMM for video indexing, such as genre detection (Huang & Wang (2000), Wang et al. (2000)) and sound effect recognition (Cowling & Sitte (2003)), implement discrete HMM models for audio. This type of HMM outputs an observation sequence characterised by discrete symbols from a finite alphabet, where

[4]During the parameter estimation phase, the initial ergodic HMM will not necessarily converge to a final ergodic representation (see Chapter 7.5.4).

each state in the HMM is modelled by a discrete PDF. One problem with this approach, is that the audio signal is continuous, and for problems such as football audio, not all content is known. To convert the audio stream into discrete form, methods such as vector quantisation are required. In this form, the audio stream becomes a sequence of discrete symbols generated from a known “code-book”. However, converting the audio stream into discrete form can result in serious degradation of the signal, and information loss (Rabiner & Juang (1993)).

An alternative approach is to employ continuous density HMM models. Continuous density HMM models are widely applied in speech recognition and other problem areas such as sound classification (Reyes-Gomez & Ellis (2003)), event detection (Xiong et al. (2003*a*)). For each state, $b_j(O)$ is represented by a continuous PDF, with the most widely applied distribution being Gaussian.

Many sound classes such as speech are often non Gaussian and are better represented by a mixture of Gaussian components. A mixture architecture can represent the variation found in similar sounds more efficiently than a single PDF, such as slight changes in environment, speaker and voice. Mixtures enhance the power of a model family by representing a complex distribution, as a combination of simple components. A simple example of this is the GMM, where a mixture architecture can be designed to fit non-Gaussian data robustly. It is common practice to use a mixture of Gaussians for modelling each state, where the same number of mixture components are normally fixed across all states. In fact, a single GMM classifier could be considered an example of a one state HMM.

4.4.3 HMM Application

Both the application and the problem to be solved, be it content analysis, structure segmentation and classification, or event detection, should be considered before applying HMM. In particular, considerations concerning HMM state selection and mixture architecture, which can all have an impact on the HMM accuracy. Application issues include the size of the model (N) and the number of mixture components per state (M), where both these issues are dependent on the test collection, the underlying data processes and the problem faced. The problem of estimating N and M is discussed further in section 4.4.3.1, outlining possible solutions.

Why and how some of these issues are addressed in the thesis are discussed in the following sections, analysing two HMM applications. The first application is content analysis and the second, multi-class classification (structure segmentation and classification, and event detection). For both these problems, estimation of N is important for effective application of the HMM. The difference in applying HMM to both these problems are mentioned.

The HMM is applied as a data mining tool for the problem of content analysis, where an optimal HMM representation is chosen for the entire data set, outlined for this problem in section 4.4.3.2. New video sequences are classified by a set of HMM classifiers for the second problem, where each HMM is a statistical representation of a known content class found using both domain knowledge, investigated further in section 4.4.3.3. First a background to the problem is presented.

4.4.3.1 Background

A crucial decision in HMM design is the selection of the model size (N). Estimating N, the number of states in a HMM is a non-trivial problem (Rabiner & Juang (1993), Chen & Gopinath (1999), Li & Biswas (2000)), where the overall system performance can be effected by selection of HMM size. Problems, such as parameter estimation, overfitting and general poor model fit can result from poor model selection. Model selection is important for generating models that are strong representations of the respective content class. However, a task such as selecting an appropriate number of states, is not straightforward, and in the case of video indexing under-researched. Both domain knowledge and training data availability have to be considered before state selection.

For ASR systems, state number estimation can be achieved using domain knowledge. For example, speech has a modular structure that can be used to limit the number of variables required for modelling (Reyes-Gomez & Ellis (2003)), where words are divided into smaller units, called phones. These units are then modelled by a 3 or 4 state Bakis HMM. To recognise speech, sequences of classified phones are then examined using language modelling techniques (Ney & Ortmanns (2000))[5].

[5]This problem is discussed in more depth in Chapter 7.2.

It would be difficult for a HMM designed for speech recognition to generalise to other types of sound classes, except speech. Content classes found in the audio track of a football video will contain inherently different structures to that of speech. For example, the audio track was reported to contain a collection of very similar types of sound other than speech, including music, crowd cheering, and sound effects (see Chapter 2).

In the video retrieval literature, this is a popular strategy for selecting the number of states in a HMM. For multi-class classification problems, the test collection is divided into known content classes that are represented by a single HMM. Each HMM is then assigned the same number of states when modelling these differing content classes. A single state being a representation of a sub-structure found in the content class. This approach is applied to problems such as genre detection (Huang & Wang (2000), Wang et al. (2000)), news segmentation and classification (Eickeler & Muller (1999)), Play/Break segmentation of football video (Xie et al. (2002)) and football event detection (Xiong et al. (2003*a*)). This strategy can be helpful when matching a known number of potential states found in the data, such as speech. However, there has been little investigation into the suitability of this strategy when applied to broad content classes, such as those found in video files.

Assigning the same number of states to broad content, such as video genre, ignores the differences in the underlying processes found in such content. Broad content classes contain various sub-structures, such as speech, music, sound effects, etc. Fixing each HMM to contain a small number of states will not explain the inter-class variation well for complex content classes. Too few states can result in insufficient discriminative information encapsulated in the HMM. Implementing HMMs with a large number of fixed states, in order to encapsulate all the complexity found within each class, is also not an appropriate solution. Too many states, for some content classes, can result in overfitting problems or poor parameter estimation, as well as an increased complexity and computational costs.

During state selection, some degree of consideration is also required, with respect to training data availability. As the number of states increase, so does the number of training samples required to estimate parameters precisely. However, precise and consistent parameter estimation is often limited by the size and quality of the training data (Jain et al. (2000)). In terms of performance and complexity, as the number of states increase the number of computations associated with a HMM grow quadratically (Rabiner & Juang (1993)). There exists a trade off in terms of better model fit, associated with larger more enriched models, versus the number of training samples available. Larger, more complex models require comprehensive training samples, which are not always widely available. Some degree of balance is required in terms of model complexity with the size of the training data available. This is taken into consideration in Chapter 8.2, where model selection is studied further.

In this thesis, HMM is applied as a content analysis tool in Chapter 7. HMM model selection for specific content classes is studied further in Chapter 8. The importance of estimating the number of states for both of these problems is now discussed.

4.4.3.2 Solution 1: Content Analysis

A particular problem with generating statistical models for recognising known content, is that the content classes and sub-classes that are found in the video file must be known. The main structures contained in football video were discovered through manual inspection, in Chapter 3.3. These structures are investigated further using a HMM as a data mining tool in Chapter 7. The goal is to generate a comprehensive knowledge of the structures found in the audio stream, with detailed knowledge of both the main content classes and the sub-units that make up each class. For this problem, each hidden state in the HMM is assumed to contain a logical meaning. e.g a type of sound.

For content analysis, a HMM is applied to model the entire audio track for football video. To apply the HMM to this problem, it is required to estimated the number of states given the data set. This is important, as it is assumed that each state in the HMM represented a semantic sub-structure found in audio track. Therefore, selecting as accurate a representation of the audio track as possible was important. This problem is solved by using a non-discriminative model selection measure (see Chapter 7 for more details). The type of structures that the HMM modelled, using this unsupervised approach, are then analysed. These structures are compared to the findings of the study in Chapter 3.3.

4.4.3.3 Solution 2: Multi-class Classification

The aim of both manual inspection and content analysis, is to identify a set of known content classes, where Figure 3.2 is an example of the main content classes found in football video. These content class can then be statistically modelled by a HMM, enabling future video sequences to be segmented and classified, using this set of HMMs (see Chapter 9 and Chapter 10).

Model selection strategies can be employed to generate robust HMMs for each class, required for accurate classification. The aim for model selection, is to select the simplest possible model that does not have a detrimental effect on overall system accuracy. As explained previously, large complex models will provide more enriched descriptions for each content class, but the larger number of parameters found in more complex models require a greater volume of training data, for accurate parameter estimation. Model selection can act as a balance between selecting a strong HMM representation and the volume of training data available, also avoiding potential problems, such as overfitting.

Therefore, for estimating both state and mixture number, three model selection methods are investigated: an exhaustive search approach with a stopping threshold, the Akaike Information Criterion (AIC) (Akaike (1974)), and the Bayesian Information Criterion (BIC) (Schwarz (1978)). This problem is addressed in further detail in Chapter 8.

4.4.4 HMM Training

The HMM can model complex processes using low-dimensional data, thereby allowing the HMM to be trained using a large amount of data. The low dimension is achieved by segmenting the data into small time steps, from which low-dimensional measurements are made. Although, the total observation space is large, in terms of observation length (number of time steps), the dimension of the observations can be kept low. A standard method for HMM training is followed, for full details on the training process please refer to Rabiner & Juang (1993).

For each content class, a training sample of representative data is manually generated.

The parameters for the HMM belonging to that content class are then estimated using an expectation-maximisation training algorithm, called the Baum-Welch algorithm. To generate a good model it is important to have good initial estimates. For parameter initialisation, the k-means clustering algorithm is applied. Afterwards, these parameters are re-estimated iteratively using the Baum-Welch algorithm until convergence, where the maximum likelihood is used as a stopping criterion.

4.4.5 Classification

To classify a new audio sequence given a set of representative HMMs, the simplest classification framework is to use the maximum likelihood criteria. This is identical to the decision process outlined for GMM (in section 4.3.3). That is, given a set of HMM models, each a statistical representation of one content class, a new data sequence is classified into the class returning the highest model likelihood score. A Dynamic Programming algorithm called Viterbi is applied for calculating the likelihood of a HMM generating a data sequence (Rabiner & Juang (1993)).

There are alternative methods to maximum likelihood decision process, such as entropic prior HMMs (Xiong et al. (2003*a*)) and dynamic programming (Huang & Wang (2000), Xie et al. (2002)). Another method, applied in continuous speech recognition, is to use a SuperHMM (Eickeler & Muller (1999)). For audio indexing there has been no comparative research into the merits of each decision process. Therefore, a number of decision processes for HMM classification are compared in Chapter 9.

4.4.6 Synthetic Data Generation

Given a set of parameters for a HMM λ, it is possible to then generate synthetic data from that model. Such data are examples of typical sequences generated from that model. This functionality was utilised for generating synthetic data from HMMs with known parameters, to test a number of assumptions during training and implementation phases[6]. This is possible, because the HMM is a statistical model of the underlying processes that generate a sound class. This functionality is important for assumption

[6]The Probabilistic Model Toolkit (PMT) written for Matlab, was used for generating synthetic data for HMM. For more details about the toolkit, see Cambridge Labs (2003)

checking and data generation. For more details of this and other HMM issues please refer to the tutorial Rabiner & Juang (1993). Examples of applications that applied synthetic data for training and evaluation purposes include Li & Biswas (2000) and Smyth (1997).

4.5 Summary

An argument was formed in this chapter as to why the GMM and HMM statistical models were chosen for audio content classification. The main points were that both frameworks were mature, well research, robust and proven to be effective through experimentation. Both the GMM and HMM were then introduced outlining the design and implementation issues that should be considered for application to the football domain. In particular, issues were discussed concerning HMM application that are not often researched in the video indexing literature, such as model design and model selection.

Chapter 5

Audio Feature Representations: A Review

5.1 Introduction

A number of audio feature representations are reviewed in this chapter, with the sole purpose of identifying a set of features that would be suitable as a front-end for both indexing problems[1], a front-end, being the process of extracting measurements from the data, for analysis. This audio parametrisation step is one of the most important phases in the development of audio-based indexing algorithms. A robust selection of measurements can help discriminate between known content and assist efficient classification. The more descriptive the information contained in the feature set is, the more accurate the representation will be. For example, features such as gender, age, hair colour, hair style, eye colour, build, height and weight provide a good list of measurements for describing an unknown person to a friend. The friend can use this list of features for recognising the described person later. However, a feature list containing only eye colour would provide a more challenging problem for the friend, given the task of recognising the described person in a crowded room.

The same problem arises for recognising known audio content in future video sequences. Given a list of known content structures to be classified, as much descriptive

[1]The two problems are: structure segmentation and classification, and event detection.

information as possible about these content classes is desired, so future sequences can be labelled accurately. An ideal solution would be a feature space with tightly grouped content classes containing as little variation as possible. Those content classes should also be well separated in the feature space. A weak selection of audio features can cause overlapping between classes, making the task of classification difficult. This is why careful selection of features is critical.

There has been a lot of research into audio feature representations for a variety of problems including Automatic Speech Recognition (ASR), speaker recognition, General Audio Data (GAD) classification, video genre detection, Music Information Retrieval (MIR) and sports event detection. In this chapter, the main audio feature representations applied to these problems are studied. From reviewing these related works, a list of candidate feature sets are then chosen, which are believed to contain the best discriminative properties for the problems faced in football audio. In the following chapter this candidate list will be examined formally, with the most accurate feature set being chosen as a front-end for the two indexing algorithms: structure segmentation and classification, and event detection.

The remainder of this chapter is as follows. In section 5.2 a background into audio parametrisation is given, while in section 5.3 feature sets that extract statistics from the audio signal are discussed. An alterative method that converts the original audio signal into a form that is perceptually similar to how the human auditory systems analyses sound is then introduced in section 5.4. Features that can be extracted from this perceptual representation of the audio signal include the Cepstral coefficients (section 5.4.2) and Mel-frequency Cepstral coefficients (section 5.5). An alternative method for extracting information from the audio signal by applying a model of speech is described in section 5.6. The merits of all these features are then investigated in section 5.8, before deciding upon what feature sets are believed to be suitable for the football audio. The chapter is then concluded in section 5.9.

5.2 Background: Audio Signal Representations

Audio features are generated either directly from the audio signal itself (often called the time domain), or the original signal is converted into a form perceptually similar to that of the human auditory system, including the frequency domain and the Mel-scale. There are also other methods for representing audio such as speech models. A commonly used example of a speech model is the Linear Predictive Coding (LPC) model. In this section these audio representations are introduced in further detail.

The time domain is a representation of the basic pressure waves produced by a sound source. This is the form the audio signal is captured (see Appendix A for a brief description of sound recording). In this form the audio sequence is described by the amplitude of the signal over time, where activity in the audio track would result in an increase in amplitude. Figure 5.1 is an example of an audio signal in the time domain that contains speech and some background crowd sound. In this representation the audio signal can be difficult to interpret, particulary when attempting to differentiate between different sounds. For example, in Figure 5.1 both the periods of speech and crowd sound are represented by an increase in amplitude.

In the time domain, statistics about the audio signal such as the average amplitude over a short period of time can be measured. These measurements are often called physical features. These statistics are often applied for recognising activity such as speech in the audio signal.

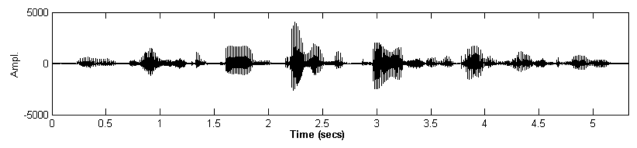


Figure 5.1: An example of an audio signal in the time domain.

In the human auditory system, pressure waves are picked up by the ear and converted into a more intelligible form, by the cochlea for translation by the brain. The audio signal can also be converted into a more intelligible form using techniques that attempt to replicate the human auditory system. A popular method for translating audio is by converting the signal into the frequency domain. The process of converting the audio signal into the frequency domain, is achieved by dividing the signal into its many frequency bands mathematically. The result can then be represented by a spectrogram. Figure 5.2 is an example of an audio sequence in both the time and frequency domain of an audio sequence captured during a football match. The audio signal in the time domain is displayed above the spectrogram.

In the spectrogram below, the y-axis represents the frequency of the signal over time (the x-axis). The points in the spectrogram relate to activity at different frequencies. There is a correlation between the energy of a pattern, and the intensity in the spectrogram, where an increase in energy will be represented by darker patterns.

In this form, content in the audio signal can be easier to translate when compared to the time domain. An experienced spectrogram reader can recognise content found in the audio signal from studying the spectrogram alone. For example, the intense wavy lines that are repeated at different frequencies, ranging from 0kHz to 4kHz in Figure 5.1, correspond to speech. These repeated lines are the harmonics of the voiced speech. In the same figure, the constant cloud like patterns between 1 and 2kHz that fluctuate in intensity, is crowd sound recorded in the stadium. At approximately 6 seconds, an event occurred that triggered a loud crowd response. This crowd response coincided with an increase in intensity in the spectrogram, at the same time, blurring the speech pattern. The distance between the speech harmonics also increased at this point, which is an indication the commentator's voice became excited during the event.

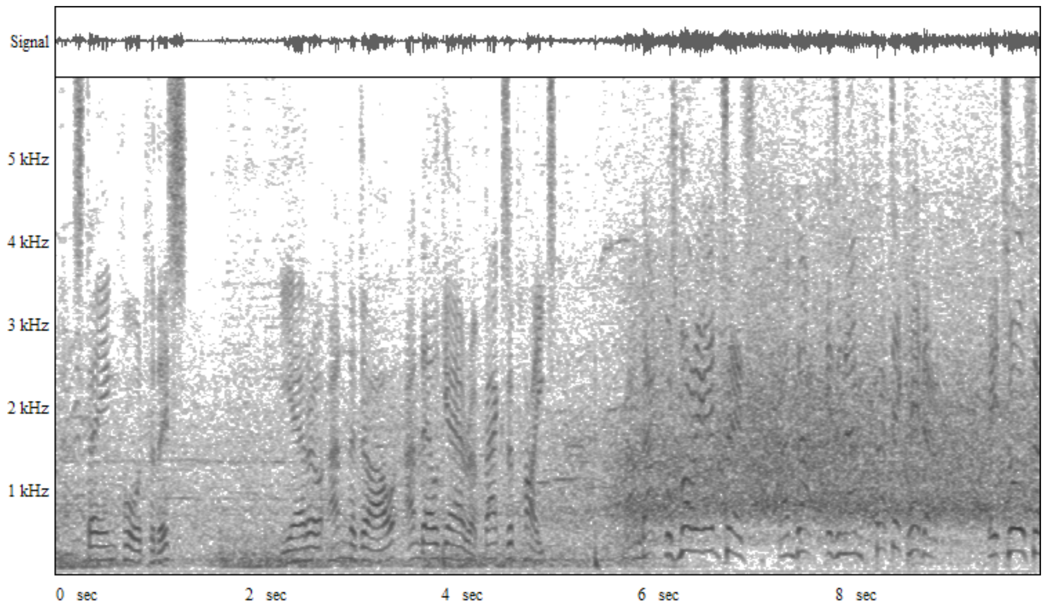


Figure 5.2: An audio signal captured from a football sequence and the derived spectrogram.

Provided with this enriched representation of the audio signal, a number of features can be derived, often referred to as perceptual features. Many of these features relate to how the human auditory system hear sound, such as the pitch, brightness or rhythm of the signal. From the frequency domain, the spectrum can also be further converted into other forms such as the perceptually scaled frequency axis of the Mel-scale (Rabiner & Juang (1993)). A popular set of features for automatic speech recognition are extracted from the Mel-scale called Mel-frequency Cepstral coefficients.

Alternate audio representations include speech models. An audio signal containing speech alone can be represented more efficiently, in terms of signal complexity, than an audio sequence containing multiple sound classes. This property is the motivation behind speech models (Rabiner & Juang (1993)), where a popular representation is the Linear Predictive Coding (LPC) model. LPC is one of the most powerful speech analysis techniques that is used for encoding good quality speech at a low bit rate by providing extremely accurate speech parameter estimates. This representation is widely applied to applications such as Telephone bandwidth speech. Coefficients can also be extracted from this audio representation, and are often applied as the front-end for ASR systems.

Given these audio representations (time domain, frequency domain, Mel-scale and LPC), the following sections review applications that apply the prominent feature sets extracted from each.

5.3 Time Domain Features

Various statistics can be extracted from the time domain that measure changes and patterns in the amplitude of a signal. These measurements are often referred to as frame level or physical features (Wang et al. (2000)). Examples of such features include the short-time energy, zero-crossing ratio (ZCR), short-time fundamental frequency and the average magnitude difference function (Liu & Chen (1998)). ZCR is a count of the number of times the signal crosses the x-axis, while the average magnitude difference is the mean signal amplitude over a short time period[2].

Physical features have been applied to various problems including music and speech

[2]For an in-depth explanation of physical features please refer to studies by Liu & Chen (1998), Wang et al. (2000) and Rabiner & Juang (1993).

discrimination (Saunders (1996), Alatan et al. (2001), Lu et al. (2002)) and event detection in sport (Chang et al. (1996), Nepal et al. (2001)). In a two-class problem such as the discrimination between speech and music sequences, very simple physical features can be employed with a high degree of accuracy. Saunders (1996) applied a front-end containing physical features for segmenting music and speech links for radio broadcast sequences, achieving 98% classification accuracy. Lu et al. (2002) also applied variations of standard physical features for classifying audio sequences into groups such as speech, music, environmental sound and silence.

For classification, an heuristic rule-based strategy was adopted, that was 98% accurate for clean speech and 85% for noisy speech. Discrimination between speech and non-speech sounds was lower, with only 61% of environmental sound sequences classified. Alatan et al. (2001) applied a similar technique for classifying film soundtracks into speech, silence or music with similar performance (see Appendix C.4.3 for further details on this algorithm).

For detecting events in sports sequences, both Chang et al. (1996) and Nepal et al. (2001) threshold the ‘loudness’ of the audio stream. Key events are assumed to be correlated with an increase in crowd reaction that would trigger an increase in signal amplitude. By thresholding the volume, the authors believed periods of crowd cheering could be recognised, where the loudness or volume feature is based on extracting statistics measuring an increase in amplitude over various sub-bands. However, this increase in amplitude can also be a result of speech, music or other environmental sound. As illustrated in Figure 5.2, both speech and crowd cheering also occur at the same period of time. Discrimination between the actual sound process that generated this increase, would be difficult using volume as a sole measure.

5.3.1 Discussion

In general, for tasks involving multi-class classification it is difficult to determine sound types using simple physical features alone. In this domain, different sound classes in the audio signal can display similar traits that can not be measured by these physical features. From studying the time domain signal in Figure 5.2 this assumption can be informally proved, where both the occurrence of speech and crowd cheering result in an increase in amplitude. Separation between both sound processes is difficult to recognise visually. The only recognisable difference is a drop or increase in activity. However, it is very difficult to determine what sound class caused the changes in amplitude, ZCR or short-time energy. Research using these features alone, require the audio classes to be, audibly, very different from each other (Saunders (1996), Alatan et al. (2001), Lu et al. (2002)) such as speech and music. These algorithms begin to fail when sound classes are similar or overlap such as speech and environmental sound (Lu et al. (2002)).

For the sports video domain, there are many environmental sounds that can cause an increase in signal volume as well as crowd cheering. Accordingly, applying physical features alone, similar to Chang et al. (1996) and Nepal et al. (2001), was ruled out for this thesis. It would be desirable to convert the audio signal into a perceptually more intuitive representation, such as the frequency domain, before feature extraction.

5.4 Frequency Domain Features

The original signal is often transformed into frequency domain, to perceive audio in a manner similar to that of the human auditory system. By doing so, different sounds are separated in the new signal representation. Figure 5.2 is an example of a frequency spectrogram containing multiple sounds. Voiced speech can be recognised in the spectrogram as the dark structured patterns with repeated harmonics. On the other hand, crowd sound is associated with the almost random patterns in the background. Intuitively, it can be assumed that the frequency domain provides a more enriched description of the sound source ready for classification.

A series of steps is required for converting an audio signal into the frequency domain. The signal is first divided into equal sized overlapping frames, a frame being a group of audio samples (see Appendix B for more details). An equal number of sequential frames are then further grouped into a window and transformed into the frequency domain using a Fast Fourier Transform (FFT). FFT is a mathematical function that separates out the component sinusoids in the original signal[3]. The final result can then be displayed as a spectrogram, Figure 5.2 being an example.

[3]This entire process is described in depth by Rabiner & Juang (1993).

5.4.1 Perceptual Features

Examples of measurements calculated from the frequency domain include the frequency centroid (a measurement of the mean of the spectrum), bandwidth (the variance of the spectrum) and the sub-band energy of the signal at specific frequencies (Wang et al. (2000)). These perceptual measurements have been applied to problems such as classifying baseball strikes (Rui et al. (2000)), speech and music discrimination (Scheirer & Slaney (1997)), general audio classification (Chaisorn et al. (2003), Zhang & Kuo (2001), Wold et al. (1996)) and genre detection (Liu & Chen (1998), Wang et al. (2000)). Rui et al. (2000) applied short time sub-band energy to generate an audio template of a baseball being hit by a bat. 81% of baseball hits were automatically recognised across a new test collection, with 14% false detections.

Scheirer & Slaney (1997) extracted 13 simple temporal and perceptual features for classifying audio into a speech or music class, achieving over 98% classification accuracy. For a similar problem, Zhang & Kuo (2001) applied an heuristic rule-based procedure to classify audio files containing speech, music, silence and combinations of both music and speech, found in TV audio. Again, a collection of perceptual and physical features were extracted for analysing new audio sequences. The experimental results indicated an average classification accuracy of 90% across each content class.

Liu & Chen (1998) evaluated a set of physical and perceptual features for the task of classifying audio sequences into the correct genre (weather reports, news, sport and adverts). Physical features, such as ZCR, were found not to contain sufficiently descriptive information, when compared to other perceptual features. Bandwidth and sub-band energy were discovered to contain the most discriminative properties overall. The same features were also integrated into a system for recognising silence, music and speech in news video (Chaisorn et al. (2003)). Perceptual features, including pitch, loudness, frequency centroid and bandwidth have also been applied to retrieve specific sound events in a database (Wold et al. (1996)).

An alternative strategy to measuring statistics such as the frequency centroid is feature extraction. This is the process of extracting coefficients directly from the frequency representation, through dimensionality reduction techniques. An example of this is the derivation of Cepstral coefficients.

5.4.2 Cepstral Coefficients

Coefficients can be directly extracted from the frequency domain often named short time Fourier transform coefficients or FFT Cepstral coefficients. For simplicity these features will be called Cepstral coefficients (CC) for the remainder of this thesis. CC are often utilised as the front-end (feature set) for speech detection or recognition systems due to the potential low dimensionality and low correlation between features.

CC are calculated by computing the inverse FFT of the logarithm of the spectrogram[4]. Dependent on the frequency bin size, the CC can be high in dimensionality. The feature space dimensionality can be reduced by combining frequency bins during the FFT process, with a small degree of information loss. This information loss is balanced with a decrease in dimensionality. The lower frequency coefficients extracted from the spectrum represent the shape, while the higher frequency coefficients relate to fast changing spectral components such as speech excitation.

CC features have been applied to a variety of audio classification problems. For example, Chang et al. (1996) detected keywords that can be correlated to exciting events in American football such as “touch down”, applying a simple template matching procedure. Keyword training samples were parameterised by 8 CC plus the perceptual log energy feature. An incoming audio sequence was then parameterised using this feature set and matched against the keyword templates. From experimentation over a small test collection, the algorithm recognised 4 out of 5 keywords, with 3 false alarms.

General audio patterns parameterised by CC have also been shown to be well modelled by a number of statistical frameworks, including the Gaussian Mixture Model (Roach & Mason (2001)) and the Hidden Markov Model (Boreczky & Wilcox (1998), Reyes-Gomez & Ellis (2003), Eickeler & Muller (1999)). Roach & Mason (2001) employed a CC feature set modelled by GMM models for classifying video sequences into genre such as Film and News. This approach was found to be more accurate than modelling visual and motion features through experimentation.

Within a HMM framework, CC feature sets have also been successful. For example, in a follow up study to Wold et al. (1996), a CC feature set applied as a front-end to a set of HMM classifiers, was found to be more accurate than a set of standard perceptual

[4]Please refer to Rabiner & Juang (1993), pp.163. for the formal derivation of the CC features.

and physical measurements (Reyes-Gomez & Ellis (2003)). The new combination of CC modelled in a HMM framework recorded a higher classification accuracy.

Boreczky & Wilcox (1998) also successfully integrated an audio-based feature set consisting of CC coefficients into a HMM shot segmentation algorithm. Appending the 12 CCs into a multi-modal feature set containing motion and visual information, improved the shot segmentation accuracy. Eickeler & Muller (1999) also employed 12 CCs as part of a multi-modal feature set for segmenting and classifying news video structure. The CC feature set was included for modelling the audio differences between the different content structures found in news video, such as report and interview.

5.4.3 Discussion

Perceptual features have been applied to a number of audio classification problems with a varying degree of success. Often these features are supplemented with physical features such as ZCR. There are, however, a number of potential problems when modelling both physical and perceptual features together in the same framework. These problems include the feature space dimensionality, managing features based on differing time scales, and the possible correlation between features. For example, features extracted from both the time and frequency domain may measure the same information.

Feature selection is a method that can be applied or used for evaluating what combination of features are appropriate for a given problem (Liu & Chen (1998)). However, there still exists the problem of combining a number of different features with varying time scales. Possible solutions to minimise this problem include vector quantisation (Wang et al. (2000)), but this method can cause a degree of information loss (Rabiner & Juang (1993)).

An alternative to calculating perceptual features is the extraction of CC. From experimentation, these coefficients have been shown to be more accurate than feature sets containing a combination of perceptual and physical features for audio classification (Reyes-Gomez & Ellis (2003)).

CCs have also been used successfully as a front-end for speech recognition and general audio classification systems. However, for speech recognition the most widely applied feature sets are the Mel-frequency Cepstral coefficients (MFCC).

5.5 Mel-frequency Cepstral Coefficients

The most widely applied feature set for automatic speech recognition systems is the Mel-frequency Cepstral coefficients (MFCC). These coefficients are extracted from the perceptually scaled frequency axis called the Mel-scale (Rabiner & Juang (1993)). This is a warped scale specifically designed to utilise the properties of human speech, which is usually found in low frequencies ranging up to 4kHz (see Figure 5.2). The conversion of the audio signal into the Mel-scale results in a higher resolution at lower frequencies, placing emphasis on those frequencies where speech is more prominent. This is the main reason why MFCC are one of the leading front-end features for automatic speech recognition (Rabiner & Juang (1993)).

To extract MFCC from the audio track, the signal is first converted into the frequency domain. The signal is then passed through a bank of triangular filters uniformly spaced in using the Mel-scale. At this point the data is reduced. A natural log transformation is then performed to separate out convoluted signal components, smoothing the signal. Finally a Discrete Cosine Transform (DCT) is applied to extract the Mel-frequency Cepstral coefficients[5].

There are a number of reasons, apart from the Mel-scale, as to why MFCC are widely applied to a variety of research problems. MFCC are compact in comparison to similar representations such as CC, where the same information can be represented with fewer parameters. MFCC coefficients are also uncorrelated, a property that occurs during the computation of the coefficients. The DCT that is applied during the MFCC extraction process, is an approximation of the Karhunen-Loeve transform or Principle Components Analysis (PCA) (Theodoridis & Koutroumbas (1999)). PCA is often utilised for reducing dimensionality by removing redundancy in a highly correlated feature set. The DCT ensures that the MFCC are orthogonal, a desirable property in a feature set.

Typically the first 10 to 16 MFCC coefficients are used in the front-end of an ASR system, assumed to contain the most representative information of speech. The first (Δ

[5]For a more in depth explanation of this process see Rabiner & Juang (1993), pp.78, 189-190.

MFCC) and second order ($\Delta\Delta$ MFCC) time-derivatives are also included in a standard ASR feature set, also known as the velocity and acceleration parameters, respectively. These coefficients measure the pitch and spectral dynamics of the signal and have been shown to improve accuracy (Young et al. (2003)).

The MFCC are gain (volume) independent except the 0^{th}coefficient, which is dependent on the energy or power level of the signal. This 0^{th}coefficient can be shown to be a representation of the averaged energies for each frequency band but is normally not added to the front-end of ASR systems because of its unreliability. However, the log energy of the signal is appended to the ASR feature set (Zheng et al. (2001)).

Apart from speech recognition systems, MFCC coefficients have also been used for other applications, such as general audio classification (Spina (2000)), excited speech detection (Rui et al. (2000)), and event detection (Adams et al. (2003), Xiong et al. (2003*b*)). Spina (2000) classified seven different radio environments including clean speech, telephone bandwidth speech, music and environmental noise using a combination of MFCC and Hidden Markov model classifiers. Classifying these environments is an important step for improving word error rate for speech transcription. The system was evaluated across a large study using over two hours of radio news achieving 80.9% classification accuracy overall. Rui et al. (2000) differentiated between excited and normal speech by applying a MFCC feature set and a statistical classifier, recognising 49 out of 63 sequences correctly.

For detecting events such as rocket launches and explosions, Adams et al. (2003) utilised a standard ASR MFCC feature set, comparing both GMM and HMM as classifiers. The HMM classifier was able to recognise with 38% precision explosion events and rocket launch events with 56% precision, across a large test collection (Smeaton & Over (2003*a*)). The same approach also classified 98% of music sequences and 89% of the speech sequences in the collection.

Xiong et al. (2003*b*) also applied a MFCC front-end for classifying content in sports video. Applying a MFCC feature set as part of a HMM classifier was found to be 94% accurate for classifying speech, music and crowd cheering classes in sport sequences.

5.5.1 Discussion

The MFCC feature set appears to be a robust front-end for a variety of research problems including speech recognition and also audio classification. The advantage of such feature representations are:

- the coefficients are designed specifically for representing speech,
- are low dimensional,
- are orthogonal.

5.6 Linear Predictive Coding Cepstral coefficients

Linear Predictive Coding (LPC) is a powerful speech analysis technique used for encoding good quality speech at a low bit rate (Rabiner & Juang (1993)). LPC can provide extremely accurate estimates of speech parameters and is relatively efficient for computation. The LPC model utilises the property that speech is found at lower frequencies, to decrease the bit rate of the signal. LPC attempts to represent how speech is formed in the vocal tract with an all pole model. The coded speech is synthesised by a time-varying all-pole filter, where the filter coefficients are obtained with an autoregressive estimation method that describe the spectral envelope of the speech signal. An example of a LPC application is Telephone-bandwidth speech, which has a sampling frequency of 8kHz.

Linear Predictive Coding Cepstral coefficients (LPCC) can be derived from the parameters in this speech production model[6]. These coefficients have been applied to applications such as speaker detection and verification system (Campbell Jr. (1997)), and word recognition (de Wet et al. (2000). From experimentation, systems employing LPCC as a front-end have also been proven to outperform MFCC (de Wet et al. (2000)). The LPCC coefficients were found to be robust to noise in comparison to MFCC, for the problem of speech recognition in noisy audio data. LPCC coefficients have also been shown through experimentation to be more reliable than the standard LPC parameters for speech recognition (Rabiner & Juang (1993)).

[6]Rabiner & Juang (1993) pp.115 provides a more in-depth explanation of the LPC and derivation of the LPCC coefficients.

5.7 Other Feature Sets

Other perceptual models, not introduced in this thesis, include the cochleogram and the autocorrelogram. Both models are popular representations in the fields of Auditory Scene Analysis and Psychoacoustics (Scheirer (2000)). However, these models are still at an early phase in development for audio classification. Deriving features from such models is computationally very expensive and at present, unproven in comparison to more established feature sets such as MFCC and LPCC.

Recently the MPEG-7 audio feature set has been developed. These features are perceptually similar to MFCC, where the signal is transformed into the frequency domain using PCA instead of DCT. Xiong et al. (2003*b*) applied MPEG-7 features as a front-end using a set of HMM classifiers for classifying content found in sports video. However, when compared with a system using MFCC features, no significant gain in accuracy was found. MPEG-7 features are relatively new and are, again, not investigated in the confines of this thesis. However, possible future research evaluating the merits of MPEG-7 features and both cochleogram and the autocorrelogram models would be beneficial, especially comparing such features against more established sets, such as CC, MFCC and LPCC coefficients.

5.8 Comparative Feature Set Investigations

There have been few comparative studies of feature sets for the problem of general audio classification. One such comparative study presented a comprehensive study of feature sets including LPCC, CC and MFCC, using a range of classifiers, such as artificial neural networks (ANN), GMM and HMM (Cowling & Sitte (2003)). The feature sets were compared across 8 different environmental sound effect classes such as, ‘footsteps’, ‘glass breaking’ and ‘coins dropping’, and a separate test collection containing musical instrument sound clips. The MFCC feature set, when applied as a front-end of a Dynamic Time Warping classifier, displayed the highest classification accuracy. However, the CC feature set was found to be more accurate than MFCC and LPCC in terms of classification accuracy, when modelled by HMM classifiers. Overall, there was no significant difference between using CC and MFCC feature sets overall, where the selection of a classification framework appeared to be a more important decision. A potential problem with the experiments was that for some models, such as the HMM the feature sets were converted from continuous to discrete using vector quantisation. This process can result in a loss of information that could have influenced results (Rabiner & Juang (1993)).

Li et al. (2001) studied a 143 features for grouping general audio into a broad range of seven categories, such as silence, speech and music and also combinations of speech and music, etc. The survey indicated that MFCC and LPCC feature sets resulted in higher discrimination accuracy than other perceptual or physical measurements including ZCR, average energy and bandwidth. A Bayes classifier was used for evaluating each feature combination.

Another comparative study, Tzanetakis & Cook (2002), classified music files into various genre, such as classical, rock, hip hop and jazz. A number of feature sets were compared, such as CC, MFCC and some physical features. k-Nearest Neighbours (kNN) and GMM classifiers were used to compare all feature set representations. It was found through experimentation that modelling classes with a GMM classifier, parameterised by MFCC features, was the best solution displaying 80% accuracy overall. However, for some music genres the CC feature set modelled by a GMM classifier produced better results.

A similar study, Marques & Moreno (1999) compared MFCC, CC and LPC coefficients for the task of instrument classification. New audio sequences were classed into one of eight different instrument classes each represented by a GMM classifier applying a MFCC, CC or LPCC front-end. The MFCC feature set provided the best discrimination between instrument classes (37% error rate) followed by the CC feature set (47%) and then LPCC (64%).

There has been one related comparative study of audio features in the sport domain (Xiong et al. (2003*b*)). In this paper both MPEG-7 and MFCC feature sets were compared across a number of content classes found in sports video, such as Baseball hit, Speech, Crowd cheering and Music. Both feature sets were employed as a front-end for HMM statistical classifiers. From experimentation, there was no significant gain from applying a MFCC or MPEG-7 audio feature set to the problem of classification.

5.8.1 Discussion

Selecting an appropriate audio feature set is important. In this chapter a number of audio feature representations were investigated that have been applied to a number of fields such as automatic speech recognition (Rabiner & Juang (1993)), general audio content analysis (Liu & Chen (1998), Li et al. (2001), Lu et al. (2002), Cowling & Sitte (2003), Wang et al. (2000), Zhang & Kuo (2001)), music retrieval (Wold et al. (1996), Scheirer & Slaney (1997)) and sports event detection (Nepal et al. (2001), Chang et al. (1996), Rui et al. (2000)).

There did not appear to be conclusive evidence to suggest that one particular feature set would be more appropriate than another, from studying a number of comparative research studies. For research problems, such as musical instrument classification, the MFCC feature set performed better than both CC and LPCC (Marques & Moreno (1999)). However, for speech recognition in noisy audio environments, LPCC has been shown to improve word error rate when compared to MFCC (de Wet et al. (2000)). Studying the related work as a whole, it was not clear what type of feature sets would be suitable for sports audio indexing. Both MFCC and LPCC were designed for the sole purpose of recognising or modelling speech. However, the CC feature set is less domain specific. This could be advantageous given the audio content found in football video, which is a combination of both speech and non-speech sound.

Provided with a wide choice of audio features to choose from, three different types of audio feature set representations to parameterise the audio track were chosen: Cepstral Coefficients, Mel-frequency Cepstral Coefficients, and LPCC coefficients. These feature sets were widely applied, well researched and proven to be effective for a number of research problems. Selection of these feature sets was based on previous research, as well as each feature set containing desirable properties, such as being well modelled by statistically and being orthogonal (CC and MFCC only).

Reviewing the related research into audio classification and a number of comparative studies, there is no conclusive evidence to suggest what feature set would be suitable for the domain of football audio. For the problem of sport, there has been little direct comparison of audio features. Xiong et al. (2003*b*) compared MPEG-7 and MFCC feature sets for parameterising the audio track of sports video for the purpose of classification. Both Nepal et al. (2001) and Chang et al. (1996) applied physical features for sports event detection. However, in related research, perceptual feature sets, such as MFCC, CC and LPCC have been shown to contain higher discriminative properties than physical features. It was therefore decided to concentrate on feature sets extracted from perceptual representations.

When applying such perceptual feature sets for audio classification problems, a popular solution is to adopt the same feature set configurations that have been successful for automatic speech recognition. For example, MFCC (Rui et al. (2000), Xiong et al. (2003*b*)) and CC (Chang et al. (1996)) feature sets have been applied to sports audio, and LPCC feature set for general audio (Li et al. (2001)) and music classification (Marques & Moreno (1999)).

However, the content found in sports audio is a collection of many different speech and non-speech sounds with varying levels of quality and noise. This could be an advantage for a feature set, such as CC, which are not domain specific when compared to MFCC and LPCC. Both MFCC and LPCC were developed with the aim of representing speech that may not generalise to other non-speech sound. However, related research into discrimination between speech and general non-speech audio, such as music have indicated that both MFCC and LPCC can be applied with a degree of success. Also, LPCC coefficients have also been shown to be robust to noise for word recognition systems (de Wet et al. (2000)).

It would be difficult to determine from the related research with a degree of confidence what feature representations would be more appropriate for the problems faced when generating indexing algorithms for football audio. Therefore it was decided to evaluate CC, MFCC and LPCC feature sets formally before finalising on one representation.

5.8.1.1 Technical Note

Each audio file was parameterised using the three feature sets using a Matlab toolkit called Voicebox. For further details please refer to Brookes (2003).

5.9 Conclusion

In this chapter, a number of audio representations were introduced, such as the time and frequency domains. Related research applying features sets generated from these domains were also reviewed. Studying this body of work as a whole, three candidate feature sets were chosen for formal analysis, using a test collection generated from football videos. The three feature sets were the Cepstral coefficients, Mel-frequency Cepstral coefficients and the Linear Predictive Coding Cepstral coefficients.

A summary of the main reasons for selecting these feature sets were:

- that these feature sets were extracted from audio representations based on the human auditory system (frequency),
- that the CC, MFCC and LPCC have proven track records in automatic speech recognition and general audio classification problems,
- that both the CC and MFCC feature sets are uncorrelated,
- that speech and other general sound classes, such as music, have been shown to be well modelled statistically when parameterised using CC, MFCC and LPCC,
- that there has been no definitive comparative study between audio feature sets for sport audio,
- that physical and perceptual measurements extracted from the time and frequency domain were not as accurate as these three feature sets in a number of studies (Li et al. (2001), Cowling & Sitte (2003) and Tzanetakis & Cook (2002)).

Provided with this evidence, it was believed that the CC, MFCC and LPCC feature sets would be suitable for the audio content found in football audio. In the following chapter, these three features are formally examined further for the purpose of selecting the one set containing the highest discriminative properties for football audio.

Chapter 6

Audio Feature Set Selection

6.1 Introduction

Three established feature sets were chosen in Chapter 5, believed to be suitable for parameterising the audio track of football sequences. These three candidate feature sets were the Cepstral coefficients (CC), Mel-frequency Cepstral coefficients (MFCC), and Linear Predictive Cepstral coefficients (LPCC). The feature sets were chosen for parameterising the audio track of football sequences, allowing for content to be modelled. By modelling content using a statistical framework such as the Gaussian Mixture Model or Hidden Markov Model, future video sequences can be recognised as part of an indexing algorithm.

In this chapter, each feature set is compared formally by measuring the classification error across a test collection generated from football videos[1]. There are two main objectives set out at the beginning of this chapter. The first goal is to discover what feature set provided the highest discrimination between each content class, minimising classification error. The second aim is to investigate the selected feature set further (MFCC), analysing the best configuration given the data set. This second objective is achieved through the comparison of a large number of coefficient combinations, again deciding upon the feature set minimising classification error.

The remainder of this chapter is structured as follows. Section 6.2 will review the

[1]Classification error is a measure of the accuracy for labelling new video sequences into the correct category, where the test collection contained three broad content classes found in football audio.

problem of feature selection, presenting the motivation behind the chapter objectives. Section 6.3 will then layout the feature selection experiment details, highlighting the results in section 6.4. The chapter is finally concluded in section 6.5.

6.2 Motivation

During the training phase for GMM and HMM when applied to the problem of classification, it is necessary to calculate the probability distributions in the feature space for each class. New video files can then be classified by measuring the likelihood of each model generating the data representation of the new sequence. A criterion such as maximum likelihood is then applied to decide what class the video sequence is most likely to belong to.

An important step in this process is deciding on a representation for the video sequence. The aim of feature selection is to identify those features that best capture the characteristics contained in each content class and providing a high discrimination between classes. To think of it as a clustering problem, it would be desirable to possess a set of tightly grouped clusters that are well separated in the feature space. The further the content classes are separated in the feature space, the higher the classification accuracy, provided that each class is efficiently modelled. The accuracy of labelling new sequences can fail dramatically if each class is poorly represented both in the feature space and the model.

One strategy for minimising poor class representation is to avoid working in high dimensional feature spaces, but it is difficult not to use a large number of dimensions if all the measurements carry limited information. Feature selection is the task of finding the optimal combination of features that minimises some criterion such as classification error. To arrive at this optimal solution involves searching through a combinatorial number of feature sets. However, as the dimensionality of the feature space increases so does the number of feature combinations to be evaluated, before discovering an optimal solution (Jain et al. (2000)). When using a criterion such as classification error, this process will involve comparing both feature combination using a statistical classifier, which is a computationally demanding solution (Jain et al. (2000), Theodoridis & Koutroumbas (1999)).

Optimal feature selection is an expensive and time-consuming procedure especially if the number of candidate features to begin with is large. There are alternative search strategies to an exhaustive search approach such as stepwise search (Theodoridis & Koutroumbas (1999)), that add or remove coefficients step by step until a criterion is reached. This method limits the number of combinations to be searched, although such a search strategy may locate only sub-optimal feature sets in terms of classification error. A common problem with feature selection search strategies such as stepwise, is that they often converge on localised optimal solutions.

The three feature sets chosen in Chapter 5 are well researched and share similar perceptual properties (Rabiner & Juang (1993)). Therefore, the objective for this thesis was not specifically to find the best subset of coefficients from the entire collection of features, but instead to select the feature set that provided the highest discrimination between content classes found in football audio. This simplified the feature selection process, minimising the number of feature combinations to be examined. Not evaluating the merits of all feature set combinations could be considered sub-optimal. However, the three candidate feature sets are well researched and perceptually share similar properties. It was believed that searching through all coefficient combinations would be computationally infeasible. Moreover, given the minimal investigation into feature set selection for such broad content classes found in video, even a sub-optimal experiment would provide an insight.

For feature selection in this thesis, two objectives were decided upon. The first aim was to evaluate the three feature sets CC, MFCC and LPCC, identifying which set minimised classification error. The second goal was to then study the chosen feature set further to find the best combination of coefficients in that feature set. For example, many ASR systems apply the MFCC feature set containing both time-derivative coefficients (MFCC Δ and MFCC $\Delta\Delta$) that indicate the degree of spectral change, and log energy that is a measure of the activity of the audio signal (see Chapter 5). This feature set combination is believed to contain the highest discriminative properties for speech and is consequently selected for other audio problems regardless of the differences in structure between speech and non speech sound (Garg et al. (2003), Adams et al. (2003), Spina (2000)). There has been little comparative investigation into the suitability of such feature set configurations for broad general audio classes found in video. It was believed that an important phase in algorithm development would be to examine the discriminative properties of different coefficient combinations further through experimentation. The overall goal is to identify the simplest (low dimensional) feature set that minimised classification error. The following sections will both outline and present the result of both feature set experiments.

6.3 Feature Selection Experiment

The three candidate feature sets chosen in Chapter 5 were CC, MFCC and LPCC. In this section, the experiment designed to compare the three feature sets is described. First a test collection was generated using three content classes found in football audio. Then the comparative measure to evaluate each feature set is defined (classification error). To calculate the classification error a framework is required for measuring the discriminative properties of the feature (Jain et al. (2000)). This methodology is explained further, outlining why the GMM was chosen for the experiment(s).

6.3.1 Test Collection

A test collection was generated to evaluate the performance of each feature set. To do so, 12 video files were manually annotated based on the findings discovered in Chapter 3.3. This involved manually segmenting and labelling the audio stream into the main content structures defined in the video model (Section 3.4). Both the broad content classes such as Advert, Studio and Game were labelled along with high-level concepts including speech, crowd cheering, music and silence, where each data sequence was assigned a label corresponding to the content class it belonged to. A description of the annotation and these labels can be found in Appendix D[2]

From this annotation, audio clips of varying length were extracted from the three dominant content classes; Advert, Game and Studio. These were the dominant content classes found in the video files, chosen to provide an insight into the feature set discrimination properties given by football audio data. The test collection contained approximately 15482 seconds of audio, over 4 hours of audio. For the Advert class there

[2]Only 12 video files were annotated fully due to time considerations and human resources. This process took over 4 weeks to complete. The remaining 36 video files were left for system development.

were 3161 seconds of audio (51 minutes), for the Game class there were 7304 seconds (121 minutes) and for Studio 5017 seconds (84 minutes). These durations were representative of how often these sequences appeared in the video file.

For each audio clip, the corresponding features were then extracted for experimentation. Subsequently, 75% of the audio clips in each sample were randomly placed in a training group and the remaining 25% assigned to a test group.

6.3.2 Classification Error

	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

Table 6.1: Confusion Matrix

A measure called classification error was used to evaluate the discriminative properties of each feature set. This section formally defines this measure.

It is common to analyse the performance of a classification algorithm using a confusion matrix (see Table 6.1). The columns of the confusion matrix correspond to the predicted class and the rows correspond to the actual class. Table 6.1 is an example of a confusion matrix for a two class problem. There are four possible outcomes in the confusion matrix. The positive class will correspond to the content class focused upon, while the negative class contains all remaining data sequences. The four outcomes are defined as:

- True Positives (TP): is the number of correctly identified data sequences from the positive class,
- True Negatives (TN): is the number of negative examples correctly classified into the negative class,
- False Positives (FP): is the number of negative data samples incorrectly classified into the positive class. This is also known as a *Type I error*,
- False Negatives (FN): is the number of positive examples classified into negative class. This is also known as a *Type II error*.

Classification error can formally be defined as,

$$\textit{Classification Error} = 1 - \frac{(TP+TN)}{(TP+FP+TN+FN)} \tag{6.1}$$

Alternatively, classification accuracy is often used to positively compare the performance of classifiers, and can be defined as,

$$\textit{Classification Accuracy} = \frac{(TP+TN)}{(TP+FP+TN+FN)} \tag{6.2}$$

Classification accuracy is often used as a measure when comparing the final performance of a classification system for audio indexing problems (Roach & Mason (2001), Wang et al. (2000)). This measure is used as a positive measure for evaluating the final performance of the indexing algorithms in this thesis, while classification error is applied for comparing different settings for one indexing algorithm.

6.3.3 Experimental Methodology

To evaluate the feature sets a two-class classification problem was used. The logic behind this decision was to compare the three feature sets across each content class separately, determining if there were any differences. Checking for this type of scenario would provide an insight into future indexing problems or limitations later, as well as help develop a more efficient indexing algorithm.

To set up the two-class problem, the three content classes were focused upon separately. For each content class, the test collection was divided into two, a positive and negative class. The positive class included data sequences from the current content class, while the negative group contained the remaining data. For example, an experiment using the Advert class would have two classes. The Advert sequences would be placed in the positive class, while the Game and Studio sequences would be grouped into the negative class.

After the two classes were formed, the test collection was randomly divided into two: a training and a test sample. 75% of the data sequences from each class were assigned to a training sample, and the remaining 25% for testing. This percentage was decided so as to allow for as much training data as possible for generating the two statistical models. Allowing for as much training data as possible would help estimate the model parameters more accurately.

For each class, two statistical models were then formed separately using training data from the respective classes (positive and negative). Two classifiers, one for each class, were required for labelling the test sample and thus measuring the classification error for each feature set. The GMM was used as a statistical representation for each class. The GMM was favoured over the HMM as it is a popular framework for this type of problem. It has been applied to the fields of video (Roach & Mason (2001)) , music genre detection (Tzanetakis & Cook (2002)), and the classification of radio news (Kemp et al. (2000)), amongst other problems. The GMM was also found to be easier to train and test when compared to the HMM. Given the number of feature combinations for achieving both objectives, and the volume of data, the GMM was considered the best option. Another reason for deciding upon the GMM was that it could be considered to be an example of a one state HMM (Chapter 4).

Using the GMM, the discriminative properties for each feature set were measured across the three content structures separately (Advert, Game and Studio). For each semantic structure, two GMM models were generated for each class (positive and negative) using the training samples. To minimise the influence of the GMM, and measure only the discriminative properties of each feature set, each GMM was fixed to have the same number of mixture components. By fixing the number of modes to be the same, the only changing variable would be the feature set.

Initial parameters were estimated using k-means clustering algorithm (Theodoridis & Koutroumbas (1999)). To limit the effect of unusually good or bad parameter estimations during the training phase, the same GMM was initialised 10 times. For training, the expectation-maximisation algorithm was employed for parameter estimation. After the 10 initialisations, the final GMM that generated the maximum likelihood was selected for testing. The reason for this whole process was to limit the possibility of a sub-optimal GMM. It was not uncommon during GMM generation for the EM to stop on a local maxima.

Once the positive and negative GMM models were formed, the test sample was classified. The data sequences in the test sample were classified using the maximum likelihood criterion. That is, each data sequence was labelled into the GMM class that returns the highest model likelihood score. The classification error is then recorded for each feature set. After the completion of the experiment, the same process was repeated 24 times randomly changing the training and test samples after each complete run. The reasoning for this was to measure the variation in performance between each run, limiting the effect of unusually good or bad experimental runs.

This process was then repeated for the next semantic content structure such as Advert or Game, where the test collection is changed accordingly.

6.3.4 Test for significance

To test for significant improvements between feature sets, the non-parametric equivalent to the ANOVA test was used, the Kruskal-Wallis test (Lehmann (1975)). This is a statistical test to check for differences in multi-group comparisons, employed to test if the mean classification error for each feature set is equal. This test was used over ANOVA to avoid problems such as the assumption of normality. The test is used to determine if there were significant differences between the different feature sets in terms of classification, where significant was p-value$\leq$ 0.05. To locate where the significant differences were between feature sets, a follow up pair-wise comparison was used.

6.4 Experimental Results

There were two aims in the experiment. The first objective was to compare the three feature sets (CC, MFCC and LPCC) across three content classes found in football video (Advert, Game, and Studio), identifying the feature set that minimised classification error. For each feature set 14 coefficients were used. This was a standard number that is considered to contain the majority of the representative information of the audio signal (Rabiner & Juang (1993)). Selecting 14 coefficients for each feature set also minimised bias favouring one feature set over another. For example, a feature set containing more feature dimensions may result in worse performance. As the dimensionality increases so does the number of parameters to estimate precisely. Fixing the number of parameters in each model would hopefully limit the affect of the GMM and hence measure only the discrimination properties of the feature set.

After the completion of this goal, the second objective was to analyse the selected feature set further. The configuration of this feature set was studied, locating the combination of coefficients that would be optimal, given the test collection. The following sections highlight the results and conclusions from the two experiments.

6.4.1 Objective1: Feature Set Comparison

Feature Set	Adverts	Games	Studio
CC	0.4331 $\pm$0.0367	0.4362 $\pm$0.0442	0.3785 $\pm$0.0403
MFCC	0.2347 $\pm$0.0308	0.2348 $\pm$0.0236	0.2979 $\pm$0.0161
LPCC	0.3704 $\pm$0.0128	0.3719 $\pm$0.0125	0.4151 $\pm$0.0154

Table 6.2: Feature selection results. The mean classification error and standard deviation for the 25 runs are displayed from each content class.

The overall results for each content class are displayed in Table 6.2. The table displays the mean classification for the 25 experimental runs and the standard deviation. Analysing the Advert class first, the MFCC feature set was found to have the lowest mean classification error out of the three feature sets. Figure 6.1 presents the results from the 25 runs of the experiment, where the circle is the mean classification error and the bars indicate the standard deviation of the 25 runs. The MFCC feature set ranged from 0.2 to 0.27 classification error, compared to CC with 0.43 mean classification error, and LPCC recording 0.37. MFCC was proved to be significantly better than the other feature sets in terms of classification error (p-value$\leq$ 0.05). A similar result was found for the Game class (Figure 6.2), where the MFCC feature set was shown to be significantly better. For both content classes LPCC was also shown to produce significantly better classification performance than CC.

For the Studio class, the MFCC feature set again significantly surpassed the remaining two feature sets (Figure 6.3). For this class though, the CC feature set displayed lower classification error than the LPCC feature, a different result in comparison to the other two content classes.

The MFCC was proven to be the best representation overall given the test collection, significantly outperforming the other two feature sets across the three content classes in terms of classification error. For two out of the three classes, the LPCC was the second most descriptive representation. A possible reason for the difference in results between the Studio and two other classes was the content. The Studio class contained high quality conversational speech and silence. The other two classes comprised of more sub-structures including speech of differing quality, music, crowd cheering and sound effects. The LPCC feature set was not a robust representation for discriminating between such classes. This could be a factor as to why the CC feature set performed better.

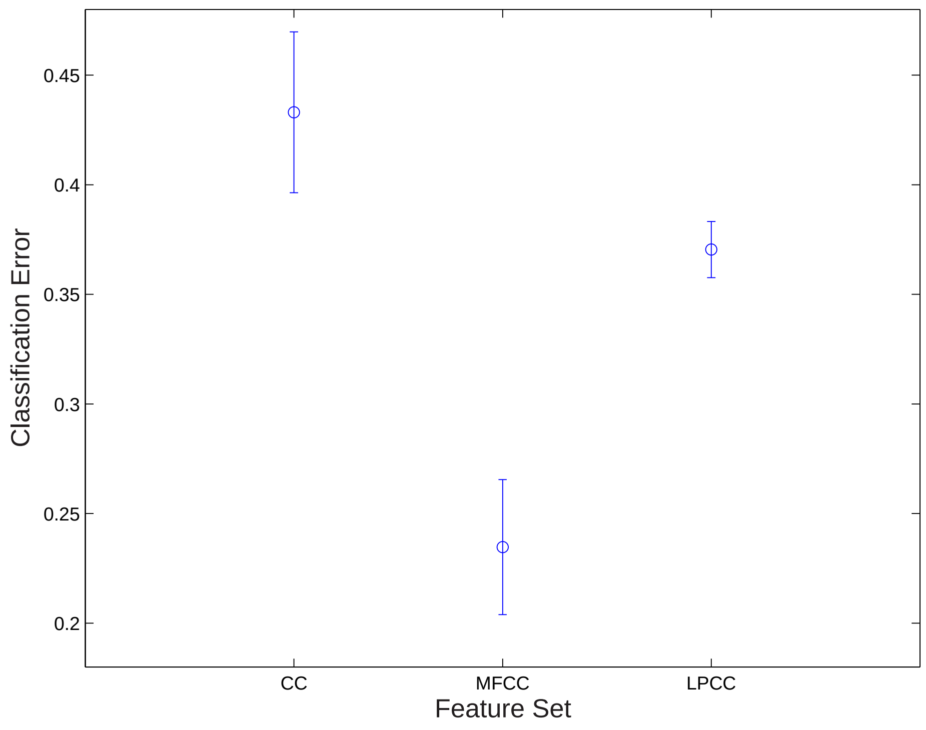


Figure 6.1: Comparison of the three feature sets across the Advert content class.

6.4.2 Objective 2: MFCC Configuration

The MFCC feature was recognised as the feature set containing the best descriptive properties. This feature set was then studied in more depth; comparing a range of settings, such as appending log energy and the MFCC first and second order time-derivatives (the feature set configurations evaluated are summarised in Table 6.3). This task would help assess the value of adding further dimensions into the feature set,

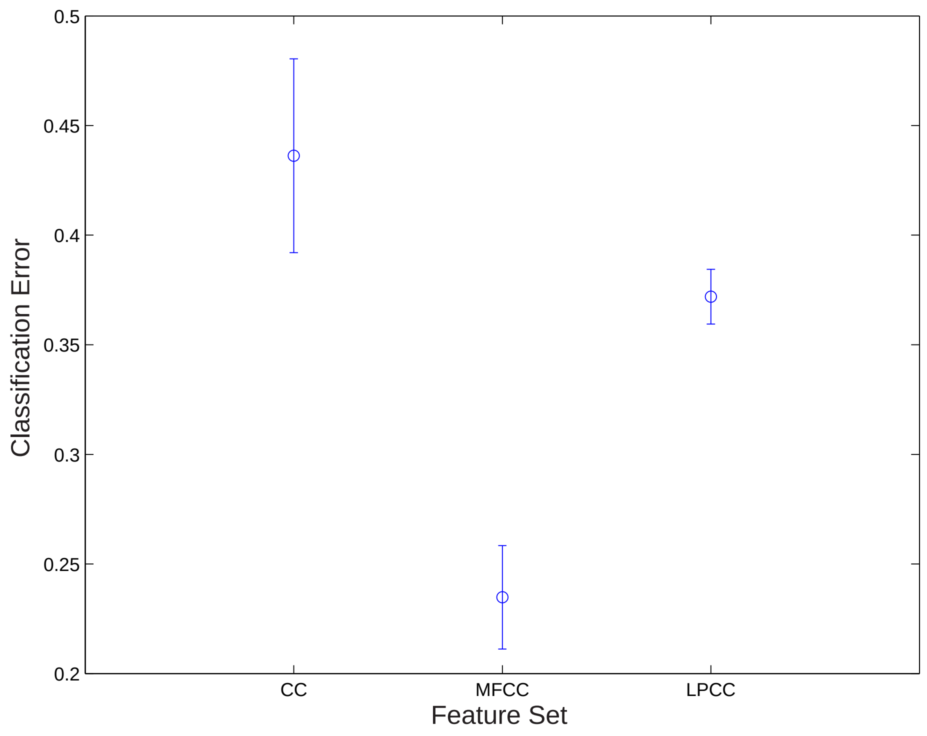


Figure 6.2: Comparison of the three feature sets across the Game content class.

evaluating whether these new coefficients contained sufficient descriptive information given the increase in dimensionality.

All MFCC coefficient combinations were again compared using the same experimental methodology (see Table 6.3). Each combination was evaluated across the three content classes and the classification error was recorded. Any differences were then formally tested for significance.

6.4.2.1 MFCC Configuration Results

Table 6.4 displays the overall results for each feature combination across the three content classes, again highlighting both the mean classification error and standard deviation of the 25 runs. For the Adverts class, adding either log energy (Set03) or the 0^{th}coefficient (Set02), or both (Set04) with the 14 MFCC coefficients improved the classification error (Figure 6.4). A significant improvement was observed in appending the 0^{th}coefficient, and the log energy and 0^{th}coefficient. Adding just the log energy to the 14 MFCC coefficients did not result in a significant gain. The best mean classifi-

Code	Feature Set Configuration	Number of Parameters
Set01	MFCC	14
Set02	MFCC + 0thcoefficient	15
Set03	MFCC + Log Energy	15
Set04	MFCC + Log Energy + 0th	16
Set05	MFCC + Δ (first-order time coefficient)	28
Set06	MFCC + Δ + 0th	29
Set07	MFCC + Δ + Log Energy(s)	29
Set08	MFCC + Δ + $\Delta\Delta$ (second-order time coefficient)	42

Table 6.3: List of the MFCC feature set combinations evaluated. Both the number of coefficients (dimensionality) and corresponding code are displayed.

Feature Set	Adverts	Games	Studio
Set01	0.2347 $\pm$0.0308	0.2348 $\pm$0.0236	0.2979 $\pm$0.0161
Set02	0.1826 $\pm$0.013	0.184 $\pm$0.0199	0.202 $\pm$0.0192
Set03	0.1884 $\pm$0.0172	0.1926 $\pm$0.0199	0.2117 $\pm$0.0139
Set04	0.184 $\pm$0.018	0.1894 $\pm$0.0172	0.2141 $\pm$0.0172
Set05	0.3006 $\pm$0.029	0.2972 $\pm$0.0256	0.2508 $\pm$0.0248
Set06	0.3069 $\pm$0.0236	0.3037 $\pm$0.0339	0.2422 $\pm$0.014
Set07	0.3107 $\pm$0.0309	0.3071 $\pm$0.0248	0.2497 $\pm$0.0166
Set08	0.3477 $\pm$0.0314	0.3512 $\pm$0.0314	0.288 $\pm$0.0282

Table 6.4: MFCC configuration results. The mean classification error and standard deviation for the 25 runs are displayed from each content class.

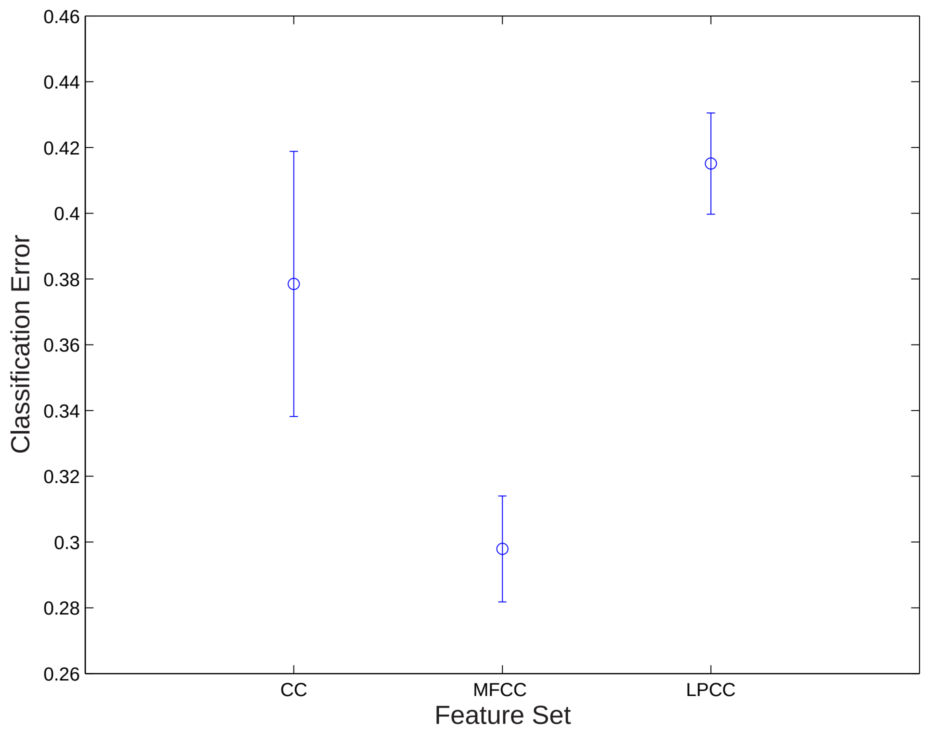


Figure 6.3: Comparison of the three feature sets across the Studio content class.

cation error was recorded by the MFCC and 0thcoefficient feature set (Set02), although the difference was minimal between the three mentioned combinations (Set02-Set04).

A combination of either the first order (Set05) or, first and second order time derivatives (Set08) resulted in a drop in performance, subsequently increasing classification error. Adding the time derivatives was significantly worse than the original 14 MFCC coefficients (Set01).

A similar trend was followed in the Game class (Figure 6.5), where adding either the 0thcoefficient or log energy (or both) into the feature set did significantly improve performance. Appending the time derivative coefficients worsened performance significantly.

This trend of results was also followed for the Studio class (Figure 6.6), where adding the 0thcoefficient, log energy or both significantly improved discrimination over the original 14 MFCC coefficients.

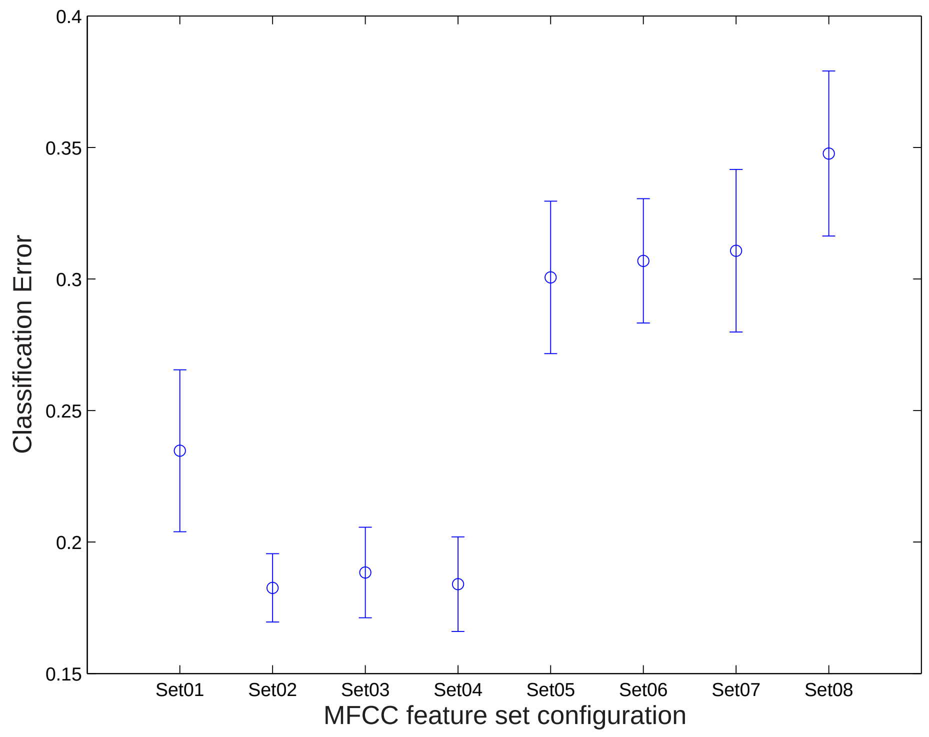


Figure 6.4: Comparison of the MFCC implementations across the Advert content class.

6.4.3 Results Discussion

The MFCC features that were designed specifically for modelling speech were found to be the most robust feature set across all sound classes. For all three classes (Advert, Game and Studio), the MFCC feature set recorded significantly lower classification error than the remaining two feature sets, in other words, it displayed the highest discrimination. For two out of the three content classes the LPCC feature set was the second best descriptive feature set, although for the Studio class the mean classification error significantly increased to 0.61. The CC feature set was the least descriptive feature set for two content classes, but performed significantly better than LPCC for the Studio class.

The three sound classes that the feature sets were evaluated across, all contained many sub-structures. These structures included varying quality of speech, music and other environmental sounds such as crowd reaction. From the experiments the MFCC feature set was a more robust measurement for these varying sub-structures when compared to the other two feature sets. A possible reason for this is that the MFCC feature was

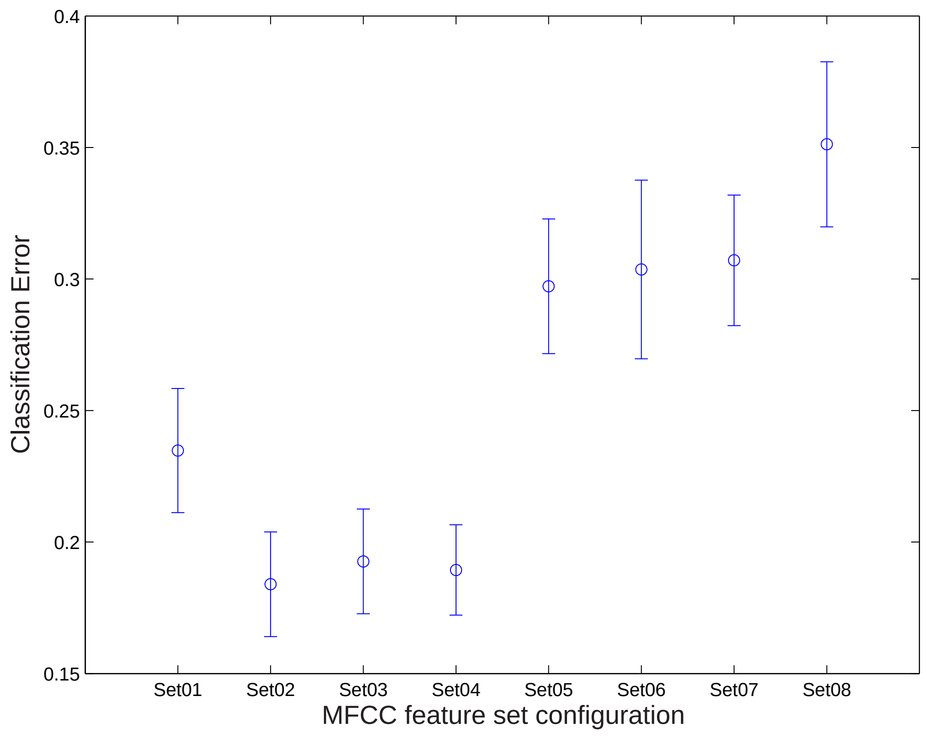


Figure 6.5: Comparison of the MFCC implementations across the Game content class.

designed specifically for representing speech. The Mel-scale was designed to allow for a higher resolution at lower frequencies where speech is found. This process would also suggest that MFCC coefficients contain higher discrimination properties between speech and non-speech sound as well. Another desirable property was that the MFCC coefficients were uncorrelated. This was a result of the DCT transformation during the feature extraction process, which generated uncorrelated coefficients and removed redundancy between the coefficients.

Investigating the MFCC feature set further, it was discovered that appending both log-energy and the 0^{th}MFCC coefficient to the original feature set increased performance. The 0^{th}coefficient is an averaged representation of the energy for all frequency bands. Because the 0^{th}coefficient is not gain independent it is not included in the standard feature set for ASR systems. It is assumed to be unreliable for representing speech (Zheng et al. (2001)). For classes containing both speech and non-speech substructures though, the inclusion of features that measure the energy or activity in the audio track improved discrimination. These coefficients were an important discriminator between content classes. A possible explanation of this is the difference in speech

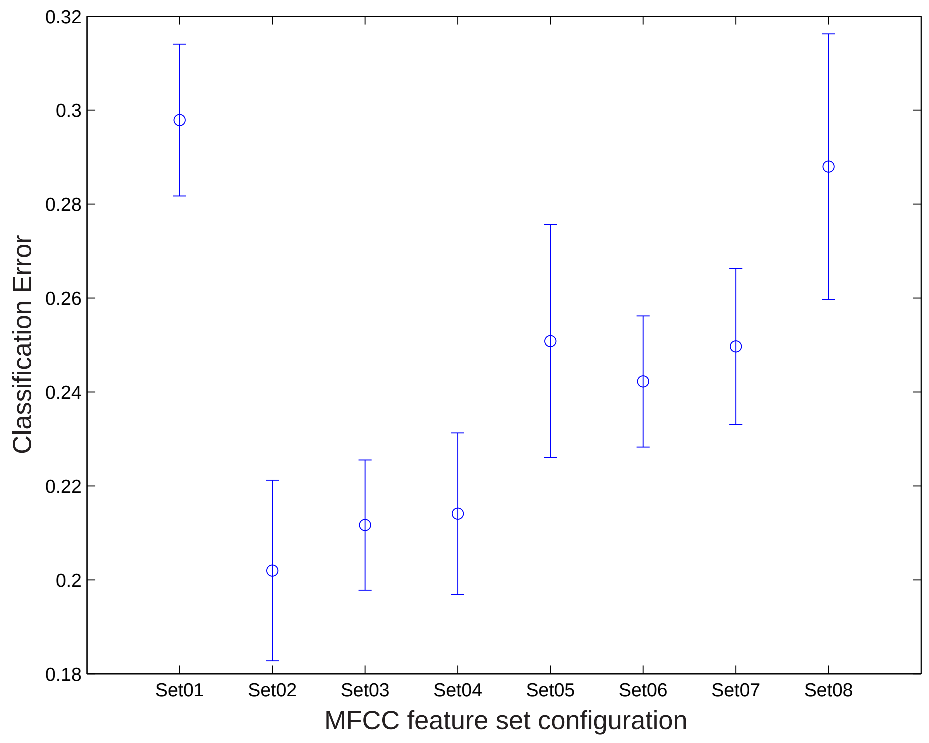


Figure 6.6: Comparison of the MFCC implementations across the Studio content class.

recording quality across the three different content classes. As discussed in Chapter 2, recording quality between Studio and Game sequences differs in bandwidth and background noise, while for Advert sequences the sound quality is further improved by post produced. Including measures of the energy of the audio signal, a further information source is added, improving discrimination.

Alternatively, by appending the time derivatives (Δ MFCC and $\Delta\Delta$ MFCC), a standard inclusion for ASR feature sets that indicate the degree of spectral change in the signal, did not improve classification error. In fact for two out of the three content classes, adding the further dimensions only resulted in a significant drop in accuracy compared to the original 14 MFCC coefficients. A possible explanation for this outcome was that all the important discriminative information was contained in the feature set containing the 14 MFCC coefficients, log energy and the 0^{th}coefficient. Appending further coefficients did not result in further discriminative information but just an increase in dimensionality. From increasing the feature set dimensionality, the number of parameters to be estimated also increased, resulting in poor parameter estimation during model training.

To summarise these results, both experiments highlighted the importance of formally analysing what features are optimal for different audio data, instead of applying standard feature sets applied for different research problems such as ASR. A consistent trend was found in the results, which indicated that the same feature settings were optimal across all three content classes. This provided sufficient evidence to suggest that the same feature set could be used for each pattern class, without significant information loss and thus drop in performance.

The optimal feature set was the MFCC, which provided the highest discrimination between content classes. Studying MFCC further it was found that appending log energy and the 0^{th}coefficient to the 14 MFCC coefficients was the best configuration. There was no significant improvement from applying log energy over 0^{th}coefficient, so both coefficients were selected for the final feature set to be used in later experiments.

Moreover, for identifying audio content in the audio stream of football video, a feature set containing 14 MFCC coefficients plus log energy and the 0^{th}coefficient is employed. The reason why log energy was also appended, was that this feature set (Set04) was no significantly worse, and log energy is considered an important feature in a variety of systems, such as ASR. The feature is a measure of the activity of the entire signal.

6.5 Conclusions

In this chapter, three audio feature sets were compared across a data set generated from football video. The MFCC feature set was found to contain higher discrimination properties than both LPCC and CC feature sets, across three content classes. Examining the MFCC feature set further, it was discovered that appending log energy and the 0^{th}coefficient improved the classification error of the MFCC feature set. However, adding the time derivatives (Δ MFCC and $\Delta\Delta$ MFCC), a standard inclusion for ASR feature sets, which indicate the degree of spectral change in the signal, did not improve classification error. This result highlighted the importance of investigating a range of feature set combinations when faced with a new test collection. Results from one research problem did not necessarily generalise across to a new problem. Hence, for recognising audio content in the audio stream of football video, a feature set containing 14 MFCC coefficients plus log energy and the 0^{th}coefficient was decided upon.

Chapter 7

Automatic Content Analysis

7.1 Introduction

A test collection was generated for developing and evaluating the algorithms introduced in this thesis. This test collection was annotated manually using information gained from inspection and domain knowledge. In this chapter, this test collection is analysed by applying automatic techniques to investigate the underlying data structures, contained in the audio track of football video.

To do so, a HMM is applied to model the entire audio stream. However, for managing such large volumes of data to model the entire audio track, the data is required to be compressed, with minimal information loss. To reduce the volume of data an experiment is planned for identifying an optimal length for dividing the audio sequence into. Once the audio track is divided into smaller units, a point estimate can be extracted from each window as an efficient representation, thus reducing the data.

Once the data is reduced, a robust HMM is generated for the audio track, with the number of hidden states in this model estimated using a measure of the fitness. This HMM representation is then applied for labelling a further test collection, observing the semantic groupings modelled. The content recognised by the HMM is then compared to previous domain knowledge, gathered through inspection, recording any similar trends or new discoveries found from thus study.

The remainder of this chapter is structured as follows. Section 7.2 provides an insight into the chapter motivation, by reviewing related research and describing the reasoning behind automatic content analysis, in greater detail. The steps required for automatic content analysis are then described. First the entire audio track is compressed to allow for the management of a large volume of data, with minimal information loss, in section 7.3. The steps for generating a HMM representation of the audio track are then presented in section 7.4. This HMM representation is applied for content analysis, in section 7.5, highlighting any interesting trends or discoveries found in the data. In this section, the video model, defined previously, is re-examined. Finally, the chapter is concluded in section 7.6.

7.2 Motivation

Automatic speech recognition (ASR) systems are developed to transcribe the spoken word accurately, but given the vast number of possible words that are contained in a vocabulary, the problem of recognising individual phrases is simplified. Speech has a modular structure that can be exploited to limit the number of variables required, when modelling words (Reyes-Gomez & Ellis (2003)). Speech is divided into smaller units called phones, a physical sound produced when a phoneme is articulated[1]. When compared to a large vocabulary of words, that is forever evolving, there are only a finite number of possible phones that can be pronounced. Therefore, ASR systems model each possible phone, instead of a large vocabulary of words, which can then be recognised by dynamically searching through patterns of classified phones (Ney & Ortmanns (2000)).

A phone is typically represented by a single HMM, generated by a large collection of training samples, which contain as many variations of the way the phone can be uttered as possible. For example, a typical training sample would contain the same phone uttered by different people with varying timing and emotion. By doing so, the HMM can be trained to encapsulate many eventualities. The development of such training data is assisted by prior domain knowledge, where there are only a finite list

[1]A phoneme can be defined as the smallest contrastive unit in the sound system of a language (Rabiner & Juang (1993)). A phoneme is a unit that serves to distinguish between meanings of words and can be pronounced in one or more ways. However, a phone is pronounced in a defined way and corresponds to a specific unit only.

of phones. Provided with this knowledge, a comprehensive training collection can be generated, allowing researchers to fine tune the complex processes involved in ASR system development.

Alternatively, for modelling non-speech audio typically found in football video, basic units such as a phone do not exist. Therefore, the problem of indexing can not be easily simplified by modularising content into smaller units. An alternative solution is to group similar types of sounds together into an individual class, such as speech, music and crowd cheering. A statistical representation for each class can then be generated for future video classification. One method of grouping similar sounds is to investigate examples in the domain, through manual inspection. This strategy was followed in Chapter 3.3.

Another problem with defining audio classes for future classification, is that not all content found the audio track will be known, especially through manual inspection alone. This can cause accuracy problems when classifying future videos. Thus, complimentary methods can be adopted, applying automatic techniques to investigate the data structures found in the audio stream. An exploratory investigation into the data can provide an insight and understanding into the underlying processes, that define the audio track, and is beneficial for model and indexing algorithm development. Automatic content analysis can be used to locate new data structures not picked up through the manual observation.

Through automatic content analysis, the same structures, identified by manual inspection, can also be compared with those patterns discovered automatically. This is beneficial for assessing what logical patterns are discovered, as well as verifying the suitability of a particular statistical representation, such as the HMM. In this chapter, it is believed that from both manual and automatic content analysis, more underlying structures will be discovered, with both strategies being complimentary to one another. As well as, providing a comprehensive study of content that can be used for further training data generation, domain understanding and algorithm development.

In the following sections, related research into this problem is reviewed, before proposing a strategy for automatic content analysis.

7.2.1 Related Work

The main objective for automatic content analysis is to group similar sounding structures, found in the audio track, together. In a sense, this is a clustering task, grouping perceptually similar sounds automatically. There are many clustering methods available, such as hierarchical, k-means and nearest neighbours clustering (Theodoridis & Koutroumbas (1999)), however, these algorithms are applied mainly for static data sets. For clustering temporal data, the HMM can be applied (Li & Biswas (2000) and Smyth (1997)). For model-based clustering involving the HMM, data items are distributed among the states, where each data item influences only the component(s) with whom it is associated with. The states in the HMM are assumed to represent a specific semantic content.

Li & Biswas (2000) apply HMM to cluster synthetically generated temporal data. It was shown through experimentation that a HMM can be trained, without supervision, to cluster the temporal data successfully. A model selection criterion was applied for identifying the optimal number of clusters in the data set. The overall clustering strategy discovered the best HMM representation, for the entire data collection, using model selection. To do so, a single HMM was trained on an unknown data set, iteratively adding hidden states until the model selection criterion was reached. In the final model, an individual cluster was assumed to correspond to a single hidden state in the HMM. From evaluating this approach using synthetically generated data, it was found that the underlying process that created the data were discovered automatically by the HMM.

Smyth (1997) also clustered temporal data using HMM, although the application of the HMM for clustering differed from that of Li & Biswas (2000). Instead of a single HMM generated for the entire data collection, a cluster was considered to be an individual HMM. Singleton HMM data representations were clustered using a hierarchical agglomerative grouping strategy, where for each new data sequence, a HMM was generated and then the log likelihood distance was measured between the current and existing HMM-based clusters. Similar HMM models, in terms of log likelihood distance, were then merged. The algorithm was again evaluated using synthetically generated data, indicating accurate unsupervised clustering of similar content.

7.2.2 Proposal

In this thesis, it was believed that an approach similar to Li & Biswas (2000) would be more appropriate for the task of automatic content analysis. The reasoning behind this assumption, was that the audio track for each video file was considered to be one complete data sequence. Therefore, modelling the entire football audio track using a HMM, would help group similar content together. For example, a HMM models the underlying processes that generate an audio stream by grouping similar structures using a series of interconnected hidden states. It is assumed that each hidden state in the HMM would correspond to a logical semantic group.

Applying an alternate strategy such as that proposed in Smyth (1997), would require dividing the audio track up into smaller segments. Since little was known about the underlying data structures found in the audio track, the task of logically dividing sequences up for clustering, was not considered suitable.

The remainder of this chapter describes the process for applying a HMM for content analysis, discussing the issues involving the implementation and the overall methodology. The results of the content analysis experiment are compared to the findings in Chapter 3.

7.3 Implementation Issues and Data Pre-processing

The main objective of this chapter, was to automatically analyse the content of the audio track for football video files. To discover content automatically, the HMM was chosen to model the underlying data processes found in the audio track. However, modelling the entire audio track for each file would be a huge burden on system resources.

To illustrate, the feature set employed throughout this thesis contained 14 MFCC coefficients, the 0^{th}coefficient and log energy. This meant the feature space contained 16 dimensions. When the audio track was converted into this feature space, 85 feature measurements were calculated per second, per dimension, thus, for one second of audio, there was a representative feature vector of size 16×85. An entire broadcast could span two and a half hours (150 minutes), hence a data sequence can equate to

9000 seconds. This meant managing huge data sequences of dimension 16×765000.

Given this volume of data, preparing the test collection into more a manageable form was required. This would allow for each data sequence to be processed efficiently. One method of minimising this large data was to divide the audio stream into equal sized windows and then extract a single point estimate per feature. For content analysis, each time unit would be clustered into a hidden state in the HMM, with a hidden state representing a semantic content cluster. For this task, there should be sufficient descriptive information contained in each time unit. It was important that these segments contained enough discriminative information for recognising content. An experiment was performed for formally analysing what window length would encapsulate sufficient descriptive information.

After this optimal unit length is found, it can be represented efficiently, by using a point estimate such as the mean. Estimating the mean value for each time step, is to assume short term stationary. In other words, all observations within the audio clip will come from a stationary process with finite moments, the mean estimate being the first moment. The problem faced is at what scale is the process being estimated stationary? For example, too wide a window, and the mean could be estimating a non-stationary process. Too short a window, can result in a poor estimate. The concern is whether the smaller number of sample means obtained when using a longer window, will be a problem in estimating the parameters of a future model i.e. the HMM.

This section examines the importance of observation length and its correlation with classification error. The aim is to find a suitable time unit, which would contain enough descriptive information for accurate classification. After identifying a suitable window length, appropriate sufficient statistics can be calculated, allowing for a more efficient data representation.

7.3.1 Related Research

Similar investigations into the problem of window size for audio, included Tzanetakis & Cook (2002), who evaluated a range of observation sizes for classifying musical sequences into genre. A set of GMM classifiers were employed for classifying new musical sequences into the correct genre class, where these audio clips were parameterised by a MFCC feature set. A range of observation lengths were evaluated using classification accuracy as a measure. It was discovered that once the audio clip length was increased to 1 second, there was no significant increase in accuracy. A shorter clip size than 0.5 seconds was found to decrease performance, and a length over 2 seconds, also resulted in a drop in accuracy. An audio clip length of a second was discovered to contain sufficient statistical information for classifying musical sequences. In a similar experiment, Marques & Moreno (1999) compared two audio clip lengths (0.2 seconds and 2 seconds) for classifying sequences into the correct musical instrument class, by using a set of GMM classifiers. It was discovered that a length of 0.2 seconds provided more accurate classification performance.

Spina (2000) also evaluated the correlation between observation length and classification accuracy, this time for the problem of radio news classification. New audio sequences were classified into seven categories including speech, music and silence. The optimal length for the audio clips was discovered to be approximately 0.5 seconds. A longer unit length provided no further information gain. In fact, increasing the length beyond one second, began to result in a drop in accuracy.

7.3.2 Observation Length Experiment Methodology

To formally identify a suitable window length for classification, the Gaussian Mixture Model (GMM) was applied. From reviewing similar research into this problem, the GMM was shown to be an effective classifier for this problem. Another reason for selecting the GMM, was that in this experiment it was assumed that the data contained in each window was identically and independently distributed, and from a stationary Gaussian process. Hence, the GMM was a more appropriate classifier than the HMM, as it does not model the temporal variation within a sequence.

The data sequences in the test collection were divided into varying lengths, ranging from $\{0.1, \ldots, 1.4\}$ seconds wide. For example, when comparing the window length of 0.1 seconds, all data samples in the test collection would be of the same length. The GMMs were both trained and tested using sequences of this length. Classification error was then used as a measure for comparing the various window lengths, where the same test collection was used for these experiments, as in Chapter 6.3.1. Again, 75%

of the data set was randomly assigned for training and 25% for testing the models. However, for this experiment, a single GMM classifier was generated for each content class: Advert, Studio and Game.

Given the length of each audio clip, it was assumed that a window would contain only one of many sub-structures found in each content class. Therefore, a number of mixture components were required for each content class, each representing content. The number of mixture components in the GMM were fixed to be the same, for each content class, for all window lengths. It was believed that this would minimise the effect of other variables in the experiment, and measure the effect of changing the window length only. The overall classification error for labelling the test collection into the correct classes was then recorded, for each window length.

For each content class, a GMM was generated using labelled training samples from the appropriate class. Each GMM was again initialised 10 times using the k-means algorithm and then trained using EM (see Chapter 4.3.2). The best initialisation, in terms of out of sample likelihood, were then used for testing. For each window length, this entire experimental process was repeated 25 times, randomly changing the training and test samples after each complete run.

7.3.2.1 Experiment Summary

The steps taken for this experiment are summarised below,

1. The test collection was randomly divided into two samples. 75% of the data into a training sample and the remaining 25 % into a test sample.

2. For both the training and test samples, the data sequences were then divided into smaller units ranging from 0.1,. . . ,1.4. For example, the 0.1 group would only contain samples of 0.1 seconds in length.

3. Separately for the three content classes (Advert, Game and Studio), a single GMM was generated following these steps:

   (a) A GMM was initialised using k-means and then trained using the EM algorithm.

   (b) The log-likelihood value was recorded for this initialisation.

(c) This process was repeated a further 9 times.

(d) The 10 log likelihood values were compared and the GMM with the maximum value, is selected for the classification phase.

4. The test sample is then classified using the three GMM classifiers. The maximum likelihood criteria is applied to determine the class of a data sequence.
5. The classification error for each window length is recorded.

The entire process was run 25 times.

7.3.3 Window Length Results

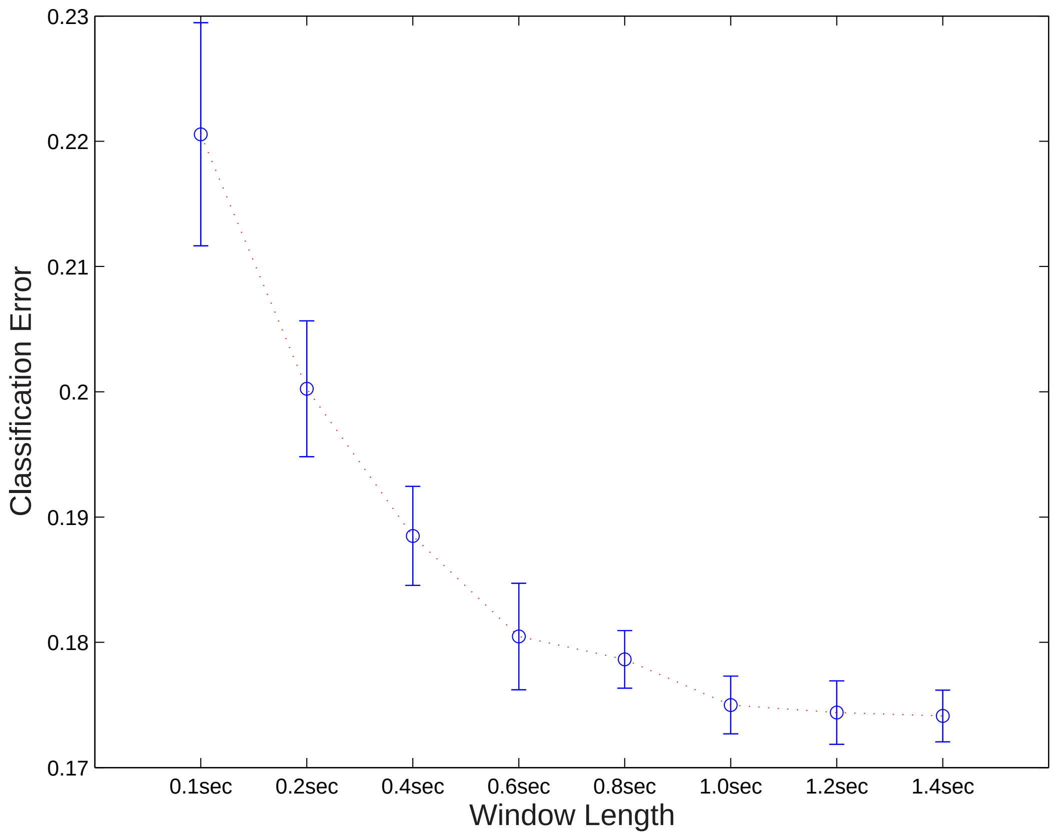


Figure 7.1: Window Length versus classification error results.

Figure 7.1 displays the results. The mean classification error is represented by the circle for each window length, and the bars represent the standard deviation. It was discovered that by increasing the window length from 0.1 second up to 1 second resulted in a drop in classification error. Up until 0.4 seconds this was found to be a significant improvement (using the Kruskal-Wallis significance test, Chapter 6.3.4). However, increasing the window length beyond 1 second, did not result in further improvement, where there appeared to be a convergence in classification error. This would suggest that there was not a significant gain in discriminative information after lengthening the window further.

Another finding discovered from the experiment, was that as the length was increased, the variation in classification error decreased. This was evidence to suggest that the longer the window length, a degree of stability was found in classification. Again, this variation stabilised around 0.8 seconds.

From these results, it was assumed that a window length between 0.6 to 1.4 seconds in length, would be ‘optimal’. For simplicity, a final window length of 1 second was chosen, as it was believed that a time scale of 1 second would be easier to manage than 0.8 or 1.2 seconds.

7.3.4 Data Compression

The complete audio track for each video file, in the test collection, was parameterised, and then divided up into one second windows, calculating the mean feature vector for each window. It was assumed that all feature measurements in a window was generated by the same Gaussian process, with the mean used as a point estimate. By estimating the mean value at each time unit, short term stationarity was assumed in the window. In other words, all data points within the window would come from a stationary process with finite moments. Calculating the mean feature for each one second window, provided a point estimate as a representation. The final result was a vastly reduced data sequences, with minimal information loss.

Figure 7.2 is an example of an original parameterised sequence, with the smoothed representation below. A number of known changes in content structure were marked on the video sequence. From analysing such sequences, tt was believed that the reduced audio sequence was an efficient representation, with minimal information loss. It can be seen in the figure, that the changes in structures also appear in the reduced audio sequence.

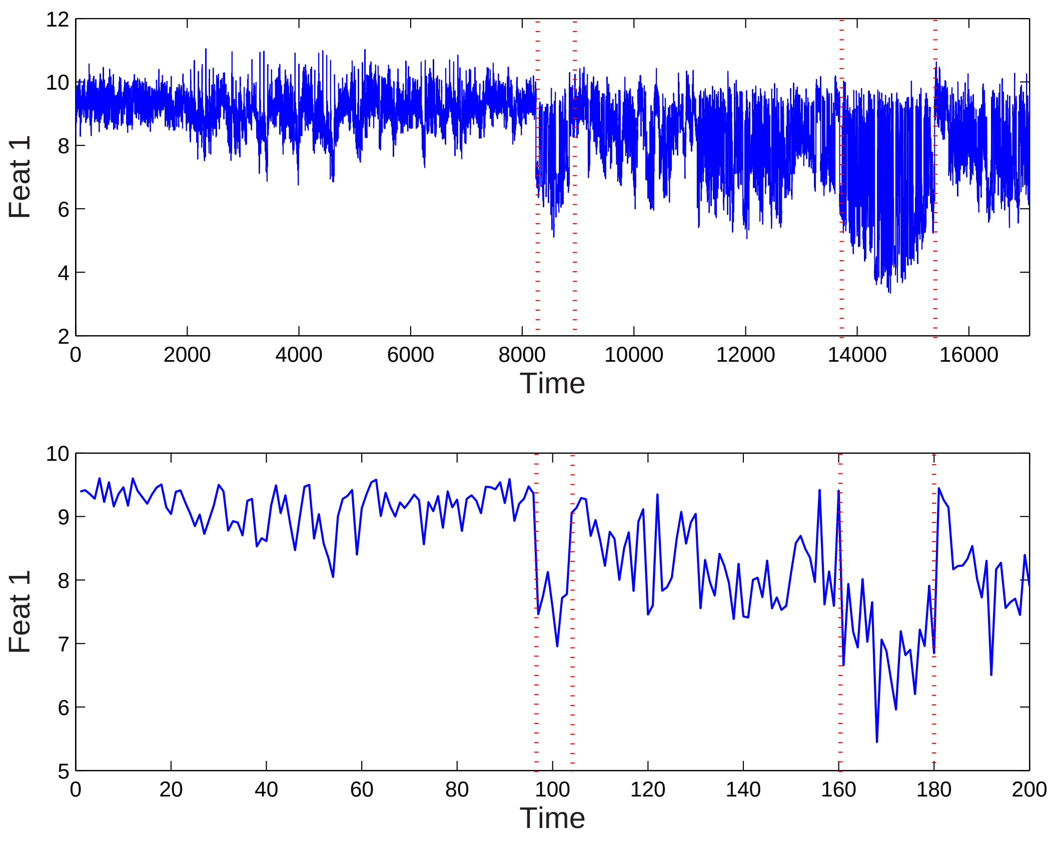


Figure 7.2: Reducing the data.

7.4 Generating a HMM model

The methodology for applying HMM to the problem of automatic content analysis, is introduced in this section. The goal is to model the underlying data structure of the audio track for football video files. By doing so, it is believed that both known and new content will be identified, assisting the indexing algorithm development. The reasoning behind this assumption is that the HMM has been proven to contain self organising properties proven (Li & Biswas (2000), Smyth (1997)). In other words, it is believed that the HMM will group similar semantic content (in terms of sound) into the same states. From generating a robust HMM representation of the audio track, new sequences can be labelled, for recognising structure, assuming that each state in the HMM will correspond to a semantic component found in football video. Analysing the typical content grouped into each state, through observation, and by comparing the results to the manually annotated meta-data (see Chapter 3.3), this assumption can be verified.

To achieve this goal, a non-trivial problem had to be solved. That is the problem of ‘optimal’ state selection, where a robust HMM is required for representing the entire audio track. This is the first issue that is addressed before content analysis. A recap of the HMM is presented below, defining the HMM used for the experiment. The problem, and suggested solution, for HMM state estimation is then discussed, finally presenting the experimental results.

7.4.1 HMM structure

Since little is known of the underlying data structures of the audio track, an ergodic HMM is selected for the experiment. An ergodic HMM is considered more appropriate than other HMM topologies such as the Bakis HMM, as no constraints are placed on the transition matrix A. This property is believed to be advantageous, making no prior assumption about the temporal relationship between content.

To improve the accuracy of many (speech) applications, a mixture of Gaussians are applied to model each hidden state (Rabiner & Juang (1993)). However, for this problem a single Gaussian per state is used. The focus of the experiment is to find an ‘optimal’ number of states that would efficiently represent the many sub-structures contained in the audio track. The reasoning behind applying multiple mixtures for each state in a HMM, is to model the variability found within each sub-structure. For example, in speech data the variability can refer to different voices or emotion. By applying a singular Gaussian PDF, it is believed that large variances within a sub-structure would be represented by a different state. States with similar semantic content can then be identified later (through inspection). The goal is not to model accuracy for future classification, but to use the HMM as a guide for discovering structure. Applying a mixture of densities per state could work at odds with this assumption (of a single semantic content per state). Multiple mixtures could allow for unrelated content to be described by the same state.

Another reason behind this decision, is to limit the number of parameters to be estimated in the HMM. Overfitting can also occur when applying Gaussian mixtures, especially when a large number of parameters are included in the model. For these reasons, a single multivariate Gaussian distribution is used to model each state of an ergodic HMM.

7.4.2 Number of hidden Markov state selection

Selecting an ‘optimal’ number of states is important, where each state in the HMM is assumed to represent a sematic content found in the audio track. Estimating the number of states in the HMM can be seen as the task of approximating the number of content (sub-)structures. It is assumed that a robust HMM, which fits the data well, can enable these structures to be identified. A measure called predictive likelihood is used to determine the number of states in the HMM, which is the probability of a new audio sequence being generated by a model.

Using the same notation as Rabiner & Juang (1993), the predictive likelihood is defined. Let $O = \{O^i : i = 1,\dots,K\}$ be the test set of K acoustic observation vectors, where $O^i = \{o^i_t : t = 1,\dots,T\}$ is a smoothed representation of an audio track. t is one time step in the acoustic vector O^i, and o^i_t is the mean feature vector for the time step t of audio sequence i.

$\Lambda = \{\lambda_c : c = 1,\dots,N\}$ is the set of candidate models to be evaluated, where N is the total number of models and λ_c is the current HMM. From this pool the final HMM representation will be selected by using the predictive likelihood as a measure. Thus, the predictive likelihood for model λ_c is defined as:

$$L_{\lambda_c}(O) = \sum_{i=1,\dots,K} \log L(O^i|\hat{\lambda}_c) \tag{7.1}$$

where $L(O^i|\hat{\lambda}_c)$ is the likelihood that model $\hat{\lambda}_c$ generated the acoustic observation sequence O^i. To calculate this measure, for the HMM, a dynamic programming algorithm called the Viterbi algorithm is employed (Rabiner & Juang (1993)).

A rise in the predictive likelihood score can be related to an increase in the model fit, or an approximation of how well the HMM represents the audio sequence, O. Starting with a simple model, more parameters (states) are added until the predictive likelihood converges. For the HMM, a stopping criterion is often applied for selecting the optimal number of states (Chen & Gopinath (1999), Li & Biswas (2000)). The reason for a stopping criterion is that, as the number of states in the HMM increases, so does the predictive likelihood score. Adding further hidden states will always enhance model fit, resulting in an increase in the predictive likelihood score $L_{\lambda_c}(O)$, until each data sequence in the training set is modelled by its own unique zero-variance state (Li & Biswas (2000)).

For this experiment, a stopping criterion was not required to determine when the HMM began to overfit the data. Instead, the relationship between predictive likelihood and state number was observed graphically, for estimating HMM size. A trial experiment was performed on synthetic data to examine this relationship. The synthetic data set was generated from a HMM with a known of states. When applying a new HMM to the data set, the pattern in predictive likelihood was observed for adding states to the model. The trend between HMM state number and predictive likelihood was noted, especially when the known number of states in the model was reached. It was assumed that this trend would be repeated for the test collection of audio sequences generated from football video.

7.4.3 Experiment

A development set of 36 video files was used to train the HMM model, and 12 labelled video files set aside for testing. The 12 files were manually annotated, labelling known content (see Appendix D for a description of these labels). A set of candidate HMM models were then generated using the training data, with states ranging from 1 up to 30. That is, the set of models $\Lambda = \{\lambda_c : c = 1, \ldots, N\}$, where c corresponded to the number of states in the HMM λ_c, and $N = 30$.

25% of the development data set was randomly held out of the training set in order to test the 'fit' of the model on the remaining data. The parameters in the HMM were initialised by the k-means clustering algorithm and then re-estimated using the Baum-Welch iterative training algorithm, until the model converged (Rabiner & Juang (1993)). The model fit was then measured using the predictive likelihood score. This overall experiment was repeated 15 times, randomly changing both samples after each complete run. This measured the variance of the HMM generation process, and limiting the effect of poor model estimation or peculiarities in the experiment, which may skew results.

A summary of the experiment is presented below.

1. Randomly divide the development data into two samples, a training O_a and test O_b sample e.g $O_a^i = \{i = 1, \ldots, 27\}$ and $O_b^j = \{j = 1, \ldots, 9\}$.
2. Train 30 HMM models, $\Lambda = \{\lambda_c : c = 1, \ldots, 30\}$, using the training sample O_a.
3. Calculate and record the predictive likelihood score for each model λ_c using the test sample e.g.

$$L_{\lambda_c}(O_b) = \sum_{j=1,\ldots,9} \log L(O_b^j | \hat{\lambda}_c)$$

This entire process was run 15 times.

7.4.4 Results

First the results using the synthetic data are presented, and then the findings of the overall test collection experiment.

7.4.4.1 Synthetic Data

The approach was first evaluated using ‘toy’ data was generated data from a 15 state HMM[2]. The main reason for this experiment, was to use it as a guide for determining the optimal number of states for the test collection containing football audio sequences. Figure 7.3 presents the results from the experiment. The solid line is the mean predictive likelihood for the 15 experimental runs[3], and the dotted line indicates when the 15th state was added to the HMM.

From the figure, it can be shown that there was a sudden increase in predictive likelihood, as hidden states were added to the HMM. This increase was particulary sharp between 1 and 10 states. The predictive likelihood then began to converge after 11 to 15 states were added. After the 15th state, the HMM began to overfit the data, with the predictive likelihood score only increasing gradually. Adding further states, did not result in further information encapsulated in the model.

[2]See Chapter 4.4.6 for further details on synthetic data.
[3]See Appendix F, Figure F.1 for a plot containing both the mean and standard deviation.

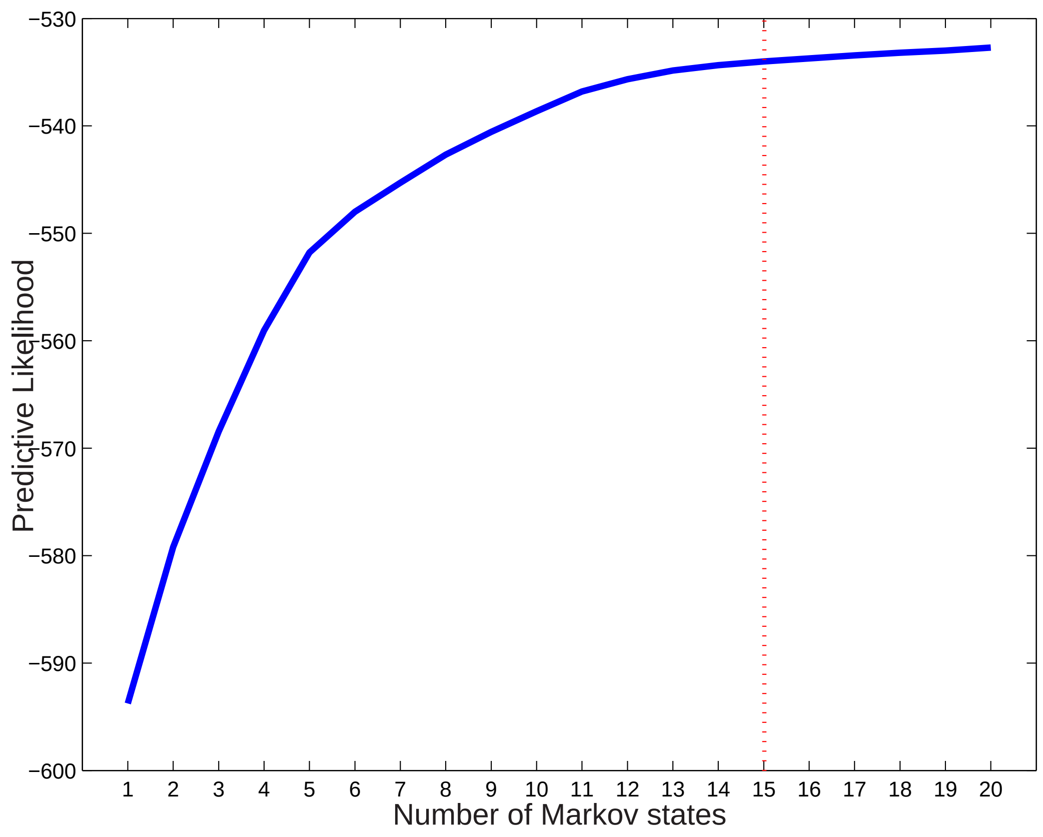


Figure 7.3: Finding the optimal number of states on the synthetic data. The blue line is the mean. The red dotted line indicates the 15th state added to the HMM.

Using this trial experiment as a guide, it was thought that a similar trend would be followed for the HMM representing the football audio test collection. Once the predictive likelihood score began to ‘level off‘, no further states would be added, and the HMM at this point would be selected.

7.4.4.2 Football Data

The same approach was then repeated for the test collection containing the smoothed football sequences. For this experiment, 30 HMM models, with state ranging from 1 up to 30 were implemented. Figure 7.4 presents the results of this experiment[4]. The mean predictive likelihood score again increased rapidly before levelling off after a number of states were added to the model. The HMM began to converge at approximately

[4]See Appendix F, Figure F.2 for a plot containing both the mean and standard deviation.

20 to 25 states. Continuing to add further states only produced a small improvement in predictive likelihood, indicating the model was overfitting the data.

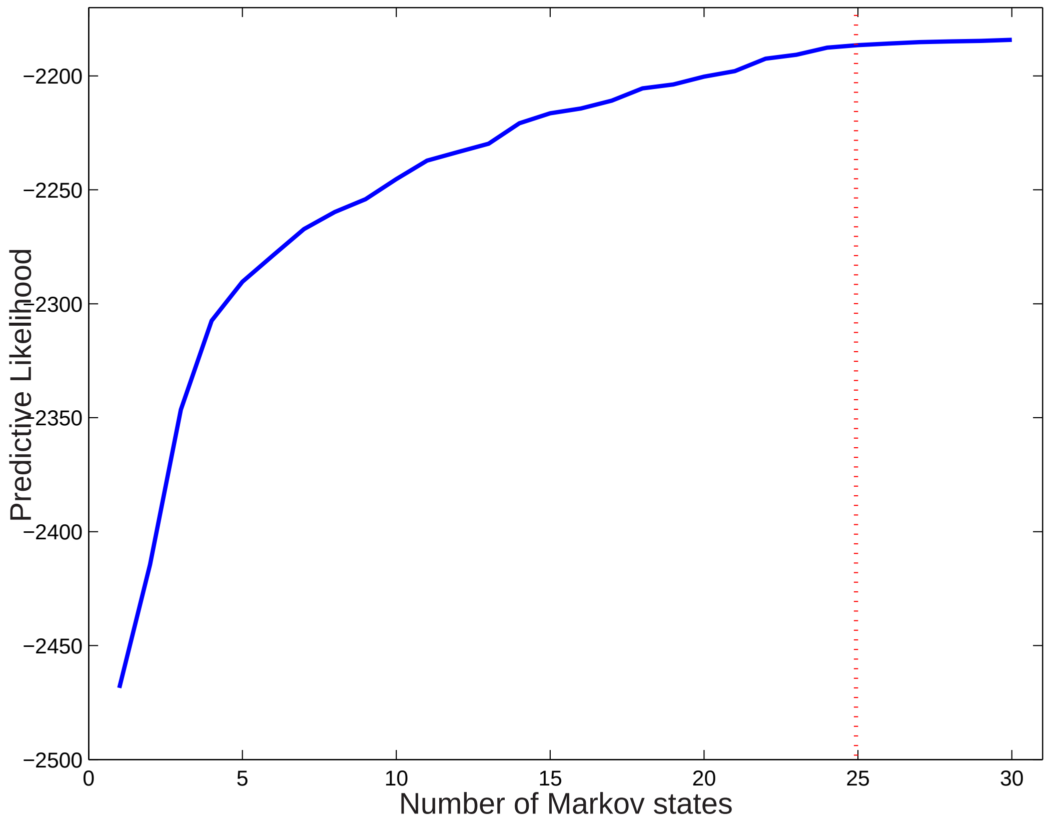


Figure 7.4: Finding the optimal number of states on the real data. The blue line is the mean. The red dotted line indicates the 25^{th} state added to the HMM.

The 25 state HMM was decided as the ‘optimal’ model for this data set. However, this estimate could be considered subjective, and is only an approximation of the ‘optimal’ solution. A HMM with states ranging from anywhere between 20 to 30 could be considered ‘optimal’. 25 was in the middle of this range and was thought of as a good starting point for discovering what patterns each state represented. This issue did highlight that an efficient method for selecting the optimal number of states, was required. Chapter 8 addressed this problem further.

7.5 Automatic Content Analysis Findings

After selecting an ‘optimal’ HMM representation of the audio track, this model was then applied for analysing the underlying data structures. This section presents the methodology and findings from this study.

7.5.1 Methodology

The HMM representation for the entire audio track was generated using the development data set. This same HMM was then used to label the 12 video files that were left out of the model generation. These 12 video files had been manually annotated previously (see Appendix D for further details).

To label the 12 video sequences using the generated HMM representation, the Viterbi algorithm was applied (Rabiner & Juang (1993)). This is a dynamic programming algorithm that assigns each time point in the audio sequence to a state. This process, essentially labels each one second time step, which is assumed to represent a logical grouping found in football audio. Figure 7.7 is an example of an audio sequence that has been assigned a state label at every time step. The top plot is the manually annotated sequence, and below is the HMM labelled version of the same sequence. In the figure, each time point in the sequence is assigned to one of the 25 hidden states of the HMM

Provided with both the HMM and manually labelled sequences, each video file was then inspected carefully, comparing the findings from both tasks. To examine the output from both manual and automatic content analysis, for each audio sequence, a number of steps were taken:

- For all 12 audio sequences, the original audio file was segmented into the one second windows, and then saved as ‘WAV’ files[5]. These audio files were then grouped according to the Hidden state label that each was assigned to. This process created 25 audio files each representing one state in the HMM. All of these audio file were then carefully studied, and the contents found in each file were recorded. Appendix F.2 presents the findings for two sequences.

[5]See Chapter B for an explanation of ‘WAV’.

- The distribution of time steps assigned to each state were also calculated (Figure 7.5).
- The distribution of each one second time step, according to the state label and the manually annotated labels, were also calculated. Figure 7.6 displays the results for one audio sequence.
- The HMM labelled sequences were also compared to the manually annotated labels, checking for similar trends (see Appendix F). Figure 7.7 is an example of an audio sequence labelled both manually and by the HMM.

7.5.2 Findings

Figure 7.5 displays the typical distribution of each one second time step across the 25 states in the HMM model, for one complete football sequence. State 7 contained the highest frequency of audio clips, while state 9 had the lowest. It was discovered that the main content grouped into state 7 was speech from Game sequences, with little to no background environmental sound. This finding was discovered by analysing the audio clips that were generated for each state. This appeared to be a logical result, as the main content structures found in football were these Game sequences (see Chapter 3.3). The commentary track in these sequences is predominant, and generally the crowd reaction throughout a match is low.

State 12 corresponded to those clips that did not contain speech in Game sequences, with little background sound. This was again a reasonable result, where state 12 was assumed to represent breaks in the commentary track found throughout the match. During these breaks in speech, little crowd sound was recorded.

Another state with a high distribution of sequences, was state 11. The content found in this state was speech recorded from the main Studio sequences. These Studio sequences were the second most predominant structure in football, so again this result was logical. Other speech recorded from Interview sequences based in a studio environment, were also grouped into this state. The quality of recording between these two content structures was similar, therefore explaining this grouping.

Summarising the rest of the states, there was a common trend found in each football

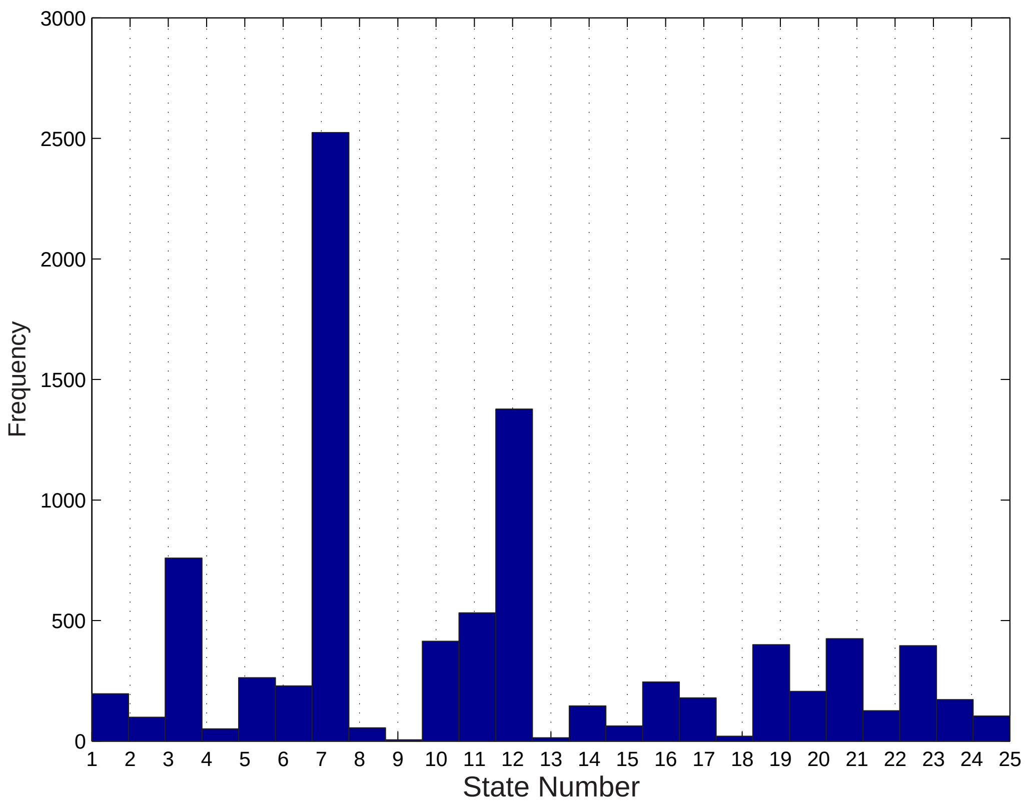


Figure 7.5: The typical distribution of audio clips, across the 25 states.

sequence, where similar content was grouped together in each state. For example, state 6 contained a lot of content related to exciting events such as crowd cheering and commentator excitement. States 21-24 contained a lot of content synonymous with Advertisements, such as music and sound effects. Content from other content structures, such as Reports and Interviews, were also grouped into the same states. For a full list of content found in each state, please refer to Appendix F.2.

The distribution of states were then compared with the manually annotated labels. Figure 7.6 is an example of the distribution of state labels from the three predominant content structures: Advert, Game and Studio. Similar trends were found for each content structure again, when comparing the output from the HMM model, and the manually labelled meta-data. For each content class, such as Studio, the HMM grouped content belonging to that class into similar states. For example, typical states for the Studio class included 10, 16, 17 and 19, while states for the Game sequences included 3, 7 and 12. Advert content was normally found in the later states, such as 20-25. This result confirmed the results found through inspection.

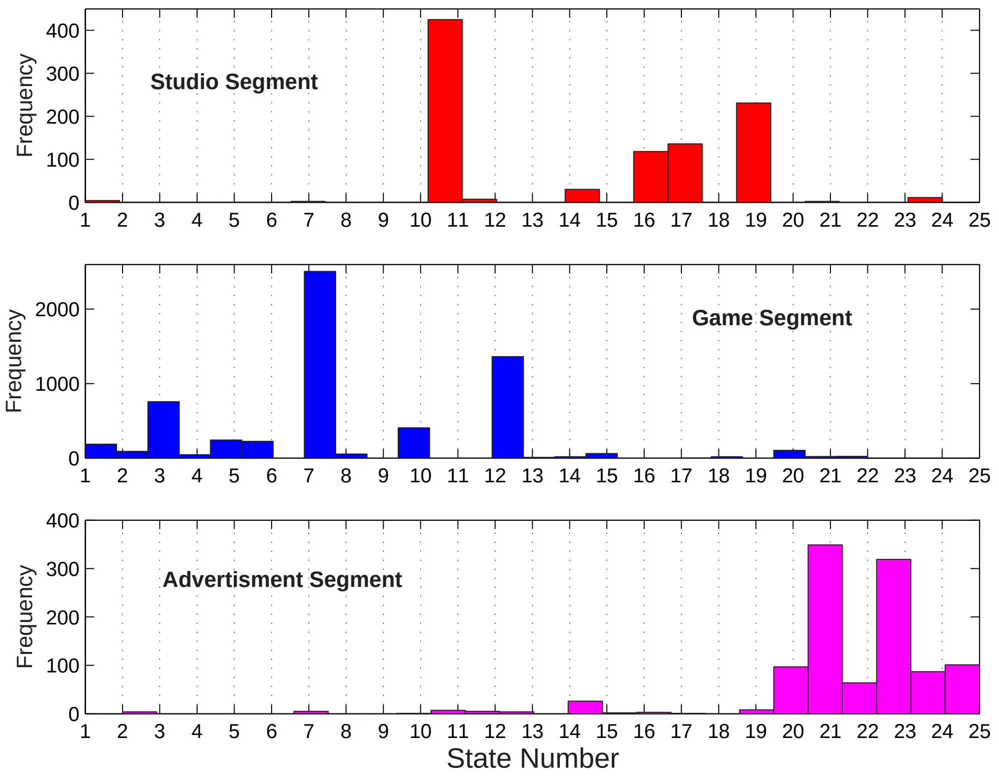


Figure 7.6: Histogram of the state frequencies per high level segments, for one video sequence. From the plot, it can be shown that the three broad classes are distributed differently.

Complete labelled audio sequences were also compared to the manual annotations. Figure 7.7 is an example of the HMM assigning each time step to a state label, for an audio sequence. The top graph corresponds to the manually labelled annotation at each time step. The bottom plot is the output from the HMM. The sequence was approximately 23 minutes extracted from one video file. It was again evident that there was a relationship between HMM state assignment and the manual annotation labels. A change in content structure from Game to Advert also corresponded to a change in state. This provided sufficient evidence that the HMM was a robust model for football audio, which can be applied for segmentation.

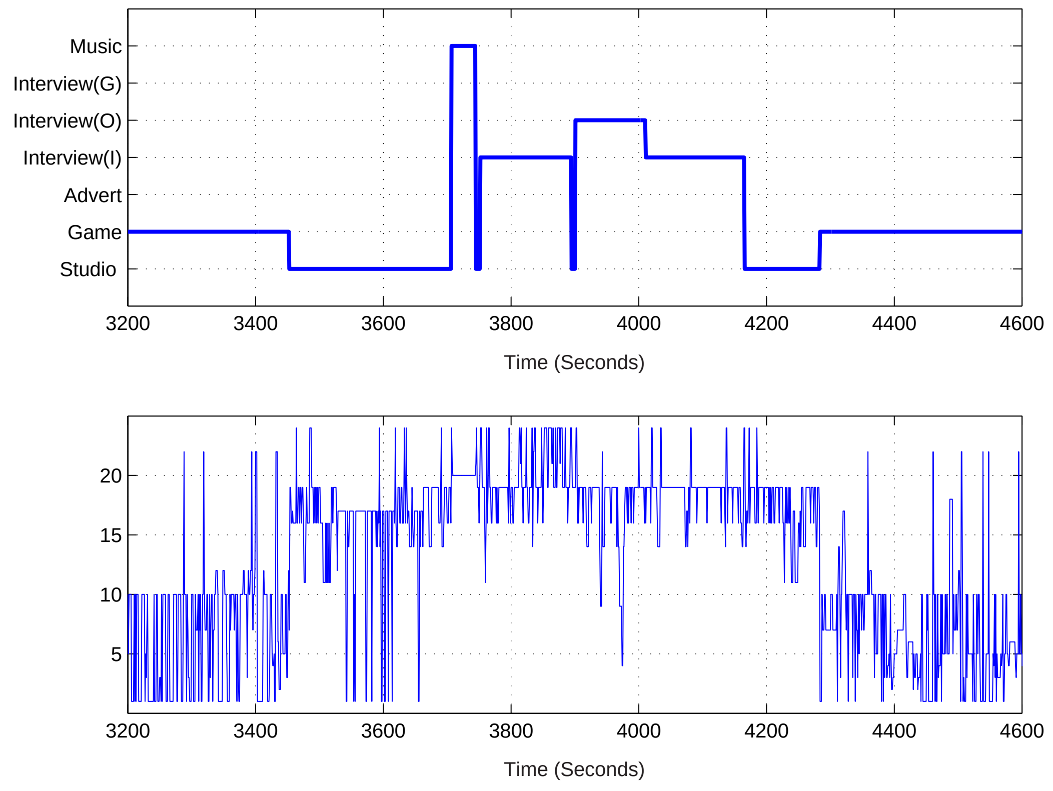


Figure 7.7: An example of an audio sequence labelled both manually and by the HMM. The top plot corresponds to the manual annotation and the bottom plot, the HMM.

7.5.3 Discussion

Examining the results from the content analysis study as a whole, it was evident that audio alone would provide a rich, discriminative information source for segmenting and labelling football video structure. There was sufficient evidence overall, to suggest that the HMM was able to recognise automatically the main structures found in football video. The many content structures within a typical football sequence, were identified and grouped accordingly by the HMM, such as modelling the differences in speech recording quality between different content structures (including Game, Studio and Advert).

Although this grouping was not entirely perfect, with a number of states sharing similar traits, there was sufficient evidence to suggest that the HMM could be successfully applied for modelling content accurately and also segmentation. With careful, supervised training, for each content structure, more robust HMM representations could be generated separately, for each class, improving accuracy.

Sub-structures within each main content class, such as Game and Studio, were also recognised. A selection of sub-structures that were grouped together, included crowd cheering periods, music, and high pitched sounds such as whistling. The differing levels of crowd sound found throughout a Game sequence were also modelled by the HMM. In Chapter 10, the sub-structures found are investigated further and applied to the problem of event detection. The recognition of crowd cheering and commentator excitement, especially useful information for developing an effective event detection algorithm.

Another interesting finding from this investigation, was that the test collection contained sequences recorded from two TV channels. There was a concern that there would be a difference in recording quality and production between the two channels, which could affect accuracy. One of the channels (BBC) did not contain advert breaks, however, the HMM was able to group similar content from both TV channels into the same states. For those sequences that did not contain Adverts, the HMM did not always use those states associated with Adverts. For example, Table F.2 represented a video file from the BBC. The 25th state, normally associated with Advert sequences, was not used at all for this sequence[6].

7.5.4 Video Topology Model

In Chapter 3.4, a video topology model was defined, which was a representation of the temporal relationship found within a typical football sequence (see Figure 3.2). From the content analysis findings, these main structures were verified. There was sufficient evidence to suggest that each main structure can be modelled efficiently by either a GMM or HMM. From the generation of a model for each content class, using supervised training, it was believed that future video sequences could be labelled according to this model. This process is described in the following chapters.

Another interesting result from the content analysis experiment was the final transition probability matrix A for the HMM. The HMM was initially ergodic with no constraints

[6]A follow up investigation comparing the differences between TV channels discovered minimal difference in audio recording (see Appendix E for further details).

between states. After training, it was discovered that there was a number of low or zero probabilities in A. This meant that the probability of moving from one state to another, in one time step, was either unlikely or impossible. Analysing the semantic relationship between states that shared such probabilities, it was found that many belonged to both Game and Advert classes. For example, state 24 to state 6 was zero. State 24 contained content mostly correlated to Advert sequences, while state 6 corresponded to Game sequences (and crowd cheering).

This provided evidence that the HMM modelled similar temporal relationships as defined manually, in the video model. In Figure 3.2 the movement from Advert sequences back to Game was restricted, and in particular movement from an Advert to an exciting event. To move from Advert content to Game, the video would first have to enter the Studio class. It was believed that the defined video model could be applied to enforce temporal restrictions between content classes during algorithm development (see Chapter 9), by doing so, improving the accuracy of the structure segmentation and classification algorithm.

7.6 Conclusions and Summary

In this chapter, the HMM was proven to be a strong representation of the audio data. However, to allow for the modelling of such large volumes of data, the original parameterised audio tracks were reduced, minimising information loss. The relationship between window length and classification error was formally evaluated, to minimise information loss. Audio sequences were then divided into one second units and the mean feature vector was taken as a representation, reducing the data.

A HMM was then generated for modelling the entire audio track, estimating the number of states graphically, using a measure called predictive likelihood. With this ‘optimal’ HMM representation, a new collection of audio sequences were then labelled, studying the results by employing a number of methods. From the findings, there was sufficient evidence to suggest the HMM modelled content logically, automatically discovering new sub-structures that will be useful for future indexing algorithm generation. It was found that the HMM, through unsupervised training, was able to model similar patterns to those identified through manual inspection. The content of the audio track for football video was effectively modelled for the purpose of content analysis, generating a robust HMM representation of the audio track. The results from the content analysis also verified the predefined video topology model. In the following chapters, these content classes will be divided up, generating optimal models and providing a framework for classifying future video sequences.

Chapter 8

HMM Model Selection

8.1 Introduction

The aim of this chapter is to investigate a number of strategies for selecting efficient HMM model representations of content found in football video. A critical process in the design of an indexing algorithm, which utilises statistical models, is the selection of robust HMM approximations of each content class. However, from reviewing the audio indexing literature, there is no proposed guide to the problem of HMM model selection. This issue is addressed in this chapter, introducing and investigating three HMM model selection strategies, and the effect each has on classification performance. The three selection strategies are: an exhaustive search approach using predictive likelihood as selection criterion, the seminal Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC).

The main reasons for investigating model selection is, to identify a method for choosing the simplest possible model, in terms of complexity, which would not have a negative impact on system performance. Model selection is also important for avoiding potential problems such as overfitting and poor parameter estimation. As the number of parameters in the model increases, so does the possibility of the model learning peculiarities found the training data, resulting in the model not generalising efficiently to new videos. For these reasons, HMM model selection is investigated in this chapter, particulary for the problems of state selection and mixture estimation.

The steps taken to investigate the problem of HMM model selection are now summarised. In section 8.2, the motivation and objectives of the chapter are discussed, identifying three model selection strategies. These three strategies are then introduced in more depth, in section 8.3. An experiment is then planned for examining whether to estimate the number of hidden states or Gaussian mixtures first, in section 8.4. In section 8.5, the results of the HMM state selection experiment are presented, and section 8.6 highlights the findings from the mixture estimation experiment. The three selection strategies are then compared over another classification problem, with the result presented in section 8.7. The chapter is then concluded in section 8.8.

8.2 Motivation

The main aim of model selection is to choose the simplest possible model, in terms of complexity, without deterioration in classification performance. This is considered especially important given the difficulty and practicality of generating large, varied training sets, that are required to estimate parameters precisely for complex models (Jain et al. (2000)). This problem is of particular interest because, thus far, there has been little research into model selection for the application of HMM, in the video indexing literature.

As previously mentioned (in Chapter 7.2), speech has a very clear modular structure that can be utilised for minimising the number of models required in an ASR system. Instead of modelling words directly, a set of HMM can be applied to represent each phone found in the vocabulary. It has been discovered, through experimentation, that these phones can be efficiently represented by a left-to-right HMM with 3 or 4 states (Reyes-Gomez & Ellis (2003)). A left-to-right HMM models the rise and decay of the pronunciation of each phone, with each of the 3 to 4 states corresponding to a phase in the pronunciation process. However, for modelling more general sound classes such as a Game sequence, there is not such a basic unit that can be modelled directly, or provide an indication towards the number of hidden states, or HMM topology.

A simple solution to the problem of HMM model selection, when modelling such broad content classes, is to choose a set of HMM models with the same number of states for each content class. This is a popular solution implemented in a number of video indexing systems (Eickeler & Muller (1999), Wang et al. (2000), Huang & Wang (2000), Xie et al. (2002), Xiong et al. (2003*a*)). This strategy can be helpful for matching a known number of potential states found in the data e.g the number of transition types that can occur during shot segmentation (Boreczky & Wilcox (1998)). There has been little research into how suitable this strategy is when applied to the broad content classes found in video.

Huang & Wang (2000) and Wang et al. (2000) employ a 5 state HMMs for modelling complete video genre, including Sport and News, ignoring potential differences in the underlying structure found within each separate video domain. Eickeler & Muller (1999) also apply the same strategy for segmenting and classifying structure in news video, and Xiong et al. (2003*a*) use a 5 state HMMs for classifying crowd cheering sequences in sports video.

A variation of this approach was employed by Xie et al. (2002), who segment and classify Play and Break sequences for soccer video, by using HMMs to model motion and colour distribution statistics. To model both classes, the authors applied a series of simple HMM models of varying size and topology, for classification. Each class was modelled by either a 3 state left-to-right or a 2 state ergodic HMM. New sequences were then labelled into the class returning the best representative HMM, in terms of likelihood. This strategy attempted to address the issue of model selection, however, it was a ‘lazy’ solution. This strategy ‘hoped’ that one of the two models generated would be a good enough representation for that class. The issue of model selection was not addressed, and in particular there was little investigation into the underlying data processes that define the content classes.

No standardised test collection exists, at present, for the majority of video domains. As a result, there has been little comparative studies between different approaches for video and audio classification. It would be difficult to infer that such approaches to model selection would provide inferior results. However, it is believed that the performance of a HMM is dependent on how well the model represented the data. The model should capture the common structure, variability and potential noise of that class, assuming that the selection of parameters will have implications concerning the future accuracy of the system. For example, the more hidden states available to the HMM, the greater the level of temporal detail.

The discriminative information required for classification can become blurred when using a small a number of states, making it difficult to differentiate between classes. If there are too many states added to the HMM, then peculiarities from the training set can be captured, resulting in overfitting problems. A HMM with too many states can also represent similarities from other groups, resulting in poor classification (see Chapter 4.4.3.1 for further details).

For the above reasons, it was believed that an investigation into HMM model selection is important, thus, two objectives are defined. The first aim is to formally investigate the possible effects on classification accuracy by ignoring model selection. The second objective is to identify a strategy for selecting HMM, considering issues such as model fit, model complexity and classification performance. Before deciding upon a possible selection strategy, a number of related works are reviewed.

8.2.1 Related Work

In Chapter 7.4, predictive likelihood was applied as a criterion for identifying the number of states in a HMM, and was an example of a non-discriminative model selection approach. The maximum likelihood is frequently used as a criterion for training HMMs, however, there is a belief that a mismatch exists with non-discriminative model selection, and the true objective of classification performance (Juang et al. (1997)). For example, predictive likelihood is a measure of the model fit, while the aim for many HMM applications is to select a set of representative models that minimise classification error. This mismatch has led to research into new discriminative criterion for both training and model selection. Discriminative measures are based on this actual target, classification performance, a popular method being the Minimum Classification Error (MCE) (Juang et al. (1997)). Discriminative model selection compares HMMs with competing classes, selecting the models that minimise the classification error (Biem (2003)). In other words, the most accurate set of HMMs found during the training phase for recognising known content.

Discriminative approaches have three major disadvantages when compared to non-discriminative criterion, based on the predictive likelihood (Gu & Rose (2002)), which are:

- the experimental design complexity is considerably increased,
- the MCE suffers greatly from poor local optima,
- the MCE does not often generalise well outside of the training set.

For these reasons, it was decided to concentrate on methods based on predictive likelihood.

Domains such as temporal data clustering (Li & Biswas (2000)), musical instrument classification (Reyes-Gomez & Ellis (2003)) and speech recognition (Chen & Gopinath (1999)) have also concentrated on non-discriminative methods for HMM model selection. These measures do not have a direct relationship to classification accuracy but are easier to implement, in practice, than discriminative techniques, as well as providing a guide as to how well the model fits the data (Gu & Rose (2002)).

Li & Biswas (2000) compared two selection criterion for determining the number of states, in a HMM, for temporal data clustering. The two measures, the Bayesian Information Criterion (BIC) and the Cheeseman-Stutz approximation, penalised the predictive likelihood. From an evaluation based on synthetic data, both strategies displayed comparable performance, and were considered more accurate, when compared to a simpler approach, which placed a stopping threshold on the predictive likelihood score. BIC was shown, through experimentation, to select comparable HMMs when compared to Cheeseman-Stutz.

Reyes-Gomez & Ellis (2003) also compared a number of non-discriminative model selection criteria, including BIC and low entropy, for finding the optimal number of states that represent general audio. These audio classes included musical instruments and sound effects. The criteria were also compared using different search framework for estimating state number. For BIC, a pool of models with states ranging from 1 to 50 were implemented, with the model producing the maximum BIC score being chosen. For low entropy, mixture components were continually added to a GMM before a stopping threshold was reached, for each model. The mixture components in the GMM were then converted to states in a HMM. The strategies were first compared, applying a single Gaussian hidden state, with a difference of 15% in classification accuracy found between BIC and low entropy (74% and 89% respectively). When further Gaussian mixtures were added per state, the difference in classification accuracy between both strategies became 5%, with the BIC method increasing accuracy to 77.8% and low entropy deceasing to 82.2%. Low entropy produced the simplest and most accurate models overall (Reyes-Gomez & Ellis (2003)).

The experiment did not appear to provide a fair comparison between the two methods (Reyes-Gomez & Ellis (2003)). The training data for each class was small, and the stopping threshold for the low entropy approach was highly set, resulting in simpler models than BIC. In one case, the low entropy selected a 4 state HMM compared to a 34 state selected by BIC. Given the small size of the training data, the simpler models resulted in better parameter estimates, than the more complex models. However, when there is little training data available, a weight can be adjusted in BIC, to select simpler models, that was not investigated in this paper. Another problem, with the experiment, was that the reasoning behind why both low entropy and BIC were not compared in the same search framework was not explained. From the results, it would seem that both the threshold estimate, and the search framework, also contributed to the difference in performance, as well as the selection criterion.

Chen & Gopinath (1999) also employed BIC for determining the optimal number of Gaussian mixtures per state, to use for modelling speech, as well as a stopping criterion for adding states to the HMM. When compared to a threshold approach, BIC was found to display comparable performance. The advantage of selecting BIC over the other method, was that it was more efficient, as well as being portable to new data sets. Also, using the BIC method allowed for fewer models to be implemented, when compared to an exhaustive approach.

8.2.2 Discussion

In other research fields, such as data analysis and time series analysis, a popular modal selection criterion is the seminal Akaike Information Criterion (Akaike (1974)). This criterion is an approximation of the Kullback-Leibler (KL) distance, which measures the distance between two probability distributions, or in this case, two models (Burnham & Anderson (1998)). In 1973, Akaike derived the AIC measure as an estimator for the KL distance and “provided a new paradigm for model selection” (Burnham & Anderson (1998)). AIC provides a simple, effective and objective way of selecting the best approximate model, given a set of candidate models. The simplicity of AIC is that it penalises the predictive likelihood, causing a peak at the ‘optimal’ model. The AIC penalty term penalises more complex models with respect to model fit, resulting in a higher penalty, helping to minimise the problem of overfitting. When compared to the computational demands of other model selection strategies, such as cross validation, the AIC was considered an appropriate method to investigate further.

The second chosen strategy was the Bayesian Information Criterion (BIC) (Schwarz (1978)). BIC is again a predictive likelihood penalty criterion, however, unlike AIC, BIC is not considered a true estimator of the KL distance between two models (Burnham & Anderson (1998)). The reason for selecting BIC, is that it is a popular method in audio classification applied to a number of problems (Chen & Gopinath (1999), Li & Biswas (2000) and Reyes-Gomez & Ellis (2003)). BIC was selected over other approaches applied in these works, such as low entropy and the Cheeseman-Stutz, because of its simplicity, and no threshold estimation is required (Reyes-Gomez & Ellis (2003)). The size of the training sample is also considered when calculating the BIC penalty term, and was shown through experimentation to be comparable to the more complex Cheeseman-Stutz criterion (Li & Biswas (2000)).

The third selection strategy is a baseline exhaustive search method, using a stopping threshold. This approach implements a comprehensive set of models, covering as many combinations as possible. Predictive likelihood is then applied to compare each model, using a stopping threshold to locate the final HMM. This approach was also compared in Li & Biswas (2000) and Chen & Gopinath (1999).

8.3 Non-discriminative Model Selection

There are three main issues for HMM model selection.

- HMM Topology,
- Number of Hidden States,
- Number of Gaussian mixture components per state.

In comparison to ASR systems, the football audio domain did not provide reliable clues on restricting the HMM topology, for each content class. For example, it would be difficult to determine that a left-to-right HMM would be more appropriate than an ergodic HMM. For ASR systems, it is known that speech will go through a series of states in a left to right manner, however, for the broad content classes in football, no prior assumption can be made on the general interaction between states. For this reason, ergodic HMMs with no restriction in the transition matrix, are initially employed.

For the remaining two issues, the three selection criteria are compared for estimating the two parameters separately (state and mixture number): an exhaustive search approach with a stopping threshold on the predictive likelihood score, the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC). These three strategies are now introduced in more detail.

8.3.1 Exhaustive Search (LIK)

In Chapter 7.4 an exhaustive search involving the implementation, training and testing of a range of models was used, to find an optimal HMM representation of the entire audio track. It was shown that the predictive likelihood score increased as more states were are added. In theory, the predictive likelihood would increase until each data sequence in the training set was modelled by its own unique zero-variance state (Li & Biswas (2000)). This would be an extreme case of the HMM overfitting the training data. To limit problems such as overfitting, a criterion is required to automatically identify the ‘optimal’ model. A popular method is to define a stopping threshold (Chen & Gopinath (1999)). The threshold is estimated for determining when the predictive likelihood has not improved by a significant amount, to warrant a new parameter being added to the model[1]. Once this threshold is reached, the current model is selected. Adopting this approach can be seen as a linear search algorithm for finding the optimal model. Parameters are iteratively added until the ‘optimal’ HMM is found. This is an exhaustive, but very costly procedure, both in terms of time and resources. Many models have to be trained and tested, before the ‘optimal’ solution is found.

The stopping threshold can be defined as follows. Given a set of HMM models

[1]For the experiments, the exhaustive search method will be referred to as LIK, which is an abbreviation of likelihood.

$\Lambda = \{\lambda_c : c = 1, \ldots, N\}$, where each model has an increasing number of states (or an increasing number of mixtures) ranging from 1 to N, the ratio of change from the previous model, to the current, is calculated using,

$$\phi_c = \frac{L_{\lambda_c}(O)}{L_{\lambda_{c-1}}(O)} \tag{8.1}$$

where $L_{\lambda_c}(O)$ is the predictive likelihood (see Equation 7.1) for the current model λ_c. λ_{c-1}, is the previous model and O is the test set of acoustic observation vectors. If the ratio of improvement is not greater than a threshold,

$$\phi_c < \varepsilon \tag{8.2}$$

then no further parameters are added to the model. The stopping threshold and is required to be estimated through experimentation.

8.3.1.1 Estimating the threshold

Note: A synthetic data set was used as a guide to define the stopping thresholds for the exhaustive approach. After experimentation, the threshold was set to be $\varepsilon = 1.001$. e.g if $\phi_c < 1.001$ then no further parameters are added.

8.3.2 AIC

An alternative method to applying a stopping threshold, is to penalise the predictive likelihood. One of the first derivations of such as measure, was the Akaike Information Criterion (AIC) (Akaike (1974)). AIC penalises complex models until adding further parameters will eventually outweigh any benefit. The predictive likelihood is penalised on the basis of model complexity, causing a peak in the AIC score. It is assumed that the optimal model is found at this maxima.

To define AIC, let $\Lambda = \{\lambda_c : k = 1, \ldots, N\}$ be the set of HMM models to be evaluated, where λ_c is the current model. AIC can be defined as:

$$AIC_{\lambda_c}(O) = L_{\lambda_c}(O) - \dim \hat{\lambda}_c \tag{8.3}$$

where $\dim \hat{\lambda}_c$ is the number of free parameters[2] or dimensionality of the current HMM model λ_c. $L_{\lambda_c}(O)$ is the predictive likelihood for model λ_c, given the test data of acoustic observation sequences O.

8.3.3 BIC

The Bayesian Information Criterion (BIC) (Schwarz (1978)) has had recent popularity in the Statistical literature for HMM parameter selection (Li & Biswas (2000), Chen & Gopinath (1999)). BIC also penalises the predictive likelihood again by a term calculated by the complexity of the model. The major differences between AIC are the derivation of this penalty term. BIC is not a true estimation of the Kullback-Leibler distance unlike AIC (Burnham & Anderson (1998)). Also, BIC factors in the size of the training sample into the penalty term, where smaller training samples will generate larger penalty scores. The original penalty term for AIC only accounts for the number of free parameters in the HMM. Again, the increase in predictive likelihood, as more parameters are added, is eventually outweighed by the penalty term, causing a peak in the BIC score. This maxima is assumed to be the ‘optimal’ model.

To define BIC, let $\lambda = \{\lambda_c : c = 1, \ldots, N\}$ be the set of models to be evaluated and $O = \{O^i : i = 1, \ldots, K\}$ be the test set of K acoustic observation vectors. BIC can be defined as:

$$BIC_{\lambda_c}(O) = L_{\lambda_c}(O) - \gamma \frac{\dim \hat{\lambda}_c}{2} \log(K) \tag{8.4}$$

where $\dim \hat{\lambda}_c$ is the number of free parameters in the c^{th} model, γ is a penalty weight, K is the number of training samples, and $L_{\lambda_c}(O)$ is the predictive likelihood (see equation 7.1). In the true definition of BIC, the penalty weight is set to $\gamma = 1$, which is followed during experimentation (Chen & Gopinath (1999)).

[2]Parameters that were counted included emission probabilities and only those initial probabilities and transition probabilities that were greater than 10^{-6}, as in Li & Biswas (2000). Probabilities close to zero were ignored as they were not considered significant enough to influence the behaviour of the HMM.

8.4 States or Mixtures first

An important issue was addressed before experimentation began, to determine what order should the parameters be addressed in. The two parameters were: the number of hidden Markov states and the number of Gaussian mixture components for modelling the emission probability densities.

Would it be more appropriate to estimate the number of hidden states *or* the number of Gaussian mixture components per state first?

In theory, a pool of HMMs covering all possible parameter combinations can be implemented. For example, if the 'optimal' number of states was S, and the optimal number of mixture components was Q. Then to arrive at this optimal solution, it would require to iteratively implement and evaluate $S \times Q$ HMM combinations to find this optimal solution, using predictive likelihood as a measure. Given the complexity of training HMM models, and the size of the data set, this was seen as an arduous task, both in terms of time and resources. It would be difficult and time consuming to examine all combinations of S and Q. As the number of states and mixture components increase respectively, so do the number of parameters to estimate, quadratically (Rabiner & Juang (1993)). To avoid searching through an unnecessary number of combinations of state and mixture numbers, an experiment on synthetically generated data was first performed.

8.4.1 Experiment

A pool of 150 HMMs, of different state and mixture combinations, were implemented on data generated from a 6 state HMM with 6 mixtures per state (see Chapter 4.4.6 for details on synthetically generated data). Each HMM had a different combination of hidden states (ranging from 1 to 15) and Gaussian mixture per state (ranging from 1 up to 10). Each HMM was generated using the same training sample, and then tested on the same, separate test sample. The predictive likelihood score, $L_{\lambda_k}(X)$, was recorded for all HMMs.

Figure 8.1 presents a summary of the predictive likelihood for each combination of HMM. Each line on the plot indicates the predictive likelihood score of HMMs with a specific number of mixture components, displaying a summary of the results for HMMs with 1, 2, 4 or 6 Gaussian mixture components per state. From the figure, it was highlighted that, as the number of states increased, so did the predictive likelihood score. After a number of states were added to the HMM, the predictive likelihood began to converge. This convergence occurred around 6 states, which was expected, given the data.

Figure 8.2, illustrates the effect of just adding mixture components. In the figure, the predictive likelihood is displayed for a 6 state HMM, as the number of mixtures was increased. Adding further mixture components also resulted in a similar trend in predictive likelihood, with an initial rise followed by convergence, around 6 mixtures. Again, there was an initial, rapid increase in predictive likelihood, followed by a levelling off around 6 mixture components.

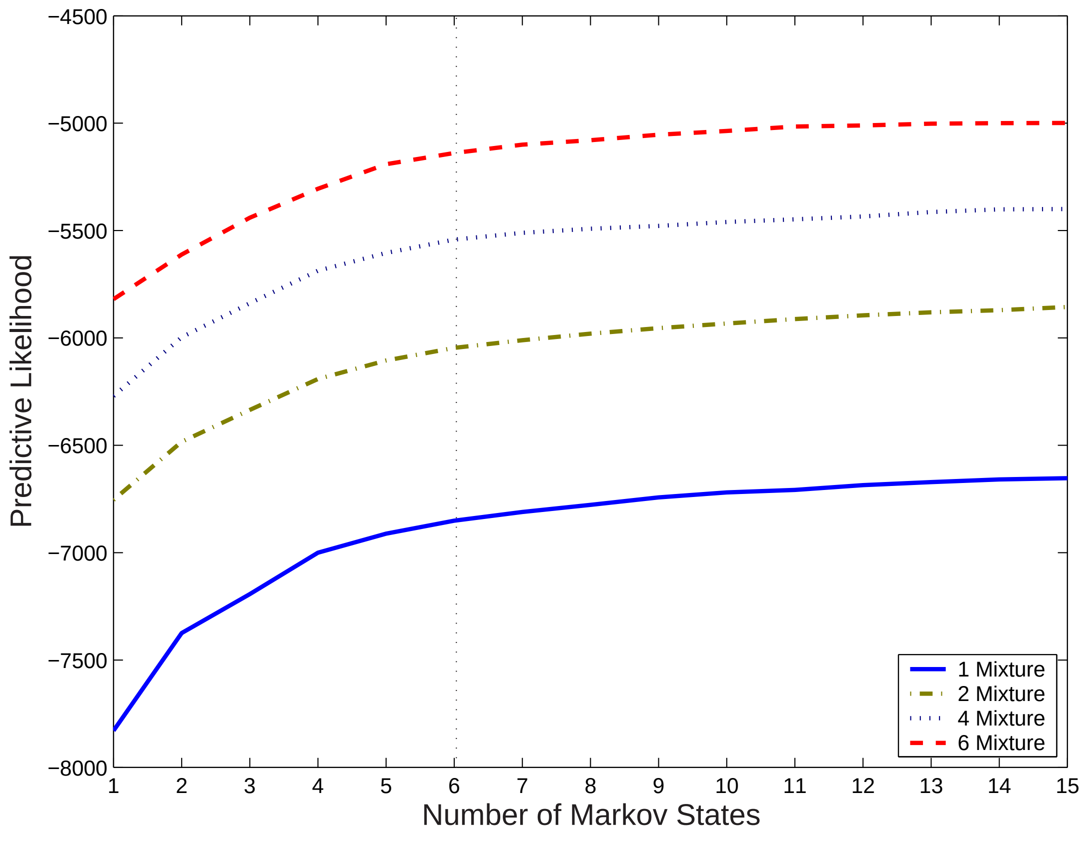


Figure 8.1: Plot of the predictive likelihood score $L_{\lambda_c}(O)$ versus the number hidden of states in a HMM model. The data set is synthetically generated from a 6 state HMM, with 6 Gaussian mixture components.

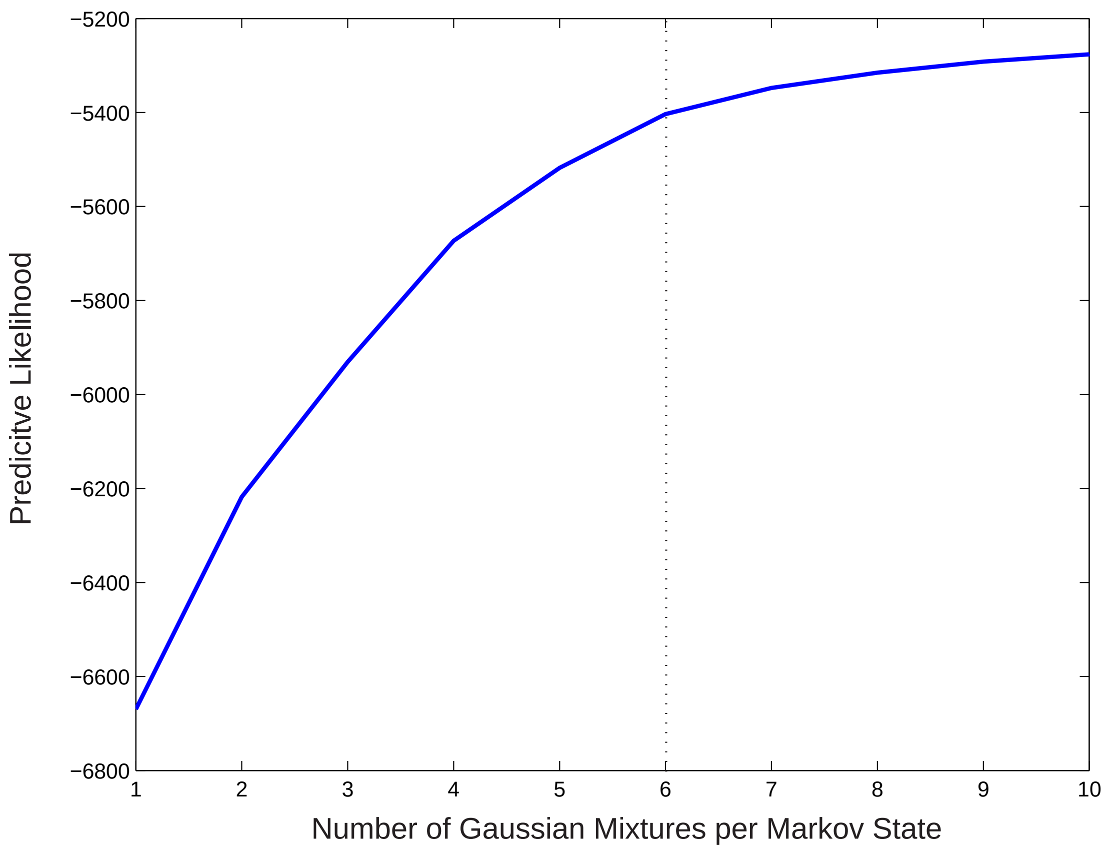


Figure 8.2: Plot of the predictive likelihood score $L_{\lambda_c}(O)$ versus the number of Gaussian mixture components in a HMM model. The data set is synthetically generated from a 6 state HMM, with 6 Gaussian mixture components.

From both Figures, it was displayed that after 6 states and 6 mixtures per state were added, the HMM began to overfit the data. As the data was generated from a HMM with 6 states and 6 mixture components per state, this result was predictable. Importantly, this result also indicated that it was more important to identify the correct number of states first.

Based on this experiment, the following procedure is followed for evaluating the model selection strategies. First, each strategy is evaluated for locating the 'optimal' number of hidden states in the HMM. This is achieved by beginning with a singular probability density function for each state, so that the correct number of states can be identified. After this is achieved, the appropriate number of Gaussian density components can be found, by adding mixtures iteratively to the HMM model. By doing so, reducing the number of HMM combinations required when evaluating the model selection strategies.

8.5 Selecting the number of Markov states

In this section, the three model selection approaches are evaluated for the task of HMM state selection. All selection strategies are evaluated using a test collection consisting of three content classes found in football video: Advert, Game and Studio (see Chapter 6.3.1 for more details). All methods are analysed across each content class separately.

8.5.1 Experiment Methodology

The same methodology was applied as in Chapter 7.4.3, however, instead of a pool of 30 HMMs, only 20 HMM models, with states ranging from 1 to 20, were iteratively implemented, separately for each content class. Again, these models were trained and tested using different samples. All HMMs were ergodic with a singular Gaussian PDF per hidden state. After the completion of one experimental run, the entire process was repeated 15 times, randomly changing the training and test samples.

Each content class contained manually labelled sequences of varying length for training and testing, using the same data set in Chapter 6.3.1. The test collection consisted of approximately 15482 seconds of audio, over four hours. The data set was divided into two. One sample was used for the model selection experiments in this section: estimating the number of states. And in the following section: estimating mixtures per state. The remaining sample was used for testing the final models, in section 8.7. The overall data set for the model selection experiment spanned over three hours of audio, and the other sample contained the remaining one hour.

In the previous chapter, a time step of one second was used for the HMM, calculating the mean of each one second unit, thus, reducing the volume of data. In this chapter, it was of interest to model each class at a finer granularity. The aim was to provide each HMM with as much information as possible, during the training phase, and to be as accurate as possible for future problems (see Chapter 9 and 10). For example, shot segmentation systems compare sequential video frames (refer to Appendix C), where there are approximately 20 to 30 frames per second. A granularity of this magnitude is preferred rather than one second. Also, working under similar time steps, would allow for easier integration between visual and audio models, for future indexing algorithms. So, for each data sequence, a time unit of one audio frame was used as an input into the HMM.

For each experimental run, the classification error was measured for all HMM state settings, for each content class. The same methodology, as in Chapter 6.3.3 was used, implementing both GMM and HMM classifiers. A GMM classifier was also generated for each content class, using a bottom up strategy, outlined in Chapter 4.3.2. The GMM was used as a baseline for comparing the performance of the HMM, trained and tested on the same data sets as the HMMs.

8.5.2 Experimental Results

The results from each content class are now presented.

8.5.2.1 Advert Class

Figure 8.3 displays the mean of the 15 initialisations for all selection strategies, with LIK being the predictive likelihood score for the 20 models. The predictive likelihood score increased as more hidden Markov states were added (Figure 8.3). First there was an initial rapid increase that levelled off around 11 states, suggesting the model was beginning to overfit the data. The stopping threshold was reached at 12 states. The maximum BIC score was found to be 10 states, and a similar pattern followed for AIC, where the maximum score was found at 15 states.

The classification error rate was also measured, see Figure 8.4. The circle represents the mean classification error for each HMM, and the bars are the standard deviation of the 15 experimental runs. The dotted line is the minimum classification error for the GMM classifier. As more states were added to the HMM, the mean classification error gradually decreased. After the 7^{th} states were added to the HMM, the model began to perform better than the GMM classifier. This result was found not to be significant, when using the Kruskal-Wallis test.

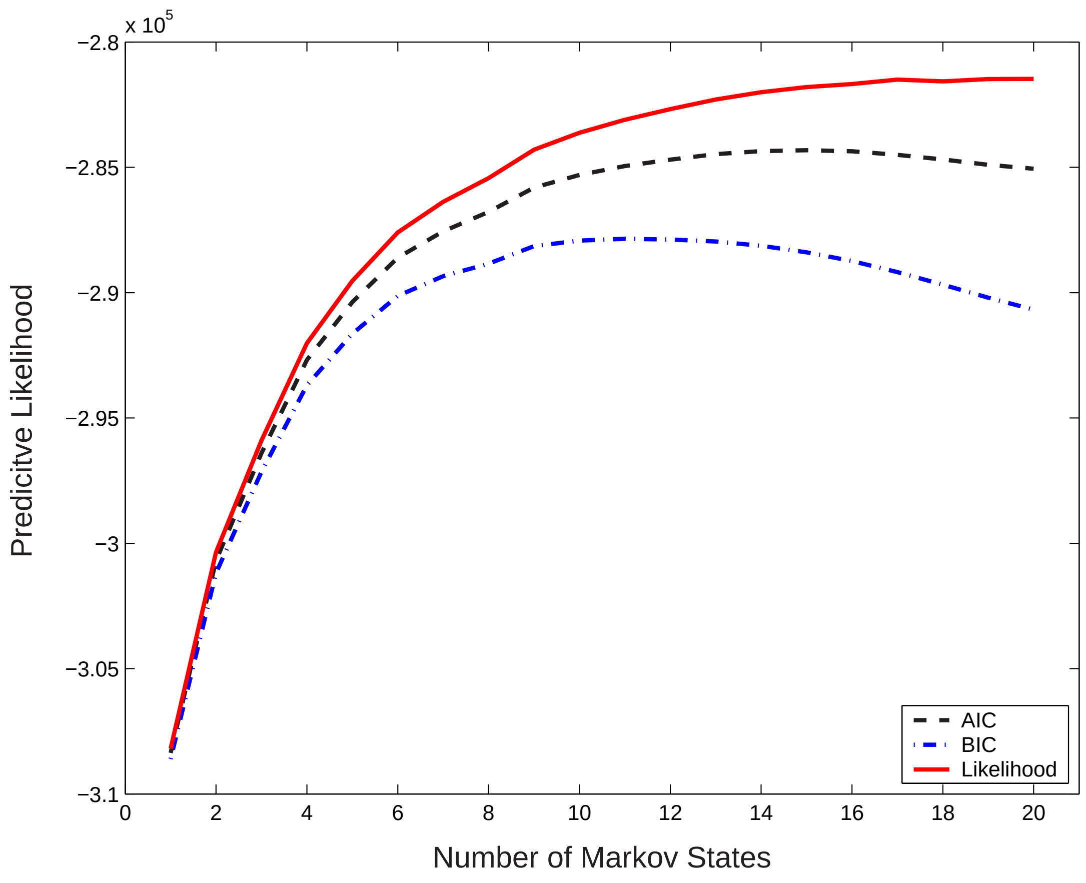


Figure 8.3: A comparison of each strategy for hidden state selection for the Advert class.

8.5.2.2 Game Class

Figure G.1 displays the mean of the 15 initialisations for all selection strategies[3]. The predictive likelihood again increased rapidly as more hidden Markov states were added. The predictive likelihood score levelled off around 14 to 20 states, suggesting the model was beginning to overfit the training data. The stopping threshold was reached after the addition of the 14*th* state. A similar trend occurred for the AIC score, where the maximum AIC score was found at 12 states, and the maximum BIC score was found at 9 states.

Increasing the number of states created an initial drop in classification error, Figure 8.5. The mean classification error gradually decreased as the number of hidden states in-

[3]The remaining predictive likelihood plots can be found in Appendix G.

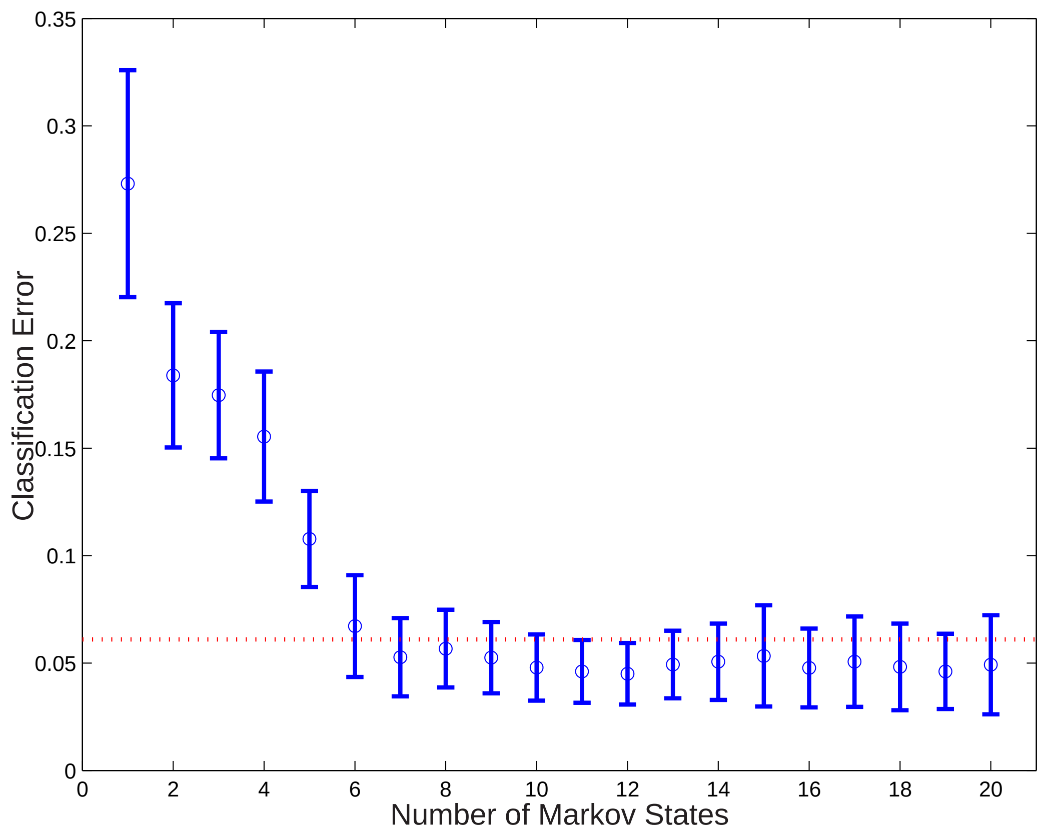


Figure 8.4: Classification error versus the number of hidden Markov states added to the HMM, for the Advert class.

creased for the HMM. After the 5^{th} hidden state was added, the HMM began to significantly outperform the GMM classifier. On average, a 12 state HMM performed best for this content class, the same model that was selected using AIC. There was no significant gain in terms of classification error after adding the 5^{th} state, and an increase in classification error was found after 14 states were added, the same HMM selected by the exhaustive search (LIK).

8.5.2.3 Studio Class

Figure G.2 displays the mean of the 15 initialisations, for all selection strategies, from the Studio class. The predictive likelihood again increased along with number of hidden Markov states, and after the initial increase there was a levelling off around 7 to 11 states. The stopping threshold on the predictive likelihood score was reached at 9

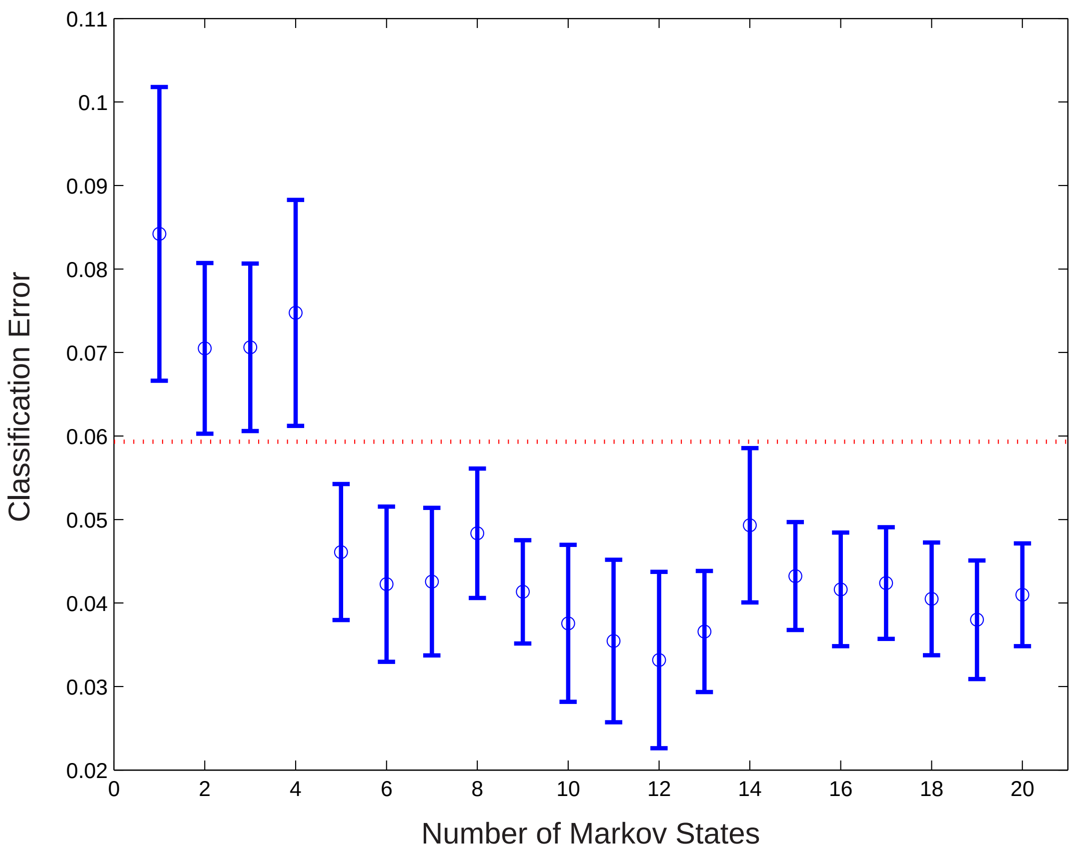


Figure 8.5: Classification error versus number of Markov states.

states. The maximum AIC score was also found at 9 states, and the maximum BIC score was discovered at 7 states.

The classification error steadily decreased as states were added to the HMM (Figure 8.6), after the 5thstate was included, the HMM began to perform significantly better than the GMM classifier. After 11 states, the improvement in classification error converged, with only a slight advantage gained from adding further states.

8.5.3 Discussion

From analysing the results, from all three content classes, there did seem to be a relationship between predictive likelihood and classification error. For both the Advert and Studio classes, as the number of states increased, the predictive likelihood score also increased, and the classification error decreased. After an initial rise, there was

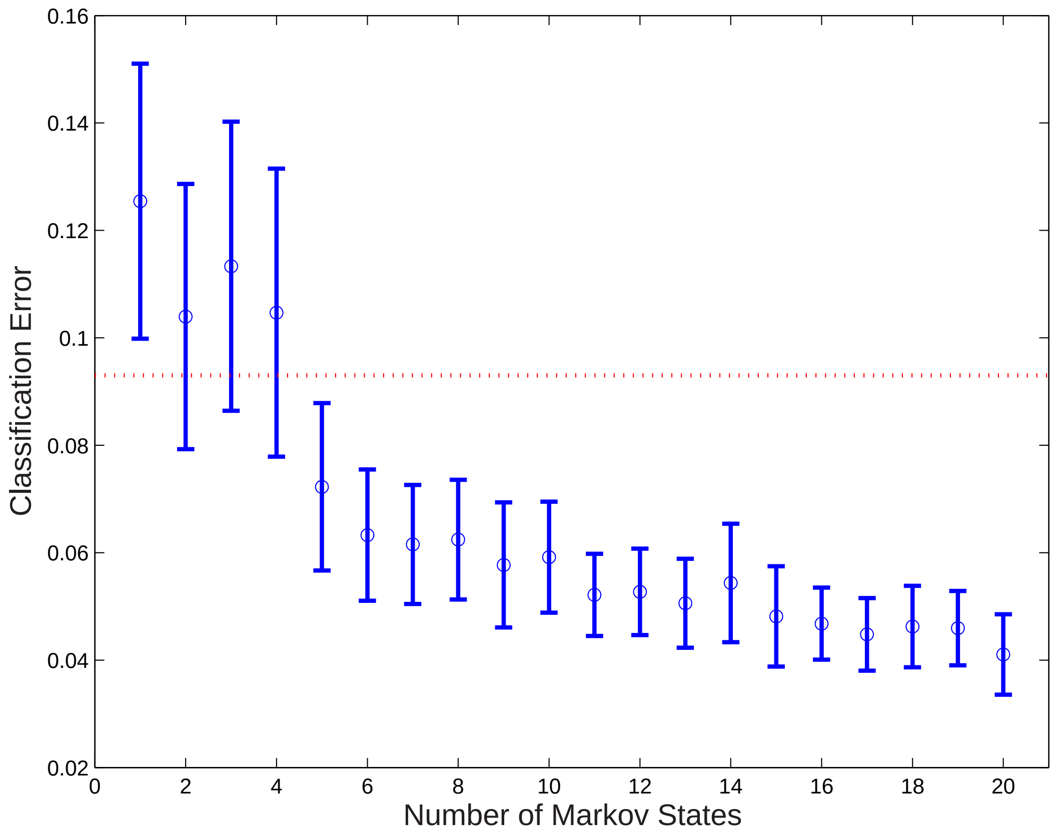


Figure 8.6: Classification error versus the number of Markov states for the Studio class.

a levelling off in both predictive likelihood, and classification error. A similar pattern occurred for the Game class, however, there was a small increase in classification error, after 12 states were added, before slowly decreasing again after 15 states.

The predictive likelihood was a good guide for identifying the optimal number of states in a HMM. The advantage of using a non-discriminative method is that HMMs can be generated in isolation, without having to use data from other classes, to determine classification error or some other related measure. Thus, simplifying the experimental process.

The BIC score selected the simplest models, out of the three methods, with no obvious drop in classification performance, with the other two methods selecting similar models, in terms of complexity. For exhaustive search, the threshold can be increased for identifying less complex models, but this issue highlighted a disadvantage with predefined threshold estimation. Approximating an appropriate threshold for each content class would be difficult. Also, the threshold does not consider the balance between model fit and complexity, which the other two methods do. BIC also had an extra advantage over AIC. The number of training samples are considered when penalising the predictive likelihood score, penalising complex models more when little training data is available.

8.6 Number of Gaussian Mixtures per Markov State

The GMM framework extends the basic single Gaussian distribution to that of a more enriched representation, which is able to model complex data by applying a mixture of PDFs (see Chapter 4.3). Likewise the HMM can also be extended under the same assumption that some content classes will be generated from multi-modal distributions (Rabiner & Juang (1993)). A mixture architecture can enhance the model by representing a complex distribution as a combination of simple components per state. For example, speech recognition systems have identified that multiple Gaussian mixtures per state are more accurate than a singular density function (Chen & Gopinath (1999)). A mixture of Gaussian components can model the inherently multi-modal emission densities that represent variation found in speech. That is, variation found within and across speakers, as well potential noise contamination.

Adding an appropriate number of mixture components to the HMM can encapsulate further discriminative information between classes, although the benefits still remain uncertain for general audio data, and have not been investigated thus far. For example, the same issues for state selection are relevant for selecting the number of mixture components. Selecting too many mixture components can result in overfitting or poor parameter estimation, given the size of the training data available. Although, a mixture architecture is worth considering given the possible potential in accuracy.

In this section the three selection strategies are evaluated for the 'optimal' number of Gaussian mixture components per hidden Markov state. The effect on classification error is also measured for extending the singular PDF per state HMM, to that of a mixture architecture. To do so, the same experiment in section 8.5 is repeated, this time implementing HMM models with increasing Gaussian mixture components per state.

8.6.1 Experiment

All three content classes were again modelled using a pool of HMMs. For all HMMs in the pool, the number of states were fixed corresponding to the simplest HMM, in terms of model complexity, found in the previous section[4]. The 10 HMMs were generated for each content class, with an increasing number of mixtures components, ranging from 1 up to 10. The covariance matrices were also constrained to be diagonal for each individual mixture. The reasoning behind this, was to reduce the number of parameters to be estimated. This was possible given the properties of the MFCC feature set, and is a standard practice for ASR systems (Rabiner & Juang (1993)). This process can also help limit overfitting (see Chapter 4).

After one complete run of the experiment, the test and training data were again randomly changed. This entire process was repeated 15 times. The following sections present the results for each class separately.

[4]For the Advert class that was a 7 state, Game a 12 state HMM, and Studio an 11 state HMM.

8.6.1.1 Advert Class

Appendix G, Figure G.3 displays the results from all three selection methods for the Advert class. The stopping threshold was reached after the addition of 9 mixtures. For the AIC score the maxima was found at 6 mixtures and the BIC score at 5 mixtures.

Increasing the number of mixtures for the Advert class resulted in a decrease in classification error (Figure 8.7). After a second mixture per state was added, the HMM significantly improved over the GMM classifier, following this trend as further mixtures were added. A possible explanation for this was the variability found in the Advert class. From the content analysis (see Chapter 7.5.2), it was noted that the Advert class contained a large variety of sub-structures including music, speech, speakers and sound effects. These sub-structures contained a lot of variation, given the infrequency of the same advert being repeated. By adding further mixtures to each state, this variability was more efficiently represented, especially when compared to the corresponding GMM representation.

8.6.1.2 Game Class

Increasing the number of Gaussian mixtures again corresponded with an increase in predictive likelihood (Figure G.4). The stopping threshold was reached at 9 mixtures per hidden. The AIC score eventually converged after 7 mixtures were added, and the maximum BIC score was found at 3 mixture components.

A different trend was discovered for the Game class, in terms of classification error. Adding further mixtures to the HMM, corresponded in a significant increase in classification error (see Figure 8.8), and poorer accuracy. After 4 mixtures per state were added, the GMM classifier performed better than the HMM. In comparison, all the model selection methods chose a HMM with a number of mixtures. However, the results from the Game class indicated that adding a mixture of Gaussians, to the HMM, did not improve the classification error rate. In fact, the classification error rate increased significantly as further mixtures were added. Analysing the results, only the BIC selected HMM with 3 mixtures per state, did not perform significantly worse than the baseline GMM, but the GMM mean classification error was higher, than this model.

The Game class consisted of a large number of sub-structures, such as speech crowd

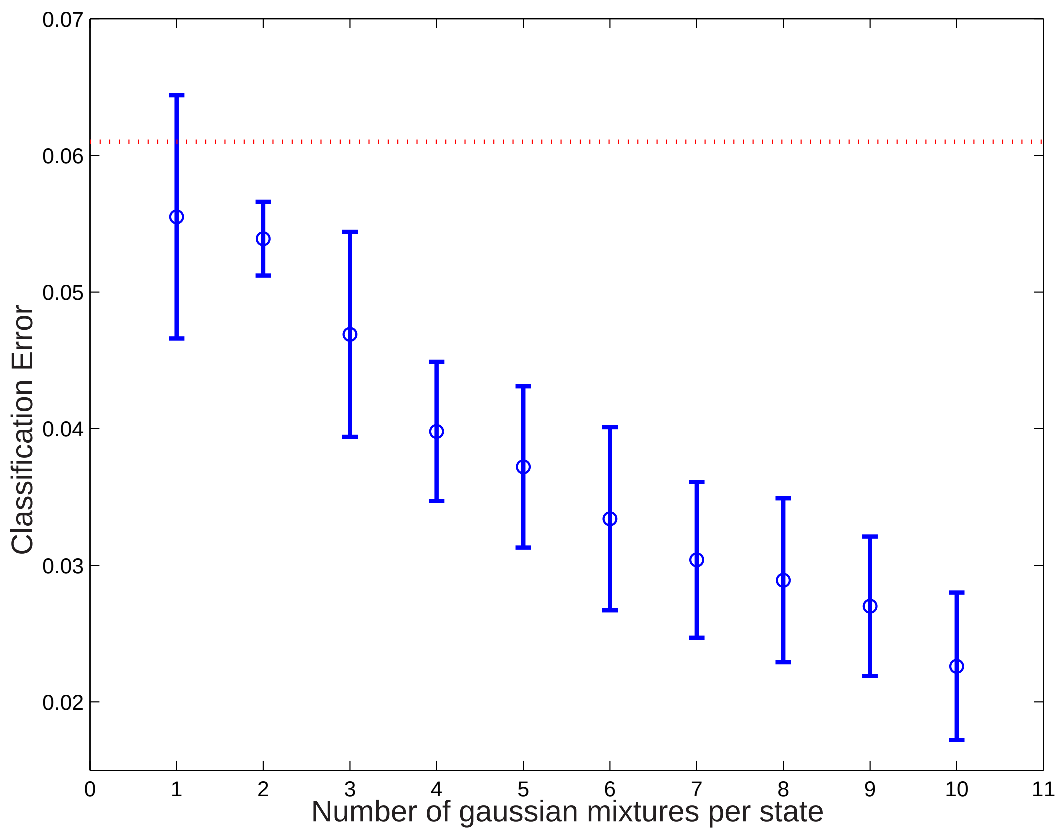


Figure 8.7: Classification error versus the number of Gaussian mixtures for the Advert class.

cheering and other environmental sounds (see Chapter 7.5.2). The results indicated that a mixture architecture did not necessarily improve the classification performance of the HMM, for representing this class. A HMM with a number of states, and a singular PDF per state, was a more efficient representation of these sub-structures.

8.6.1.3 Studio Class

BIC selected the simplest HMM, with 3 mixture components per state (Figure G.5). The maximum AIC score was at 7 mixtures per state, and 8 mixtures were added before the stopping threshold for the predictive likelihood score was reached.

In terms of classification error, a similar trend to the Game class occurred for Studio. Increasing the number of mixtures for the Studio class resulted in an increase in classification error (Figure 8.9), after adding further mixtures to the singular Gaussian

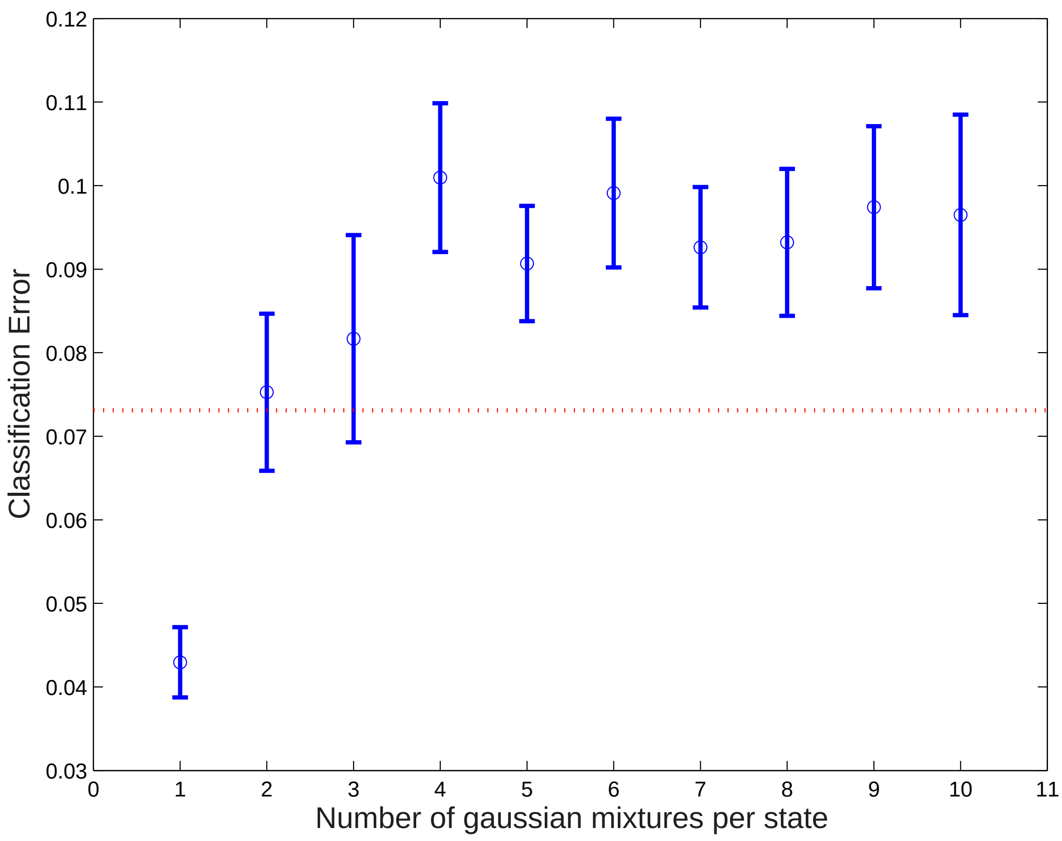


Figure 8.8: Classification error versus the number of Gaussian mixtures for the Game class.

density per state resulted in a drop in performance. This could be explained by the little variation found in Studio segments. The content of Studio classes mostly contained clean speech from a small set of people. It would appear that this content was robustly explained by a number of states modelled by a single Gaussian PDF.

The trend in classification error was not follow the three selection strategies (see Figure G.5). All three selection criterion indicated that a mixture architecture was more efficient, when compared to a singular density HMM. Increasing the number of mixtures, resulted in both the LIK, AIC and BIC selected HMMs to display a higher mean classification error than the original singular density HMM. All chosen HMMs were significantly more accurate than the GMM classifier though.

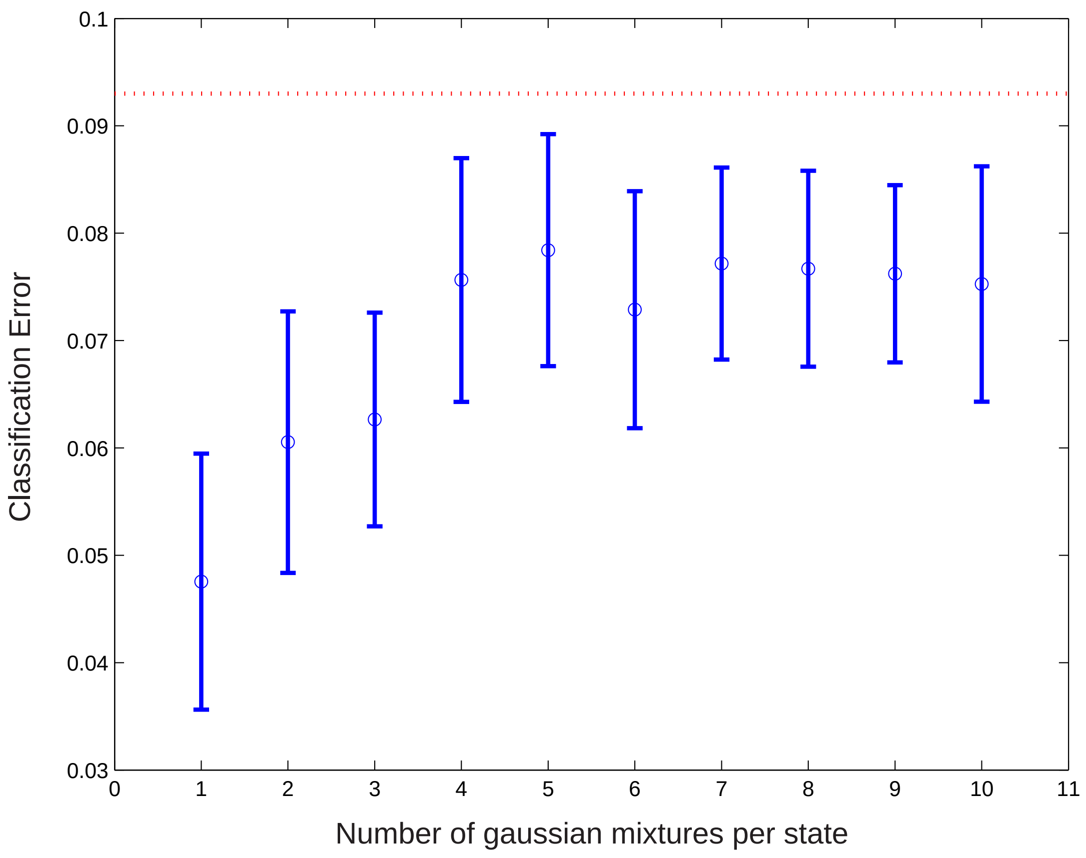


Figure 8.9: Classification error versus the number of Gaussian mixtures for the Studio class.

8.6.2 Discussion

From the experiments it was discovered that for two out of the three content classes, implementing a fixed mixture architecture for the HMM was not an ‘optimal’ solution. It was found that by increasing from a singular Gaussian representation to a mixture components per state, resulted in an increase in classification error. For the Game and Studio classes the assumption that data, in each state, followed a multi-modal distribution did not necessarily hold. In comparison, adding further mixtures to the Advert class, resulted in an overall improvement in performance. The variation that is contained in the Advert class was more efficiently represented by a number of mixtures per state.

A possible explanation for this phenomena, was that fixing the same number of mixtures per state was not ‘optimal’. For example, one state in the HMM represents speech that contains a lot of variation, such as different speakers, emotion, gender and background sound, with another hidden state represented only silence periods, or breaks in speech. This sound type did not generate a lot of variation. Therefore, a number of mixtures was efficient for one hidden state, which contained a lot of the variation, but not for the other state, which contained little variation.

This experiment indicated that further investigation into HMM mixture architecture was required, and in particular, investigation into strategies for selecting different mixture architecture for each HMM state. Fixing the number of mixtures to be equal, per hidden state, often resulted in a sub-optimal representation for some broad content classes. From the results, a HMM with a singular PDF per state was ‘optimal’ for the Game and Studio classes (Figures 8.8 and 8.9), while the same HMM for the Advert class, resulted in a significant drop in accuracy (Figure 8.7).

Studying the results of the three selection strategies investigated, there appeared to be differing performances across each class. The BIC method selected the simplest models that were closest to the minimum classification error scores. The remaining two methods chose more complex models, and in the case of the Game and Studio class, generating significantly worse models in terms of classification error.

For the exhaustive strategy, the stopping threshold was set too high for mixture selection, where an increased threshold would have selected simpler models. This again highlighted the value of methods such as AIC and BIC, which can generalise better to different content classes. Estimating a predefined global threshold does not transfer to all content classes.

The findings in this section, indicated the importance of investigating the merits of HMM design for individual content classes, with the ideal solution being an automatic HMM model selection process finding an optimal model. Again the BIC selection strategy selected the simplest models, however, even this method selected a mixture architecture for the Game and Studio classes. Further investigation into this problem and identifying a more robust search strategy would be advantageous.

For state selection there did appear to be a relationship between predictive likelihood and (minimum) classification error. But this relationship did not hold for the task of estimating mixture component number, for two out of three classes.

8.7 Classification Experiment

In this section, HMM models selected by each strategy, were evaluated formally for a three class classification problem. Using a three class problem, each model selection strategy is compared across a more difficult problem, providing further insight into the model accuracy. This would also indicate how effective each is strategy was for selecting HMM models automatically, by testing each method over a new test set.

8.7.1 Experiment Methodology

The three HMM selection strategies were again applied to generate three ‘optimal’ HMMs, using the entire data sample applied in the previous two sections. A HMM representation for each content class (Advert, Game and Studio) was generated using all model selection methods (LIK, AIC and BIC). A GMM classifier for each class was also generated.

The classification accuracy for each strategy was then formally compared across a new test set[5]. The new sample contained labelled sequences 10 seconds in length from each of the three content classes; Advert, Game and Studio. The overall test sample was approximately one hour in length. The maximum likelihood criteria was applied for classification, where a new test sequence was labelled into the content class that produced the highest model likelihood score.

8.7.2 Results

Table 8.1 provides a summary of the results. The overall classification accuracy (numbers in bold) and the percentage of false classifications (remaining numbers), for each class, are presented in a confusion matrix. The results highlight how accurate the models chosen by each strategy were in classifying the new test collection.

[5]Classification accuracy (see Equation 6.2) is the inverse of classification error. To evaluate each HMM selection strategy, the classification accuracy measure was used. This measure was applied rather than classification error because for this problem it was of interest to identify the criterion that produced the most positive results, in terms of classification performance. This measure is widely used for audio classification problems (Roach & Mason (2001), Wang et al. (2000)).

Classification Accuracy(%)												
	Game				Studio				Advert			
	LIK	BIC	AIC	GMM	LIK	BIC	AIC	GMM	LIK	BIC	AIC	GMM
Game	**89.6**	**92.7**	**90.4**	**90.4**	1.8	1.1	1.0	2.9	8.5	6.2	8.6	6.7
Stud	4.6	5.2	5.1	2.9	**89.1**	**86.8**	**87.6**	**90.3**	6.2	8.0	7.2	6.8
Advt	1.1	1.0	1.1	1.4	3.5	3.4	3.0	3.7	**95.4**	**95.6**	**95.9**	**94.9**

Table 8.1: Confusion matrix. The % of correctly classified observations are in bold.

The Advert classifiers displayed the highest accuracy for all content classes, with the HMM selected using AIC recording a 0.03% improvement over BIC, and over 0.05% compared to the HMM selected using the exhaustive search method (LIK). The AIC selected HMM also showed a 1% improvement over the corresponding GMM classifier. From the Advert content classes, the majority of false classifications were sequences that contained very clean speech. These sequences were wrongly labelled as Studio sequences. A small percentage of clips containing sound-effects were labelled into the Game class. A possible reason was that these sound-effects were not contained in the training collection, and could not be explained by the Advert classifiers.

For the Game class, the BIC chosen HMM was over 2% better than all other classifiers, with the threshold selected HMM the worse, with 89.6% accuracy. The majority of false classification from the Game class included sequences containing speech that contained little to no background noise. These were often mistaken as speech sequences belonging to the Advert class. Over 8% of such sequences were wrongly classified for both the threshold and AIC selected HMMs. Other false classification of Game samples into the Advert class included, music played inside the stadium, and other peculiarities, such as stadium announcements. Again, these samples may not have been represented in the training collection.

The Studio classifier produced the worst performance overall, with the GMM classifier recording the highest classification accuracy. The GMM classifier for Studio was over 3% more accurate than the BIC selected HMM, and the LIK HMM being the best performing HMM. Inspecting the results closely, the majority of false classifications were found to be speech wrongly labelled into either the Game or Advert classes.

To summarise the experimental findings, the results highlighted that there was no significant difference, in terms of classification accuracy, across all three model selection strategies, for each content class (Table 8.1). There was also no significant improvement in choosing a HMM over GMM for this three class problem.

8.7.3 Discussion

The BIC selected HMM models were again the simplest models in terms of complexity. Over the three content classes, these HMMs performed comparable to the other, more complex HMMs selected by the other two strategies. HMM models selected by BIC displayed no decrease in classification accuracy overall, and for the Game class, did recorded the highest classification score. On the other hand, the same strategy did provide the worse result for the Studio class. The AIC model selection approach chose the less complex models than the threshold approach. The threshold can be increased to select simpler models, but the experiments indicated that it is difficult to set a threshold that is portable across many different classes.

An advantage with using the AIC BIC criteria was that not all HMMs have to be implemented to find the optimal model. Since these methods penalise the predictive likelihood score, the final curve is concave. Therefore, a bisection search method can be used to find the maxima of the curve, or another equivalent. Comparing the two selection strategies in terms of performance, the BIC method is more desirable than AIC, as it factors in the size of the training data, as well as model complexity.

Comparing the HMM and GMM classification frameworks there was no significant evidence to suggest one framework was significantly better than the other, for the three-class problem. For the Studio content class, the GMM produced better results than all HMM models. There was no sufficient evidence from this experiment to suggest that the HMM framework was better than the simpler GMM classifier, with both frameworks displaying comparable results.

8.8 Conclusions

In this chapter, the importance of model selection was highlighted across a number of experiments, where a gain in performance was found from selecting models methodically. For many applications of HMM in video indexing, this type of approach is not adopted (Eickeler & Muller (1999), Wang et al. (2000), Xie et al. (2002)), suggesting optimisation issues with each system. A particular problem with current accepted solutions was, that by selecting too few or too many hidden states, a drop in classification performance can result. This was illustrated through experimentation.

Three model selection strategies were evaluated: an exhaustive search approach applying a stopping threshold, the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC). Both the AIC and BIC strategies have two obvious advantages over a stopping threshold approach. First, the number of models required to find an optimal solution can be reduced, eliminating an iterative addition of parameters, with both AIC and BIC creating a maximum peak. Therefore, an efficient search strategy such as a bisection method (Newton-Raphson) can be employed to find the maxima of the curve. For example, a changing upper and lower bound is placed on the curve until the maxima is found. Secondly, both criterion do not require a rigid stopping threshold to be estimated, which was not to generalise across different content classes.

Overall, for the problem of selecting the ‘optimal’ state number for the HMM, it was discovered that BIC was the more efficient HMM selection method. BIC consistently selected less complex HMM models, without affecting classification accuracy. However, there was no conclusive proof that BIC would be more robust than AIC in the long term. The experiments did indicate though that the optimal number of states for one class, was not always suitable for another content class. However, using model selection was beneficial for generating robust HMMs.

From experimentation another important discovery was the effect of increasing the number of mixture components from a singular density, had on classification accuracy. For differing classes, the effect was either positive or negative. For both the Game and Studio class, adding two or more mixtures caused a detrimental effect on performance. On the other hand, adding a mixture architecture to the Advert class, improved accuracy.

To summarise the findings from the experiments, the results did highlight a problem with mixture selection, where fixing a set number of mixture components per state, is not the best solution for representing the underlying data processes. For each hidden state in the HMM, would possibly be better represented by a differing number of Gaussian mixtures. Future investigation into methods for identifying the ‘optimal’ number of modes per state independently, rather than fixing the same number for each state, would be advantageous.

Finally, for the task of classification, the HMM framework was not proven to be significantly better than the simpler GMM framework. This presented a doubt over the suitability of adopting a HMM framework, over GMM, for classification of broad audio sequences.

Chapter 9

Structure Segmentation and Classification

9.1 Introduction

Video structure can include components, such as shot units, scenes, news reports and other related semantic content, which is edited together to form a logical sequence of information the viewer can understand (see Appendix C for more details). Automatic recognition of these low level semantic structures can also reduce both the time and workload for manual annotation, and in general, is important for efficient storage, browsing and retrieval of video. Automatically mapping semantic components can enable the viewer to jump straight to points of interest, in indexed video, or even skip advertisement breaks during playback[1]. More importantly, automatic labelling of each semantic unit will facilitate effective indexing of higher level concepts, such as event detection.

A main objective defined at the beginning of this thesis, was to both automatically segment and classify football videos into high level semantic structure[2]. In the previous chapters, the test collection used for model generation and evaluation had been man-

[1] See Appendix K for details on such an application.

[2] Figure 3.2 is a video model of these structures. This video model was defined through manual inspection (Chapter 3.3) and then verified by automatic content analysis (Chapter 7.5.2). The content classes, and the many sub-structures that define each class, were discovered by applying a HMM to model the entire audio track.

ually segmented and annotated. Manual segmentation is precise, allowing for robust model development and accurate evaluation, but the process is time consuming. In this chapter, the aim is to investigate methods that can automate the segmentation process without affecting precision.

To solve this problem, seven algorithms that automatically segment and classify structure found in football video, using audio information alone, are defined. There are a number of strategies for segmenting audio, given a pool of statistical models, with two main algorithm categories investigated: Model-based segmentation and Metric-based segmentation. Model-based segmentation algorithms are widely used for audio segmentation and classification, while Metric-based algorithms are popular for segmenting speech and non-speech content, for Automatic Speech Recognition systems. The Metric-based algorithm is specifically tailored for the problem of structured segmentation and classification, improving the efficiency and accuracy of the original implementation.

The reason a number of algorithms are selected, is to formally compare and evaluate the current state of the art over a large test collection, a limitation with current research into audio-based segmentation and classification. Algorithms from both strategies are evaluated across the same test collection, identifying which approach is more precise for this problem.

The remainder of this chapter is structured as follows. In section 9.2 related work into audio-based segmentation is reviewed, as well as discussing the motivation behind the selection of each algorithm. In section 9.3, a simple Model-based segmentation and classification algorithm, using maximum likelihood criteria, is introduced. In section 9.4, a Dynamic Programming approach is defined, which can be applied as a decision process for Model-based segmentation. In section 9.5, a technique called SuperHMM is outlined that concatenates a group of single HMMs into a unified framework. In section 9.6 a robust Metric-based segmentation algorithm called BICseg is introduced. Section 9.7 then presents the results of a comparative experiment, finally summarising the chapter in section 9.8.

9.2 Audio Segmentation: Motivation

The aim of this chapter is to apply statistical models to both segment and classify the structure of football video, into semantic units. There are, however, a number of issues surrounding the application of statistical models for this problem. Issues for both GMM and HMM have been addressed separately, throughout the thesis, such as model generation, parameter selection and classification. Another important problem to investigate, is how to segment the audio stream into the homogeneous units, which can then be classified by the set of models. It is believed that audio segmentation is an important phase in algorithm development, which should be addressed with care.

Provided with a set of robust model representations for known content, found in the audio track, the process of recognising the transition between one content structure and another, is important. An inefficient segmentation algorithm can create accuracy problems for the set of model-based classifiers, thus affecting future system accuracy. Therefore, a precise segmentation algorithm is required, as well as a robust set of classifiers. However, investigation into this problem for sports video, and for video indexing in general, has been limited.

In this section, a number of possible strategies are reviewed with the aim of selecting a number of algorithms, to be applied for structure segmentation and classification of football video. After studying related research into this problem, a number of segmentation algorithms are chosen. The motivation behind the selection of these algorithms is also discussed in this section.

9.2.1 Related Work

Audio segmentation algorithms can be grouped into three main categories: Energy-based, Metric-based and Model-based segmentation algorithms. In this section, these three segmentation categories are introduced, analysing the potential suitability to structural segmentation and classification.

9.2.1.1 Energy-based/Decoder Segmentation

Periods of silence in an audio stream can indicate a change in scene, breaks in commercials or a change in speaker. Energy-based segmentation algorithms segment the audio stream at the location of these periods of silence, by directly measuring a feature value, such as signal energy. A dip in energy, below a specified threshold, would indicate a silence period. Segments are then generated by cutting the input at these silence points, ready for classification.

Li et al. (2001) applied an energy-based segmentation algorithm for dividing the audio stream into silence, speech and music segments, using a two step process. The audio stream was divided into signal and silence segments, by thresholding the volume of the audio track. This approach has also been integrated into news story segmentation algorithms, such as Browne et al. (2003) and Hauptmann & Witbrock (1998) (see Appendix C for more details). Again, a threshold was used to determine significant drops in the energy or volume of the audio stream, to recognise silence breaks.

The disadvantage with applying an Energy-based approach is that these algorithms are reliant on thresholds, and are very much domain specific. For example, changes in audio quality from one video file to another would have to be reflected in the threshold setting. A low quality audio file would require a different threshold setting to that of a high quality recording, with little background noise. Consideration for threshold estimation is needed for each video file.

Within a sound controlled environment such as a Studio sequence, changes in speaker or breaks in speech are represented by periods of silence. On the other hand, during a Game sequence, the constant environmental sound would result in speaker changes accompanied with background noise.

To Summarise, Energy-based segmentation algorithms:

- are useful for segmenting changes in speaker for audio files with little background sound,
- would identify many false segments for more difficult audio environments,
- are not suitable for complex problems such as structure segmentation of high level content.

9.2.1.2 Metric-based Segmentation

Traditionally, Metric-based algorithms divide the audio track into overlapping groups of frames, called windows. Employing a window of audio frames to identify segment change provides a greater source of evidence than comparing individual frames. A distance measure is then applied to determine the similarity between two adjacent windows, where, if at a local maxima the distance value exceeds a threshold, a segment change is found. To determine segment change, a variety of distance measures can be applied, such as the symmetric Kullback-Leibler distance (KL2) (Siegler et al. (1997)) and the Likelihood Ratio Test (Gish et al. (1991)).

Metric-based algorithms are popular in ASR systems because they are relatively efficient, in terms of computation, and can be employed in real-time applications (Siegler et al. (1997)). Such algorithms have also been recently applied to problems, such as speaker detection and radio news story segmentation (Kemp et al. (2000)). However, one of the main disadvantages with such algorithms is the reliance on threshold estimation, to determine segment change. For noisy soundtracks, such as football audio, estimating a global threshold flexible to changing environments, is difficult. Many divergence measures, applied to measure the difference between windows are also noisy as well (Chen & Gopalakrishnan (1998)). Traditional Metric-based algorithms overall, do not generalise to challenging audio sequences, and estimating a threshold would not be suitable for the problem of structure segmentation and classification.

In an attempt to develop a more robust Metric-based segmentation algorithm, not reliant on threshold estimation, Chen & Gopalakrishnan (1998) converted the difference between frames into a model selection problem. The Bayesian Information Criterion (BIC), a model selection criterion[3], was applied to determine the distance between two windows, assumed to be generated from two separate Gaussian PDFs. The algorithm, although more computationally intensive than standard Metric-based solutions, was shown to be robust to difficult audio streams, for the problem of speaker segmentation (Chen & Gopalakrishnan (1998)).

[3]See Chapter 8.3.3 for further discussion on BIC.

9.2.1.3 Model-based Segmentation

For Model-based segmentation, the audio track is classified frame by frame, or by groups of audio frames, using a set of model classifiers. Boundary changes are then marked at locations where there is a change from one content class to another. The advantage of Model-based algorithms is that once the set of models for an audio stream are defined, future audio sequences can be segmented and classified in a single pass. However, for an effective algorithm, the content classes within the audio stream must be known, where new content not explained by the set of models, can generate false errors[4].

Examples of Model-based segmentation algorithms applied to segment high-level audio and video content include, Roach & Mason (2001), Tzanetakis & Cook (2002) Wang et al. (2000), Spina (2000) Xie et al. (2002), Liu et al. (1998), and Eickeler & Muller (1999). Tzanetakis & Cook (2002) use GMM classifiers to segment music into different genre such as rock and pop. Roach & Mason (2001) apply GMM models to audio to classify and segment video into genre, such as news and music content. Both algorithms classify equal sized window of frames, sequentially, using GMM classifiers. Grouping frames provides the classifier with a greater information source for determining class.

Wang et al. (2000) also used a window of audio frames for segmentation and classification, using HMM classifiers instead of GMM. The audio stream was divided into equal sized windows of overlapping audio and then HMMs were used to distinguish between News, Sport and Weather categories. For all of these algorithms, a maximum likelihood criteria was used to classify each audio unit. These segmentation algorithms are, however, only as effective as the models generated. If the model set contain inefficient representations of each content class, this can have an effect on classification accuracy, where misclassification will result in segmentation error.

To minimise the effect of false classification on segmentation accuracy when using the maximum likelihood criteria, Spina (2000) applied a median filter. A set of GMM classifiers labelled each audio into one of seven classes, such as clean and noisy speech and silence, using maximum likelihood. A smoothing technique was then applied to

[4]This is another contributing factor for content analysis and domain knowledge (Chapter 7).

correct any false classifications, where each individual frame was relabelled into the same class as the majority of surrounding frames, within a filter.

Applying a Dynamic Programming (DP) algorithm to determine the best path through the model likelihood scores, is consider a more "intelligent" decision process to maximum likelihood (Huang & Wang (2000), Xie et al. (2002)). This strategy is an example of applying a Markov chain to decide the most likely class path through a set of likelihood scores. Huang & Wang (2000) reported that using such an approach improved the segmentation results, when classifying video genre. However, there was no direct comparison with a maximum likelihood algorithm to qualify this assumption. Xie et al. (2002) also applied the same strategy for segmenting football video, which was discovered to improve segmentation, over a maximum likelihood, algorithm by 2%. The maximum likelihood algorithm did not apply a filtering process however.

A general video topology can also be developed to explain the relationship between contents classes for the purpose of segmentation and classification. Eickeler & Muller (1999) apply a further unified HMM framework to explain the relationship between a pool of HMMs, each a representation of a specific content class in news video. By using a probabilistic grammar to combine the relationship between each single HMM, a unified HMM framework called SuperHMM was defined. Then using a DP search algorithm, news video was segmented and classified in one pass. Huang & Wang (2000) discovered that applying a SuperHMM decision process improved segmentation results over a standard maximum likelihood approach. However, the results were over a very small and limited test collection.

9.2.2 Discussion

At present, there has been little comparison of audio-based segmentation techniques for the problem of structure segmentation and classification. From reviewing the literature, the most popular applied method for this problem was Model-based algorithms The only direct comparison between segmentation algorithms, from each category, has been for the problem of speech/non-speech segmentation for ASR systems, itself an open research problem. It would be difficult to determine the potential of each algorithm category for the problem of structural segmentation and classification, given the limited algorithm comparison for related problems. It was believed that an analysis of a selection of well researched algorithms, from each category, would be beneficial, providing an insight into algorithm accuracy. Thus, three Model-based approaches were chosen and one Metric-based algorithm. The three Model-based algorithms were: a Maximum likelihood, a DP search algorithm and a SuperHMM algorithm; where all three algorithms have been applied to the problem of structure segmentation and classification for video.

Model-based segmentation algorithms are the most widely applied algorithms for audio and video segmentation of content providing an elegant solution. However, there is one believed "principal drawback" when such algorithms are applied to segmentation (Dietterich (2002)). Due to the first order Markovian property, any relationship that exists between values separated by more than one time point, must be "communicated via the intervening" values. According to the Markovian assumption, the probability that the model will move from one state (or class) to another, during one time step, is governed only by the transition probability of moving from the state at the previous time point.

Therefore, as both a comparison to Model-based segmentation techniques, and to evaluate this assumption, a robust Metric-based algorithm was also implemented, specifically adapted for the problem of structure segmentation and classification. However, an identified problem with traditional Metric-based algorithms was that many implementations did not generalise across all types of audio, especially when a threshold estimate was applied to determine segment change. Chen & Gopalakrishnan (1998) extended and improved the tradition Metric-based approach by turning it into a model selection problem. This meant the decision process was determined by a model selection criterion rather than a global threshold. It is believed that this approach would be more robust for the problem of segmentation and classification, therefore the Metric-based segmentation algorithm called BICseg was also implemented, as a comparison to the three Model-based algorithms.

For each algorithm, both the HMM and GMM model frameworks were also compared as classifiers, integrated into each algorithm. These model-based classifiers are required for labelling the segmented units, with 7 algorithms compared overall. In the following sections, each strategy will be introduced, in further detail, before presenting the findings of the algorithm comparison experiment.

9.3 Maximum Likelihood Segmentation

Maximum likelihood (ML) is a standard decision process for classification. For ML, the model likelihood scores from a pool of models are returned, given a new data sequence. These scores are used to determine which model was most likely to have generated that sequence. The sequence is then classified into the class returning the highest score.

For the purpose of structural segmentation and classification, employing a ML is also straightforward. The audio stream is first divided into equal sized groups of audio frames, called windows. Each window is then classified using ML. Segment boundaries are identified at points, where there is a change from one content class to another.

For example, the classified sequence $\{SSSSSGGGGG\}$, contains a segment change at the 5^{th} window, where S represents a window classified as Studio, and G represents a window classified into the Game class.

A disadvantage with this approach is classification errors. A window that has been labelled into the wrong class, at even one time point, will result in a false boundary detected. To limit the effect false classification can have on segmentation accuracy, a number of steps can be taken with filtering being one such method. Spina (2000) applied smoothing techniques to correct classification errors that effect segmentation accuracy. For example, the sequence $\{SSSSGSSSSS\}$ has a false classification at time $t = 5$. A median filter can be applied to correct this wrongly labelled window. A median filter is another “window” that sequentially slides across each (windowed) group of frames. The current group is then relabelled into the majority class in the median filter “window”.

This simple step is also applied to correct false classifications, in the ML segmentation algorithm implemented in this thesis.

9.3.1 Implementation

There are a couple of issues surrounding the implementation of an ML algorithm. The first issue is the number of audio frames to group together for classification. This is a similar problem to what was faced in Chapter 7.3.2. The aim was to find a suitable window length that contains enough descriptive information for accurate classification. If the window is too small, not enough statistical information will be contained for accurate classification. If the unit is too large, each window could contain more than one content class. Given the results from Chapter 7.3.2, a length of one second was considered appropriate.

A second operation value is also required to be estimated; the length of the median filter window. Again, too small a window will mean small localised classification errors will not be identified. To large a window, could result in small segments being overlooked. It was decided that a median filter length of 30 seconds would be appropriate. This setting contained 15 one-second windows either side of the current window, which was believed to be wide enough for the filter, and was the same length applied by Spina (2000).

9.4 Dynamic Programming with GMM and HMM (DP)

A ML solution to segmentation and classification of structure has several disadvantages. For example, each window of audio frames is treated as an independent observation rather than a sequential stream of related units e.g. the temporal relationship between sequential windows is not considered. A popular method in video indexing to address this problem, is to apply a Dynamic Programming algorithm (DP) as part of the decision making process (Huang & Wang (2000), Xie et al. (2002)). Instead of selecting the maximum likelihood score for each time point, a DP algorithm attempts to find the best class path sequence, through the set of model likelihood scores for an entire audio sequence. This is an assumed advantage of applying a DP decision process (Xie et al. (2002)).

A popular DP algorithm is the Viterbi algorithm (Rabiner & Juang (1993)), the same algorithm that is employed for locating the optimal path through a HMM state sequence.

Applying the Viterbi algorithm to find the optimal path through a series of model likelihood scores, is an example of a first order Markov process. Instead of modelling the interaction between a set of interconnected hidden states, the DP algorithm models the temporal behaviour, or interaction, between content classes. The most likely content class is chosen rather than the class returning the maximum likelihood score.

Examples of a DP decision process employed for video indexing include Xie et al. (2002) who applied a DP algorithm to segment Play/Break sequences in football video. A DP algorithm was used to determine the best path through the model likelihood scores outputted by a HMM for the two classes. Huang & Wang (2000) also applied a DP algorithm for segmenting video files into genre. The DP algorithm employed for both papers was the Viterbi decoding algorithm. This same DP algorithm was also implemented for the problem of structure segmentation and classification, and is described in Appendix H.

9.4.1 Modelling Class Behaviour

The DP algorithm can define the behaviour between classes in a similar manner to the way restrictions can be placed in the state transition matrix, A, of a HMM (see Chapter 4.4.2.1). This is helpful, if the behaviour between content classes is known. It was recognised previously that during football video (see Figure 3.2), there is no transition from Adverts to Game sequences. During the video production, the broadcast is returned to the Studio part of the show after an Advert sequence e.g. the Studio segments act as a segue between content.

This relationship can be reflected in the DP algorithm. To do so, the transition probabilities between classes that do not directly interact can be set to zero. For example, a simple illustration would be the DP example in Figure H.1. At each time point, the DP algorithm can move from one class to another. However, given that the relationship between the three classes is known and follows the topology in Figure 9.1, where there is no direct interaction between classes 1 and 3, only an interaction between class 2 first. The DP algorithm can be implemented to reflect this relationship by setting the transition probabilities between each class to zero e.g $a_{13} = a_{31} = 0$. The DP algorithm will then follow the behaviour defined in Figure 9.2.

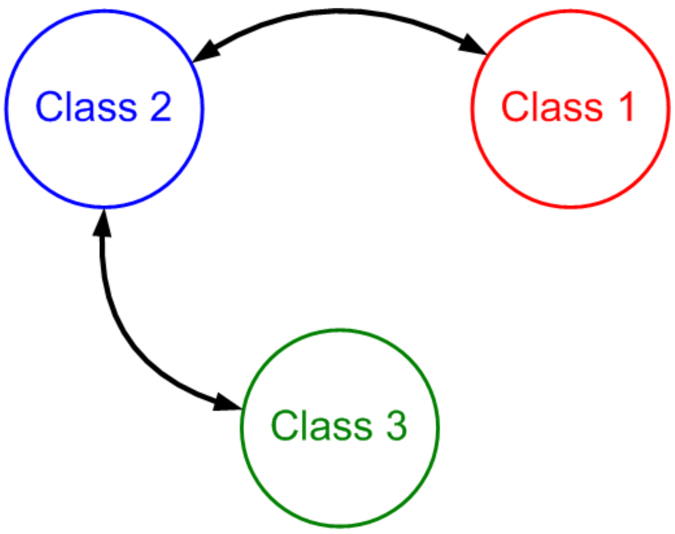


Figure 9.1: The Topology between three classes

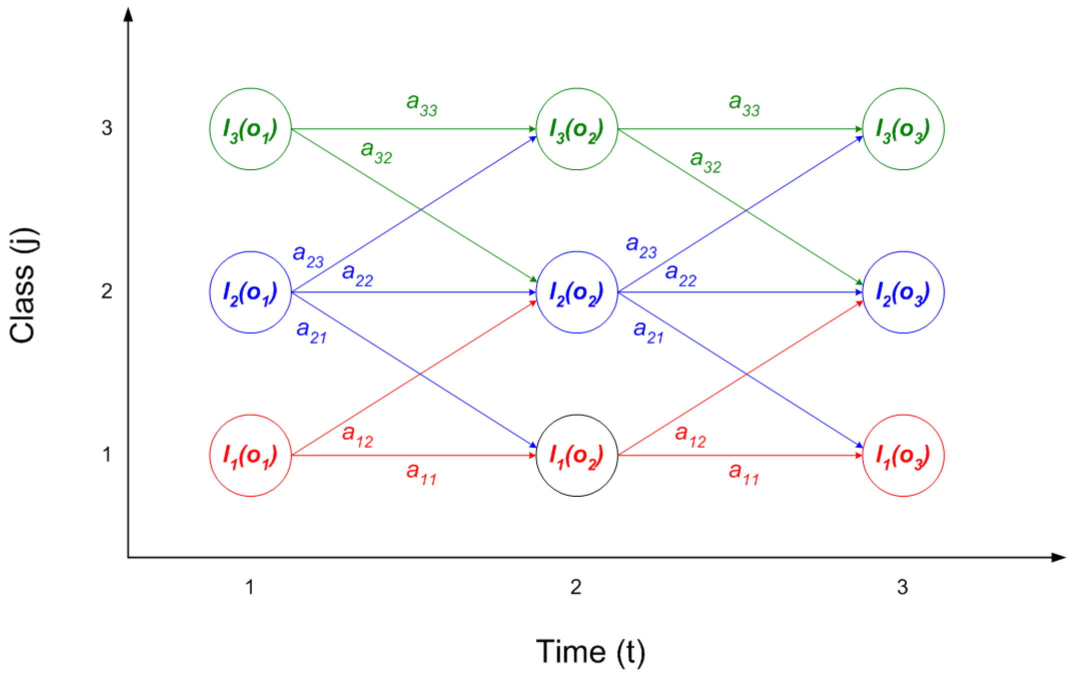


Figure 9.2: DP algorithm with restriction between class 1 and class 3.

This same procedure is followed to model the video topology identified in Figure 3.2, maximising the DP process for football video. This is another assumed advantage, the DP algorithm has over a simple ML decision process for segmentation and classification.

9.5 Super HMM

An alternative solution to the problems faced when using ML for HMM-based segmentation and classification, is to combine the classifiers into a single unified framework, also known as a Superior (Eickeler & Muller (1999)) or Super HMM (Ney & Ortmanns (1999), Huang & Wang (2000))[5]. This approach is popular in continuous speech recognition (Ney & Ortmanns (1999)), where the SuperHMM is applied for recognising complete words (or sentences) from a collection of classified phones, modelled by single HMMs. The algorithm has also been applied to the problems of video genre recognition (Huang & Wang (2000)) and news structure segmentation (Eickeler & Muller (1999)).

SuperHMM is the process of combining the single HMM models that represent each content class, into one large unified HMM. A DP search algorithm then finds both the best state and class path sequence through the SuperHMM, for a given observation sequence. For example, the phoneme models for continuous speech recognition systems are concatenated by applying a pronunciation lexicon (Ney & Ortmanns (2000)). For video indexing problems, the class models are concatenated into a SuperHMM using a preconceived video topology model of the entire video file, illustrating the relationship between content classes (Eickeler & Muller (1999)). This procedure is also followed by using the video model identified from both domain knowledge and content analysis (Figure 3.2).

This solution can be considered an optimised example of the HMM generated for representing the entire audio track (see Chapter 7). For automatic content analysis, a single HMM was generated using unsupervised training techniques, with the HMM automatically learning the structure in the audio data of football video. In the following chapters, these content structures were divided into individual classes. HMMs for

[5]For simplicity, throughout the remainder of the thesis the algorithm will be named SuperHMM.

each content class were then generated, using supervised training and model selection (see Chapter 8). The SuperHMM can now be considered a process of concatenating these separate models into one unified HMM, for the purpose of segmentation and classification. Similar to the single HMM applied for content analysis, the SuperHMM will then be able to automatically segment and label new video sequences, however, with improved accuracy.

9.5.1 SuperHMM: HMM Concatenation

The principle behind the SuperHMM, is to combine each single HMM content class representation into one large unified model. A DP search algorithm then finds both the best state and class path sequence through the SuperHMM for a given observation sequence. This process was followed, using the video model identified through both domain knowledge and automatic content analysis (Figure 3.2).

To combine the collection of HMMs $\Lambda = \{\lambda_1, \ldots, \lambda_C\}$, each representing a particular content class, domain knowledge was first applied. Figure 3.2, is an outline of the known low level structures found in a typical football broadcast video. This topology can be converted into a SuperHMM, where each state is represented by a corresponding HMM classifier 'optimally' generated. The advantage of this solution is the ability to incorporate the temporal flow of the video into the segmentation process, which is assumed will limit segmentation error.

For example, restricting movement from the Advert to Game segments (Figure 3.2), can be mirrored in the state transition matrix of the SuperHMM e.g. the state transition matrix defines the temporal flow of the video model. An input and output state is also included into the transition matrix to mark the beginning and end of a video sequence [6].

To illustrate this point, a simple example is used (see Figure 9.1). For a three class problem such as this, a fully ergodic class transition matrix A, would be:

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

[6]Each video file was recorded using a scheduling system. As a consequence, there was usually content at the beginning and end of the video file that was not part of the programme. For any evaluation, these sections were manually segmented to avoid noise in the experiment.

A is learned in a preprocessing stage using training data using equation H.2 (see Appendix H). However, in the video topology, there is no direct interaction between classes 1 and 3 (Figure 9.1). As defined in the topology model movement should be restricted within these classes. Hence, A will become:

$$A = \begin{bmatrix} a_{11} & a_{12} & 0 \\ a_{12} & a_{22} & a_{23} \\ 0 & a_{32} & a_{33} \end{bmatrix}$$

A can then be applied as a probabilistic grammar to join each single HMM into a unified SuperHMM[7]. Segmentation is then achieved by identifying both the best state and class sequence path through the SuperHMM, where each time unit in the audio sequence is assigned to a state label belonging to a specific class. Segment change is identified when the audio stream moves from one class of states to another.

For finding both the best state and class path sequence through the SuperHMM, the One-Pass dynamic programming search algorithm, defined in Ney & Ortmanns (1999), was applied. Other, more complex, search algorithms could be employed to find the best class path (Ney & Ortmanns (2000)), however, the same algorithm was effectively used by Huang & Wang (2000) for segmenting video into genre. Eickeler & Muller (1999) also applied this method for the high-level segmentation of news video. Hence, for comparison purposes with related research, this algorithm was selected.

Instead of finding the best word sequence through a sequence of phones, in a large vocabulary continuous speech problem, the algorithm was implemented to locate the best class path sequence for structural segmentation and classification of football audio. Appendix I introduces the search algorithm.

9.6 Metric-based Segmentation: BICseg

Applying a modelling framework to the problem of audio structure segmentation provides an elegant solution, but there is one believed “principal drawback” when a first order Markov process is applied for segmentation (Dietterich (2002)). Due to the first

[7]For HMM concatenation a concatenation function in the Probabilistic Modelling Toolkit was used (Cambridge Labs (2003)). For implementation reasons, each single HMM was fixed to the same number of Gaussian mixtures per hidden state, allowing for simpler mixture combining in the SuperHMM.

order Markovian property, any relationship that exists between values separated by more than one time point must be “communicated via the intervening” values. According to the Markovian assumption, the probability that the model will move from one state to another during one time step, is governed only by the transition probability of moving from the state at the previous time point. For example, any relationship that occurs during values at the time steps $t = 4$ and $t = 6$, must be communicated by $t = 5$. That is according to the Markovian assumption, the probability that the model at state j, at time $t = 6$, is governed only by the transition probability a_{ij}, where i is the true state at time $t = 5$.

Provided with this information, a suitable alternative to the Model-based segmentation was also implemented. By doing so, the algorithm could be applied to evaluate the assumption about Model-based segmentation algorithms and the Markovian property (Dietterich (2002)), as well as present an insight into whether Model-based algorithms are rightly justified as the most popular algorithms for segmentation of audio content.

A Metric-based algorithm was chosen as a comparison to the Model-based algorithms, called BICseg (Baillie & Jose (2004)). The motivation for selecting this algorithm was the success of this method as part of an event detection system for football video. The algorithm was adapted from the original implementation by Chen & Gopalakrishnan (1998), improving the efficiency and accuracy of the algorithm. From experimentation, it was discovered that BICseg was more effective than Model-based algorithms for segmenting events. The algorithm was also shown through experimentation to be robust to difficult sound environments, and is described in Chapter 10.

To apply BICseg to problem of football video structure segmentation, the algorithm required alterations. The original algorithm was designed purely for the problem of speaker and speech segmentation, with the algorithm being developed to recognise subtle changes in the audio track that would indicate the start and end of speech segments. For the problem of segmentation and classification, it was required to segment at a higher level rather than purely transitions between speech, speaker and silence. To allow the BICseg algorithm to segment high level concepts, such as transitions from Studio to Game segments and vice versa, the audio sequence was smoothed, minimising localised differences.

In the following sections, the original algorithm is first introduced (Chen & Gopalakrishnan (1998)), discussing its known limitations. Then the new implementation of the algorithm is outlined, explaining how BICseg was improved (in terms of efficiency and accuracy), and adapted for the problem of structural segmentation and classification of football audio.

9.6.1 The BIC Algorithm

As previously mentioned in Chapter 4, there are some disadvantages with applying traditional Metric-based segmentation algorithms. These disadvantages include the reliance on estimating both a threshold and the width of the search window. For these reasons Chen & Gopalakrishnan (1998) developed a more robust Metric-based algorithm, applying model selection techniques to solve the thresholding problem. The Bayesian Information Criterion (BIC) (Schwarz (1978)) was employed as a decision process to determine segment change. The advantage of this approach over other Metric-based segmentation schemes is that it is threshold free, and hence its ability to generalise. The algorithm also implemented an expanding search window to include as much information as possible for model selection, thus avoiding the problem of window length estimation. On the downside, the algorithm is not as efficient as traditional Metric-based algorithms, and can not be so easily integrated into real time system (Meinedo & Neto (2003), Ramirez et al. (2004)).

Other model selection criterion, such as the Akaike Information Criterion (AIC) or Minimum Description Length (MDL) could be used instead of BIC, for the decision process. However, in a large scale experiment, Cettolo & Federico (2000) compared six model selection criteria including BIC, AIC and MDL for the problem of audio segmentation. No significant gains in accuracy were achieved from implementing an alternative criterion to BIC. It would be inaccurate to assume that this result would be transferable to other test applications, such as that used in the thesis. But it was considered that BIC was a good starting point before possible future research into alternative selection criteria.

Now the original algorithm by Chen & Gopalakrishnan (1998) is introduced, before explaining how it can be adapted for the problem of structural segmentation and classification.

9.6.1.1 Detecting Segment Change with BIC

The same notation as Chen & Gopalakrishnan (1998), is now used to introduced the algorithm.

The audio sequence X, where $X = \{x_i \in \mathbb{R}^d, i = 1, \ldots, N\}$, is a sequence of feature set vectors extracted from an audio clip with length N. It is assumed that the sequence X is drawn from an independent and identically distributed (i.i.d) multivariate Gaussian process,

$$X = x_i \sim N(\mu_i, \Sigma_i)$$

where μ_i is the mean vector and Σ_i is the covariance matrix of the sequence X. To illustrate how to detect one segment change, a hypothesis test for a change occurring at time i is defined. The null hypothesis is that the sequence X does not contain a segment change and was therefore generated from a single Gaussian process. The alternative hypothesis is that the audio sequence X was generated from two separate Gaussian processes, where the change from one process to another is found at frame i.

$$\begin{aligned} &H_0 : x_1 \ldots x_N \sim N(\mu, \Sigma) \\ &H_1 : x_1 \ldots x_i \sim N(\mu_1, \Sigma_1); \quad x_{i+1} \ldots x_N \sim N(\mu_2, \Sigma_2) \end{aligned}$$

The maximum likelihood ratio statistic to determine segment change is,

$$R(i) = N \log |\Sigma| - N_1 \log |\Sigma_1| - N_2 \log |\Sigma_2| \tag{9.1}$$

where Σ is the sample covariance matrix for the model of all the data $\{x_1, \ldots, x_N\}$, and Σ_1 and Σ_2 are the covariances[8] for the two independent segments $\{x_1, \ldots, x_i\}$ and $\{x_{i+1}, \ldots, x_N\}$. The maximum likelihood estimate of the changing point can then be defined as,

$$\hat{t} = \arg\max_i R(i) \tag{9.2}$$

[8]To avoid parameter estimation problems with the original algorithm, if the number of audio frames for model estimation was less than or equal the dimension of the model, the covariance matrices were constrained to be diagonal.

This hypothesis test can be viewed as a model selection problem, where there are two competing models. The first model is a single Gaussian representation of the entire data sequence (H_0). The remaining model (H_1) is two separate multivariate Gaussians, where each Gaussian is a representation of a non-overlapping homogeneous segment. Using the BIC criterion, the difference between the two models can be expressed as:

$$BIC(i) = R(i) - \gamma P \tag{9.3}$$

where the likelihood ratio $R(i)$ is defined by (equation 9.1) and γ is the penalty weight. According to BIC (Chen & Gopalakrishnan (1998)) the penalty P can be defined as:

$$P = \frac{1}{2}(d + \frac{d(d+1)}{2}) \log(N) \tag{9.4}$$

N is the size of the decision window and d is the dimension of the feature space. The BIC criterion is a replacement for an empirically set threshold of the log-likelihood distance (Gish et al. (1991)). The new threshold being automatically chosen by equation 9.4. If equation 9.3 is positive, modelling the data as two separate Gaussian models would be optimal. A segment change can then be determined when,

$$\{\max_i BIC(i)\} > 0 \tag{9.5}$$

where the boundary change will be found at frame,

$$\hat{t} = \arg\max_i BIC(i) \tag{9.6}$$

Before outlining how the algorithm can be used for locating multiple segments within an audio sequence, the problems surrounding the original algorithm are now discussed.

9.6.2 Adapting the Algorithm

There were a number of limitations with the original implementation of the BICseg algorithm that were addressed (Chen & Gopalakrishnan (1998)). The original algorithm suffered from a number of problems including efficiency and its tendency to over segment (insertion errors)[9]. Also many boundaries (deletion errors) were missed between adjacent segments that were below a certain length. These small segments contained unrelated content that was thought to be one continuous segment and was overlooked by the segmentation algorithm.

9.6.2.1 Algorithm Efficiency

The original algorithm employed an expanding search window to find segment change, BIC was then used to detect segment change for each frame in the window. If segment change was not found within the search window, the window would expand by adding the next sequential frame. After each boundary search cycle, the search window would grow, frame by frame, until a segment change was found. Once a segment change was found, the search window would be reset at the location of the boundary change. This procedure allowed for as much data as possible to be used in the decision process. On the downside, the algorithm was slow and computationally expensive.

A number of steps were taken to address the original algorithm efficiency. The first step was to place a limit on the maximum window length. For the original algorithm the search window sequentially expanded after the completion of each cycle until a segment change was found. A new audio frame is added to the search window until a boundary is found. In practice, the algorithm could linearly searching through a potentially huge window. By placing a limit on the size of the search window, the algorithm searching through extremely large windows can be prevented. In the new implementation, the window is prevented from exceeding w_{max} frames.

Expanding the window, frame by frame, was also expensive. Searching through large volumes of audio data, such as football audio, is very time consuming. Instead of sequentially increasing the window, one data vector x at a time, more than one frame was added to the search window after the completion of each search cycle. The search window now added $w_{skip} > 1$ data vectors after each search cycle.

[9]An insertion error is when a segment change is detected that did not exist. A deletion error is when a segment boundary is missed.

9.6.2.2 Algorithm Accuracy

Insertion and deletion errors were often the result of poor parameter estimation in the Gaussian models. Poor estimation was usually a result of limited data required to produce robust statistics. For example, the algorithm searches for segment changes in a data sequence $X = \{x_i \in \mathbb{R}^d, i = 1, \dots, N\}$, using a search window $[w_l, w_r]$, where $w_l = \{1, \dots, N-1\}$ and $w_r = \{2, \dots, N\}$, $w_r > w_l$. There was no minimum limit enforced on the size of the search window. When the search window was too small, segments were missed because of insufficient data to adequately develop each Gaussian model. The accuracy of the BIC decision process is dependent on how well each Gaussian model represents its corresponding segment.

A number of alterations were made to the original algorithm to avoid parameter estimation problems associated with the original algorithm. First, situations were minimised when model generation occurred using small amounts of data. To do so, a limit was placed on the minimum size of the search window. The window would be initialised and following each boundary change thereafter, to length w_{min}.

A limit was also placed on searching at the beginning and end of the window. i.e. the extremes of the window were not searched. This would allow for a minimum buffer of audio frames to be used for estimating the Gaussian model(s), when searching at the start and end points of each window, w_{buff} seconds from each extreme. Each opposing Gaussian model would then have a minimum number of audio frames ($w_{buff} > 1$) to estimate Σ_1 and Σ_2, as opposed to potentially one audio frame.

To further limit the problem of poor parameter estimation. The covariance matrices were also constrained to be diagonal due to the limited data often available for estimation. The adapted BICseg algorithm for detecting multiple segment boundaries in an audio sequence is now formally introduced.

9.6.3 The adapted BICseg Algorithm

1. Initialise the interval $[w_l, w_r]$
   $w_l = 1$; $w_r = w_r + w_{min}$
2. Using BIC, search for a segment change in the window

```
   $[w_l + w_{buff}, w_r - w_{buff}]$
3. **if** there is no segment change in the window
   $[w_l + w_{buff}, w_r - w_{buff}]$;
       **if** $(w_r - w_l) \leq w_{max}$
           $w_r = w_r + w_{skip}$;
       **else**
           $w_l = w_l + w_{skip}$;
           $w_r = w_r + w_{skip}$;
   **else**
       set $\hat{t}$ as a segment change boundary;
       $w_l = \hat{t}$; $w_r = w_l + w_{min}$;
   **end**
4. **goto** 2. until **end**
```

9.6.4 Variable Estimation

There are a number of operation variables for the new BICseg algorithm to be estimated, such as the minimum and maximum length of the search window $[w_l, w_r]$. Before these variables were estimated, the audio sequence was smoothed, minimising the effect of localised differences.

9.6.4.1 Smoothing the data

The original algorithm introduced by Chen & Gopalakrishnan (1998), was designed to segment speech from non speech segments as part of an ASR system. In previous work, the adapted BICseg algorithm was also applied as part of an event detection system (see Chapter 10). For both these applications, the segmentation algorithm was employed to identify subtle changes from one homogeneous content class to another, such as speech and silence, or changes from one speaker to another. This was achieved by comparing the differences found in audio frames. However, for the problem of structure segmentation and algorithm, it was of more interest to detect changes from one semantic structure to another, such as those content classes found in Figure 3.2.

To minimise the effect subtle changes would have on the algorithm (such as changes in

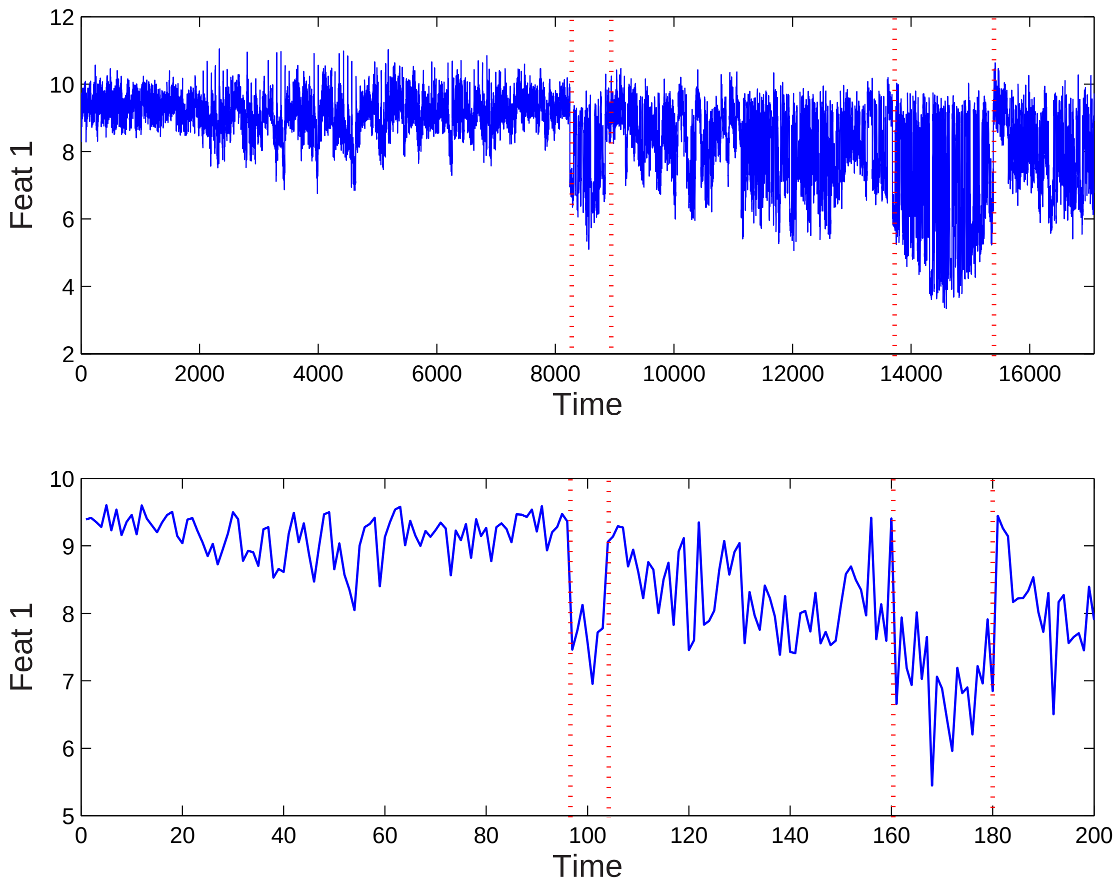


Figure 9.3: Smoothing the data.

speaker), the audio sequence was smoothed. The same approach was employed as in Chapter 7.3, by dividing the audio stream into equal sized one second units. The mean of each one second group of audio frames was then taken as a representation for each window.

The BICseg algorithm was then applied to locate the differences in the sequence of mean data vectors $\bar{x}$, each a point estimate of a one second window. It was assumed that by using the mean measurement, would smooth the effect local variation has on the algorithm. This was considered important for the task of segmentation and classification, because the BICseg was originally designed for detecting breaks between segment changes of fine granularity, such as changes in speaker.

Smoothing the data also minimised the effect these localised differences might have on the algorithm accuracy, when applied to this problem. Figure 9.3 is an example of an audio sequence that has been smoothed. The top plot is the audio stream after feature extraction. The bottom plot is the same audio sequence displaying the mean data vector for each one second unit. From analysing this figure, it can be assumed that smoothing the audio stream had minimal effect on locating segment boundaries (shown by the dotted lines). Smoothing the data also minimised the localised differences within these segments and compressed the data sequences, aiding the efficiency and accuracy.

Finally, once a potential boundary was found, the BICseg algorithm was then applied to the one second unit locating the exact audio frame the boundary is located at, retaining the same accuracy as the SuperHMM. Both the ML and DP Model-based segmentation algorithms were not as precise at finding the exact frame location as this algorithm.

9.6.4.2 Algorithm Variable Estimation

To estimate the BICseg algorithm variables, statistics from 12 files set aside for system development were calculated[10]. Each audio file was manually segmented into relevant content classes (Figure 3.2) and then the distribution of segment lengths for the entire test collection were calculated (Figure 9.4). The sample of videos contained 134 segments overall, approximately 11 per video file. The minimum segment length was 16 seconds, an interview straight after the game. The maximum length was 3556 seconds, 1 hour in length. This was one half of a football match. The median segment length was approximately 3 minutes, but it is clear from the exploratory analysis (Figure 9.4), that there was two main groups; the Game sequences (the group on the right of the histogram), and the remaining content classes (the group on the left). The Game sequences were 50 minutes on average in length, while all other segments ranged from 30 seconds to 4 minutes on average. The main group contained only one non Game sequence, a Studio sequence that was 50 minutes in length. The build up to an important match.

9.6.4.2.1 Operation Variables Using this as a guide, the minimum window (w_{min}) size was set to 30 seconds in length. The window would be wide enough to contain only two small segments, but also large enough to contain sufficient discriminatory information for model generation. It was noticed that the smaller segments of 30 seconds

[10]This data set was not included during the final evaluation of all segmentation and classification algorithms, to avoid overlap and algorithm bias.

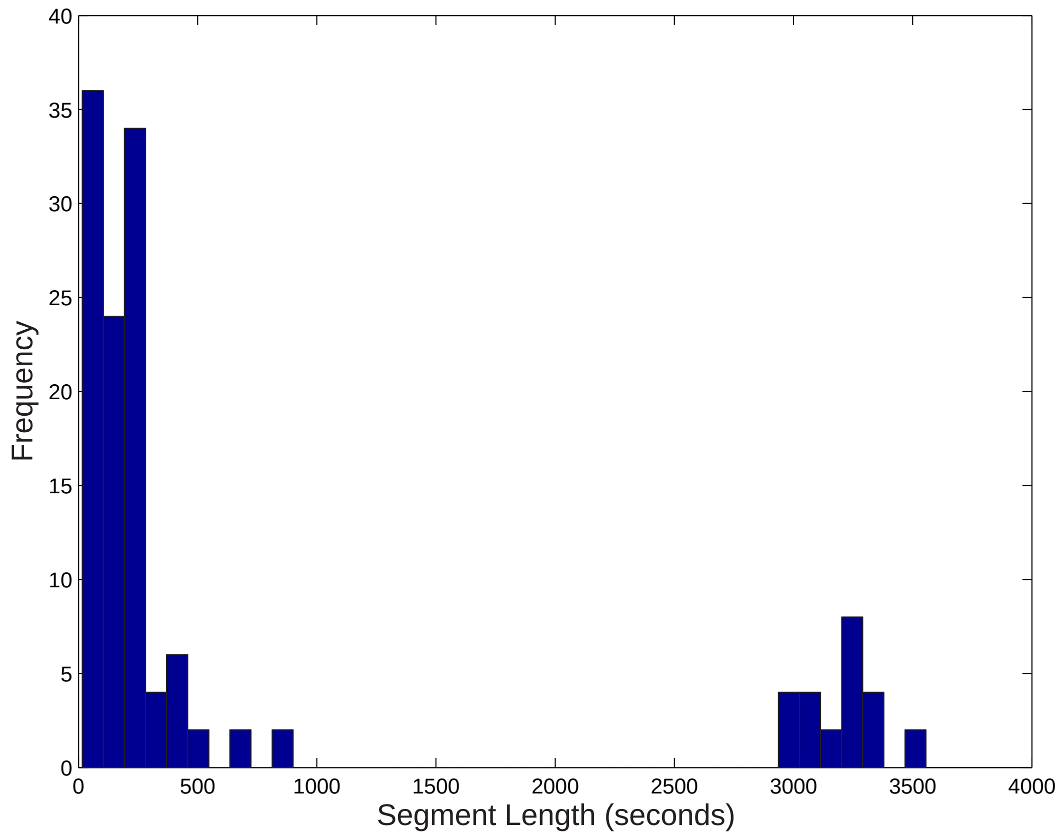


Figure 9.4: Segment length distribution for 12 video files.

or less in length, were always sandwiched between longer segments. These smaller segments belonged to Studio sequences linking Advert, Game or Interview sequences. The maximum window length (w_{max}) was set to 3 minutes, or 180 seconds. The median length of the segments was used as a maximum limit for the window size to grow.

Efficiency or speed was another concern of the original BICseg algorithm. The window would move sequentially from one time point to another, in this case second to second. It was assumed that the smallest segment would not be smaller than 5 seconds. It was thought that 5 seconds would be the minimum length of time to convey important information to the viewer. w_{skip} was set to 5 seconds. Searching at the extremes was restricted in the first and last 8 seconds (w_{buff}) of the search window. This was believed to be the absolute minimum amount of information required for adequate estimation. 8 seconds was half the length of the minimum segment length.

9.6.4.2.2 Penalty Weight Estimation After defining these algorithm parameters, an appropriate value for the penalty weight γ was investigated. By adjusting γ, greater or less emphasis is placed on the effect of model complexity. For the original algorithm, Chen & Gopalakrishnan (1998) set $\gamma = 1$, which is the strict definition of BIC. Tritschler & Gopinath (1999) investigated the algorithm further for speaker segmentation and discovered that $\gamma = 1.3$ produced better results for speaker segmentation. Delacourt et al. (1999) also found increasing $\gamma > 1$ improved segmentation for the same problem.

The goal for our segmentation system was to find changes in semantic content, such as a transition from a Game to Studio sequence. The overall aim of the algorithm was to minimise the problem of over-segmentation[11], which the original algorithm suffered from, but not at the expense of missing important segment boundaries. It was assumed that over-segmentation could be corrected during classification. For example, a false boundary change within a Game segment caused by a decrease in crowd sound could be corrected during the classification phase, if both segments are then classified into the same class.

So to estimate γ, truth data was generated from three video files. The segment boundary locations for each file were manually marked, where a boundary was identified as a change from one content class to another. The BICseg algorithm was then applied for each video file, incrementally increasing the parameter γ in 0.1 steps, e.g. $\gamma = \{1.0, 1.1, \ldots, 2.0\}$. The number of insertion and deletion errors were noted for each parameter value.

Figure 9.5 presents the results from the experiment for each video file. The solid blue line represents the insertion error rate, while the red dotted line represents the deletion error rate. The x-axis represents the increase in the penalty weight γ.

The insertion error rate was high for initial values of γ, which then steadily declined as the penalty weight increased. After $\gamma = 1.6$ the deletion error began to rise, as γ was increased (Figure 9.5). It was discovered that $\gamma = 1.6$ maximised the desired balance between insertion and deletion errors. It was of interest to limit the number of deletion errors, because it was believed that insertion errors could be corrected during segment classification. Increasing γ generated further deletion errors, while decreasing

[11]Over-segmentation is when the algorithm detected a false transition.

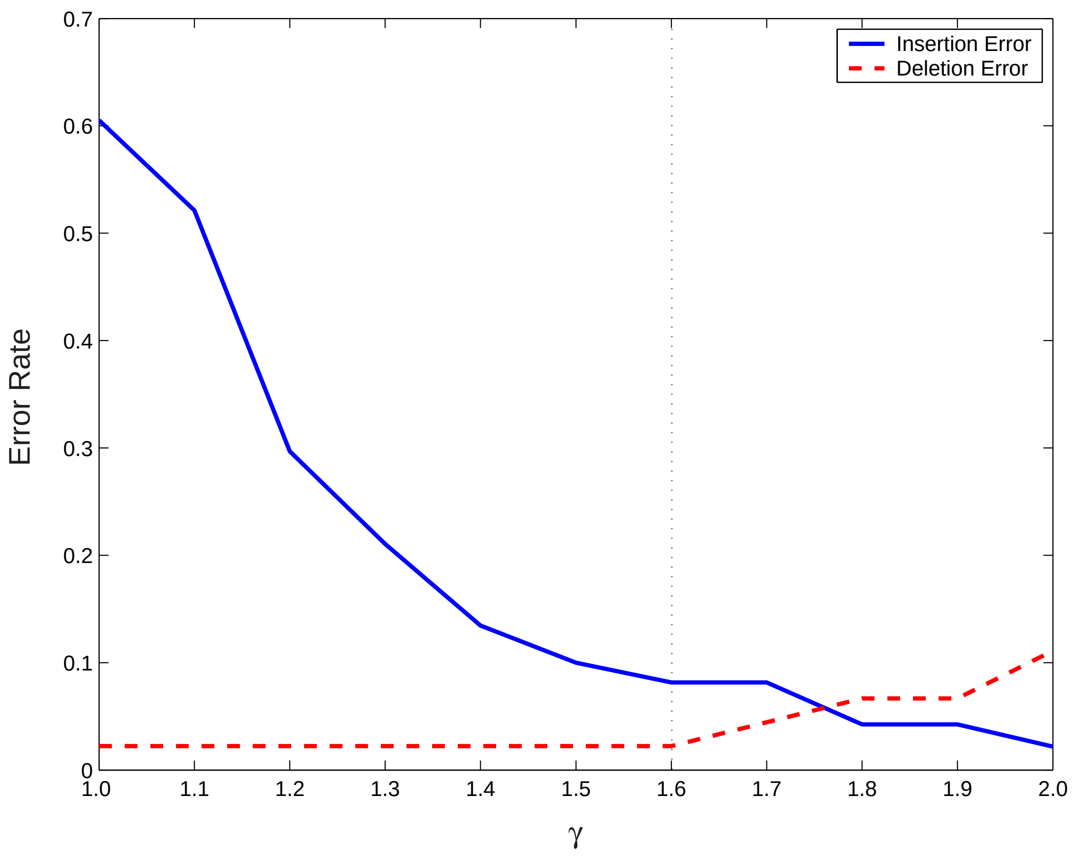


Figure 9.5: The overall effect on error rate by adjusting the penalty weight γ for the BICseg algorithm.

the penalty weight, increased the number of insertion errors.

9.6.5 Classification

After segmentation, either a set of GMM or HMM models can be used to label each segment. A simple maximum likelihood decision process was used for classifying each segment, where the segment would be labelled into the class returning the highest likelihood score.

9.7 Algorithm Comparison Experiment

In this section, the experimental procedure to compare all algorithms across the same test collection is defined. First the content classes that are to be segmented, in the test collection, are defined, in section 9.7.1. The experimental is outlined in section 9.7.2, and then the final results are presented in section 9.7.3.

9.7.1 Video Model

For each video file, the number of content classes were reduced, removing both the opening and closing titles. This was done for two reasons. The first reason was, that for the test collection video files were recorded from two different broadcast channels. Therefore, the opening and closing sequences differed. The second for reason was that the video files were recorded using a scheduler, similar to an automatic recording device on a home VCR. Often, the start or end of the football broadcast was missed as a result of this schedular.

In this experiment, it was of interest to recognise the broad content structures in a video sequence, and not specific sub-structures. It was thought that sub-structure, such as exciting events and national anthems would be more accurately recognised after this first phase of segmentation, by applying domain knowledge (see Chapter 10). Therefore, these sub-structures were integrated into one overall Game class.

The final video model used to compare each algorithm is displayed in Figure 9.6. Each video file was edited so that it started and finished on the opening and closing studio sequences, also removing the opening and closing titles, and other content not related to the football broadcast. The arrows indicate the relationship between each class. For example, at the end of an Advert sequence, the temporal flow of the video would not enter any other class than Studio.

Both the DP and SuperHMM algorithms were developed using this video model for defining the temporal behaviour between content classes. Also, for each content class displayed in Figure 9.6, a set of both GMM and HMM models were generated for classification.

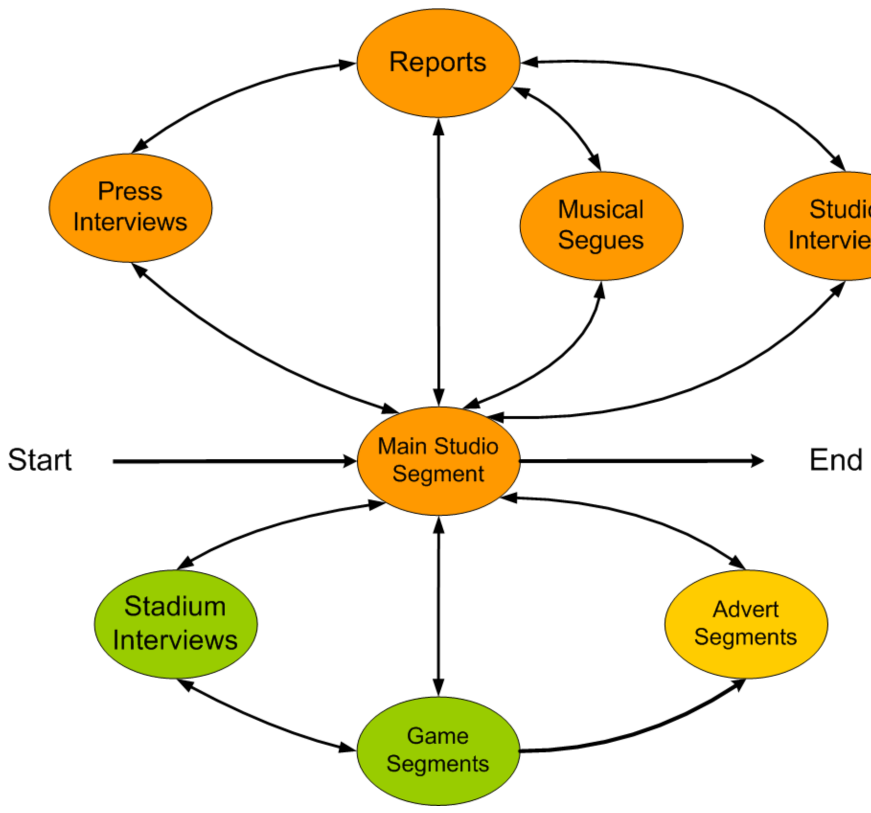


Figure 9.6: The redefined video model used for the segmentation and classification experiment.

9.7.2 Experiment

Seven segmentation and classification algorithms were evaluated across 12 new video files. Each segment change was manually annotated and the content class recorded for each segment. Segment boundaries were examined fuzzily as slight inaccuracies were expected due to the precision of the manual tagging, and alignment differences in the actual algorithms. The exact location of some segment changes was subjective, so each algorithm was given a small allowance of frames between detected boundaries. It was assumed that a system could tolerate a small degree of inaccuracy between segments. For the experiment, 2 seconds of error was allowed for each segment change, where a correctly detected boundary change had to be within 2 seconds of the true segment change that was manually tagged in the test collection. Increasing the window beyond 2 seconds would improve the accuracy of algorithms that over-segmented, possibly providing a false impression of the algorithm accuracy.

For a fair comparison, both the GMM and HMM used for each algorithm were gen-

Scheme	Code
Segmentation with HMM maximum likelihood	HMMmax
Segmentation with GMM maximum likelihood	GMMmax
Segmentation with HMM and Dynamic Programming	HMMDP
Segmentation with GMM and Dynamic Programming	GMMDP
Segmentation with Superior HMM	SuperHMM
Segmentation with BICseg and classification using HMM	BICHMM
Segmentation with BICseg and classification using GMM	BICGMM

Table 9.1: Segmentation and classification legend for all techniques.

erated using the same training collection. For each class, outlined in Figure 9.6, both GMM and HMM models were optimally generated from labelled data gathered from the video files set aside for system development. The GMM classifiers were generated using a bottom up approach outlined in Chapter 4. For the HMM models the BIC model selection was applied, Chapter 8.

The number of insertion and deletion errors were recorded to compare each algorithm. An insertion error is when a segment change is detected that did not exist. A deletion error is when a segment boundary is missed.

9.7.2.1 Algorithms

Seven algorithms in total were compared across the same test collection. Table 9.1 provides a summary of these algorithms.

9.7.3 Results

The results of the experiment are presented in Table 9.2. In the first column are the overall insertion error rate for all 12 video files. The second column is the overall deletion error rate. For both BICseg algorithms (BICHMM and BICGMM), the error rate before classification is presented on the left, and after classification on the right.

The most accurate segmentation algorithms were found to be the two BICseg algorithms, both in terms of insertion and deletion errors. Both BICseg algorithms per-

Scheme	Insertion Error	Deletion Error
HMMmax	0.4714	0.2062
GMMmax	0.4409	0.1856
HMMDP	0.3379	0.2062
GMMDP	0.4031	0.1856
SuperHMM	0.5698	0.0979
BICHMM	0.2945 / 0.1222	0.0773
BICGMM	0.2945 / 0.1060	0.0773

Table 9.2: Results for the segmentation algorithm comparison experiment).

formed better than the Model-based segmentation algorithms, either using GMM or HMM as a labelling framework. Even before the classification phase, the BICseg algorithm produced lower insertion and deletion errors.

In terms of deletion errors, it was discovered that SuperHMM was the best performing Model-based algorithm. This is the more important measure, as missed true boundaries are more difficult to resolve with further processing than insertion errors. The SuperHMM was approximately 9% more accurate than both the DP and ML algorithms. However, the SuperHMM algorithm tended to over-segment more than both the DP and ML algorithms, 23% more than HMMDP. There did appear to be a gain in accuracy by searching at the state level for the SuperHMM, however, this did result in problems in precision.

In terms of deletion error, the GMMDP algorithm was more accurate than HMMDP, although GMMDP did over-segment more. Both the ML algorithms displayed similar performance, in terms of deletion error, when compared to the corresponding DP algorithm. However, both the ML algorithms produced a higher insertion error rate, than the DP counterpart, 47% compared to 34% for the HMM algorithm (HMMmax versus HMMDP), and 44% error rate compared to 40% for the GMM algorithm (GMMmax versus GMMDP). There was not a considerable improvement in performance applying the more complex DP algorithm, compared to the simpler ML strategy, thought to be a more intelligent decision process (Huang & Wang (2000), Xie et al. (2002)).

9.7.3.1 Missed Boundaries

A large number of the insertion errors were caused by Musical Segues or pre-produced reports, containing both music and speech. These sequences were often classified as Advert sequences, resulting in a false transition. Another source of segmentation error was music played inside the stadium, such as the anthems of each competing nation. The occurrence of national anthems were often classified as Advert sequences. It was thought that the training data, for the Game classifiers, did not fully represent this occurrence, resulting in these segments being more similar to Advert sequences.

A large proportion of missed boundaries were also created by transitions from a Studio sequences to pre-recorded Interview sequences. The similarities in sound quality and environment resulted in these transitions being overlooked. This problem would not have a great impact on future system accuracy as the semantic relationship between these components is close. If these components were not so directly related then this would result in future problems.

9.7.4 Results Discussion

For the sole purpose of segmentation and classification of football video structure, Model-based algorithms were shown not to be as effective as BICseg, a robust Metric-based algorithm. For video indexing, Model-based algorithms are widely applied (Huang & Wang (2000), Eickeler & Muller (1999), Wang et al. (2000) and Xie et al. (2002)), where Metric-based algorithms are rarely (if at all) applied for such a problem. These results would indicate that a more in-depth comparison of both Model-based and Metric-based algorithms is required, with a robust Metric-based algorithm performing considerably better. This could be due to the "principal drawback" when applying a first order Markov process for segmentation Dietterich (2002), where any relationship that exists between values separated by more than one time point must be "communicated via the intervening" values.

Another advantage of the BICseg algorithm is that it does not require previous domain knowledge to segment structure. The algorithm compares only disparages between sequential audio frames within a search window. Therefore, the algorithm can generalise to future football sequences more accurately. Model-based algorithms rely on a complete knowledge of the content classes that constitute a video sequence. Hence, new content, not described by the set of models, will often result in segmentation error.

Applying a segmentation scheme first, has its advantages over a model-based segmentation approach. For example, over segmentation problems can be addressed by joining wrongly segmented content into one unified segment during classification. Whereas, employing a Model-based scheme alone for both classification and segmentation means that segmentation errors will occur during false classification. Both the ML and DP algorithms required the audio stream to be divided into equal sized groups of frames. Many of these groups will contain more than one content class, creating potential classification errors that can only be corrected with further processing or filtering.

An assumed disadvantage of the SuperHMM, over using DP, is the dimensionality of the transition matrix in the SuperHMM (Xie et al. (2002)). For a SuperHMM, the transition matrix A will be larger when compared to a DP algorithm, employed to search through a series of model likelihood scores (section 9.4). For the DP algorithm, only the relationship between content classes is modelled, however, for the SuperHMM, both the relationship between each class, and the hidden state interaction within each content class is searched. Therefore, the search space for the SuperHMM, for the same problem, is considerably larger.

Applying a DP algorithm, over a ML decision process, did improve the accuracy of the Model-based algorithms, but only by a small percentage. The SuperHMM algorithm identified more correct segment transitions than the other Model-based algorithms but had a tendency to over-segment. This result challenged the claim that by searching at the state and class level in a SuperHMM will not be as accurate, when compared to a DP solution (Xie et al. (2002)). In terms of insertion error, the DP algorithm produced better results, but missed considerable more true boundaries. This would have a more adverse effect on overall system accuracy.

Overall, Model-based algorithms were found to be only as successful as the weakest classifier. As shown from the results, this problem can result in a high rate of misclassification between classes and over segmentation.

A final, interesting, result was that for this problem, the GMM classifiers were found to be as accurate, and in many cases more so, than applying a HMM model set for labelling content. Thus, for the purposes of classifying broad content classes, such as those found in video, the GMM framework would appear to be a more suitable starting point, than the more complex HMM.

9.8 Summary

In this chapter, seven segmentation and classification algorithms were compared. It was found that a Metric-based algorithm adapted to the problem of structural segmentation and classification of football video, performed better than widely applied Model-based algorithms. The algorithm was improved on the original implementation, reducing the computational costs and improving accuracy. The advantage of this new robust Metric-based algorithm, named BICseg, was that during the segmentation process no assumption on content was required. The Metric-based algorithm is not reliant on all content classes being covered by the model set for accurate segmentation. For Model-based algorithms content classes must be known and have strong representations for accurate segmentation. This result indicates that applying a robust Metric-based segmentation algorithm will provide a more precise solution than the more widely applied Model-based strategies.

Chapter 10

Event Detection

10.1 Introduction

In the previous chapter, the main structure that defines a football video file was segmented and then labelled according to a predefined topology, mapping the general content. This was an important pre-processing step that is required before the indexing of higher-level concepts can begin. This point is illustrated in this chapter, where the goal is to develop an event detection algorithm specifically designed for Game sequences. Event detection is important for automatic highlights generation, as well as indexing content for reuse[1].

Event detection is the process of locating the key moments in a football sequence including incidents, such as a goal, an exciting passage of play, a caution, etc (see Chapter 3.5 for further details). Concentrating specifically on Game sequences can enable the event detection algorithm to be optimised using domain knowledge. The main reason for this assumption, is that it would be difficult to apply domain knowledge across an entire video file. For example, many similar patterns can be generated from unrelated sequences, such as Studio and Advert. These similar patterns can affect the accuracy of the event detection algorithm, where a sequence of 'canned laughter' (or equivalent), can be wrongly classified as a key event. Applying an sports event detection algorithm on content such as advertisements is also inefficient, in terms of time and resources.

[1]Appendix K provides an illustration of an application integrating an event detection algorithm.

For the task of events detection, a statistical approach is adopted for recognising audio-based patterns that are related to excited crowd reaction. It is assumed that supporters inside the stadium react to different stimuli, such as a goal, an exciting passage of play or even a poor refereeing decision during a match by cheering, shouting, clapping or booing. Also, changes in the emotion of the commentator(s) voice is another important information source, when recognising key events. As described in Chapter 3.5, both of these clues are important reference points that can be correlated to key events.

In this chapter, six audio pattern classes are introduced, which are considered important for event detection. These classes are defined using the findings of the automatic content analysis experiment (see Chapter 7.5.2). For each class, statistical models are then generated comparing both GMM and HMM frameworks, using training data manually segmented for each class. Manual segmentation of the audio track ensured precise training samples, allowing for robust statistical representations as possible, for each class.

Manual segmentation can provide precise training samples, but for event detection, an automatic segmentation algorithm is required, hence three audio segmentation algorithms are evaluated. These algorithms divide Game sequences into homogeneous segments that can be labelled into one of the six predefined groups. The first two segmentation algorithms are Metric-based, widely used in Automatic Speech Recognition (ASR) systems (Siegler et al. (1997), Chen & Gopalakrishnan (1998)). Both segmentation algorithms are investigated and adapted for the problem of segmentation in nosy audio environments, associated with Game sequences. One algorithm applies a distance measure to compare sequentially sliding windows of audio frames. A robust threshold is defined that is a function of the activity of the audio signal. This is an improvement over current global threshold implementations.

The second Metric-based segmentation algorithm, is an implementation of the BICseg algorithm introduced in the previous chapter. The remaining technique is a Model-based segmentation and classification algorithm, popular in current event detection systems (Rui et al. (2000)). The segmentation accuracy of all algorithms, is then evaluated over a test collection of Game sequences.

To provide an insight into the remainder of this chapter, the structure is as follows. In section 10.2, previous research into event detection for sport is reviewed. In section 10.3 the predefined content classes (or sub-structures) found in Game sequences are then defined in detail. The two statistical models GMM and HMM are also formally compared for each content class in this section. The problem of automatically segmenting noisy Game sequences accurately, before event detection, is addressed in section 10.4, comparing three segmentation algorithms. Finally, a number of event detection approaches, integrating these segmentation algorithms, are evaluated in section 10.5. The chapter is concluded in section 10.6.

10.2 Related Work

The have been a number of investigations into audio-based event detection for Sport. These include heuristic frameworks (Nepal et al. (2001), Chang et al. (1996)), template matching (Chang et al. (1996)) and model-based segmentation and classification algorithms (Rui et al. (2000), Baillie & Jose (2003), Xiong et al. (2003*b*), Xiong et al. (2003*a*)). Nepal et al. (2001) defined a set of heuristic rules for detecting scoring events in Basketball video sequences. Periods of crowd cheering were recognised by placing a threshold on the ‘loudness’ of the audio track. This is a simple solution, although the strategy can be problematic e.g. placing a threshold on the ‘loudness’ of audio can generate a lot of error. As highlighted in Chapter 5, the ‘loudness’ feature by itself can not discriminate well between various content, and in particular, between complex patterns such as speech, crowd cheering and music. A set of heuristic rules were then defined acting as a guideline for the detection of scoring attempts. These rules included, that a scoring event will occur “three seconds surrounding a crowd cheer”. The disadvantage with defining such rules, is that both the heuristic rule set and the threshold parameter will not generalise across different Basketball games. For example, defining a loose guideline such as “three seconds” will not find a precise location of scoring events, to the closest audio frame. Also, the precision of the algorithm is questionable, especially during extended periods of crowd reaction, or other sound sources, which increase the ‘loudness’ feature value beyond the predefined threshold. Such occurrences are not always correlated to scoring events. From evaluation, 15 out of 16 goal events were detected using the crowd cheering detection algorithm for two test sequences, although the precision of the algorithm was found to vary from 55% to 64% between the two sequences.

Chang et al. (1996) also defined a heuristic set of rules for detecting “touchdowns” in American football, where events were detected by recognising crowd cheering sequences and keyword spotting. Periods of crowd cheering were found automatically if the energy of the audio signal exceeded a predefined threshold. Again, the disadvantage of placing a threshold on physical feature value will not allow for discrimination between varying sounds, which increase the energy. This approach would not discriminate between crowd cheering and other high-energy sounds such as speech, music and crowd noises not related to key events. Classes not directly related to key events can all increase the energy of the audio signal, and as a result, probably trigger false events. Applying a global threshold will also result in generalisation problems. Due to changes in environmental and background noise across different games, the global threshold for one match will not always generalise in others.

Template matching was also employed to spot keywords associated with key events, such as a “touchdown” (Chang et al. (1996)). Keyword spotting is a difficult task, especially in noisy audio tracks such as those found in sport (Spina (2000)). One particular problem is that the vocabulary of words that could be associated with key events is large, and will vary between games and commentators. Keywords, such as “goal” or “touchdown”, are often spoken throughout the game, and are not limited to specific key moments. However, keyword spotting applied alongside other information sources such as the excitement in the commentators voice, and crowd cheering detection, could help accuracy (see Chapter 3.5). Both are significant evidence sources for key events. Chang et al. (1996) evaluated their crowd cheering recognition and keyword spotting (only ”touchdown” was tested) across a small test collection, locating 4 out of 5 key events, with 1 false detection.

Rui et al. (2000) implemented a Model-based segmentation and classification algorithm for detecting excited speech in Baseball sequences. The authors first distinguished between speech and non-speech segments by dividing the audio stream into overlapping windows. Then, comparing three classification frameworks (kNN, GMM and SVM), each window was classified into a speech or non-speech group. The same three algorithms were then employed to label each speech segment into a further two groups: excited speech or not. Events were identified at those periods labelled as excited speech. For both problems the SVM classifier achieved the highest discrimination, however, there was no significant gain from applying one classifying framework over another. For example, the SVM detected 49 out of 66 excited speech segments, compared to 46 out of 66 for the GMM. It would be difficult to predict how accurate this event detection algorithm was, as there was no record of how precise either classification framework was e.g. how many false detections were found.

Another limitation of the event detection algorithm, defined by Rui et al. (2000), was that the soundtrack of baseball sequences is not made up entirely of two classes: speech and non-speech. Ignoring those segments that contain excited crowd sound or other possible environmental noises will have an effect on the event detection algorithm accuracy. Also, from a previous study (Baillie & Jose (2003)), it was identified that the distinction between speech and non-speech segments can become blurred during periods of intense crowd sound. The phenomenon of crowd sound was not investigated in this work.

For the problem of event detection, HMM-based classifiers have been utilised for modelling crowd cheering segments (Baillie & Jose (2003), Xiong et al. (2003*b*), Xiong et al. (2003*a*)). In a previous study, we developed a set of HMM classifiers that recognised audio segments which correspond to crowd cheering (Baillie & Jose (2003)). To do so, the audio track was first divided into overlapping sequential units, which were then classified into one of seven categories, using a maximum likelihood decision process. An “event window” filter then identified the location of key events, by analysing the distribution of labelled audio segments into each pattern class. Xiong et al. (2003*b*) also applied a similar approach, comparing both maximum likelihood and maximum entropy decision processes for HMMs, when classifying crowd cheering, music, speech and applause. Maximum entropy HMMs was found to be optimal when applied alongside a MPEG-7 feature set, while the maximum likelihood decision process was found to be optimal for HMMs applying a MFCC feature set. There was, however, no significant gain found from using one strategy over the other. Both approaches were found to be approximately 94% accurate for classifying sequences containing speech, music and applause.

This MPEG-7 feature set was then applied, alongside entropic prior HMM classifiers, for classifying crowd cheering clips extracted from soccer, and “ball strikes” for golf and baseball video (Xiong et al. (2003*a*)). From evaluation, the classifiers were found to show varying accuracy across four audio sequences (2 golf sequences, and a soccer and baseball sequence). For the task of detecting when a golf ball was “struck”, 66.7% of all related sequences were recognised correctly. For recognising when a bat hit a baseball, the classifier was 87% accurate, and for classifying crowd cheering in soccer audio, the same algorithm was 40% accurate. For the problem of soccer, it was claimed that the algorithm found it difficult to discriminate between crowd cheering and other types of crowd sound, not related to key events. This same problem occurred in an earlier study we undertook for event detection, where crowd reaction, such as singing or chanting are not directly correlated to key events (Baillie & Jose (2003))[2]. These periods are often falsely classified as key events.

Another limitation of the event detection approach, described in Xiong et al. (2003*a*), was that the algorithm required the audio track to be manually segmented. For the experiments, an automatic segmentation algorithm was not integrated into the system, with each audio clip classified was manually segmented. However, extending the HMM classification framework for both segmenting and labelling the audio track is possible, as highlighted in (Baillie & Jose (2003)), although, assigning Model-based algorithms for both segmentation and classification can create accuracy problems. These algorithms require the audio stream to be divided into equal sized, overlapping windows of frames. A particular problem is that some windows can contain more than one pattern class, generating classification errors. To minimise this problem, the window size can be decreased to limit multiple classes occurring in one window. By doing so, the level of sufficient information available in each window decreases, again affecting classification accuracy.

In this chapter this issue was addressed further. In the following sections three algorithms for the segmentation of Game sequences before event detection are introduced. But first, the different content classes that constitute a Game sequence are defined.

10.3 Defining the Content Classes

From previous investigations into football audio, it was discovered that Game sequences differ from other semantic components, such as Studio and Advert. These differences include the occurrence of background environmental sound associated with

[2]This study is described in more detail in section 10.3.1

large stadium crowds, and recorded speech at lower bandwidths. The stadium atmosphere, found in Game sequences, is replicated using microphones strategically placed at pitch level, which is then mixed with the commentary track. The microphones that record the commentary track also differ from those employed in other content structures, due to sound generated inside a stadium. Lower bandwidth microphones are used because they are less sensitive to noise (see Appendix A).

The audio track of a live Game sequence is a complex and noisy environment. Each sequence can contain similar sounding sub-structures, which are all potential sources of error for an event detection algorithm. An illustration would be the discrimination between crowd cheering and singing. It is assumed that crowd cheering can be correlated to exciting key events (Chapter 3.5). Crowd singing, on the other hand, is not so directly related. During a match it is not unusual for periods of singing from supporters. These periods coincide with the start and end of the game, as well as after important events, such as a goal. Singing and chanting can also occur during lulls in the match where supporters vocally encourage their team. For precise event detection, distinction between these two sound classes is crucial and is often overlooked by existing algorithms (Chang et al. (1996), Nepal et al. (2001), Rui et al. (2000), Xiong et al. (2003*a*)).

Code	Description
F0	Prepared speech
F1	Spontaneous speech
F2	Low fidelity speech including telephone channel speech
F3	Speech with background music
F4	Speech with the presence of background noise
F5	Speech from non-native speakers
FX	All other speech.

Table 10.1: The F-conditions for speech.

In this section, the main pattern classes that constitute a Game sequence are defined, which is advantageous for a number of reasons. When applying Model-based segmentation and classification algorithms, such methods require the correct number of classes to be known, prior to segmentation. Classes that are not represented by a model will affect segmentation accuracy. For example, an unexplained sound event appearing in a new video sequence could be wrongly classified as a key event. It is beneficial to explain as much content as possible by the set of model classifiers. Separating the audio data into well-defined groups can also enable high discrimination between those segments that are correlated to key events, and those that are not.

Another important reason for defining the main content sub-structures is that for future indexing, such as Automatic Speech Recognition (ASR), locating the start and stop of each speech segments is an important task. Being able to transcribe speech automatically is beneficial, providing a rich information source for future indexing of content. However, standard ASR algorithms do not adapt well to changes in speech style and audio environment, such as those found in Game sequences. These noisy environments often resulted in an increased word error rate (Spina (2000)). There has been recent investigation into classifying different speech styles and audio environments to address this problem, where a set of content classes called F-conditions have been defined for radio broadcast news (Chen et al. (2002)).

Table 10.1 is an example of the F-conditions, which contain a number of different styles of speech spoken in differing audio environments. Classifying each speech segment into one of these F-conditions is believed to improve word error rates during transcription. The detection of audio environment change can enable the ASR algorithm to be adapted to counteract potential problems. Instead of defining F-conditions for Game sequences, it was thought that by recognising speech segments with differing levels of crowd noise could assist future transcription, as well as assist future indexing techniques, such as speaker change detection and speaker clustering (Chen et al. (2002)).

Before defining the content classes found in Game sequences, an initial investigation is now introduced that outlines the problems experienced when defining sub-structures found in Game sequences (section 10.3.1). Using this experience, alongside the information gained during the data mining phase (Chapter 7), enabled the definition of a more accurate set of pattern classes (section 10.3.2). The reasoning behind the selection of each class is explained, in particular those pattern classes associated with key events. These pattern classes were then modelled by both GMM and HMM, comparing the accuracy of each representation, in terms of classification, before deciding upon one framework for event detection (section 10.3.3).

10.3.1 Initial Investigation

Code	Class Description
S-l	Speech and Low Levels of Crowd Sound
N-l	Low Levels of Crowd Sound
S-m	Speech and Medium Levels of Crowd Sound
N-m	Medium Levels of Crowd Sound
S-h	Crowd Cheering and Speech
N-h	Crowd Cheering

Table 10.2: Initial pattern classes.

An initial set of pattern classes were introduced from a previous study into event detection (Baillie & Jose (2003)). Six representative pattern classes were defined (see Table 10.2), that were identified through manual inspection and visualisation of the data. The audio data set was first divided into pattern classes containing speech or not, and within both groups, a collection of more complex sub-classes was defined. There were three classes that contained speech (S-) and three that did not (N-), with each sub-class corresponding to the level of crowd sound found in a Game sequence, ranging from low to high. The first two sub-classes, ‘S-l’ and ‘N-l’ represented a ‘lull’ during the match. Throughout these periods, there was little or no crowd sound recorded inside the stadium. Sub-classes ‘S-m’ and ‘N-m’, represented periods that contained crowd sounds, such as singing, and the remaining two sub-classes, ‘S-h’ and ‘N-h’, defined those periods of crowd response triggered by a key event.

Each class was then represented by a HMM, and the accuracy of these classes was evaluated across a test collection of Game sequences. A maximum likelihood decision process was used for classifying audio clips in the new test collection into the correct class. If the likelihood scores for all HMMs exceeded a threshold for one data sample, it was placed in an outlier class. Table 10.3.1 presents the results from the experiment, in the form of a confusion matrix, where the bold numbers represent the classification accuracy for each class. The remaining numbers are the percentage of false classifications between classes.

Many of the HMM representations classified less than 80% of the new test samples correctly (Table 10.3.1). The S-h and N-h classifiers, both important for event detection,

Classification Accuracy (%)						
Class	S-l	S-m	S-h	N-l	N-m	N-h
Outliers	1.21	4.43	7.12	5.02	4.3	10.76
S-l	**85.43**	13.32	1.11	3.21	0.01	0
S-m	9.5	**75.78**	4.3	0.2	2.94	1.2
S-h	3.5	4.21	**74.1**	0.65	0.24	4.21
N-l	0.33	1.23	0	**79.2**	5.32	5.34
N-M	0.03	0.4	1.22	9.6	**76.54**	9.81
N-H	0	0.63	12.15	2.12	10.65	**68.68**
Total	*100%*	*100%*	*100%*	*100%*	*100%*	*100%*

Table 10.3: Confusion matrix for the HMM-based classification results, for the initial investigation.

were found to be 74.1% and 68.7% accurate. Both of these classes were important for event detection, containing high levels of crowd sound correlated to key events. It was discovered that there was a high degree of misclassification between the two classes, 12.15% and 4.21%. There were also a high percentage of samples wrongly placed into the outlier group from both classes, 7.12% and 10.76% respectively. 9.81% of the N-h samples were also wrongly labelled as 'N-m', the crowd singing and chanting category. Missing this high proportion of samples correlated to key events could have a negative effect on event detection accuracy.

Drawing a conclusion from this study, there was evidence to suggest that a considerable overlap existed between classes. This had a detrimental effect on the final event detection accuracy. It was believed that the definition between many of the classes was blurred creating a number of problems. For example, during the manual annotation of audio samples for both training and testing, it was believed accurate labelling was difficult and possibly created an overlap between classes. For example, distinction between classes with, varying levels of crowd sound, was difficult to determine for some audio samples. The overall result being that the HMM models for some classes shared similar traits, because the training data contained similar samples, thus causing poor discrimination between those classes.

Another problem identified, was the false classification of audio samples that contained unusual sounds. A threshold was introduced to label samples that could not

Code	Class Description
CN	Low to medium levels of Crowd sound
SN	Speech with low to medium levels of crowd sound
SC	Crowd chanting with / without Speech
CC	All Crowd Cheering related to exciting events
MS	National Anthem / Music in Stadium / Announcements
W	High pitched Sounds such as Whistle

Table 10.4: Redefined audio-based pattern classes.

be explained by any class, into an outlier group. However, segments such as signal interference, stadium announcements, music inside the stadium and the high pitched whistling, were often classified as crowd cheering. To address these problems, further investigation into the definition of the pattern classes was required.

10.3.2 Redefining the Pattern Classes

In an attempt to improve event detection accuracy, the original pattern classes were redefined using both the experience gained from the initial investigation, alongside the findings from automatic content analysis (see Chapter 7.5.2). The new pattern classes defined, are presented in Table 10.4.

The first step was to add two new classes, with one class being a representation of the high-pitched sounds (W) and the other class, representing music (MS). From the initial investigation, high pitched sound, such as the referee whistle were often wrongly identified as crowd cheering. Studying the findings from the content analysis experiment (see Appendix F, Tables F.2 and F.2), high pitched sounds including the referee's whistle, whistling from the crowd, booing and signal interference, were often found to be grouped together (state number 22). Therefore, the 'W' class was introduced to address the problem of false classification, containing high-pitched sounds.

Another class defined was 'MS'. It was discovered through investigation that music was often played inside the stadium, normally before the start of a match. In the case of international matches, the national anthems of two competing countries would also be played. In some game sequences, music was also played inside the stadium as part of a goal celebration. From the data analysis findings, it was discovered that music samples were grouped together into the same state(s) alongside announcements (over a tannoy system) inside the stadium (Tables F.2 and F.2; states 8, 13, 14 and 20). Again, during the initial investigation, samples from both these groups were wrongly classified as crowd cheering. Therefore, the ‘MS’ class was introduced to address this limitation, with the class consisting of music played inside the stadium alongside any stadium announcements.

Another limitation found from the initial event detection investigation was the process of generating precise training data. It was believed that human error played a significant part in the low discrimination between classes. Labelling training samples into the appropriate class was a difficult task, especially between classes with varying crowd sound levels and reaction. The cut off from one class to another was not clear, and was reflected in training data similarities for separate classes. This resulted in the HMM-based classifiers sharing similar traits. To address this problem, clear distinct classes were defined that limited overlapping. To assist the labelling process, some of the existing classes were either grouped or redefined.

From the initial investigation, it was discovered that distinction between speech and non-speech segments became blurred, during high levels of crowd cheering, triggered by a key event. During a period of crowd cheering there is an increase in energy that can distort the speech segment. Often during these periods, the commentator(s) voice also became excited resulting in increased energy. This was reflected in the content analysis findings, where patterns from ‘S-h’ and ‘N-h’ were found in the same group (Table F.2 and F.2; state 6). Excited speech was also found in this group. Therefore, crowd cheering with or without speech (‘S-h’ and ‘N-h’) was grouped into a single class: ‘CC’ (see Table 10.4). The new crowd cheering class (‘CC’) contained a mixture of crowd cheering, applause, shouting and excited speech, triggered by a key event.

Grouping the two original classes (‘S-h’ and ‘N-h’) also eased problems, during the labelling process, when generating training data. During this annotation phase, all samples correlated to key events were placed into the one class. Focus was on the exact location of key events, and not the extra segmentation of speech and non-speech clips during these periods. Given the infrequency of key events compared to other sound classes, this also meant a limited volume of data for both training and testing.

Merging the two groups increased the volume of data for both training and testing[3]. Separation of speech and non-speech segments could then be achieved, after this initial classification step.

Those audio samples that contained crowd chanting or singing (‘S-m’ and ‘N-m’) were also grouped into one single content class, called ‘SC’. Content belonging to ‘SC’ included periods during a match that contained crowd sounds, not directly related to a key event, such as singing, chanting or clapping. It was found that many of these sounds were grouped together during the data mining phase (Table F.2 and F.2; states 2, 4, 10, 17, 18). To avoid overlapping problems, and difficulties arising during annotation, it was believed that grouping these sounds into one class would be easier than defining an individual class for each. The identification of these periods that were not directly related to crowd cheering, was considered more important. Speech and non-speech segments during these periods were also grouped into this one class (‘SC’), as distinction between speech and non-speech was sometimes blurred. Again, the further segmentation of speech and non-speech could be achieved after classification.

The remaining classes represented speech (‘SN’) and non-speech segments (‘CN’) that did not contain high levels of crowd sound, such as cheering or chanting. These classes were found during general moments of a match, with minimal background noise. Periods were divided into speech and non-speech for two reasons. For future indexing, such as automatic speech transcription, the location of speech segments especially with different background audio environments, is important (Chen et al. (2002), Spina (2000)). It was believed that speech transcription during periods of minimal background crowd noise would be simpler than during periods of increased reaction. Therefore, automatically classifying speech segments during these periods would be beneficial for future indexing. Classifying speech during quiet spells would also minimise false classification between speech and crowd cheering, which occurred during the initial investigation.

Given the defined classes, both the GMM and HMM were compared formally as statistical representations, for each class, for the purpose of event detection. For each framework, the classification accuracy of all classes was measured. This experiment is now described.

[3]For accurate model parameter estimation, a large and varied training sample is required (Jain et al. (2000)).

10.3.3 Classifier Comparison Experiment

For all pattern classes, both HMM and GMM models were trained and then evaluated using manually annotated test collection generated from 12 football broadcasts. Audio sequences were segmented and then labelled accordingly, using the predefined pattern classes as a guide (see Table 10.4). Table 10.5 presents a summary of the number of samples belonging to each content class. The test collection was then randomly divided into two samples: 75% of the data, from each class, were randomly placed into a training sample, and the remaining 25% for evaluation.

Pattern Class	Total no. of Samples	#Training Sample	#Test Sample
CN	2238	1678	560
SN	1053	789	264
SC	438	328	110
CC	708	531	177
MS	1734	1300	434
W	486	364	122

Table 10.5: The test collection.

For each pattern class, a HMM was generated using the Bayesian Information Criterion model selection approach defined (in Chapter 8.3.3), and a GMM was generated using the bottom up approach outlined (in Chapter 4.3.2). The test sample was then classified using a maximum likelihood decision process. The results from both frameworks are now described separately, and then comparing the accuracy of each directly.

10.3.3.1 HMM results

Table 10.6 presents the results from the experiment for the HMM-based classifiers. The table is presented in the form of a confusion matrix, displaying the classification accuracy for each classifier (see Chapter 6). The classification accuracy for each classifier is displayed as bold numbers. The remaining numbers are the percentage of false classifications across each class. These numbers allow for the similarities between classes to be measured, analysing if an overlap exists. For example, a high false percentage of false classification between two classes would indicate that sequences between these classes are difficult to discriminate between.

Classification Accuracy (%)						
Class	CN	MS	SN	SC	CC	W
CN	**81.5**	2.77	1.14	0.68	0.0	5.35
MS	4.16	**87.02**	0.57	0.0	0.0	1.23
SN	4.02	5.54	**89.74**	4.11	1.27	3.5
SC	4.69	3.11	6.84	**95.21**	0.0	1.85
CC	4.96	0.87	1.42	0.0	**98.31**	0.41
W	0.67	0.69	0.28	0.0	0.42	**87.65**
Overall	*100%*	*100%*	*100%*	*100%*	*100%*	*100%*

Table 10.6: Confusion matrix for the HMM classifiers.

Concentrating on the HMM results first (Table 10.6), the important classifier for event detection (‘CC’) was found to be 98% accurate. This was evidence that the new classifiers were able to discriminate well between ‘CC’ and the other classes. This was found to be a weakness of both related work (Xiong et al. (2003*a*)), and from the initial investigation (Table 10.3.1), where the two content classes for crowd cheering (S-h and N-h) recognised approximately 70% of the crowd cheering sequences. This low accuracy affected the overall event detection results.

There was also high discrimination between those samples that contained crowd chanting or singing (‘SC’), and crowd cheering (‘CC’). This result was important for accurate event detection, as those samples in the ‘SC’ class were not directly correlated to key events. Overall, 95% of the ‘SC’ sequences were correctly classified, with 4.11% falsely labelled into the speech with low to medium crowd noise class (‘SN’).

Interestingly, 4.96% of the general crowd sounds (‘CN’) was falsely labelled as ‘CC’. This could be a potential source for false event detection, where these wrongly labelled sequences could result in falsely detected events. However, it was believed that a small number of false events would be acceptable as part of an event detection system. These false events could be verified manually during the final annotation process. On the other hand, missing true events would have a greater impact on event detection, as a result of poor accuracy of the ‘CC’ classifier.

Interesting results from the other classifiers included, 5.5% of the ‘MS’ samples that were wrongly classified as ‘SN’. A possible explanation for this was that many of these wrongly labelled samples contained singing from a player, close to a microphone, during the national anthem. This singing was mistaken for speech. The ‘CN’ class also produced a large source of error across many other groups such as ‘MS’, ‘SN’, ‘SC’ and ‘CC’. This could be a result of the large variance found within this class.

10.3.3.2 GMM results and comparison

In comparison, the GMM results were not favourable (see Table 10.7). 86% of the ‘CC’ test samples were correctly classified by the GMM model, over 10% less than the HMM equivalent. The ‘MS’ classifier was only 56% accurate, with the same HMM model classifying 87% segments correctly. Over 4% of samples from both ‘MS’ and ‘SN’ were falsely identified as crowd cheering (‘CC’). This was strong evidence that the HMM framework was more appropriate for this problem. The GMM classifiers did not appear to be as accurate as the HMM, at this level of granularity.

Classification Accuracy (%)						
Class	CN	MS	SN	SC	CC	W
CN	**71.31**	4.39	6.89	2.05	0.42	4.94
MS	6.43	**56.06**	0.0	2.74	0.85	0.82
SN	3.35	17.75	**73.63**	4.79	11.86	9.47
SC	8.31	7.91	6.18	**76.03**	0.0	2.26
CC	2.01	4.22	4.75	0.0	**86.02**	0.21
W	8.58	9.67	8.55	14.38	0.85	**82.3**
Overall	*100%*	*100%*	*100%*	*100%*	*100%*	*100%*

Table 10.7: Confusion matrix for the GMM classifiers.

10.3.4 Discussion

In this section, the pattern classes found in Game sequences were defined. The final definition for each class was a result of both an initial investigation into event detection, and from the findings of the content analysis experiment in Chapter 7. Although not directly comparable, there was strong evidence that the redefinition of the content classes helped discrimination. Also, from experimentation it was discovered that the

HMM was a more accurate modelling framework than the GMM for this problem. A possible explanation for this result was that the HMM was able to represent the subtle difference between each class more efficiently.

The predefined pattern classes are integrated into an event detection algorithm, with the HMM selected as a statistical representation. However, before event detection, the issue of segmenting the audio track is investigated. The findings are discussed in the following section.

10.4 Automatic Segmentation Algorithm Evaluation

In this section, a number of audio segmentation algorithms are evaluated with the aim of automatically parsing the Game sequences into homogeneous units. Once a Game sequence is segmented, these segments can then be classified into one of the 6 predefined pattern classes using the set of HMM-based classifiers, generated in the previous section. Three algorithms widely used in ASR systems are evaluated: two Metric-based, and a Model-based segmentation algorithms[4].

10.4.1 Algorithm Selection Motivation

As previously discussed in Chapter 9.2, audio segmentation algorithms can be divided into three categories: Model-based segmentation, Metric-based Segmentation, and Energy-based Segmentation algorithms. Given that the soundtrack of Game sequences have a constant background of crowd or stadium sound, it was believed that Energy-based segmentation algorithms would not be suitable. These algorithms segment the audio stream at periods of silence. It was assumed that many segment changes would not be separated by silence at all; therefore this category of algorithms was overlooked. The two remaining algorithm categories were investigated instead.

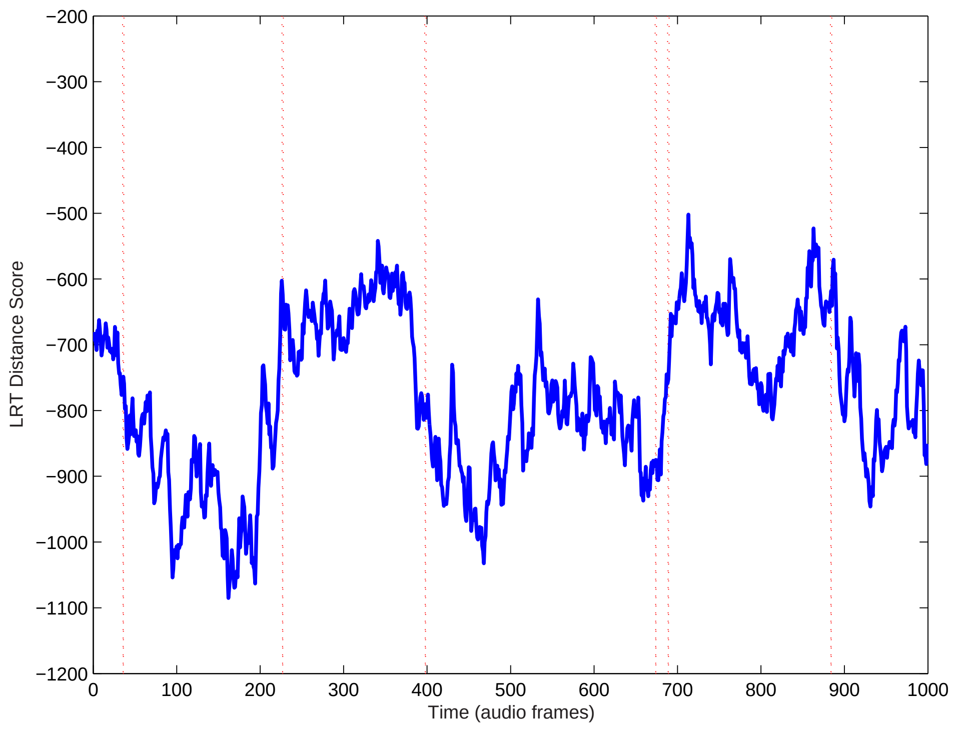
Traditional Metric-based algorithms are widely applied in ASR systems because they are computationally inexpensive and can be implemented in real-time systems (Siegler et al. (1997), Pietquin et al. (2001), Meinedo & Neto (2003), Ramirez et al. (2004)).

[4]See Chapter 9.2 for further details on audio-based segmentation algorithms.

These algorithms are not so computationally demanding when compared to other, more complex Metric-based algorithms, such as BICseg (see Chapter 9.6). Such algorithms vary in the distance measure adopted for evaluating segment change, popular distance measures being the generalised likelihood ratio test (LRT) (Gish et al. (1991)) and the symmetric Kullback-Leibler (KL2) divergence measure (Siegler et al. (1997)). In this section, a KL2 Metric-based segmentation algorithm is investigated that employed a threshold decision scheme. The reason for selecting a KL algorithm over the LRT, resulted from an exploratory investigation using a small collection of data extracted from a Game sequence.

Figure 10.1 is an example of both the LRT and KL2 measure calculated on the Game sequence. Examining both plots, it was argued that the constantly changing audio environment of the Game sequences affected the LRT measure. For both algorithms, a predefined threshold was normally applied when indicating segment change (Gish et al. (1991), Siegler et al. (1997)). It would be difficult to estimate a global threshold for the LRT measure, which would result in many segment changes missed, and many false transitions identified. In comparison, the KL2 distance appeared to be more robust. For the KL2 score, large peaks in the score were discovered at true segment changes (in Figure 10.1 these change were indicated by a dotted line). For this reason, it was decided to evaluate the KL2 algorithm further. A shortfall of applying such an algorithm is this reliance on estimating a global threshold for segment change, so a more dynamic alternative to estimating a global threshold is also investigated.

The remaining two algorithms are, a robust Metric-based algorithm and a Model-based algorithm. Given the success of the Metric-based segmentation algorithm BICseg in the previous chapter, this algorithm is also investigated for the problem of Game sequences. For this problem, the algorithm is adapted for segmenting content found in Game sequences. The last algorithm compared is a HMM Model-based segmentation algorithm. A popular strategy for event detection algorithms, and audio-based classification in general, is to apply a Model-based approach to event detection (Rui et al. (2000), Baillie & Jose (2003)). Therefore, a HMM-based approach to segmentation is investigated as a comparison to the other two algorithms.



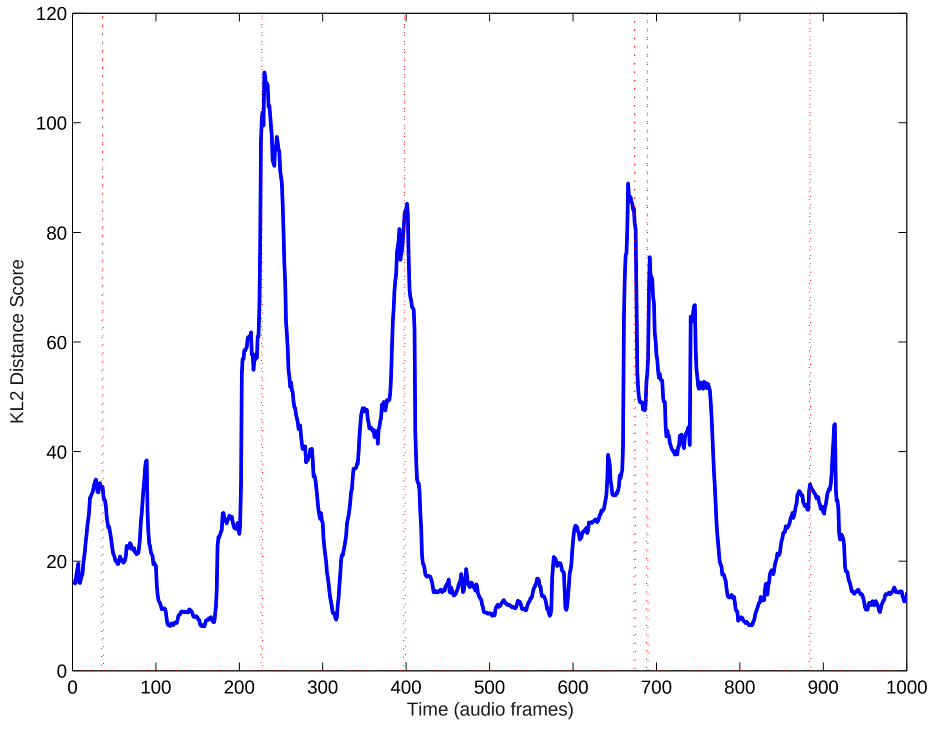


Figure 10.1: An example of the LRT and KL2 distance measures for a 'Game' sequence. A red-dotted line indicates a true segment change.

10.4.2 Kullback-Leibler Algorithm

The KL2 algorithm was first used by Siegler et al. (1997) to segment news broadcast audio data into speech and non-speech segments. The KL2 is an information theoretic measure, also known as Relative Cross Entropy. To introduce the algorithm, the same notation as Siegler et al. (1997) is used.

10.4.2.1 The KL2 Measure

A Game sequence $X = \{x_i : i = 1, \ldots, N\}$ is divided into neighbouring, equal sized sliding windows, A and B. It is assumed that the two windows A and B are both generated by a multivariate Gaussian probability density function (PDF), where at frame i the KL2 can be calculated by,

$$KL2(A;B) = \frac{1}{2}(\bar{\mu_B} - \bar{\mu_A})^T(\Sigma_B^{-1} + \Sigma_A^{-1})(\bar{\mu_B} - \bar{\mu_A}) + \frac{1}{2}tr(\Sigma_A^{-1}\Sigma_B + \Sigma_B^{-1}\Sigma_A - 2I) \quad (10.1)$$

where μ_A is the mean vector and Σ_A is the covariance matrix for the PDF of window A, and μ_B is the mean vector and Σ_B is the covariance matrix for the PDF of window B. The greater the value of KL2, the greater the distance between the windows A and B. If KL2 is zero, then A and B share the same PDF, where $KL2(A;B) \geqslant 0$.

10.4.2.2 Thresholding the KL2 measure

Segment change for the KL2 algorithm is normally found at a local maxima in the KL2 distance (equation 10.1) that exceeds a predefined threshold. However, this method of using a global threshold is a limitation of many of Metric-based algorithms (Siegler et al. (1997), Meinedo & Neto (2003)). A globally set threshold will not often be suitable for a constantly changing audio environment, such as that found in Game sequences (Figure 10.1 is an example of this). For example, a high threshold would miss many segment changes but limit the number of false boundaries. Lowering the global threshold would increase the number of true segment changes picked up but would also increase the number of false transitions. Another problem is that a global threshold for one sequence may not generalise to other sequences. Thus, an adaptive, more intelligent threshold system was investigated.

Examples of alternative methods to applying a global threshold include Pietquin et al. (2001), who employed two threshold parameters. The first is a global threshold to determine segment change, while the second parameter is a minimum delay between two segment changes i.e. two segment changes cannot occur within a predefined number of frames. This approach limited false change detection from smaller localised maxima that can sometimes occur surrounding a segment transition, but this did not address the problem of predefined global thresholds.

Ramirez et al. (2004) also employed two predefined ‘optimal’ thresholds, this time for changing audio conditions. One threshold was defined for periods when there were high levels of background noise found in the audio signal. The remaining threshold was utilised during periods of low-level noise, with noise being determined by analysing the signal-to-noise ratio of the audio signal. This solution did address the constant change in background sound as estimation of both thresholds was still required.

This decision process was the inspiration that formed basis for the definition of a new dynamic threshold scheme, for the KL2 measure. However, instead of estimating predefined thresholds, it was believed that the threshold for the KL2 distance could be defined as a function of the energy of the audio signal. That is, during high-energy periods, the threshold would increase, while during low-energy periods, the threshold would decrease. It was assumed that estimating the threshold by such a function would minimise segmentation errors, adjusting to localised changes in the audio track.

To detect segment change, the KL2 score was normalised for each Game sequence to be mean zero and unit variance. The log energy feature parameter was then used to calculate the current threshold during a Game sequence, where the log of the signal energy measures the activity during the audio sequence (see Chapter 5). A linear function of the relationship between log energy and segment change was defined to estimate the threshold at the current frame i in an audio sequence X, defined as:

$$\psi(i)_{KL2} = \alpha + \beta * \log en(i) \tag{10.2}$$

where β is the slope and α is the intercept of the linear function. $\log en(i)$ is the log energy value at the current frame i. The function parameters α and β can be estimated through experimentation. A segment change occurs if at a local maxima,

$$KL2_i(A;B) > \psi(i)_{KL2} \tag{10.3}$$

It was assumed this threshold would adjust to the constantly changing environment both within and between Game sequences. The threshold was now a function of the energy of a sequence, which could adapt accordingly to localised changes, as well as differences across different audio files.

10.4.2.3 KL2 Window Size

Another limitation with traditional Metric-based segmentation algorithms is the need to estimate an appropriate length for the neighbouring windows. The Gaussian PDFs have to be estimated with the available information in each window. If the window is too small, the feature vectors could contain insufficient information for producing robust statistics. If the window is too large, many different segments can be found in the window. To minimise this problem many applications implement varying window sizes according to the given data. Siegler et al. (1997) estimated that a window length of 2 seconds was optimal for broadcast news audio, while Ramirez et al. (2004) employ smaller windows for noisier speech soundtracks. Meinedo & Neto (2003) implemented three time units for each window; 0.5, 1.0 and 4.0 seconds. A decision process was then determined by combining the results of each window using weighted evidence.

Overall, it would appear that a general rule for window estimation is that small windows obtain a high degree of time accuracy, while longer windows are less accurate but are able to detect slower transitions. For an initial solution, a window length of 1 second was employed, estimated using the results of the optimal window experiment in Chapter 7.3. This time unit was believed to be wide enough to estimate the Gaussian PDF for each window, but not so large that windows could contain many segments. The 1 second was derived using the findings of the experiment in Chapter 7.3.

10.4.3 The BIC Segmentation Algorithm

As previously mentioned, there were some limitations with implementing traditional Metric-based segmentation algorithms. These include both threshold estimation and search window length estimation. For these reasons, and the success of the BICseg algorithm for structure segmentation and classification[5], the robust Metric-based algorithm was analysed for this problem. The advantage of the BICseg algorithm over other Metric-based segmentation schemes is that it is threshold free, hence its ability to generalise. The algorithm also applies an expanding search window to include as much information as possible for detecting segment change, thus avoiding the problem of window length estimation. On the downside the algorithm is not as efficient as KL2.

For the problem of event detection, the BICseg algorithm applied for structure segmentation and classification was required to be changed. In the following section these changes are discussed.

10.4.3.1 BICseg Modification and Variable Estimation

The main difference between applying BICseg for structure segmentation and classification, and for the problem of event detection, was the audio sequence (see Chapter 9.6). For structure segmentation and classification, the audio sequence was smoothed, but for this problem, segmentation was required at a finer level of granularity. Segmentation of concepts, such as speech and crowd cheering was required so after parametrisation, the audio tracks for Game sequences were not smoothed but left in the original form. This allowed the BICseg algorithm to be applied on a frame-by-frame basis.

Let X be the parameterised audio sequence from a Game unit, then $X = \{x_i \in \mathbb{R}^d, i = 1, \ldots, N\}$, where x_i is a single data vector for the frame i, N is the length of the entire Game sequence, and d is the dimensionality of the feature space.

There were also a number of algorithm variables to be estimated in the new version of the BICseg algorithm, such as the minimum and maximum length of the search window $[w_l, w_r]$. To estimate these parameters, sample statistics from a small test set of Game sequences was produced (see Table 10.5). The minimum segment length

[5]For details on the BICseg algorithm please refer to Chapter 9.6.

was found to be 0.4 seconds. The largest segment length being over 5 seconds, and the median segment length was approximately 1.1 seconds. Using these statistics as a guide, the new variables were set to $w_{min} = 0.5$, $w_{max} = 5$, $skip = 0.05$ and $w_{buff} = 0.1$ seconds in length.

10.4.4 Model-based Algorithm

A HMM Model-based segmentation algorithm was also implemented. For the HMM model-based segmentation algorithm, the audio stream was divided into sequentially, overlapping windows of audio frames. Using the maximum likelihood criterion, each window was then classified by the set of HMM models generated from each pattern class (see section 10.3.3). A segment boundary was then found where there was a change from one pattern class to another. This Model-based segmentation algorithm is described in more detail in Chapter 9.3.

10.4.5 Segmentation Algorithm Comparison Experiment

All algorithms were then formally compared across a test collection generated from 4 Game sequences, approximately 30 minutes of audio data. For each audio sequence the segment boundaries were manually tagged, where a segment change was considered to be a change from one content class to another (see Table 10.4). For the experiment, the segment boundaries were examined fuzzily, as slight inaccuracies were expected due to the precision of the manual tagging, and alignment differences between algorithms. The exact location of some segment changes was subjective, so each algorithm was allowed a small allowance of frame error, between detected boundaries. It was believed that an event detection system could tolerate a small degree of inaccuracy between segments. For the experiment, 10 audio frames were decided upon.

10.4.5.1 Operational Parameters

Each segmentation strategy was run over a range of operational parameters to help provide an insight into the accuracy of each approach. The goal of segmentation was

Code	Algorithm	Operational Parameters
$KL2_{th}$	KL2 with global threshold	Global Threshold $= 0.1, 0.2, \ldots, 1$
$KL2_{lin}$	KL2 with linear threshold	$\alpha = 0, 0.05, \ldots, 0.5$; $\beta = 0, 0.05, \ldots, 0.5$
BICseg	BIC Segmentation	$\gamma = 1, 1.1 \ldots, 2$
HMMseg	HMM model-based	Window Size=$0.2, 0.4, \ldots, 2secs$

Table 10.8: Segmentation algorithms and the evaluated operational parameters. The range for each parameter is provided.

to minimise over-segmentation (insertion errors) but not at the expense of missing important segment boundaries (deletion errors)[6]. Evaluating the algorithms over a wide range of value, would help provide a greater insight into the algorithm accuracy. For example, a chosen parameter selection for one algorithm could result in sub-optimal performance, while a setting for a comparative algorithm, could result in optimal performance. Thus, the comparison of both algorithms will be skewed when using one set of variables. The full details of the operation parameters and the range of values evaluated, for each algorithm, can be found in Table 10.8.

For the KL2 algorithm the α and β parameters were varied simultaneously over numerous combinations. As a baseline, a global threshold was implemented over a range of values ranging from low to high. This would help evaluate the new dynamic threshold scheme, with an existing global threshold approach, still applied in many systems.

For the BICseg algorithm, the penalty weight, γ, was varied across a range of values. Chen & Gopalakrishnan (1998), set $\gamma = 1$, which is the strict definition of BIC, however, Tritschler & Gopinath (1999) investigated the penalty weight further for speaker segmentation, and found that the penalty weight set to $\gamma = 1.3$ resulted in higher segmentation accuracy. In this experiment, γ was evaluated over a wide range of values.

For the HMM Model-based algorithm, both the window size and overlap length for grouping frames were varied before classification. The window length is important, as a small window size may not contain sufficient information for classification, while a longer window could contain more than one pattern class.

[6]An insertion error is when a segment change is detected that did not exist. A deletion error is when a segment boundary is missed.

10.4.5.2 Results

The results are presented in Figure 10.2 for each algorithm over all operation parameters. For all algorithms, both the insertion and deletion error rates were measured. The closer a point on the figure is to (0,0), the more accurate the segmentation algorithm was at that operation setting. The overall aim of the segmentation algorithm is to limit the number of deletion errors first, as insertion errors, it was assumed, could be corrected during segment classification, or by filtering. For example, a median filter could be applied to correct errors for a Model-based segmentation algorithm (see Chapter 9.3). For a Metric-based algorithm, during the classification phase, over-segmentation errors can be corrected if the two adjoining segments are classified into the same group.

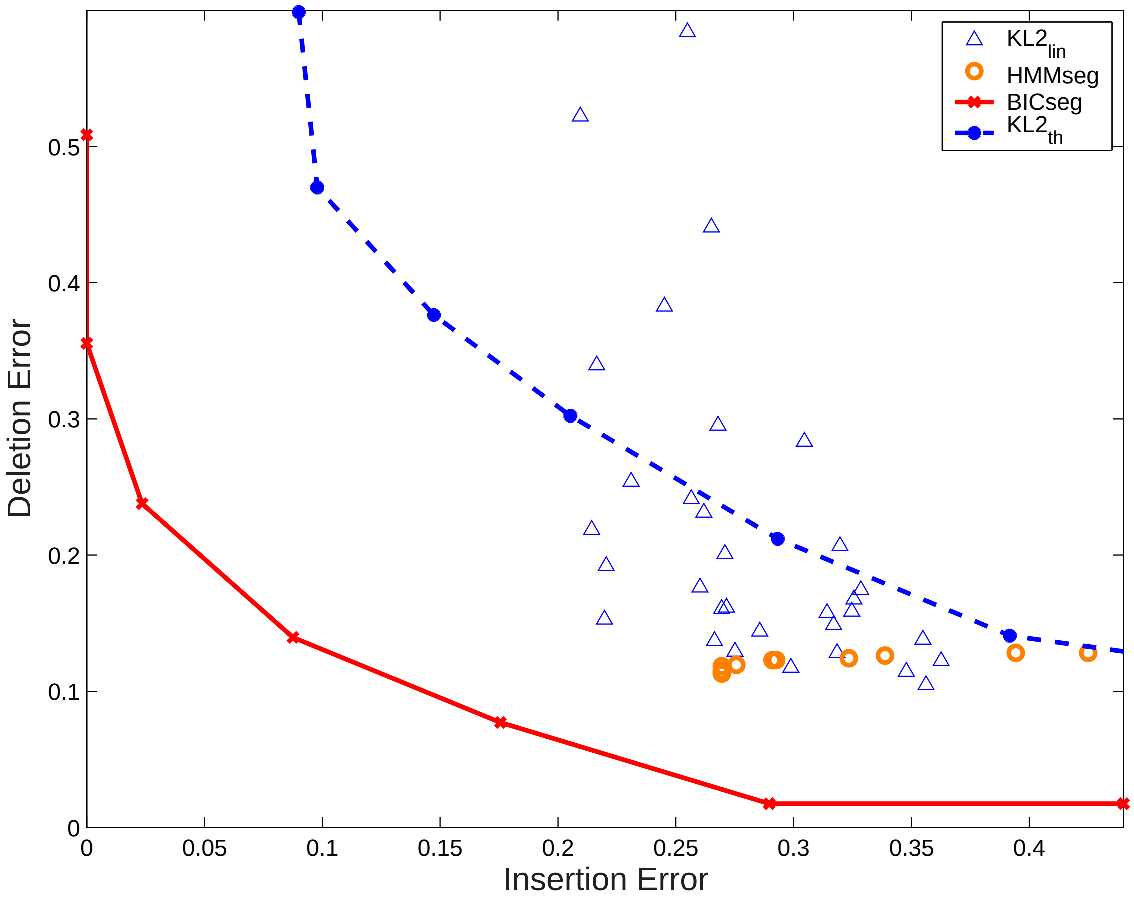


Figure 10.2: Audio Segmentation Results

From the results, Figure 10.2, it was discovered that the BICseg was the most accurate algorithm, when applying a penalty weight of $\gamma = 1.3$. Changing the penalty weight had an effect on the algorithm, from increasing the deletion error, for a weight less than 1.3, to increasing the error rate after this value.

Both the KL2 dynamic threshold ($KL2_{lin}$) and HMM Model-based (HMMseg) segmentation algorithms produced comparable results at their optimal settings, although both algorithms were not as accurate as BICseg. The $KL2_{lin}$ did experience a lot of variation in accuracy across all operation values. By increasing the intercept parameter β over 0.2, dramatically increased the number of deletion errors. This result indicated the value of evaluating such an algorithm over as wide a range of parameter values as possible.

The optimal settings for $KL2_{lin}$ were $\alpha = 0.1; \beta = 0.15$. For the majority of parameter settings, this dynamic threshold function was found to be more accurate than employing a standard global threshold. Overall, the $Kl2_{th}$ was discovered to be the worse performing algorithm.

Increasing the window size from 0.4 to 1.2 seconds, for HMMseg, had minimal effect on segmentation accuracy. Although, increasing the window size over 1.2 seconds, resulted in a higher number of insertion errors recorded. A window length of 1 second with an overlapping shift of 0.25 seconds was found to be optimal.

The experiment, when taken as whole, did not indicate that there was a considerable difference between the three segmentation algorithms, $KL2_{lin}$, BICseg and HMMseg. Therefore, in the following section, these three segmentation algorithms were evaluated as part of a complete event detection system.

10.5 Event Detection Results

In this section, five event detection approaches, each integrating either the $KL2_{lin}$, BICseg or HMMseg segmentation algorithms, were evaluated. The overall accuracy for the event detection algorithms were measured across eight new football games, unseen during the previous development phase. To generate a realistic data set, official match reports and detailed game statistics for each Game sequence were gathered from the FIFA 2002 World Cup web-site (FIFA (2002)), the world governing body for the sport[7]. Important events were considered to be goals, scoring attempts, cautions or other key incidents highlighted in the match report. For each event, the start and end point were recorded. As a guide, the match reports indicated approximate time points for all events, which aided this labelling process. This information formed the ground truth against which all event detection algorithms would be compared. The total number of events in each video file are reported in Table 10.9.

To measure the accuracy of each algorithm, a correctly identified event was determined to be: "if a classified crowd-cheering ('CC') segment overlapped the location of a true event, at any time-point". If there was an overlap between an actual event and a 'CC' segment, a correct detection was noted. If there was no true event during a classified segment, a false detection was recorded.

The different event detection strategies are first defined in section 10.5.1. The results are then presented in section 10.5.2, and then the findings and conclusions of the experiment, are discussed in section 10.5.3.

10.5.1 Event Detection Strategies

Five strategies are compared overall. The first two integrated either the BICseg or $KL2_{lin}$ segmentation algorithms. Applying a Metric-based segmentation scheme first has its advantages over the HMM model-based segmentation algorithm. For example, over segmentation problems can be addressed by joining wrongly segmented content into one unified segment, during classification. On the other hand, employing a Model-based scheme, for both classification and segmentation, can result in segmentation errors occurring during false classification, which can only be corrected with further processing or filtering.

Therefore, for event detection, both approaches were evaluated, with the audio track being parsed into homogeneous units, using either BICseg, $KL2_{lin}$. Each segment was then classified using the HMM-based classifiers generated in section 10.3.2, and events were found at a segment classified as 'CC'.

The remaining three strategies were based on the HMM model-based segmentation and classification algorithm. An "event window" decision process was employed to

[7]See Appendix J for an example of a match report.

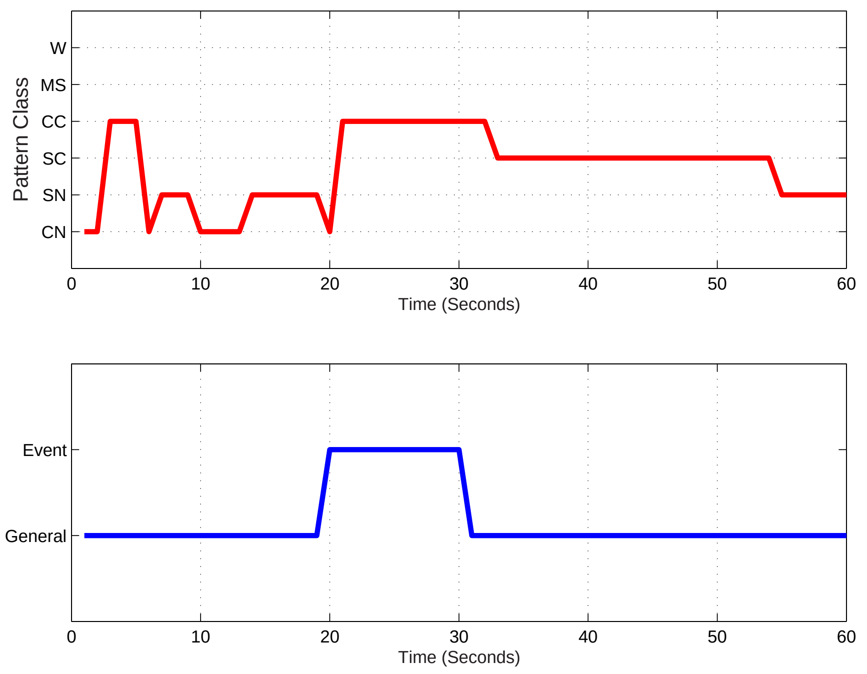


Figure 10.3: Example of a detected event using the "event window"

.

determine event locations (Baillie & Jose (2003)). The "event window" is a filtering process, applied to limit the effect that false classification has on accuracy. For example, a key event triggers a crowd response that normally lasts longer than 1 second in length. Therefore, a key event is flagged if *n* sequential audio clips were classified as 'CC'. Figure 10.3 is an illustration of a detected event, where the top graph is a 60 concurrent audio segments grouped into one of the 6 categories. The bottom graph indicates the location of a true event. In this example, it is assumed that the "event window" is set at 10 seconds in length. The Game sequence enters the 'CC' class at 20 seconds and exits at 33 seconds (13 seconds later), thus flagging a key event. In the same sequence, a potential false event is avoided when samples between 3 and 6 seconds were classed as 'CC'.

The three strategies employed a different length for the "event window", to evaluate the accuracy of different window sizes. For example, the first algorithm (Win1), detected events if only 1 audio sample was classified as 'CC'. i.e no event window filter was applied. The second scheme (Win5), flagged an event if a minimum of 5 sequential audio clips were classified 'CC'. The final approach (Win10), 10 sequential audio clips were required for an event to be flagged.

Approach		KL2		BICseg		Win1		Win5		Win10	
File	#Events	Recall	Prec	Recall	Prec	Recall	Prec	Recall	Prec	Recall	Prec
BRA-CHN	27	1.00	0.63	1.00	0.71	1.00	0.14	1.00	0.36	1.00	0.56
BRA-TUR	21	0.86	0.91	0.86	1.00	0.86	0.44	0.86	0.86	0.71	1.00
CRC-BRA	18	1.00	0.72	1.00	0.78	1.00	0.21	0.94	0.47	0.89	0.67
CRC-TUR	48	0.54	0.43	0.54	0.54	0.54	0.09	0.54	0.23	0.54	0.35
FRA-SEN	16	0.88	0.84	0.88	1.00	0.88	0.47	0.88	0.94	0.81	1.00
GER-IRE	48	1.00	0.29	1.00	0.32	1.00	0.06	1.00	0.15	0.92	0.23
GER-SUA	29	0.97	0.88	0.97	0.97	0.97	0.16	0.97	0.44	0.97	0.88
POR-KOR	56	0.98	0.21	0.98	0.26	0.98	0.05	0.98	0.13	0.95	0.21
Overall	**263**	**0.90**	**0.61**	**0.90**	**0.70**	**0.90**	**0.20**	**0.90**	**0.45**	**0.85**	**0.61**

Table 10.9: Event Detection Results

For each segmentation algorithm, the optimal parameter settings discovered in section 10.4.5.2 were used.

10.5.2 Event Detection Results

Table 10.9 presents the event detection results for all algorithms. The precision and recall of each algorithm was calculated, common measures applied in Information and Video retrieval. Recall is the ratio of the number of relevant events identified to the total number of events. Precision is the ratio of the number of relevant events identified to the total number of irrelevant and relevant events identified.

It was discovered that all the algorithms produced comparable accuracy, recording a recall of 90%, except Win10 that produced a recall of 85%. The algorithms did differ in terms of precision, with the BICseg integrated event detection algorithm being the more precise. This result reflected the findings from the segmentation experiment (section 10.4.5.2), where BICseg was the more precise algorithm. The event detection algorithm integrating KL2, was the second most precise algorithm (61%). Win10 also recorded the same precision, however, this was at the expense of drop in recall. Win1, the HMM-based algorithm applying no filter, was the worst performing algorithm in terms of precision with 20%.

It was interesting to note that some false events detected by each algorithm did contain periods of crowd cheering. These false events were passages of play such as a flamboyant skill by a player or an important defensive play, producing a crowd response not recorded in the truth data. This was exaggerated in games of high importance, such as GER-IRE and POR-KOR. In these games the crowd were extremely vocal in comparison to other matches, thus producing a higher rate of false detections. This was an issue to be addressed during future truth data generation.

Events that were not picked up by all algorithms, included exciting action that did not coincide with a high level of crowd response in the stadium. This often occurred when a team was supported by a small section of the crowd in the stadium. The small support generated little crowd response resulting in the event not being recognised.

10.5.3 Results Discussion and Conclusions

The BICseg event detection algorithm was shown to be the most effective for recognising events in Game sequences. The main benefit of the algorithm is its ability to recognise patterns that are correlated to key events. By applying HMM-based classifiers to the problem, the need for defining a heuristic set of rules was eliminated (Chang et al. (1996), Nepal et al. (2001)). The performance of the HMM-based classifiers was encouraging given the difficult nature of the Game soundtrack. There was strong evidence that the redefined pattern classes allowed for greater discrimination between classes, as well as easier training data generation for model training. For this problem, HMMs were found to be more accurate than using a GMM framework.

Both KL2 and BICseg, when integrated into an event detection system improved accuracy over standard Model-based segmentation and classification approaches. Segmenting each Game sequence accurately into homogeneous segments limited classification errors and was shown to improve event detection precision, when compared to HMM Model-based segmentation and classification, widely used in current event detection systems (Baillie & Jose (2003), Xiong et al. (2003*b*), Rui et al. (2000)).

The choice of segmentation algorithm employed for event detection was also important. From experimentation, BICseg was found to be the most accurate segmentation algorithm. Both $KL2_{lin}$ and HMMseg produced similar results but suffered from higher error rates. However, the results from the experiment highlighted that there was no obvious gain from selecting one segmentation algorithm, over the other. The advantage of applying a BICseg algorithm, over the other algorithms, is the expanding search window. BICseg search window grows until a boundary is found, therefore using as much information as possible. The KL2 and HMMseg algorithms require the audio stream to be divided into equal sized groups of frames.

This is an advantage over the other two algorithms, in terms of accuracy, but also a downside. The BICseg algorithm is more computationally algorithm. Running BICseg over large data sets can be very time consuming. Cettolo & Vescovi (2003) have investigated software engineering solutions for improving efficiency, by saving the number of computations made through estimation of the covariance matrices on partially shared data.

On the other hand, KL2 is widely employed in real time ASR systems because of its low computational cost. The linear thresholding function introduced for KL2 improved accuracy over a standard global threshold. It should be noted, that the relationship between log energy and KL2 is not directly linear. Thus, future investigation into a more suitable representation could improve accuracy further.

During event detection, applying a Metric-based segmentation scheme first had its advantages. For example, over segmentation problems can be addressed, by joining wrongly segmented content into one unified segment during classification. Whereas employing a model-based scheme alone, for both classification and segmentation, does not allow for segmentation errors to be corrected as classification occurs before segmentation. Metric-based algorithms are also portable as they do not require the correct number of pattern classes to be know for segmentation, unlike HMMseg. This could be a possible explanation for the low precision of the HMM-based algorithm, as Game sequences will eventually contain segments that cannot be explained by one of the predefined pattern classes. An “event window” filter was required to limit these problems. Such a filter could also be applied to improve event detection precision for Metric-based algorithms. For example, if we assume an event must span a minimum time length, segments smaller than this limit can be ignored.

10.6 Summary

In this chapter, an event detection algorithm for football video was developed. To develop this algorithm, a number of steps were taken. First a set of pattern classes, each an important content found in Games sequences, were defined. Then a number of audio segmentation algorithms were evaluated formally, with BICseg algorithm minimising both insertion and deletion errors. A number of event detection algorithms were then evaluated across a test collection of 12 football video files. It was found through evaluation that an algorithm applying BICseg for segmentation alongside a set of HMM-based classifiers, was the most precise algorithm. However, a more efficient KL2 using a dynamic thresholding scheme was also implemented, and was found to be as accurate as BICseg for detecting events. This algorithm did suffer from over segmentation problems, but could be an alternative option for realtime systems.

Chapter 11

Conclusions and Future Work

11.1 Contributions and conclusions

In this section, I list the contributions that this thesis has made and outline the main conclusions.

11.1.1 Contributions

At the outset of the thesis, I outlined three problems that were considered important for the automation of video indexing. The first problem was a lack of domain understanding, particularly for sports video. Many investigations into the sports video domain, did not utilise the advantages of prior knowledge during the development of indexing algorithms, which were shown through experimentation to have an affect on system accuracy.

Two indexing problems were also defined: *structure segmentation and classification*, and *event detection*. The first indexing problem is important for mapping video content, which can enable efficient browsing, retrieval of specific sequences, and support further high level indexing algorithms. Mapping content can allow for the removal of advertisements during playback, or for further indexing analysis. The second indexing problem (event detection) is crucial for automating highlights generation, and in general, providing detailed video summaries.

A number of proposals were suggested to address each problem. To focus the thesis, each problem concentrated on applying audio as an information source only. Analysis of related research into video indexing had identified audio as an important information source for video indexing. It was also proposed to focus on applying statistical models for representing audio content. The reasoning behind this proposal, was that when applied to other video domains, statistical models were more accurate than ad hoc techniques.

In the following paragraphs, I list the contributions this thesis has made towards solving these problems. I first list the individual contributions that this work has achieved. Then the overall conclusions that I believe have been reached, are summarised, when the work of the thesis is taken as a whole.

11.1.1.1 Generation of a large test collection

A current limitation in the field of sports video indexing is that no standardised test collection at present exists. A standard test collection can help compare different approaches, for each indexing problem, as well as provide an information source for gathering domain knowledge. It was important to generate a large test collection for evaluating the indexing algorithms developed in this thesis. Therefore, 48 complete football matches were recorded and stored in MPEG-1 format. A large cross section of the collection was annotated manually, for both algorithm development and evaluation. This test collection will also be a useful resource for fellow researchers who are part of the local IR group[1]. To the best of my knowledge, this thesis is the first comparative study, over a large test collection, for sports video.

11.1.1.2 Domain Knowledge and Content Analysis is important for algorithm development

From reviewing related research into sports video indexing, it was apparent that limited integration of domain knowledge, the indexing process, was a weakness of existing works. This weakness was a result of poor understanding of content, production of the sport, and the rules. It was believed that more robust indexing techniques could

[1]Copyright laws prevent open distribution of such a collection.

be developed by gathering as much information about the domain as possible, with all aspects of the video domain being analysed.

To address the limitation of domain understanding, Chapter 3 analysed the main content structures found in football video and in particular, the audio differences between those structures were recognised. From this investigation, a video model of the relationship between each structure was defined. This video model was utilised for both manual annotation of test data, and algorithm development.

In Chapter 7, the entire audio track for football video was statistically modelled using a single HMM. This model was a robust representation of the underlying processes that generate the audio data. Those structures discovered by the model were then related back to the original video files, checking for similarities with content found in Chapter 3. Content analysis of the underlying data structures provided an insight into the video domain, and help formulate assumptions (automatic content analysis as a contribution is described further in section 11.1.1.4).

The findings from both the manual and automatic content analysis were compared, where a common trend in content emerged when comparing the findings from both studies. Different hidden states in the HMM corresponded to different sub-structures found in each content class, defined during manual inspection. The major benefit from automatic content analysis was the discovery of various sub-structures contained in the main semantic units not identified through inspection. To illustrate, the differences in recording quality for speech from one semantic unit to another was recognised automatically by the HMM.

Identification of such sub-structures was advantageous for the development of both the segmentation and classification, and event detection indexing algorithms. For example, the difference between crowd reaction, and other unrelated sound classes, were recognised by the HMM. For event detection, this distinction is important particularly for discrimination between crowd singing and crowd cheering, where only crowd cheering can be directly correlated to key events. This problem limited of existing research into audio-based event detection (Chang et al. (1996), Nepal et al. (2001), Xiong et al. (2003*b*)).

The results of the automatic content analysis also satisfied the assumption that audio found in football video, could be well modelled statistically by the HMM.

11.1.1.3 Careful selection of Audio features can improve accuracy

Audio feature selection was an identified weakness for many related audio-based indexing algorithms. Previous research into this problem included the selection of audio features extracted from the time domain (Chang et al. (1996), Nepal et al. (2001)), or a reliance on selecting standard feature sets from automatic speech recognition (Rui et al. (2000)). There was little formal investigation into the merits of each feature set, for the problem of broad audio content classification.

Three candidate feature sets for parameterising the audio track of football video were chosen, by reviewing the state of the art in a number of research fields. These feature sets were the: Mel-frequency Cepstral coefficients (MFCC), Linear Predictive Coding Cepstral coefficients (LPCC) and Cepstral coefficients (CC). The three candidate feature sets were formally evaluated, comparing the classification error of each set. To do so, GMMs using each feature set were generated for three content classes found in football video. The accuracy of each feature set was then calculated for classifying new test sequences, with MFCC found to minimise classification error.

A further experiment into the configuration of the MFCC feature set was run. It was revealed that by parameterising the audio stream using the first order and 0^{th} MFCC coefficients, and by appending log energy, achieved the highest discrimination. This result differed from standard ASR systems, which avoid using the 0^{th}coefficient. For football audio, the addition of this coefficient improved classification error. Another interesting discovery was that classification error increased significantly when the time derivative MFCC coefficients were added to the feature set. This configuration is typically applied in standard ASR systems, and is also mirrored in many applications to general audio, as well as for sports video. However, it was found through experimentation that the simpler feature set was more accurate for representing broad content classes containing both speech and non-speech sounds.

11.1.1.4 HMM can be successfully applied for automatic content analysis

The audio track for football video was statistically represented by a single HMM for the purpose of automatic content analysis. After feature extraction, the data representation of each audio file was large, so compressing this data was considered a viable option for managing such sequences, for modelling the entire audio track. Data compression was achieved by dividing the audio stream into equal length windows[2], and then calculating the mean feature vector. The audio files were then compressed, by using the mean feature vector as a representation for each one second time step.

With a reduced data sequence for each football video, a HMM was generated. To select a robust HMM representation of the entire audio track, a series of HMMs were generated with varying number of hidden states, with the goal of selecting the best candidate model. Predictive likelihood, a non-discriminative measure of the fit of each model, was used as a selection guide. The final HMM model was chosen when the predictive likelihood score converged once 25 states were added.

Employing the Viterbi decoding algorithm, each one second time step for a new collection of football sequences, was then assigned to a hidden state label in the chosen HMM. Each state was assumed to represent semantic content found in the audio track. These automatically labelled football sequences were used as a comparison between the results from manual inspection, as well as discovering new sub-structures found in the data. To the best of my knowledge, this is the first time such an application of HMM has been applied to audio indexing, for content analysis.

11.1.1.5 HMM Model selection is an important phase in algorithm development

A proposed objective for this thesis was to investigate methods for HMM model selection. HMM is widely used for modelling audio data but is often applied in an ad hoc manner. For audio content classification, in particular, a standard approach for HMM model selection is to initialise similar models for each content class, even if those classes vary dramatically. This was not believed to be a scientific solution, especially given the very broad and varying content classes found in video. In fact, this approach lead to inferior models.

The importance of model selection was highlighted from experimentation, where a gain in performance was reported from selecting models methodically. Selecting too few or too many hidden states resulted in poor classification performance. Another important discovery was the effect of increasing the number of mixture components,

[2]The optimal length of this window was discovered through experimentation, where one second was found to contain sufficient information for modelling.

from a singular Gaussian density, had on classification error. For two out of the three content classes, increasing the number of mixtures was detrimental, whereby the addition of further Gaussian mixture components resulted in both overfitting and poor parameter estimation. This result alone qualified the assumption that applying similar models for each content class would generate problems.

It was believed that a selection strategy for HMM would help identify suitable models for each content class. So three HMM model selection strategies were investigated to address this problem, providing a method for automatically selecting HMMs. The three strategies were an exhaustive search method, the classical Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC). Through experimentation, BIC was found to select the simplest models without deterioration in classification performance. There was no significant improvement gained from using either selection method, however, in terms of classification accuracy. An assumed advantage of BIC over AIC, was that it factored in the training sample size during selection. Thus, for those content classes with little available training data, BIC favoured simpler models.

An advantage both AIC and BIC had over the exhaustive search method was that they did not rely on predefined thresholds. Setting a global threshold is difficult and will not generalise across different content classes. AIC and BIC are terms that penalise the predictive likelihood score instead, causing a maximum peak, with the 'optimal' model located at this maxima. An advantage is that the number of models required to find an optimal solution can be reduced for both AIC and BIC, avoiding an iterative addition of parameters, required by the exhaustive search approach. A bisection search strategy, or equivalent, can be applied instead for finding the maximum score during HMM selection.

11.1.1.6 Formal investigation is advantageous before deciding on statistical frameworks

In Chapter 8, both HMM and GMM models were generated for three broad content classes found in Football video: Advert, Game and Studio. Results from related work into audio content classification suggested that the HMM outperforms GMM, when directly compared. This finding did not transfer to broad content classes found in football audio, with no significant difference found in classification error when comparing both the GMM and HMM as classifiers. This provided evidence that selecting a complex statistical representation such as the HMM, before investigating simpler models (such as GMM), would not necessarily translate into improved classification performance.

In Chapter 10, both GMM and HMM were again compared, this time modelling the many sub-structures found in Game sequences (for the purpose of event detection). HMM performed considerably better than the GMM for classifying sub-structures such as ‘crowd cheering’ and ‘music and speech’. At this finer granularity, experimentation indicated that the HMM was a more accurate representation than the GMM.

Summarising both results, it was clear that careful consideration was required before selecting statistical classifiers. Particular investigation into the type of content classes found in the data can provide gains in performance. Simpler statistical representations can provide sufficient performance in comparison to more complex models.

Drawing conclusions from both experiments, it would seem that for broad content classes, which contained many sub-structures, the GMM was an appropriate statistical representation. In comparison, the HMM provided a higher discrimination accuracy between the sub-structures found in Game sequences. It was discovered that the HMM can model these sub-structures more effectively than the GMM (in terms of classification accuracy).

11.1.1.7 BICseg was more accurate than Model-based audio segmentation algorithms

Several Metric-based and Model-based segmentation algorithms were evaluated for both indexing problems; structure segmentation and classification, and event detection. It was discovered that a robust Metric-based algorithm called BICseg, adapted specifically to both problems, were more accurate than Model-based algorithms. This was an interesting result for structure segmentation and classification especially, where Model-based algorithms are more popular. The same result was also repeated for event detection, where the BICseg algorithm was again adapted for the finer level of difference between content classes.

The BICseg algorithm was adapted from the original algorithm by Chen & Gopalakrishnan (1998), with a number of improvements being made to the original algorithm to improve efficiency and accuracy. There are a number of possible explanations as to why the adapted BICseg algorithm performed better for each indexing problem. The algorithm segmented the audio stream into homogenous units before classification, instead of dividing the audio track into equal sized units. This process assisted accuracy, and also enabled false segmentation to be corrected during the classification phase.

An advantage with using the BICseg algorithm is that a comprehensive set of models is not required for accurate segmentation, and does not rely on a specific modelling framework. This was found to be a limitation of employing a Model-based segmentation algorithm. For such an approach, a comprehensive set of models, each a representation of the content found in the audio track, was required for accurate segmentation. When a new audio track contains unexplained content not covered by the model set, then the Model-based segmentation algorithm failed. For BICseg, no prior assumption of content is made before segmentation.

Another advantage the BICseg algorithm has over the other segmentation algorithms evaluated, was the search window. Instead of grouping audio frames into equal sized units, BICseg employed an expanding search window for locating segment change. This search window allows for as much information as possible to be included in the decision process. The search window also avoids the need for setting a length for grouping audio frames, which may not generalise within and across video files. BIC was then used to determine segment boundaries in the search window, another feature that made the algorithm robust when compared to traditional Metric-based segmentation algorithms. When compared with traditional Metric-based algorithms, no threshold was required during the decision process.

11.1.1.8 A robust and effective structure segmentation and classification algorithm

In Chapter 9, seven algorithms that map the main structure found in football video (through domain knowledge and automatic content analysis) were compared. Each algorithm was set the task of segmenting and classifying content classes outlined in a video topology model (Chapter 9.7.1, Figure 9.6). It emerged through experimentation that utilising a combination of both the BICseg algorithm and GMM classifiers, was the most accurate strategy, although the difference in accuracy between GMM and

HMM was minimal (2%), when applied as classifiers.

Applying a combination of both BICseg and a set of statistical classifiers was advantageous, compared to the other strategies. This was because many segmentation errors produced by BICseg were corrected during the classification phase, approximately 19% of all insertion errors. This algorithm overall, was able to map the main structures found in the football video using audio alone, with a high degree of accuracy (with only 7% of segment changes missed).

11.1.1.9 An effective event domain algorithm integrating domain knowledge

Chapter 10 illustrated how an indexing algorithm can be enhanced using domain knowledge. The audio-based event detection approach outlined in this chapter was shown to be effective for recognising key events, reported in the official game statistics. The main benefit of this algorithm was the ability to recognise patterns that indicate high levels of crowd response, correlated to key events. Also, by applying HMM-based classifiers to this problem, the need for defining a heuristic set of rules that determine event detection, was eliminated (Chang et al. (1996), Nepal et al. (2001)).

The pattern classes found in Game sequences were redefined from a previous study into event detection (Baillie & Jose (2003)), allowing the event detection algorithm to discriminate between segments that are correlated to exciting events, and those that are not. The content classes were redefined using the results from both domain knowledge and automatic content analysis, improving the accuracy of each classifier.

Applying an adapted version of the BICseg segmentation algorithm improved classification accuracy, and thus event detection precision. BICseg when applied at this finer resolution, often where there are only subtle transitions from one content class to another, again performed better than a Model-based approach. Segmenting the audio track into these predefined content classes first, using BICseg, improved the precision of the event detection algorithm.

A traditional Metric-based segmentation algorithm was also implemented using the Kullback-Leibler distance. Both a global threshold and a dynamic threshold were implemented, to determine segment change. Estimating a global threshold is difficult and will not always generalise both within and across video files, and was the motivation behind the dynamic threshold. The dynamic threshold was a linear function of the energy of the audio track, as the energy increased (an indication of increased activity) so did the threshold, minimising the effect noise had on segmentation. This decision process improved over the global threshold and showed similar accuracy to a Model-based approach to event detection. However, the algorithm when applied to event detection was not as precise as BICseg. This algorithm is more efficient in terms of computational complexity, when compared to BICseg, and could be applied to systems that require real time solutions.

11.1.2 Thesis Conclusions

The results of this thesis have demonstrated that audio can be a powerful information source for indexing football video, particularly when utilised with domain knowledge. With careful consideration into the many implementation and application issues such as audio parametrisation, segmentation and modelling, effective indexing algorithms can be developed. However, it is essential that these issues are focused upon for each specific video domain, where it was discovered that results from one video domain do not necessarily transfer to others. Hence, investigation into the underlying data structures is advantageous before algorithm development.

As part of this thesis, I have rigorously evaluated a number of widely applied techniques across a large and varied test collection for two specified indexing problems. The reasoning behind this, was to facilitate a shortage of comparative research into techniques for audio-based content indexing, across a common test collection. Comparing a number of approaches over a large test collection, provided an insight into which techniques were successful, for both indexing problems. The experiments demonstrated that:

- The main content classes in football video can be mapped automatically using audio information alone.
- The key events of a football match can be recognised by identifying audio patterns correlated to commentator and crowd excitement.

A number of sub-conclusions were also demonstrated from experimentation:

- Mel-frequency Cepstral coefficients (MFCC) feature set is more robust for parameterising the audio track of Football video than Linear Predictive Coding Cepstral coefficient and Cepstral coefficients.
- Appending log energy and the 0^{th} cepstral coefficient to MFCC provides a more discriminative feature set than the standard configuration used in speech recognition systems.
- HMM is a useful tool for automatic content analysis, by modelling the underlying processes that generate the audio track. From observing the output of each state, those structures that were modelled by the HMM, can be recognised in new sequences.
- When modelling individual content classes, careful selection of both hidden state number, and the number of mixture components per hidden state, can improve classification accuracy.
- For modelling broad content classes in football audio the simpler GMM is as effective as HMM, in terms of classification accuracy.
- For modelling the sub-structures found contained in the main semantic content classes, such as Game, HMM was discovered to be a more accurate statistical representation than GMM.
- BICseg, a robust Metric-based segmentation algorithm, was shown to be more precise than Model-based and traditional Metric-based segmentation algorithms, when applied to both indexing problems.

To summarise, the results of this thesis demonstrated that by utilising and modelling audio patterns found in football video, important information can be indexed such as low level video structure and key events. The three main problems defined at the outset of the thesis were addressed with a degree of success. The first problem was the gathering of domain knowledge, an important step towards effective indexing algorithms. Also, two specific indexing problems were studied in this thesis: *structure segmentation and classification*, and *event detection*.

I believe this thesis will be a useful guide for developing indexing algorithms for other video domains, where through only careful consideration and investigation into the application of many techniques, will there be advances in video indexing.

11.2 Further Investigation and Future Research Directions

In this section, I will outline various techniques or experiments that were not covered in this thesis. These issues will be discussed in the further investigation section. This thesis covered the problem of video indexing for football video, comparing the classification accuracy of various algorithms. The next stage in development would be to integrate these algorithms into video retrieval, browsing and annotation systems. New research directions, as a result of this thesis, are discussed in the future work section.

11.2.1 Further Investigation

In this section I outline the various “loose ends” that were not completely addressed within the confines of this thesis. These “loose ends” might be worthy of further investigation.

11.2.1.1 Annotation of the entire test collection

Given the scarcity of large test collections, especially for algorithm comparison and development, it would be beneficial to complete the annotation of the entire collection.

11.2.1.2 GMM model selection

For modelling broad content classes, the GMM was found to be as accurate as HMM, in terms of classification accuracy. The GMM was compared against HMM models selected using the BIC criterion. A follow up investigation into model selection strategies for GMM, may provide further gains in accuracy for GMM. Possible model selection strategies include AIC, BIC, and cross validation. By comparing such strategies against the bottom-up method, chosen in the thesis for GMM (Vlassis et al. (1999)), would also verify the assumption for selecting this approach.

11.2.1.3 HMM Model Selection Strategies

There were a number of model selection strategies that were not evaluated within the thesis (Chapter 8). Given the importance of model selection, further investigation of other strategies would be of interest. Strategies including cross validation, discriminative measures and other penalty criterion. One particular method that is of interest, as part of a follow up experiment, is AIC_c (Burnham & Anderson (1998)). This is a variation of the traditional AIC model selection criterion, which is often applied when only a small training sample is available. AIC_c factors in the training sampling size into the penalty term, thus biasing against more complex models. A possible advantage over the BIC criterion, is that AIC_c is an approximation of the Kullback-Leibler distance (Burnham & Anderson (1998)).

11.2.1.4 Investigation into the Support Vector Machine

In Chapter 4 the SVM was outlined as a potential classification framework for audio. It would be interesting to compare the SVM against the two frameworks selected for each indexing problem (GMM and HMM). Both GMM and HMM are parametric algorithms that were applied to the problem of multi-class classification with a high degree of accuracy. Comparing such algorithms, with a state of the art non-parametric algorithm, for this problem would provide an insight into the accuracy of both frameworks.

11.2.1.5 Feature Set Combinations

A possible limitation in the feature set experiments in Chapter 6, was the number of feature set combinations evaluated. For the first phase of experimentation, comparing CC, MFCC and LPCC feature sets, these feature sets were not evaluated with appended log energy or time derivatives. Further experimentation would provide an insight into the effect of appending log energy to LPCC or the CC.

Recently MPEG-7 audio features have also been developed, where the audio signal is transformed into the frequency domain by applying a PCA transformation, instead of DCT. Xiong et al. (2003*b*) and Xiong et al. (2003*a*) applied these features to a HMM event detection algorithm, for sports video. However, when compared to a HMM modelling MFCC features there were no significant difference in performance found. The MPEG-7 feature set is relatively new and was not investigated as part of the thesis, however, possible future analysis could include the evaluation of this, and other feature sets, against the more established measurements.

11.2.1.6 Feature Selection for Event Detection

The experiment for selecting the audio feature set concentrated on broad content classes such as Game and Studio. It would be of interest to repeat this experiment for those content classes found in Chapter 10, such as crowd cheering.

11.2.1.7 Further Experimentation

Evaluating each algorithm over a wide range of operational parameters can help provide an insight into the accuracy of each algorithm over various conditions. By doing so, the goal is to provide as much information about each algorithm under various settings, which can be 'tweaked' to improve accuracy. For chapters 9 and 10, further experimentation comparing each algorithm over a more comprehensive set of operation parameters would be of future interest. For example, in Chapter 9 not all operational parameters were investigated for each algorithm, as of yet. Further gains in accuracy might be found from such an investigation.

11.2.1.8 Other criterion for BICseg algorithm

For the BICseg algorithm, the BIC model selection criterion is used to determine the difference between two Gaussian probability density functions (PDF), each a representation of a neighbouring window of audio frames. This is essentially a model selection problem, where BIC is used to determine the distance between the two Gaussian PDFs. One area that can be investigated for this algorithm further, is to evaluate a number of other model selection criterion such as AIC, AIC_c (Burnham & Anderson (1998)) and MDL (Cettolo & Federico (2000)).

11.2.1.9 Audio Segmentation efficiency

In Chapter 10, an alternative method to thresholding was investigated for a traditional Metric-based algorithm, using the symmetric Kullback-Leibler distance (KL2). A dynamic threshold, which was a linear function of the energy of the audio track, was used. As the energy increased (an indication of increased activity) so did the threshold, minimising the effect noise had on segmentation. This decision process improved accuracy over a global threshold, but was not as accurate as BICseg as a chosen criteria.

This approach to event detection has one advantage over BICseg. In terms of efficiency, applying KL2 with a dynamic threshold for segmentation involves less computation. This is a desirable property for real time systems, such as logging events as they occur. Therefore, further investigation into developing a more efficient segmentation algorithms would be beneficial.

The linear thresholding function introduced for KL2 improved accuracy over a standard global threshold. However, the relationship between log energy and KL2 is not directly linear. Future investigation into a more suitable representation could improve accuracy of these segmentation algorithm further.

A similar strategy was investigated for video shot segmentation by Vasconcelos & Lippman (1997) was modelled shot length distribution and the relationship it had with shot transition. The time period between shots closely followed that of the Weibull distribution. The longer the time period after the last shot transition, the more likely the probability of a shot change. Prior knowledge of the expected ‘arrival rate’ of a new shot was then integrated into the decision process, for a standard shot segmentation algorithm. To determine shot transition, the threshold was dynamically calculated as a function of the shot length distribution. For example, the longer the time period since the previous shot, the lower the threshold would become. The algorithm was further extended to include both shot length and shot activity[3] as part of the threshold function (Vasconcelos & Lippman (2000)). Overall segmentation results were improved when compared to a standard global thresholding algorithm, where the analysis was recorded over a small test collection.

[3]Shot activity is the histogram distance between neighbouring frames. High shot activity can be a result of object or camera motion. An increase in motion will also result in an increase in shot activity.

A similar investigation into developing such a threshold for audio segmentation may improve segmentation precision.

11.2.1.10 Further Decision Processes for Event Detection

Only the maximum likelihood criteria was investigated for event detection, as a decision process for classification. Given the success of the maximum likelihood method, further gains may be discovered by investigating more complex decision processes, such as dynamic programming and SuperHMM. For example, it was highlighted that games of high importance produced an exaggerated level of crowd response when compared to other matches. Other sources of information, such as crowd cheering duration and, the pattern classes that immediately follow the detected event, might be an important indicator for event type and importance. Also, segments classified as containing crowd singing, not directly related to a key event, might be important clues for the location and the importance of key events. A more robust decision process could integrate these clues for an improved event detection algorithm..

A number of these false detections were also not tagged as exciting events, but the passage of play did produce a crowd response. A more intelligent decision process strategy could improve the precision of the event detection algorithm, minimising human involvement in the indexing process. With the current algorithm, there were a number of events not in the test collection that were picked up by the system. Human involvement would be required for verifying those false detections as either events or not, thus making the annotation process semi-automatic.

11.2.1.11 Utilise the classes in the Event Detection algorithm further

For event detection, only the crowd cheering class was utilised. However, the other classes could also provide indication and description about the game. For example, the whistle class was found to be accurate 87%. Therefore, locating the referee whistle could assist with finding the start and stop points of the match. Also locating music in the stadium can be considered important, e.g. national anthems and goals.

11.2.2 Future Work

In the following paragraphs, I outline a number of directions for possible future research. These directions either describe various aspects of the thesis, that might be worthy of further investigation, or stem as a consequence of the findings of the thesis.

11.2.2.1 Selection Strategy for mixtures in HMM

In Chapter 8, it was identified that all the HMM selection strategies experienced problems for finding the optimal number of mixtures per state. A possible reason suggested for this, was, that not all states in the HMM could be represented by the same mixture architecture e.g. the same number of mixture components per state. Therefore, further investigation into methods for selecting the ‘optimal’ number of mixture components per state could generate a more robust HMM representation for broad audio content classes.

11.2.2.2 Integration of further modalities

Given the success of applying audio as an information source for indexing football video, the next stage in algorithm development would be to extend the indexing algorithms to integrate further modalities such as vision, motion and textual information. For other video domains, such as News and Film (see Appendix C for further details), the inclusion of multi-modal features has improved indexing accuracy.

11.2.2.3 Semi-automate the annotation process for algorithm development

Given the success of both indexing algorithms for recognising content found in football video, these techniques can be integrated into an annotation tool for building future test collections. The workload for annotation would then be reduced by providing a content guide for new video. Therefore automatising some of the more laborious tasks required for annotation. Also, the generation of large training and test collections is not cheap, therefore, some degree of automation in the annotation process is beneficial. A similar annotation system has been developed by IBM called VideoAnnEx for general video (IBM (2004)). However, for sports video, an initial prototype system has been proposed and implemented, outlined in Appendix K.

11.2.2.4 Event Classification

This thesis has demonstrated that key events can be detected by recognising audio patterns such as crowd cheering and commentator excitement. The next stage in algorithm development would be to classify these events into categories, such as goal, shot and foul. To do so, it would be necessary to investigate further modalities. One rich source of evidence is the commentary track. Transcribing the commentary track would provide clues into the type of event that has occurred. By identifying key phrases, at the location of recognised events, event classification might be possible. Babaguchi et al. (2002) have attempted such approach using closed caption text generated for sports events. However, these closed caption transcripts are not so readily available for many broadcast channels. Speech transcription is also affected by noisy audio environments such as Game sequences, where word error rate increases dramatically. The event detection algorithm proposed in this thesis, provides the facility to recognise difficult sound environments, which will assist future speech recognition applications. By identifying and differentiating between speech segments that contain a lot of noise or not, word error rate could improve (Chen et al. (2002), Spina (2000)). Speech transcription algorithms could be tailored to adapt to changing audio environments such as those found in Game sequences. Part of this solution would be to pre-identify high levels of noise that would affect standard speech transcription systems.

Other modalities such as visual and motion information would also enhance event classification. Related research into visual based event classification includes Assfalg et al. (2002) and Ekin & Tekalp (2002). These modalities are important for detecting events that are not recognised through audio.

11.2.2.5 Further investigation in audio parametrisation

From the thesis, the audio features chosen for experimentation were selected from reviewing related work in other fields, and in particular automatic speech recognition, where feature sets such as MFCC and LPCC are well researched. However, each feature set was developed with the goal of recognising speech. For non-speech sound classes it would be beneficial to investigate other forms of parametrisation. Recently for the problem of modelling and extracting information from Music, Scheirer (2000) investigated other non-speech related forms of audio parametrisation. A similar investigation would be fruitful for the audio content found in Sports video.

11.2.2.6 Indexing other content classes

It would appear, just from this investigation, that content classes other than the Game sequences also contain important information. For example, Studio sequences contain highlight replays of the current match, which are analysed by a panel of experts. Locating highlight replay sequences in these Studio units would be easier than locating the same replays from Game sequences. Studio units contain easier to detect reference points, such as shots of the studio panellists. Algorithms based on news anchor-person detection (Hauptmann & Witbrock (1998), Browne et al. (2003)) could be applied to detect these highlights sequences, rather than concentrating on the more difficult game units (van Beek et al. (2001)). The contrast between the playing field surface and a studio set would be easier to detect, with the added bonus that highlight replays in studio segments will contain events that are important for the studio panellists to chat about. Detecting highlight sequences from the Studio units could also be a useful tool for locating the actual event, and where it occurred in the Game sequence.

Bibliography

Adams, W. H., Iyengar, G., Lin, C.-Y., Naphade, M. R., Neti, C., Nock, N. J. & Smith, J. R. (2003), 'Semantic Indexing of Multimedia Content using Visual, Audio and Text cues', *EURASIP Journal on Applied Signal Processing* **2**, 1–16.

Aigrain, P., Joly, P. & Longueville, V. (1997), Medium knowledge-based macro-segmentation of video into sequences, *in* 'Intelligent multimedia information retrieval', MIT Press, pp. 159–173.

Akaike, H. (1974), A new look at the statistical model identification, *in* 'Transactions on Automatic Control', Vol. AC-19, IEEE, pp. 716–723.

Alatan, A. A., Akansu, A. N. & Wolf, W. (2001), Multi-modal Dialogue Scene Detection using Hidden Markov Models for Content-based Multimedia Indexing, *in* 'International Journal on Multimedia Tools and Applications', Vol. 14, Kluwer Academic Publishers, pp. 137–151.

Assfalg, J., Bertini, M., Colombo, C. & Bimbo, A. D. (2002), Semantic Annotation of Sports Video, *in* 'IEEE Multimedia', Vol. 9, IEEE, pp. 52–60.

Assfalg, J., Bertini, M., Colombo, C., Bimbo, A. D. & Nunziati, W. (2003), Semantic Annotation of Soccer Videos: Automatic Highlights identification, *in* 'Computer Vision and Image Understanding', Vol. 92, Academic Press, pp. 285–305.

Babaguchi, N., Kawai, Y. & Kitashi, T. (2002), Event based indexing of broadcasted sports video by intermodal collaboration, *in* 'Transactions on Multimedia', Vol. 4, IEEE, pp. 68–75.

Baggenstoss, P. M. (2002), Statistical Modelling Using Gaussian Mixtures and HMMs with Matlab, Technical report, Naval Undersea Warfare Center. http://www.npt.nuwc.navy.mil/Csf/index.html.

Baillie, M. & Jose, J. M. (2003), Audio-based Event Detection for Sports Video, *in* '2nd International Conference of Image and Video Retrieval (CIVR2003)', Il, USA, pp. 300–310.

Baillie, M. & Jose, J. M. (2004), An Audio-based Sports Video Video Segmentation and Event Detection Algorithm, *in* 'CVPR Event Mining Workshop', Washington DC.

Biem, A. (2003), A Model Selection Criterion for Classification: Application to HMM Topology Optimization, *in* 'The Seventh International Conference on Document Analysis and Recognition (ICDAR 2003)', IEEE.

Boreczky, J. S. & Rowe, A. L. (1996), Comparison of video shot boundary detection techniques, *in* 'Storage and Retrieval for Still Image and Video Databases 1V', SPIE, pp. 170–179.

Boreczky, J. S. & Wilcox, L. D. (1998), A Hidden Markov Model framework for video segmentation using audio and image features, *in* 'Proceedings of the IEEE International Conference on Acoustic, Speech and Signal Processing', Seattle, WA, pp. 3741–3744.

Brookes, M. (2003), 'VOICEBOX: Speech Processing Toolbox for MATLAB', Web.
URL: *http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html (Last visited December 2003)*

Browne, P., Czirjek, C., Gaughan, G., Gurrin, C., Jones, G. J. F., Lee, H., Marlow, S., McDonald, K., Murphy, N., O'Connor, N. E., O'Hare, N., Smeaton, A. F. & Ye, J. (2003), Dublin City University Video Track Experiments for TREC 2003, *in* 'In E M Voorhees and D K Harman, editors, Proceedings of the Twelve Text REtrieval Conference (TREC-12)', Gaithersburg, MD.

Browne, P., Smeaton, A., Murphy, N., O'Connor, N., Marlow, S. & Berrut, C. (2000), Evaluating and Combining Digital Video Shot Boundary Detection Algorithms, *in* 'IMVIP 2000 - Irish Machine Vision and Image Processing Conference', Belfast, Northern Ireland.

Brunelli, R., Mich, O. & Modema, C. M. (1999), 'A Survey on the Automatic Indexing of Video Data', *Journal of Visual Communication and Image Representation* **10**, 78–112.

Burges, C. J. C. (1998), A Tutorial on Support Vector Machines for Pattern Recognition, *in* 'Data Mining and Knowledge Discovery', Vol. 2, pp. 121–167.

Burnham, K. P. & Anderson, D. R. (1998), *Model Selection and Inference: A Practical Information Theoretic Approach*, Springer-Verlag.

Cambridge Labs (2003), 'The probabilistic model toolkit (pmt)', Web.
URL: *http://crl.research.compaq.com/projects/vision/pmt.html (Last visited December 2003)*

Campbell Jr., J. P. (1997), Speaker Recognition: A tutorial, *in* 'Proceedings of the IEEE', Vol. 85, IEEE, pp. 1437–1462.

Cettolo, M. & Federico, M. (2000), Model Selection Criteria for Acoustic Segmentation, *in* 'Automatic Speech Recognition: Challenges for the new Millennium (ASR2000)', Paris, France, pp. 221–227.

Cettolo, M. & Vescovi, M. (2003), Efficient Audio Segmentation Algorithms Based on the BIC, *in* 'International Conference on Acoustics, Speech, and Signal Processing (ICASSP03)', Hong Kong.

Chahir, Y. & Chen, L. (1999), Automatic Video Segmentation and Indexing, *in* 'Intelligent Robots and Computer Vision XVIII: Algorithms, Techniques, and Active Vision', Vol. 3837, SPIE, Washington, USA.

Chaisorn, L., T.-S.Chua, C.-K, Koh, Zhao, Y., Xu, H., Feng, H. & Tian, Q. (2003), A Two-Level Multi-Modal Approach for Story Segmentation of Large News Video Corpus, *in* 'In E M Voorhees and D K Harman, editors, Proceedings of the Twelve Text REtrieval Conference (TREC-12)', Gaithersburg, MD.

Chang, Y.-L., Zeng, W., Kamel, I. & Alonso, R. (1996), Integrated Image and Speech Analysis for Content-Based Video Indexing, *in* 'Proceedings of the International Conference on Multimedia Computing and Systems', pp. 306–313.

Chen, S. S., Eide, E. M., Gales, M. J. F., Gopinath, R. A., Kanevsky, D. & Olsen.,

P. (2002), ‘Automatic Transcription of Broadcast News’, *Speech Communication* **37**(1), 69–87.

Chen, S. S. & Gopalakrishnan, P. S. (1998), Speaker, Environment and Channel Change Detection and Clustering Via the Bayesian Information Criterion, *in* ‘Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop’, DARPA, pp. 127–132.

Chen, S. S. & Gopinath, R. A. (1999), Model Selection in Acoustic Modeling, *in* ‘Proceedings of Eurospeech 99’, Eurospeech, Hungary.

Cowling, M. & Sitte, R. (2003), Comparison of techniques for environmental sound recognition, *in* ‘Pattern Recognition Letters’, Vol. 24, Elsevier, pp. 2895–2907.

Dartfish (2004), ‘Website’. Last visited July 2004.
URL: *http://www.dartfish.com/*

de Wet, F., Cranen, B., de Veth, J. & Boves, L. (2000), A comparison of LPC and FFT-based acoustic features for noise robust ASR, *in* ‘Proceedings of Eurospeech 2001’, Aalborg, Denmark, pp. 865–868.

Del Bimbo, A. (1999), *Visual Information Retrieval*, Morgan Kauffman Publishing, San Francisco, California. pages 203-230.

Delacourt, P., Kryze, D. & Wellekens, C. J. (1999), Speaker-based segmentation for audio data indexing, *in* ‘Proceedings of the ESCA workshop: Accessing information in spoken audio’, Cambridge University, pp. 78–83.
URL: *http://svr-www.eng.cam.ac.uk/˜ajr/esca99/ (Last visited July 2004)*

Dietterich, T. G. (2002), ‘Machine Learning for Sequential Data: A Review’, In T. Caelli (Ed.) Lecture Notes in Computer Science. Springer-Verlag.
URL: *http://www.cs.orst.edu/ tgd (Last visited July 2004)*

DivXNetworks Inc. (2004), ‘Divx codec’, Website.
URL: *http://www.DivX.com (Last visited July 2004)*

Duda, R. O., Hart, P. E. & Stork, D. G. (2000), *Pattern Classification*, Wiley-Interscience Publication.

Eickeler, S. & Muller, S. (1999), Content-based video indexing of TV broadcast news using Hidden Markov Models, *in* ‘Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP99)’, IEEE, Phoneix, USA.

Ekin, A. & Tekalp, A. M. (2002), A Framework for Tracking and Analysis of Soccer Video, *in* ‘Proceedings of the Visual Communications and Image Processing (VCIP)’, SPIE.

Ekin, A., Tekalp, A. M. & Mehrotra, R. (2003), ‘Automatic Soccer Video Analysis and Summarization’, *IEEE Transactions on Image Processing* **7**(12), 796–807.

FIFA (2002), ‘Official Japan-Korea World Cup 2002 Website’, Website.
URL: *http://www.fifaworldcup.com (Last visited March 2004)*

FIFA (2003), ‘Laws of the Game’, Website.
URL: *http://www.premierleague.com (Last visited December 2003)*

Fischer, S., Lienhart, R. & Effelsberg, W. (1995), Automatic recognition of film genres, *in* ‘Proceedings of the third ACM international conference on Multimedia’, ACM Press, pp. 295–304.

Gall, L. D. (1991), ‘MPEG: A Video Compression Standard for Multimedia’, *Communications of the ACM* **34**(4), 46–58.

Garg, A., Naphade, M. & Huang, T. S. (2003), *The Handbook of Video Databases Design and applications*, CRC Press, chapter Modeling Video using Input/Output Markov models with application to multi-modal event detection.
URL: *http://www.ifp.uiuc.edu/ ashutosh/publications.html (Last Visited July 2004)*

Gish, H., Siu, M. H. & Rohlicek, R. (1991), Segregation of Speakers for Speech Recognition and Speaker Identification, *in* ‘IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP91)’, pp. 873–876.

Gong, Y., Lim, T. & Chua, H. (1995), Automatic Parsing of TV Soccer Programs, *in* ‘Proceedings of the IEEE International Conference on Multimedia Computing and Systems’, pp. 167–174.

Grimble, A. M. (2004), A Sports Video Browsing and Annotation Tool, Msc, Computing Science Department, University of Glasgow.

Gu, L. & Rose, K. (2002), Sub-state Tying With Combined Parameter Training and

Reduction in Tied-Mixture HMM Design, *in* ‘Transactions On Speech and Audio Processing’, Vol. 10, IEEE, p. 2002.

Guo, G. & Li, S. Z. (2003), ‘Content-based Audio Classification and Retrieval by Support Vector Machines’, *IEEE Transactions on Neural Networks* **14**(1), 209–215.

Haering, N., Qian, R. & Sezan, I. (1999), ‘A Semantic Event Detection Approach and Its Application to Detecting Hunts in Wildlife Video’, *IEEE Transactions on Circuits and Systems for Video Technology* pp. 78–112.

Hanjalic, A. (2002), Shot-Boundary detection: Unravelled and Resolved?, *in* ‘IEEE Transactions on Circuits and Systems for Video Technology’, Vol. 12, IEEE, pp. 90–105.

Hanjalic, A., Lagendijk, R. L. & Biemond, J. (1997*a*), A novel video parsing method with improved thresholding, *in* ‘Third Annual conference of the Advanced School for Computing and Imaging (ASCI’97)’.

Hanjalic, A., Lagendijk, R. L. & Biemond, J. (1997*b*), Detection of global story units in full-length movies, *in* ‘IEEE Workshop on Content-based Access of Image and Video Libraries’.

Hauptmann, A. G. & Witbrock, M. J. (1998), Story segmentation and detection of commercials in broadcast news video, *in* ‘ADL-98 Advances in Digital Libraries Conference.’, ADL, Santa Barbara, CA.

Hornby, N. (2000), *Fever Pitch*, Penguin Books.

Huang, J. Liu, Z. & Wang, Y. (2000), Joint Video Scene Segmentation and classification based on Hidden Markov Model, *in* ‘Proceedings of the International Conference on Multimedia and Expo (ICME2000)’, IEEE.

IBM (2004), ‘The VideoAnnEx System’, Website.
URL: *http://www.research.ibm.com/VideoAnnEx/ (Last visited July 2004)*

Intille, S. S. (1997), Tracking using a local closed-world assumption: Tracking in the football domain, *in* ‘Proc SPIE Storage and Retrieval for Image and Video Databases’, pp. 216–227.

Jain, A. K., Duin, R. P. W. & Mao, J. (2000), ‘Statistical Pattern Recognition: A

Review', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**(1), 4–37.

JPEG (2004), 'Joint Photographic Experts Group (JPEG), The JPEG Standard', Web. **URL:** *http://www.JPEG.org (Last visited July 2004)*

Juang, B. H., Chou, W. & Lee, C. H. (1997), Minimum Classification Error Rate Methods for Speech Recognition, *in* 'Transactions In Speech and Audio Processing', Vol. 5, IEEE, p. 1997.

Kemp, T., Schmidt, M., Westphal, M. & Waibel, A. (2000), Strategies For Automatic Segmentation of Audio Data, *in* 'Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP00)', IEEE, pp. 1423–1426.

Kobla, K., Doermann, D. & DeMenthon, D. (2000), Identification of sports videos using replay, text, and camera motion features, *in* 'Proceedings of the SPIE Conference on Storage and Retrieval for Media Databases', Vol. 3972, pp. 332–343.

Koprinska, I. & Carrato, S. (2000), Temporal Video Segmentation: a Survey, Signal processing: Image communication, Institute for Information Technologies, Sofia, Bulgaria.

Lehmann, E. L. (1975), *Nonparametric Statistical Methods Based on Ranks*, McGraw-Hill, New York.

Li, C. & Biswas, G. (2000), A Bayesian Approach to Temporal Data Clustering using Hidden Markov Models, *in* 'Proceedings of the International Conference on Machine Learning', ICML, Stanford, California, pp. 543–550.

Li, D., Sethi, I. K., Dimitrova, N. & McGee, T. (2001), Classification of general audio data for content-based retrieval, *in* 'Pattern Recognition Letters', Vol. 22, Elsevier, pp. 533–544.

Lin, T. & Zhang, H. J. (2000), Automatic video scene extraction by shot grouping, *in* 'Proceedings of the International Conference on Pattern Recognition (ICPR'00)', Barcelona, Spain.

Liu, Z., Huang, J. & Wang, Y. (1998), Classification of TV Programs Based on Audio Information Using Hidden Markov Model, *in* 'Proceedings of 1998 IEEE international Conference on Image Processing (ICIP98)', Vol. 3, Chicago, IL, pp. 526–530.

Liu, Z., Y. W. & Chen, T. (1998), 'Audio Feature Extraction and Analysis for Scene Segmentation and Classification', *Journal of VLSI Signal Processing Syst. for Signal, Image, and Video Technology* **20**(2), 61–80.

Lu, L., Zhang, H.-J. & Jiang, H. (2002), 'Content Analysis for Audio Classification and Segmentation', *IEEE Transactions on Speech and Audio Processing* **10**(7).

Marques, J. & Moreno, P. J. (1999), A Study of Musical Instrument Classification using Gaussian Mixture Models and Support Vector Machines, Tech Report CRL 99/4, Cambridge Research Laboratory, Massachusetts, USA.
URL: *http://www.crl.research.digital.com (Last visited October 2003)*

Meinedo, H. & Neto, J. (2003), Audio segmentation, Classification and Clustering in a Broadcast News Task, *in* 'Proceedings of the ICASSP2003 Workshop on Spoken Document Retrieval (MSDR2003)', Hong Kong, China, pp. 95–100.

Meng, J., Juan, Y. & Chang, S. F. (1995), Scene Change Detection in a MPEG Compressed Video Sequence, *in* 'IS & T / SPIE Symposium Proceedings on Digital Video Compression: Algorithms and Technologies', Vol. 2419, San Jose, California, pp. 14–25.

Microsoft (2004), 'Windows Media Codec', Web.
URL: *http://www.microsoft.com/windows/windowsmedia/format/default.aspx (Last visited July 2004)*

Nepal, S., Srinivasan, U. & Reynolds, G. (2001), Automatic detection of Goal segments in basketball videos, *in* 'Proceedings of the ninth ACM international conference on Multimedia', ACM Press, Ottawa, Canada, pp. 261–269.

Ney, H. & Ortmanns, S. (1999), 'Dynamic Programming Search for Continuous Speech Recognition', *IEEE Signal Processing Magazine* **16**(5), 64–83.

Ney, H. & Ortmanns, S. (2000), 'Progress on Dynamic Programming Search for LVSRC', *IEEE Signal Processing Magazine* **88**(8), 1224–1240.

Ogg Vorbis (2004), 'Xiph.org foundation, ogg vorbis codec', Web.
URL: *http://www.vorbis.com/ (Last visited July 2004)*

O'Toole, C., Smeaton, A., Murphy, N. & Marlow, S. (1999), Evaluation of Automatic

Shot Boundary Detection on Large Video Test Suite, *in* ‘Proceedings of Challenges in Image Retrieval’, Newcastle, UK.

Pan, D. (1995), A Tutorial on MPEG Audio Compression, *in* ‘IEEE Multimedia’, Vol. 2, pp. 60–74.

Pan, D. Y. (1993), ‘Digital Audio Compression’, *Digital Technical Journal* **5**(2).

Pan, H., Li, B. & Sezan, M. I. (2002), Detection of slow-motion replay segments in sports video for highlights generation, *in* ‘IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP02)’, Salt Lake City, Utah.

Petkovic, M., Mihajlovic, V. & Jonker, W. (2002), Multi-Modal Extraction Of Highlights From Tv Formula 1 Programs, *in* ‘IEEE’, International Conference on Multimedia and Expo, Lausanne, Switzerland.

Pickering, M. J. & Ruger, S. M. (2001), Multi-timescale video shot-change detection, *in* ‘In E M Voorhees and D K Harman, editors, Proceedings of the Tenth Text REtrieval Conference (TREC-10)’, Gaithersburg, MD.

Pietquin, O., Couvreur, L. & Couvreur, P. (2001), Applied Clustering for Automatic Speaker-Based Segmentation of Audio Material, *in* ‘Belguim Journal of Operators Research Statistics and Computer Science’, Vol. 41, JORBEL, pp. 69–81.

Porikli, F. & Haga, T. (2004), Event Detection by Eigenvector Decomposition Using Object and Frame Features, *in* ‘Submitted to the CVPR Event Mining Workshop’, Washington DC.

Rabiner, L. & Juang, B. H. (1993), *Fundamentals of Speech Recognition*, Prentice Hall, Englewood Cliffs, NJ, USA.

Ramirez, J., Segura, J. C. & Benitez, C. (2004), ‘A New Kullback-Leibler VAD for Speech Recognition in Noise’, *IEEE Signal Processing Letters* **11**(2).

Real Networks (2004), ‘Real Player Codec’, Web.
URL: *http://www.realnetworks.com (Last visited July 2004)*

Reyes-Gomez, M. J. & Ellis, D. P. W. (2003), Selection, Parameter Estimation, and Discriminative Training of Hidden Markov Models for General Audio Modeling, *in* ‘Proceedings of the International Conference on Multimedia and Expo (ICME2003)’, IEEE, Balitmore, Maryland.

Roach, M. & Mason, J. (2001), Classification of Video Genre using Audio, *in* ‘Proceedings of Eurospeech-01’, Aalborg, Denmark.

Rui, Y., Gupta, A. & Acero, A. (2000), Automatically extracting highlights for TV Baseball programs, *in* ‘Proceedings of the eighth ACM international conference on Multimedia’, ACM Press, Marina del Rey, California, United States, pp. 105–115.

Rui, Y., Huang, T. S. & Mehrotra, S. (1999), ‘Constructing Table-of-Content for Videos’, *Multimedia Systems* **7**(5), 359–368.

Saunders, J. (1996), Real-time discrimination of broadcast speech/music, *in* ‘Proceedings International Conference on Acoustics, Speech and Signal Processing (ICASSP96)’, Vol. 11, IEEE, Atlanta, GA, pp. 993–996.

Scheirer, E. D. (2000), Music-Listening Systems, PhD thesis, Massachusetts Institute of Technology, USA.

Scheirer, E. & Slaney, M. (1997), Construction and Evaluation of a Robust Multifeature Speech/Music Discriminator, *in* ‘International Conference on Acoustics, Speech, and Signal Processing (ICASSP97)’, IEEE, Munich, pp. 1331–1334.

Schwarz, G. (1978), Estimating the dimension of a model, *in* ‘Annals of Statistics’, Vol. 6, pp. 461–464.

Seo, Y., Choi, S., Kim, H. & Hong, K.-S. (1997), Where Are the Ball and Players? Soccer Game Analysis with Color Based Tracking and Image Mosaick, *in* ‘Proceedings of the 9th International Conference on Image Analysis and Processing-Volume II’, Springer-Verlag, pp. 196–203.

Shahraray, B. & Gibbon, D. C. (1995), Automatic Generation of Pictorial Transcripts of Video Programs, *in* ‘Proceedings of Multimedia Computing and Networking’, Vol. 2417, SPIE.

Shin, T. S., Kim, J. G., Kim, J. & Ahn, B. H. (2000), A statistical approach to shot boundary detection in an MPEG-2 compressed video sequence, *in* ‘Visual Communications and Image Processing’, Perth, Australia.

Siegler, M. A., Jain, U., Raj, B. & Stern, R. M. (1997), Automatic segmentation, classification and clustering of broadcast news, *in* ‘DARPA Speech Recognition Workshop’.

SKY+ (2003), ‘Sky website’, Website.
URL: *http://www.sky.com/ (Last visited* 24th *April 2003)*

Smeaton, A. F. & Over, P. (2003*a*), The TREC-2002 Video Track Report, *in* ‘NIST Special Publication: SP 500-251 The Eleventh Text Retrieval Conference (TREC)’, Department of Commerce, National Institute of Standards and Technology.

Smeaton, A. F. & Over, P. (2003*b*), TRECVID: Benchmarking the Effectiveness of Information Retrieval Tasks on Digital Video, *in* ‘2nd International Conference of Image and Video Retrieval (CIVR2003)’, Il, USA, pp. 19–27.

Smeaton, A. F. & Over, P. (2004), The TREC-2003 Video Track Report, *in* ‘NIST Special Publication: The 12th Text Retrieval Conference (TREC)’, Department of Commerce, National Institute of Standards and Technology.

Smyth, P. (1997), Clustering Sequences with Hidden Markov Models, *in* M. C. Mozer, M. I. Jordan & T. Petsche, eds, ‘Advances in Neural Information Processing’, Vol. 9, MIT Press.

Snoek, C. G. M. & Worring, M. (2002), A Review on Multimodal Video Indexing, *in* ‘Proceedings of the International Conference on Multimedia and Expo (ICME02)’, IEEE, Switzerland.

Spina, M. S. (2000), Analysis and Transcription of General Audio Data, PhD thesis, M.I.T. Laboratory for Computing Science.
URL: *http://www.sls.csail.mit.edu/spina/*

Sudhir, G., Lee, J. C. M. & Jain, A. K. (1998), Automatic Classification of Tennis Video for High-level Content-based Retrieval, *in* ‘Proceedings of the 1998 International Workshop on Content-Based Access of Image and Video Databases (CAIVD ’98)’, IEEE Computer Society, p. 81.

Theodoridis, S. & Koutroumbas, K. (1999), *Pattern Recognition*, Academic Press.

TiVo Inc. (2003), ‘Tivo website’, Web.
URL: *http://www.tivo.com/ (Last visited 24th April 2003)*

Tritschler, A. & Gopinath, R. A. (1999), Improved Segmentation and Segment Clustering using the Bayesian Information Criterion, *in* ‘Eurospeech 1999’, Budapest, Hungary.

Tzanetakis, G. & Cook, P. (2002), ‘Musical Genre Classification of Audio Signals’, *IEEE Transactions On Speech and Audio Processing* **10**(5), 293–302.

Uchihashi, S., Foote, J., Girgensohm, A. & Boreczky, J. (1999), Video manga: Generating semantically meaningful video summaries, *in* ‘Proceedings ACM Multimedia’, ACM Press, Orlando, Florida, pp. 383–392.

van Beek, P., Pan, H. & Sezan, M. I. (2001), Detection of slow-motion replay segments in sports video for highlights generation, *in* ‘International Conference on Acoustics, Speech and Signal Processing (ICASSP2001)’, Salt Lake City, Utah.

Vasconcelos, N. & Lippman, A. (1997), A Bayesian Video Modeling Framework for Shot Segmentation and Content Characterization, *in* ‘Proceedings of the 1997 Workshop on Content-Based Access of Image and Video Libraries (CBAIVL ’97)’, IEEE Computer Society, p. 59.

Vasconcelos, N. & Lippman, A. (2000), ‘Statistical Models of Video Structure for Content Analysis and Characterization’, *IEEE Transactions on Image Processing* **9**(1).

Veneau, E., Ronfard, R. & Bouthemy, P. (2000), From video shot clustering to sequence segmentation, *in* ‘Proceedings of the International Conference on Pattern Recognition (ICPR’00)’, Barcelona, Spain.

Vlassis, N., Papakonstantinou, G. & Tsanakas, P. (1999), ‘Mixture density estimation based on maximum likelihood and test statistics’, *Neural Processing Letters* **9**(1), 63–76.

Wang, Y., Liu, Z. & Huang, J. (2000), Multimedia content analysis using both audio and visual clues, *in* ‘IEEE Signal Processing Magazine’, IEEE.

Wold, E., Blum, T., Keislar, D. & Wheaton, J. (1996), Content-based classification, search, and retrieval of audio, *in* ‘IEEE Multimedia’, Vol. 3, IEEE, pp. 27–36.

Wolf, W. (1997), Hidden Markov model parsing of video programs, *in* ‘Proceedings of IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP97)', Munich, Germany, pp. 2609–2611.

Xie, L., Chang, S.-F., Divakaran, A. & Sun, H. (2002), Structure Analysis of Soccer Video with Hidden Markov Models, *in* 'Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP02)', Orlando, Florida.

Xiong, Z., Radhakrishnan, R., Divakaran, A. & Huang, T. S. (2003*a*), Audio-based Highlights Extraction from Baseball, Golf and Soccer Games in A Unified Framework, *in* 'Proceedings of the International Conference on Acoustics Speech and Signal Processing (ICASSP2003)', Hong Kong, China.

Xiong, Z., Radhakrishnan, R., Divakaran, A. & Huang, T. S. (2003*b*), Comparing MFCC and MPEG-7 Audio Features for Feature Extraction, Maximum Likelihood HMM and Entropic Prior HMM for Sports Audio Classification, *in* 'Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP2003)', Hong Kong, China.

Xu, P., Xie, L., Chang, S.-F., Divakaran, A., Vetro, A. & Sun, H. (2001), Algorithms and Systems for Segmentation and Structure Analysis in Soccer Video, *in* 'Proceedings of the IEEE International Conference on Multimedia and Expo (ICME01)', Tokyo, Japan.

Xvid (2004), 'Xvid codec', Web.
URL: *http://www.Xvid.org (Last visited July 2004)*

Yeung, M. & Yeo, B.-L. (1998), Segmentation of Video by Clustering and Graph Analysis, *in* 'Computer Vision and Image Understanding', Vol. 71, pp. 94–109.

Young, S., Jansen, J., Odell, J., Ollason, D. & Woodland, P. (2003), *The HTK Book*, Web, Entropic Cambridge Research Laboratory.
URL: *http://htk.eng.cam.ac.uk/ (Last visited November 2003)*

Yow, D., Yeo, B. L., Yeung, M. & Liu, B. (1995), Analysis and Presentation of Soccer Highlights from Digital Video, *in* 'Proceedings of the Second Asian Conference on Computer Vision', Vol. II, pp. 499–503.

Zhang, H. J., Smoliar, S. W. & Wu, J. H. (1995), Content-Based Video Browsing Tools, *in* ‘Proceedings of the SPIE Conference on Multimedia Computing and Networking’, San Jose.

Zhang, T. & Kuo, C. C. J. (2001), ‘Audio Content Analysis for Online Audiovisual data segmentation and classification’, *IEEE Transactions on Speech and Audio Processing* **9**(4), 441–447.

Zheng, F., Zhang, G. & Song, Z. (2001), ‘Comparison of different implementations of mfcc’, *Journal of Computer Science and Technology* **16**(6), 1–6.

Appendix A

Sound Recording Basics

Sound can be defined as a continuous flow of pressure waves created by a single or a mixture of multiple sources. The sound-generating source, be it a human voice box, a musical instrument or a footballer striking a ball, generates these pressure waves. A sound receiving device such as the human ear or a microphone will then pick up these pressure waves, which are then converted into a form ready for interpretation. A simple example of this process would be that of speech. When a human speaks air is pushed from the lungs and released out through the vocal tract, becoming speech. A combination of both vibration and relaxation from the vocal chords and the shape of the vocal tract determines which sounds are made. The speed of the vocal vibration also determines the pitch of the voice. For example, woman and child tend to have faster vibration, hence a higher pitched voice compared to that of an average adult male. Finally, the volume of air released from the lungs determines the volume or loudness of the voice

Once a phrase has been released, the ear will pick up these pressure waves. The human auditory system contains an organ called the cochlea found in the inner ear. This organ has the ability to separate out the pressure waves into interpretable information, where each component from the varying air pressure resembles that of a sinusoid. A sinusoid is a mathematical function that traces out the simplest repetitive motion in nature. A single sound such as a speech utterance contains many sinusoids with varying frequency and amplitude. The frequency of a signal is how many repetitions there are in a second while the amplitude is the strength of the signal[1]. From one end of the cochlea to the other, each section of the cochlea is sensitive to different frequencies. The cochlea separates out the signal and translate its various components. For example, inside the cochlea there is a reaction to the frequencies found in the signal, sending nerve impulses to the brain for interpretation and translation. This same process occurs when translating other sounds, as well, such as music.

A microphone on the other hand, picks up and transforms these sound waves into electrical vibrations. There are two popular methods of doing this for a microphone, called Dynamic and Condenser. A Dynamic microphone has a small diaphragm with a coil of wire attached to its apex that contains an electrical current. This diaphragm is moved by the changing sound waves, thus moving the coil and causing the current to be cut, converting the signal into electrical energy. The signal can then be sampled and stored as described in Appendix B. The advantage of dynamic microphones are that they are very rugged and can withstand extremely high sound levels without damage or excessive distortion. The disadvantage is that they do not respond well to extremely low frequencies.

Condenser (or capacitor) microphones use a lightweight membrane and a fixed plate, that act as opposite sides of a capacitor. Sound pressure against a thin polymer film causes it to move. This movement changes the capacitance of the circuit generating a changing electrical output. Condenser microphones, are preferred for their very uniform frequency response, and ability to respond with clarity to transient sounds. The low mass of the membrane diaphragm permits extended high-frequency response, while the nature of the design also ensures outstanding low-frequency pickup. The resulting sound is natural, clean and clear, with excellent transparency and detail.

The advantage to the condenser microphone is that has an extremely wide frequency response. It has also has good pick up sensitivity. The disadvantage is that its easy to overload and distort the audio signal if placed to close to a high intensity sound source. If the microphone is going to be exposed to severe environments such as outdoors, difficult sound environments, etc. Then dynamic microphone is probably the best choice. If the microphone is going to be used in controlled environments then the condenser microphone is a more suitable choice.

[1]The human ear can interpret frequencies up to around 20kHz, while the majority of speech information is contained in the frequency band up to approximately 4kHz.

Appendix B

Digital Video Media Representations

B.1 Introduction

Storage and transmission of digital video is costly, both in terms of disc space and bandwidth. Large collections of digital video material can take up a vast amount of disc space. For both efficient transmission and storage, video sequences are often compressed, minimising the volume of storage space required, to hold vast collections. In this section, I will introduce the basics of audio and visual streams, and how digital compression can be used for efficient storage and representation, with particular attention to audio and the compression algorithms used to store the test collection (MPEG-1 and WAV).

B.2 The MPEG video compression algorithm

Until recently, the most popular and widely accepted encoding standard was from the Motion Picture Experts Group (MPEG) (Gall (1991)). Compression algorithms such as MPEG-1 and MPEG-2 reduce the original video stream into a collection of smaller bit tokens providing efficient storage and transmission. The has been a recent development of competing codecs[1] such as the Microsoft version of MPEG-4 (Microsoft (2004)), Real Media (Real Networks (2004)), and the open source codecs; DivX (Di-

[1]A term used to describe the encoding and decoding algorithms of a compression format.

vXNetworks Inc. (2004)) and Xvid (Xvid (2004)). These codec's are now competing with MPEG, particulary across mediums such as the internet.

There are two popular compression algorithms from the Motion Picture Experts Group (MPEG) (Gall (1991)); MPEG-1 and MPEG-2. MPEG-1 is optimised for applications such as the CD-ROM, while MPEG-2 has uses in the high definition Digital TV domain and Digital versatile disc (DVD) production. The MPEG standard allows for the compression of 'raw' video into a stream of smaller tokens or bits. This is achieved through encoding algorithms using techniques to capture temporal and spatial redundancy.

The video stream is converted into a bit-stream of smaller tokens. Redundant information within the decompressed video is effectively removed, allowing for a high compression rate. Steps in the compression algorithm include separating frames into independent groups. Within these groups are three types of frames, I, P and B. The I being an anchor frame encoded using the JPEG compression standard for images (JPEG (2004)). The P frame is a further compressed frame predicted from the previous I frame, while the B frames are predicted solely on surrounding I or P frames.

Hence a decoded stream has to encode a group of I, P and B frames, rather than iteratively decoding sequential frames. The MPEG algorithm also outlines the decoding process for playback. The coded bits are mapped from the compact representation into the original, 'raw' format of the image sequence. For example, a flag in the coded bit-stream signals whether the following bits are to be decoded with a DCT (Discrete Cosine Transformation) algorithm or with a prediction algorithm.

Each P and B frame is further divided up into units called macroblocks. At this level, spatial and temporal redundancies are also removed to decrease file size. Temporal redundancy is reduced using motion compensation applying predictive and interpolative techniques. The prediction error signal is further compressed using spatial redundancy reduction techniques. It is in the compression of the P and B frames that the compression gains are found. For full details on MPEG please refer to Gall (1991).

B.3 Digital Audio Compression

The storage of audio as part of a video stream, or, as a separate audio file is also costly. It is desirable to compress the original stream into a more efficient format. Current encoding formats include AIFF, AU, WAV (all Pan (1995)), Real Audio (Real Networks (2004)), the Windows Media Format WMA (Microsoft (2004)), Ogg Vorbis (Ogg Vorbis (2004)), and probably the most widely used, MP3 (Pan (1995)).

The conversion from an analog audio stream to digital is achieved by sampling the audio input in regularly timed discrete intervals (a basic introduction into sound can be found in Appendix A). These samples are quantised into 8 or 16-bit integer values. The digital audio will then consist of a sequence of binary values which represent the number of ‘quantizer’ levels for each sample. All samples are then represented with an independent code word. This is called pulse code modulation (PCM), often referred to as uncompressed audio.

High quality CD recordings contain uncompressed digital audio consisting of a 16-bit sampled audio stream (sometimes 8 bits per sample is used for lower quality recordings), sampled at a rate of 44100Hz (or samples per second). Digital Versatile Disc (DVD) formats can allow for a maximum of 96kHz sampling. The higher the sampling rate, the better the sound quality. On the downside, one minute of 44100Hz, 16 bit Stereo CD quality audio can require up 10.3MB of disc space.

Formats such as the Microsoft and IBM’s WAV apply variations of PCM. The WAV format applies a technique called Adaptive Differential Pulse Code Modulation. Given the similarity between neighbouring samples, each sample is not treated independently but as a difference. ADPCM is a more efficient version of the PCM digitising method. It converts analog signals to digital data and is used to transmit analog voice over digital channels. ADPCM uses a lower bit rate than PCM. Both the difference between samples and the coding scale can be dynamically changed to compensate for amplitude and frequency variations (see Pan (1993) for an in-depth introduction).

Current popular compression formats including Real Audio, WMA, Ogg and MP3 standards, are able to compress the audio stream further. This allows for efficient storage of large collections and transmission across mediums such as the internet. The most popular digital audio encoding algorithm being MP3. MP3 takes advantage of the perception limitations of the human ear, removing perceptually irrelevant parts of the audio signal that the human ear may not pick up. MP3 audio compression is considered a high-complexity, high-compression and hence very high quality encoding algorithm. For more details see a tutorial by Pan (1995). The MP3 is the current standard of choice for many applications such as music storage and video compression. Therefore the audio track is encoded using this format. For the analysis of the audio stream, the audio track is decoded into WAV.

Appendix C

An Overview of General Video Indexing

C.1 Introduction

The process of indexing video content is an essential step towards the efficient archiving of large media collections. Archived video is often reused for a variety of reasons, so it is desirable be able to query content and locate relevant video sequences. Content-based indexing can assist this search and retrieval process.

Indexing can also be beneficial for the creation of current interactive services provided by digital TV broadcasters. Interactive advancements now allow viewers to watch highlight replays of the current match as it happens. However methods employed to provide detailed indexing of video content are largely manual. It has been estimated that manual annotation can take a team of trained librarians up to ten hours to fully index one hour of video (Del Bimbo (1999)). Automatic indexing tools are a viable alternative to the labour intensive methods currently in practice. In this chapter, I provide an overview of the state of the art for automatic video indexing.

The first step towards the development of automatic indexing systems, is the recognition of low level video structure. Produced video is created by editing a combination of structures together in a logical way to create a meaningful story (Figure C.1). The smallest of these structures is called the camera shot, defined as a continuous non-

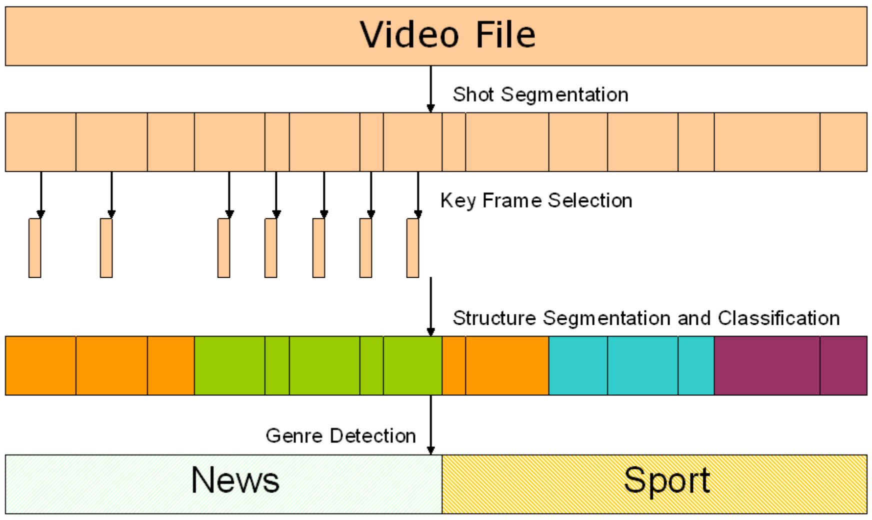


Figure C.1: The associated methods for indexing video structure.

breaking camera sequence. Shots are edited together using a number of transitions, the simplest being the cut. A sharp transition over one or two frames, joining two shots together. Detection of these transition locations is one of the most important steps in a video indexing system. Shot segmentation is the starting point for many video indexing and retrieval systems. For example, vital statistics can be extracted about the editing process by recognising shot structure. The volume of information to process for future techniques can also be reduced, by extracting a single representative keyframe from each shot.

The next structure layer is the scene. By using an analogy of a book, the video can be further broken up into chapters, or scenes. An individual shot can be thought of as one paragraph. Just like paragraphs, shots are then ordered in a specific way by the editor to create a logical sequence. In cinematic terms, these logical units are subdivisions of the overall story called scenes. Detecting the location of scene transitions is an important step towards identifying the overall video structure. Similar to the table of contents of a book, the overall video structure can be presented to the viewer as a collection of scenes. This presentation can provide an instant overview of the video content, as well as enabling instant access to specific reference points.

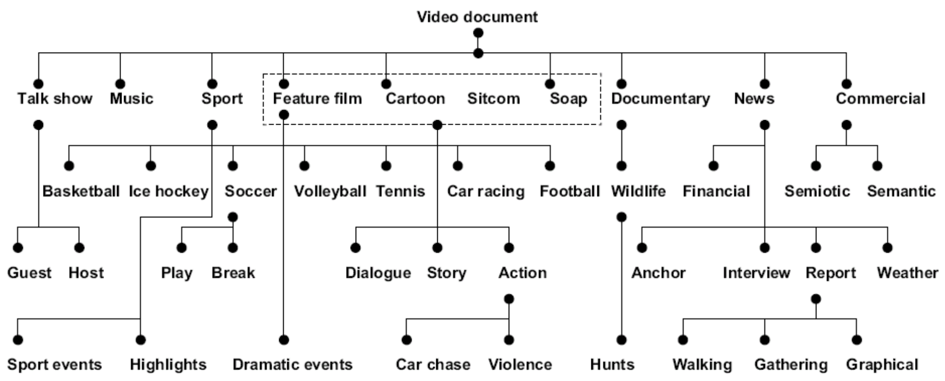


Figure C.2: This is a flow chart of the current video domains being investigated. The first level is the main genre such as news and sport. The second level is the sub-genre. The third level is the typical scene structures found in each domain, and the fourth level are typical events that can be extracted for summarisation.

Dependent on the video domain, the identification of scene structure is often referred to as scene or story segmentation. For domains such as news and sport, scenes can also be automatically provided with appropriate labels such as news or weather report (Figure C.2). The overall process of detecting these logical units is called structure segmentation and classification. Snoek & Worring (2002) provided an overview of the typical structures currently being analysed in video retrieval (Figure C.2), across a variety of domains.

Video files can be divided up into a number of main domains such as news, sport and film (Figure C.2). Within each of these domains, there are number of sub-genre. Content-based indexing algorithms can de designed utilising domain specific knowledge. A particular problem with current media management, is that often a single video file can contain a collection of programmes from different domains (Figure C.1). It is therefore of interest to provide automatic labels for each video programme. This process is referred to as genre detection. For efficient archiving it is helpful to assign each programme into a specific category. This can enable similar files to be stored together under the same genre. Automatic labelling of content into genre, is also useful for more in depth indexing of higher level concepts, such as sporting or dramatic events.

To provide an automatic summary of the highlights in a video sequence, it is important to detect important events. These key events can differ across genre (Figure C.2). For example, event detection in cinematic film could include incidents of car chases or other dramatic events. For sports video, event detection can cover scoring sequences and other such exciting incidents. The process of event detection is useful for instant summarisation of important video content, as well as the generation of automatic replays.

As part of this chapter, I provide an overview of all of these techniques (Figure C.1). Where the structure of this chapter is as follows. I first introduce current methods for shot segmentation in section C.2. Keyframe selection is an important process for higher-level indexing tasks, as well as an important method for viewing video content, described in section C.3. In section C.4, I provide detailed descriptions of structure segmentation and classification algorithms. I then describe techniques for genre detection (section C.5) and event detection (section C.6). Finally summarising the Appendix in section C.7.

C.2 Shot Segmentation

The structure of post-produced video is a collection of carefully organised camera sequences, called shots. These shots are edited together to create one logical semantic unit. The most common join between each shot is the cut. A sharp transition between adjacent shots spanning one or two frames. Other types of transition between shots occur gradually over a period of frames. These include the fade, wipe and dissolve, as well as other special effects editing. Manual indexing of these joins would be time consuming and impractical. Therefore, for the majority of video indexing systems, the first step is to capture this low-level structure, identifying the start and end point of each shot component, Figure C.1. This process is the basic foundation for many Video Retrieval tools and applications. For example, shot segmentation can assist the extraction of vital shot statistics required for higher-level structure indexing such as the mapping of scenes. The importance of shot segmentation can be highlighted by illustrated by the volume of research dedicated to this problem. The Text Retrieval Conference (TREC) have a dedicated track to video retrieval, where one of the main tasks is shot segmentation over a large collection (Smeaton & Over (2003*a*), Smeaton & Over (2004)). The video retrieval track is referred to as TRECvid.

Automatic shot segmentation algorithms can be grouped into two main categories. Those algorithms that concentrated on the raw video stream and those that are designed specifically for compressed video. Compression standards such as MPEG (Gall (1991)) have allowed for the parsing of video without fully decoding the video stream. Thus improving time complexity. This is also the disadvantage of these algorithms. Once a certain compression standard has been superseded, shot segmentation algorithm designed for that standard will also become redundant. Another disadvantage with compression-based algorithms is accuracy. Research by Boreczky & Rowe (1996) and Browne et al. (2000), have identified that algorithms working on uncompressed video are more precise. From the experiments across a large video collection in TRECvid (Smeaton & Over (2003*a*), Smeaton & Over (2004)), those algorithms analysing uncompressed video are more accurate. The information lost during the compression process is a major cause of this result.

Accuracy and longevity are the advantages of developing algorithms in the uncompressed domain. No information loss is experienced due to compression and no reliance on particular compression formats. The obvious downside being algorithm speed and efficiency. For the above reasons, in this section I focus on uncompressed shot segmentation algorithms. The majority of current, precise shot segmentation algorithms use a combination of three features; pairwise frame pixel comparison, histogram-based comparison and block matching. To determine if patterns in these features are the result of shot transition, a variety of decision processes are used. From standard thresholds to statistically modelling shot transitions. In the following sections, I examine these issues in more detail.

C.2.1 Pixel Comparison

The adhering relationship between algorithms applying pixel based techniques, is the comparison of corresponding pixel differences between successive frames. A high degree of difference between successive frames, would be an indicator for shot transition. Statistics concerning the overall change in pixel value from one frame to another, be it colour or intensity, are extracted. Shot transitions are then tagged if the overall change is greater than a predefined value. Implementation examples of these algorithms can be found in Boreczky & Rowe (1996), Brunelli et al. (1999) and Browne et al. (2000).

One of the most popular techniques within this subgroup is analysing frame differences between ‘Edge’ images (Del Bimbo (1999)). Edges are the boundaries or outlines in each frame, where there are strong intensity contrasts i.e. a jump in intensity from one pixel to the next. Converting frames into edge images is useful for the identification of gradual transitions such as fades and dissolves. The number of entering or exiting edge pixels, from a series of frames, can be counted. Gradual transitions are associated with a high degree of change in edge pixels.

The main downfall of pixel based algorithms is motion. These algorithms suffer from a sensitivity to camera or object motion. For example, a large shift in pixel values from left to right as the result of a camera pan, will result in a false transition. In particular, localised changes in pixel values as a result of object motion can often trigger false transitions. Another problem is the occurrence of ‘flash’ lights. These flashes create a sudden change in pixel values, resulting in false transitions.

C.2.2 Histogram-based

Histogram-based algorithms measure the distribution of colour or intensity between successive frames. For example, each pixel in a image has a value in a finite range. The distribution of pixels across the various levels of brightness or colour, can be displayed in a histogram. A distance measure can then be used to compare distributions from two frames. Distance measures such as chi-squared, correlation or cosine similarity. If the distance between neighbouring frames is large, a new shot is declared. Examples of systems that implement these algorithms include Boreczky & Rowe (1996), Boreczky & Wilcox (1998), O’Toole et al. (1999) and Chang et al. (1996).

The advantage of histogram-based comparison is the effect of object or camera motion is limited. The histogram distribution is taken for the overall frame. So localised spatial changes within the frame, will have little effect on the overall colour or intensity distribution. However, any sudden change in the distribution will be reflected in changes in the histogram. These changes can be the result of flashing lights or explosions. A solution to the problem of sudden changes not related to shot transition, is to examine histogram distributions at longer temporal intervals (Pickering & Ruger (2001)). Comparing frames at greater temporal ranges has been shown to limit the effect of these sudden changes. This approach is also useful for identifying gradual transitions across a number of frames, which are not picked up by frame by frame comparison.

Histogram-based algorithms are robust to object and camera movement overall. These algorithms do ignore spatial information though. In theory algorithms will fail when two different neighbouring shots contain visually similar distributions. In practice however, this problem has a very small likelihood of occurring. For the above reasons this is why histogram-based algorithms are widely applied for shot segmentation.

C.2.3 Block-based

Block-based algorithms divide each frame into a collection of sub-frames called blocks. These blocks are then analysed separately through the use of either pixel comparison or histogram-based techniques. By doing so, existing techniques become more robust to camera or object motion. Where the effect small localised object movement can have on accuracy is minimised.Using a block-based approach, camera or object motion can be estimated by comparing neighbouring blocks. Motion is estimated by matching blocks between successive frames. A block movement from one frame to another would indicate object or camera motion. This framework can also extend histogram-based algorithms to include spatial information. Where differences between frames with similar over all distributions can be found. Examples of systems that implement these algorithms include Boreczky & Wilcox (1998) and Chahir & Chen (1999).

To determine shot transition, the number of matching blocks within a localised neighbourhood can be counted. A cut is then declared if there is not a significant number of matching blocks between successive frames. An alternative method to determine shot transition, is to count the number of similar blocks overall between adjacent frames. Block-based algorithms can improve accuracy in existing pixel or histogram algorithms. Although, the computation and time complexity of the algorithms will increase dramatically.

C.2.4 Decision Process

Given a comparison between two sequential frames, what methods are applied to determine shot change? The simplest step is to employ a predefined global threshold. This cut off value can be used to determine if the difference between successive frames is large enough to signify shot transition. There are however a number of disadvantages with applying global thresholds. Thresholds can be particulary sensitive to localised differences found in video such as excessive camera or object motion. For example, if the threshold is low, a lot of false cuts would be identified as a result of object or camera motion. While if the threshold is high, a number of subtle cuts would be missed. A global threshold will become even more sensitive across (and between) video genre. Genre such as sport, is characterised by fast object motion against an unchanging background (the playing field). While News broadcasts are more static, with little camera or object motion. A global threshold suitable for all genres would be impractical. In fact, a global threshold for one video file may not be transferable to another, no matter of genre.

An active area of research for shot segmentation is the development of more intelligent decision processes. Such as a dynamic threshold that adapts to the current video state. One suggested solution is the ‘sliding window’ or ‘adaptive’ threshold. This technique can adjust the threshold in response to localised fluctuations in frame difference cause by motion. To calculate the dynamic threshold, frame differences within a localised area (or window) are compared. Large frame differences in relation to the window average, would indicate shot transition. This localised difference ratio will increase or decrease relative to the current window characteristics, caused by camera or object motion. From evaluation, it has been shown that algorithms that implement this type of decision process improve accuracy over standard global thresholds (Browne et al. (2000), Hanjalic et al. (1997*a*), Meng et al. (1995), Shin et al. (2000)). One particular disadvantage with this approach is the choice of the window size. Small windows in length will not provide a true reflection of the variation across a shot. While a large window will ignore localised changes. Selection of the window size could also be dependent on video genre.

An alternative to thresholding is to model the frame differences. Boreczky & Wilcox (1998) implemented a Hidden Markov Model (HMM) framework which modelled patterns in histogram and block-based frame differences associated with shot transition. Each hidden Markov state in the HMM was a representation of a shot transition, such as a cut or fade. A video sequence parsed into shot units, by entering the sequenced into the HMM. Where each frame is allocated to a state label such as cut, fade, camera motion or normal state. This approach was evaluated across a small test collection and was found to be more effective than a standard global thresholding strategy.

Vasconcelos & Lippman (1997) modelled shot length distribution and the relationship it has with shot transition. The time period between shots closely followed that of the Weibull distribution. The longer the time period after the last shot transition, the more likely the probability of a shot change. Prior knowledge of the expected 'arrival rate' of a new shot was then integrated into the decision process for a standard shot segmentation algorithm. To determine shot transition, the threshold is dynamically calculated as a function of shot length distribution. For example, the longer the time period since the previous shot, the lower the threshold would become. The algorithm was further extended to include both shot length and shot activity[1] as part of the threshold function (Vasconcelos & Lippman (2000)). Overall segmentation results were improved when compared to a standard global thresholding algorithm, where the analysis was recorded over a small test collection.

It would appear that Vasconcelos & Lippman (2000) can provide a significant step forward for shot segmentation algorithms. However, to really quantify the effectiveness of combining both shot length and shot activity as priors in the decision process, further comparison across a large and varied test collection is required. At present, there has been no direct comparison of both this strategy and the HMM framework (Boreczky & Wilcox (1998), against other techniques such as dynamic thresholding. Especially over a larger test collection such as TRECvid (Smeaton & Over (2003*b*)).

As part of TRECvid, research groups run shot boundary algorithms over a large dataset consisting of 4 hours and 51 minutes of video. The video files ranged in quality from old archived video to high quality production. Overall results would indicate that algorithms which implemented a combination of techniques, such as histogram difference, block-based motion compensation and edge frame comparison performed best. Specific modules for problems such as flashlight detection also increased accuracy. Sig-

[1]Shot activity is the histogram distance between neighbouring frames. High shot activity can be a result of object or camera motion. An increase in motion will also result in an increase in shot activity.

nificant gains were also found in algorithms comparing frame differences at various temporal distances, using a sliding window decision process. For full details on the results please refer to Smeaton & Over (2003*a*) and Smeaton & Over (2004).

C.2.5 Summary

Systems that employ a combination of features alongside a dynamic thresholding scheme provide the most accurate shot segmentation results. Comparing frames at longer temporal intervals can also minimise the effect of camera and object motion. The overall accuracy of current shot segmentation algorithms is impressive in domains such as news (Smeaton & Over (2004)), with a variety of different algorithms research high precision and recall across a large test collection. The future for shot segmentation algorithms would now appear how to modelling the different information sources in an intelligent framework, which can adapt to the various problems found between video genre. Problems such as constant changes in object and camera motion. Algorithms such as Boreczky & Wilcox (1998) and Vasconcelos & Lippman (2000) might provide the solution. However, these algorithms have not been rigourously evaluated across large test collections as of yet. For further details for the problem of shot segmentation, please refer to the review papers by Boreczky & Rowe (1996), Brunelli et al. (1999), Koprinska & Carrato (2000) and recently Hanjalic (2002).

C.3 Keyframe Selection

Once the video has been segmented into shots, a widely applied technique is to extract a representative frame(s) from each shot (Figure C.1), known as keyframes. Keyframe selection is important to applications such as content browsing (Uchihashi et al. (1999), Rui et al. (1999), Yeung & Yeo (1998)), video skimming (Del Bimbo (1999)), shot grouping (Rui et al. (1999), Yeung & Yeo (1998)), genre identification (Fischer et al. (1995)) and event detection (Adams et al. (2003)). Keyframes can be used to visualise the shot content, an important technique for both indexing and browsing. Selecting keyframes that are good representations of the entire can be a form of compression for future indexing algorithms. Analysing a single keyframe(s) representation of a shot will reduce the volume of data to process to compare similarity between shots. However it is important to select strong shot representations in order to minimise information lose. Techniques that process the entire number of frames in a shot are expensive in terms of time and resources. Therefore many indexing techniques rely on a selection of keyframes from each shot that reflect its content.

The simplest solution to keyframe selection is to choose the first or last frame in a shot (Shahraray & Gibbon (1995)). The obvious advantage of this approach is its simplicity, which is also its downside. By selecting the first or last frame, the possibility of noise is high from gradual shot transitions, camera movement or even digitisation errors (Boreczky & Wilcox (1998)). For many shots, the content does not become clear until several frames into the shot, especially those that contain a fade, camera pan or zoom. The first and last frames can become blurred or contain little visual information about shot content. Selecting the middle frame instead can minimise these problems. Assuming that the overall shot content will appear towards the centre of the shot, the middle frame would appear an appropriate selection. For TRECvid experiments (Smeaton & Over (2003*a*)), the middle keyframe is selected for each shot.

This assumption will not always hold, especially when working with long shots. For example, the middle frame of a long camera pan shot from one object of interest to another, may not be the best representation of that shot. The middle frame will only include the halfway point of the sequence, therefore missing out important detail. A simplistic solution for longer shots is to extract several keyframes at specific intervals. This type of keyframe selection is applied for IBM's MPEG-7 annotation tool, VideoAnnEx (IBM (2004)). The I frames from the MPEG-1 stream are extracted as keyframes (see Appendix B).

The problem with the above approaches is that they work under the assumption that more times than not, a correct representation from each shot will be selected. An almost blind faith. An alternative to becoming a believer, is to develop intelligent solutions. Yeung & Yeo (1998) extracted a sub-sample of frames from each shot, using non-linear temporal sampling. The sample size extracted is related to the amount of motion or variation within each shot, using a shot activity feature e.g. the average colour histogram difference between successive frames. A small number of key-frames are extracted during a static shot. As the shot activity increases, so does the number of keyframes selected. The advantage of this selection process is that information such as shot activity and shot length are taken into consideration.

An alternative solution to sampling is to select the frame that is visually closest to all frames in the shot (Zhang et al. (1995)). For example, the frame that has the closest representation of the average histogram distribution for entire shot is selected. Conversely, Brunelli et al. (1999) selected keyframes at various points along a shot where there was a high degree of motion. Motion is again detected by analysing the shot activity. At these periods of increased motion, the frame difference will increase. A keyframe is selected at the local maxima of the motion period as well as the beginning and end. This will provide a complete representation of the motion sequence, addressing the problem of camera and object movement in relation to strong shot representation.

Overall it is more effective to apply intelligent techniques for keyframe selection. Keyframes can be selected that provide an optimal representation for each shot, minimising information loss as a result of the data compression. These techniques are however both time consuming and computationally expensive in comparison to the more ad hoc approaches. Many real time systems implement these naive methods such as the middle frame (Smeaton & Over (2003*a*)) or I frame (IBM (2004)) selection for speed and efficiency. Ignoring the possible trade off in poor shot representation. At present, there has been little research into the effect of this trade off, for both content browsing and video indexing.

C.4 Structure Segmentation and Classification

Once a video sequence has been parsed and representative keyframes selected from each shot. It is then possible to identify higher-level structures. The next level of video structure being the scene (Figure C.1). A scene is characterised by a collection of semantically related shots edited together within a video sequence. Shots can be related by time, location, characters, content, context, etc. Across video genre there are different representations of what a scene represents (Figure C.2). In cinematic video this would be the process of segmenting the start and end point of each scene. For news video, the start and end points for each news article would be located. In sport a scene can be a tennis rally, a single play in American football, or an entire half in soccer. These semantically related units can then be labelled through content analysis. I refer to this complete process of identifying logical units as structure segmentation and classification, although for specific domains the task has been called as scene (film) or story segmentation (news).

This process of segmenting scene structure is an important indexing step and as a result, a very active research problem. The importance of this problem is highlighted by TRECvid. In 2003 one of the tasks included a news story segmentation experiment over a large test collection (Smeaton & Over (2004)). Parsing a video file into scenes will enable the retrieval and playback of coherent blocks. Segmentation of video into logical units is useful for mapping structure. By segmenting and labelling structure a table of contents can be generated, providing the user with an overview of the entire video content. Also, when querying a news collection, independently segmented stories can be returned from a single query rather than the complete video file. An entire video file can be many hours in length. A user browsing or querying a video file(s) will not want to view the entire video file, just the relevant sequences. It would be insufficient to return the entire program when querying a specific subject.

Structural segmentation and classification algorithms are still at an immature phase in development. Currently there has been little large scale comparison of techniques except for the TRECvid 2003 news story segmentation track (Smeaton & Over (2004)). However these results are solely applicable to the domain of news. News story segmentation algorithms benefit from the availability of closed captions and speech transcriptions. The integration of audio and visual information sources can improve existing text segmentation algorithms.

For domains such as sport, automatic transcriptions or closed captions are not so readily available for a variety of reasons. As I illustrate in later chapters, the soundtrack for sport is very noisy. The speech found in sport is in conversational form rather than scripted, well spoken speech found in news. News readers by profession must have clear and concise pronunciation. Current automatic speech transcription systems decrease dramatically in accuracy when faced with difficult soundtracks such as sport. Therefore using audio and visual information has more importance in domains outwith News.

In this section, I concentrate on existing research into structure segmentation and classification using audio and visual information. These techniques can be grouped into a number of categories. These include scene segmentation algorithms based on linear searching and clustering algorithms. Segmentation algorithms using production values and structural segmentation and classification algorithms that apply statistical models.

C.4.1 Linear Search and Clustering Algorithms

These techniques work under the main assumption that semantically related shots from the same structure will contain visual similarities. As part of cinematic production visually similar shots are repeated to display aspects such as character dialogue. Both linear search and clustering algorithms compare the similarity of shots, grouping those shots that share similar characteristics. For linear search algorithms, a 'window' is introduced to determine how far ahead the algorithm should compare shots to locate similar patterns. The main concept is to compare the current shot(s) in a scene with a future number of shots. A scene transition is then located, when the following shots are not visually similar to the existing shots in the scene.

Examples of linear searching techniques include an 'expanding window' (Lin & Zhang (2000)), and the 'Likelihood Graph' (Hanjalic et al. (1997*b*)). The 'expanding window' algorithm compares the visual similarity of each new shot to existing shots in a current scene. If there is a strong resemblance then the shot is absorbed into that scene. Shot similarity is measured by comparing the correlation of the dominant colour histograms found in the scene, weighted by the frame distance between the two shots. The further apart the two shots are the less likely they are to belong to the same scene. If no new shots are absorbed at the limit of the expanding window, a scene transition is flagged. Hanjalic et al. (1997*b*) implemented an alternative approach to the 'expanding window'. A 'likelihood graph' was calculated, which contains the visual 'likeness' of neighbouring shots. To build a likelihood graph, the current shot is compared to a following number of shots. Scene transitions are represented by sharp peaks in the graph, where there is little visual similarity between neighbouring shots.

Rui et al. (1999) group shots using a two stage unsupervised linear clustering algorithm. The first stage of the clustering algorithm is to place each shot into an existing or new group, based on visual similarities. Time is used as a weighting measure to bias visually similar shots that are temporally far apart. New shots are then compared against the last shot merged into each group. The value for the most similar matching shots is calculated and measured against a group threshold. If the value is above the threshold, the shot is merged into the group. Otherwise a new group is created. The second stage of the process is to compare groups using a scene threshold. If the new group similarly measure is above the scene threshold, a scene transition is recorded.

Hierarchical clustering algorithms can also been applied to structure segmentation. Yeung & Yeo (1998) cluster shots with respect to visual and motion characteristics, where time is used as a constraint during the clustering phase. This constraint prevents visually similar shots that are temporally far apart to be grouped together. The constraint is in the form of a threshold, were temporally far apart shots have their similarity measure reset. The hierarchical clustering is stopped when the distance between the next two shots exceeds a predefined threshold. Where shot similarity is calculated by comparing the histogram difference between representative keyframes. A Scene Transition Graph (STG) is then generated to display the relationship between shots and scenes. The STG displays visually similar clustered shots as well as the interactions between clusters. Scene transition is identified when there is a temporal transition from one group of clusters to another. Veneau et al. (2000) implement the same approach, using time not as a constraint but as a weight in the similarity scores. By doing so, they can avoid experimentally define a threshold for the time constraint.

There are many problems associated with linear search and clustering algorithms. An obvious problem is the assumption of shot repetition. For the algorithms to work, it is important that the assumption of shot repetition as a production value does hold. These algorithms fail when a shot belonging to the current scene does not visually resemble the previous shots from that scene. During a scene there is a strong possibility that camera angles will not be repeated such as an establishing or cut-away shot. Without shot repetition false transitions will be marked. Subtle changes in a scene, including the introduction of a new character, camera position or a telephone conversation will also result in false transition. Scene boundaries can also be missed if two semantically independent structures contain visually similar shots. To solve these problems it is important to inject further domain knowledge into the algorithms, as well as other information sources such as audio.

C.4.2 Ad Hoc Segmentation Based on Production Clues

An alternative strategy to grouping shots based solely on visual similarity, is to utilise prior knowledge of production clues in the video generation process. For cinematic video these clues can include the appearance of a fade, dissolve and changes in background sound in the audio track. Aigrain et al. (1997) implemented a technique to segment video into scenes through a set of heuristic rules based on production values. It was assumed that each scene transition was marked by a collection of descriptors. For example, a shot fading into black would be an indicator of scene change. During shot segmentation, gradual transitions were tagged as the start and end points of each scene unit. The algorithm was only successful if these simplistic production rules were followed. Given the artistic nature of cinema, production rules such as these are often broken. Although as part of an integrated algorithm using other indicators, these clues can be an important clue for structure identification.

In specific domains such as News, heuristic rules based on production values can be employed with a higher degree of confidence. An important clue for news story segmentation is the detection of the anchor-person. The role of the anchor-person (or people) is to create a smooth transition between reports, introducing context. Thus the detection of anchor-person shots is an important indicator to the location of each news report boundary. Anchor-person shots are filmed in a studio and remain static throughout a news broadcast, using the same location and camera settings. As a result, there is a high degree of similarity between anchor-person shots.

To detect anchor-person shots, Browne et al. (2003) cluster all shots in a news programme by calculating the visual keyframe similarity. Shots are grouped if the corresponding keyframes exceed a cosine similarity threshold. Similar to the Veneau et al. (2000) shot clustering algorithm, the similarity measure is weighted by the temporal distance between shots. A set of heuristic rules are then applied to identify those clusters that represent the anchor-person group(s). To label a group as an anchor-person cluster, the cluster statistics must meet a set of predefined thresholds. These statistics are the mean temporal distance between shots, the mean visual similarity measure and the mean shot length. Anchor-person shots are spread out across the news broadcast, showing a high degree of visual similarity and span a long enough time period to introduce the next news report.

Anchor-person shots contain a person speaking directly into a camera, so a face detection sensor can also be applied in the detection process (Browne et al. (2003)). The sensor analysed keyframes for skin pixel values, outputting a probability score. A high score indicating a high degree of confidence that a shot contained a human face. A third information source that can be used for anchor-person detection is shot activity. It can be assumed that an anchor-person shot will be static, with little object or camera motion. Browne et al. (2003) measured shot activity by analysing the motion vectors extracted from the MPEG-1 stream (see Appendix B)). Across a shot, the P-frame with the lowest ratio of P-macroblocks was used to represent shot activity.

The output from all three features were then combined using a Support Vector Machine (SVM). The SVM is a non-parametric classifier that combine evidence from a number of information sources. SVM is applied to determine which shots are anchor-person shots. Once the anchor person shots are labelled, the news show is then segmented into reports. Boundaries are assumed to be the start and end points of each anchor-person shot. For example, an individual news report will span from the end of one anchor-person shot to the beginning of the next. These visual features (Browne et al. (2003)) were shown to improve segmentation accuracy over a standard text-based segmentation algorithm using automatically transcribed speech scripts. Across the large TRECvid collection, segmentation boundaries were detected with 33% recall and 49% precision. However, integrating the visual features with a text segmentation algorithm had a detrimental effect (recall 29% and precision 45%).

Hauptmann & Witbrock (1998) apply similar common production traits to segment news structure, using a combination of both audio and visual clues. These features also compliment an existing text-based segmentation algorithm using closed caption and speech transcripts. Anchor-person detection is implemented by grouping visual similar shots using colour similarity. A face similarity sensor is also applied to group shots further. News stories boundaries are then detected using the anchor-person shots as a guide. The difference between systems is that Hauptmann & Witbrock (1998) also detect the start and end point of advertisement segments. For efficient news retrieval, it is important to annotate news with non-news stories and remove advertisement sequences.

The start of each advert sequence is normally accompanied by a series of black frames and a dip in volume. Hauptmann & Witbrock (1998) implemented a sensor to detect black frames, using a threshold to determine how close a current frame is to black. By analysing the signal to noise ratio of the audio stream silence breaks are also detected. Where a threshold is applied to determine how quiet the audio track is. However, black frames accompanied by silence, are often found at the start and end points of other structures such as news reports. So to identify and label advertisement segments a set of ad hoc heuristic rules were outlined. Advert sequences are labelled if there is an occurrence of short, regular intervals of black frames and silence, over a period of time. It is assumed that each advert will be of a minimum length, accompanied by these features. On the other hand, news reports will span longer time intervals. Across 13 complete news shows, the algorithm missed 15% of news story boundaries.

The advantage of these news story segmentation algorithms (Browne et al. (2003), Hauptmann & Witbrock (1998)) is that they utilises prior information to detect structure. Unlike the clustering-based algorithms for cinematic video, assumptions based on news production values will often hold. Using clues found in show production, important structure can be mapped. The downside of these algorithms are their reliance on predefined thresholds and rules. Thresholds do not generalise well within and across news programmes. A threshold value for one news report may not often transfer across to another. For example, assuming the anchor-person shot is a rigid guide to the start and end point of each news report, can result in a lot of useful information being lost. Information such as the story introduction and context. This information is ignored alongside individual stories that are contained solely within the anchor-person shot. A possible solution to the rigid, inflexible nature of heuristic rules, is the process of statistical modelling.

C.4.3 Statistical Modelling

Clustering and linear search algorithms can discover basic structures by grouping visually similar shots (Rui et al. (2000), Yeung & Yeo (1998),Veneau et al. (2000)). These grouping algorithms however experience problems when recognising complex structures such as a scene or story unit, if the assumption of shot repetition does not hold. To improve such algorithms, it is possible to apply further heuristic rules based on production value assumptions to identify structure and label content. Through the understanding of video production, systems can utilise domain knowledge to improve accuracy. The domain of news is an example of this, where a set of ad hoc rules can be defined to map structure (Hauptmann & Witbrock (1998), Browne et al. (2003)).

The disadvantage with using a set of ad hoc rules is that often these algorithms fail due to variation found across video. To avoid the problems associated with the above techniques statistical modelling algorithms have been applied. Statistical modelling can be employed to recognise patterns found in video associated with known content. The advantage with statistical modelling is that a degree of content variation can be captured in the model. It is assumed that statistical models can recognise common patterns and structure in spite of the inevitable variations found across video (Wolf (1997)). Employing these models can be seen as an extension of defining rules based on prior knowledge of video production.

Wolf (1997) developed a very simplistic stochastic modelling framework to parse cinematic video structure into dialogue scenes, where a dialogue scene was considered to be an on-screen conversation between two or more people in a single location. First shots were classified using a face sensor into three types; close up, establishing and master shot. A detected face covering a large ratio of the current key frame was labelled as a close up shot. If the detected face(s) was smaller than this ratio, the shot was labelled as a master shot. A shot with no detected face was labelled as an establishing shot. These shots are employed to provide the viewer with some scene context such as location information. A three state HMM modelled the temporal relationship between these three shot types in relation to a dialogue scene. Each state corresponded to a pattern associated with a dialogue. For example, it is common for dialogue scenes to begin with an establishing shot, followed by close up shots of people talking, interspersed with master shots. This framework provided a simple method of recognising dialogue scenes but the accuracy of the HMM framework was not empirically evaluated.

A follow up study integrated further audio and visual information to detect dialogue scenes (Alatan et al. (2001)). Alongside the face detection sensor, visual patterns were analysed to determine a change in location. The audio stream content at each shot was also labelled as either silence, speech or music, by thresholding the audio energy. A token generater combined the output of the three sensors and then entered into a

HMM. Where the HMM segmented new video sequences into dialogue, transitional or establishing scenes. To do so, the HMM was trained on manually labelled video sequences learning the patterns associated with each scene. Combining the audio and face detection features produced the most accurate segmentation results, with the audio feature on its own indicating favourable accuracy. The overall value of using the HMM framework for segmentation and classification was the ease in which features from different modalities can be integrated to segment and label structure. The scene patterns from many video files could also be used to train the HMM.

HMM have also been applied for news story segmentation and classification. Eickeler & Muller (1999) segment broadcast news sequences into semantic units modelling patterns found in the visual and audio stream. A HMM is generated for each specific content found in a news program such as a news report, anchor-person, weather segments and interviews. Where each single HMM is a statistical representation of the feature patterns associated with a specific content class. Examples of the visual information used include frame colour distribution and motion intensity. 12 Cepstral Coefficients were also extracted from the audio stream (see Chapter 5.5) to be used as part of the feature set. Each single HMM was then combined into a unified Superior HMM framework, which segmented and classified new video sequences. Across a small data set, this approach was accurate in detecting and labelling news video structure from various TV networks, with an accuracy over 90%. Anchor-person, news reports and interview segments were labelled 99%, 96%, and 92% correctly, however correct classification of weather segments was not so accurate (80%). This was a possible indication that either the visual and audio features, or HMM was not as well tuned for identifying those segments.

As previously discussed, in the 2003 TRECvid (Smeaton & Over (2004)), a dedicated experiment focused on segmenting and labelling news from non-news stories (e.g. advertisements, weather reports). Many of those algorithms applied ad hoc rules based on common production traits for structure segmentation. In comparison to Browne et al. (2003), Chaisorn et al. (2003) applied a two layer framework for news story segmentation. Shots from the news video were first labelled into one of 12 categories, which constitute a news broadcast. After shot classification, a HMM segmented the video file into individual news reports. Similar to Eickeler & Muller (1999), the list of shot categories is comprehensive, covering anchor-person, speech, interview and weather shots, as well as shots containing financial, sport, commercial and live reporting sequences.

Unlike Eickeler & Muller (1999), instead of directly generating single HMMs that represent the audio and visual patterns for each class, Chaisorn et al. (2003) used a decision tree to first classify each shot. After shot classification, the entire shot label sequence for a news video is entered into a HMM for segmentation. The input vector for the decision tree contained 6 discrete features; the output of an audio-based neural network classifier, labelling content into speech, silence, background noise or music; shot activity; shot duration; a pixel skin-tone face sensor; camera shot type sensor similar to Wolf (1997); and a superimposed text caption sensor. Using a training collection, the decision tree classifier recognised the feature patterns associated with each shot type.

A news video file was then segmented into reports using a combination of a four state HMM and heuristics. First the HMM found the transition locations between one logical sequence and another. Input into the HMM was a two dimensional discrete feature vector, where an appropriate shot tag and a binary location change code (yes or no) was associated with each time point. Similar to the dialogue scene HMM (Alatan et al. (2001)), this news HMM models the typical patterns associated with a transition from one scene to another. After segmenting these logical units, heuristic rules are then applied to determine if scene unit is a news report or not by analysing the first and last shots of each candidate sequence. If the first shot is of an anchor-person followed by shots that are not labelled as advertisements, then a news report is tagged.

Across the 2003 TRECvid collection, Chaisorn et al. (2003) were able to detect the news report boundaries with a high degree of accuracy (73% recall and 76% precision), using both the audio and visual features. The system significantly outperformed Browne et al. (2003), where the best run was a 33% recall, and 49% precision. Overall, Chaisorn et al. (2003) was the best performing news story segmentation algorithm. It would appear that the occurrence of both audio and visual features in a framework such as HMM, improved results over heuristic approaches (Browne et al. (2003)). It would be difficult to assess if the significant gains in accuracy were a result of audio inclusion, a classification framework such as the decision tree or by the HMM segmenter. Although the authors did note that both the face and audio features were found to be most important for discrimination.

C.4.4 Discussion

From reviewing the literature, it would appear that using techniques such as shot grouping or clustering alone, are not flexible enough for the majority of video domains. Defining rules based on video production clues can improve the accuracy of these techniques, injecting a degree of prior knowledge about video generation. However these heuristic rules do not generalise well within and across domains. Recently heuristic rules are being replaced by statistical modelling of video content for structure segmentation. These frameworks are the starting point for algorithms that can begin to capture variation in video, as well as integrate features from different modalities. Integration of audio information into the modelling framework provides greater discrimination between structures (Chaisorn et al. (2003)), and in some cases is more accurate than using visual features (Alatan et al. (2001)). Statistical models such as HMM, can also provide a way of modelling the almost random and constantly changing nature of video. For example, assumptions such as visually similar shots will not always hold. Therefore a flexible decision process is required that can integrate some degree of leverage into the decision process, modelling the variation found across video.

Although there has been little direct comparison between shot grouping and statistical-based techniques, there is some evidence that they are more accurate. Currently there has been no large scale comparison of techniques except for the TRECvid story segmentation track (Smeaton & Over (2004)), solely for the domain of news. However, statistical modelling does allow for a less rigid solution to structure segmentation in comparison to defining a set of ad hoc rules. Systems that utilise statistical models are still at an immature phase in development. For example, the advantage of the system by Eickeler & Muller (1999) over Chaisorn et al. (2003) is that individual segments such as news report, weather forecast and interview can be segmented and labelled in a single pass, with a high degree of accuracy. There is no real evidence to suggest this HMM implementation is more accurate than a two stage approach, where the TRECvid experiments are the only current direct comparison between structure segmentation algorithm across a large and varied test collection. This is an area I wish to investigate in some depth as part of the thesis. In particular how to model the wealth of information found in audio appropriately, for structure segmentation and classification of sports video. Many works indicate that audio was a vital feature source, and in some cases the most important component in the system.

C.5 Genre Detection

In the manner in which current video data is archived, it is very common for video files to contain programmes from more than one genre. It is desirable to label the genre for each programme first. This classification is important for a variety of reasons such as efficient storage by category, and for indexing of high level concepts, where algorithms can be optimised using domain knowledge. As illustrated in the previous section, algorithms that map video structure will differ across domain. For example, clues in production vary across cinematic video, news and other domains. News will follow a very rigid set of rules that can be used to identify structure, while cinema will allow for relaxation of set rules. This segmentation and labelling of video programmes into appropriate categories is called genre detection (Figure C.1).

Shot length and activity are important information sources genre detection. During the editing process, stylistic features can be integrated into the video such as rhythm. By editing a number of quick, short fast shots together, the editor can build a sense of excitement. While longer, static shots can bring life to an emotional dialogue. Many advert sequences are edited together with a number of quick shots to hold the viewers interest. On the other hand, documentaries or news footage is edited together with more static shots, which convey information to the viewer avoiding distraction. All these clues are useful for identifying genre.

Vasconcelos & Lippman (2000) visualised cinematic movies in a two dimensional feature space of shot length and activity. In cinema, rhythm is an important tool to portray emotion. To build suspense or tension, long slow moving camera shots are employed. Quick short shots, with a lot of object and camera activity, are employed to portray excitement or action. Calculating the mean shot length and activity statistics from a collection of movie trailers, films were mapped in the 2-dimensional feature space. Each film was provided with manually labelled tags from one of three categories; action, romance/comedy or other. From the results, the authors discovered that the majority of trailers from each category, were found to have populated different areas in the feature subspace. This would suggest classifiers could be developed to automatically label future trailers into these simple categories. It would be interesting to find how effective this feature space technique would be to label entire films accurately, where it is likely that shot length and activity would vary over the course of a film.

Audio and visual information have also been applied to label genre. Fischer et al. (1995) use audio, visual as well as motion information alongside features such as shot length and activity. Integrating sensors that detect the occurrence of speech and music, video programmes are labelled into news, motor racing, tennis and cartoons categories following a set of ad hoc heuristic rules. These rules are based on previous knowledge of the production techniques for each genre, where a style profile of the features from each genre was created. For example, motor racing will contain high shot activity and background sound. The results illustrated a high degree of discrimination between genre when labelling new video sequences. It would be of interest however to discover how accurate this technique would be across a selection of more varied genre. The current categories appear to be hand-picked for their very different production styles. To illustrate this point, the style profiles may be too rigid to adapt to genre with less subtle differences between them such as news and documentaries, or tennis and football.

Roach & Mason (2001) also classified video files into five broad categories; sport, cartoon, news, advertisements and music videos. A statistical classifier named the Gaussian Mixture Model (GMM) was applied to model both audio and rhythm features. Test sequences of 25 seconds in length were classified, from a test collection containing 8 minutes of video sequences from each of the 5 classes. Using audio alone as an information source was shown to be effective in differentiating between Sport and News files but not so accurate in labelling sequences from the remaining three groups. Integrating rhythm information improved classification performance for cartoon and commercial groups, but results deteriorated for the remaining genres. It would be of interest to analyse how accurate applying a Hidden Markov model framework to model the temporal properties of both the dynamic and audio features would be in comparison to the GMM, where a GMM classifier can be considered a single state HMM.

Wang et al. (2000) begin to answer this question by applying a HMM to label video sequences into categories such as news, sport and weather. A HMM from each genre is generated modelling simple statistics gathered from the audio stream. This approach was able to discriminate well between sub-genre of sport (basketball and American football) but was not so accurate labelling news or weather sequences. This could be due to a number of factors such as the audio sequences being divided into small segments and then classified. These small segments might not contain sufficient enough information for accurate classification. The audio statistics extracted may not be suitable for discriminating between complex classes such as news and weather. The parameter selection for each HMM model also appeared to be ad hoc, where only a small number of hidden states were selected to model large, complex and varied genre.

The framework was then extended to include simple visual and motion features such as colour and shot activity (Wang et al. (2000)). Separate HMMs for each genre class were generated for both visual and motion features. Using the visual information alone, classification accuracy was increased for American football and advertisement sequences, but deteriorated for the remaining classes. In the case of news, dramatically, where the majority of clips were labelled as advertisements. It would appear that the dominant green pitch found in American football was important for improving discrimination between other classes. Advert sequences are also very colourful in nature, an indicator as to why many other categories were falsely labelled into this group. Using motion features alone improved the classification of news reports over the other modalities. Motion appeared to be a more accurate information source compared to visual features but not audio.

The HMMs from each modality were finally combined into a single classifying framework (Wang et al. (2000)). For each new sequence the likelihood score from the three HMMs representing the different modalities were summed together. New video sequences would then be labelled into the genre class that produced the highest overall likelihood score. By integrating the features, classification accuracy for the majority of classes was improved, however classifying weather sequences was found to be more accurate when using audio alone.

The system by Wang et al. (2000) is at a very early stage where simplistic HMMs were generated for each class using a variety of features. It is clear that by even combining the features from different modalities using a simple framework, gains in accuracy can be achieved. It would be of interest to investigate in more detail HMM model generation and feature selection for each modality separately first. Greater accuracy could well be achieved through the identification of features that contain greater discriminant information, which are modelled well by optimally selected models. By also investigating the suitability of a range of statistical models, different information sources may be better modelled by different frameworks. It is clear that audio is an important role information source for content-based indexing. For this thesis, I concentrate on this task, where audio appears to contain a high level of discriminant information that can be well modelled by statistical frameworks. These different modalities, once optimised, can then be integrated into a powerful indexing tool.

C.6 Event Detection

Event detection is the process of identifying important incidents within a video file. This process is very much domain specific, where the notion of key events will differ across genre. For example in surveillance video, recognising threatening or suspicious human behaviour is important. Alarm systems can be developed by detecting unusual behaviour such as a car driving in the wrong lane in a busy road (Porikli & Haga (2004)). On the other hand, detection of scoring sequences would be the main area of interest for sports video. The common factor across domains is the detection of key incidents that can be used to summarise video. By doing so, limiting the volume of information a user has to browse. The aim of event detection algorithms is to identify moments or incidents that are important for summarisation and highlights generation, encompassing many methods and solutions.

Event detection can also be useful for both ‘query by example’ and keyframe selection. For example, sample clips of a rocket ship taking off can be used to find other similar incidents in a large test collection (Adams et al. (2003)). During keyframe selection, frames can be selected at points where events occur, providing an intelligent highlights summary to the user (Figure C.3). In this section, I provide an overview of a number of event detection algorithms from a variety of video genre.

C.6.1 Event Detection Systems

Haering et al. (1999) extract colour, textual and motion information to detect animal hunt events in wild life documentaries. To do so, low level visual features are used

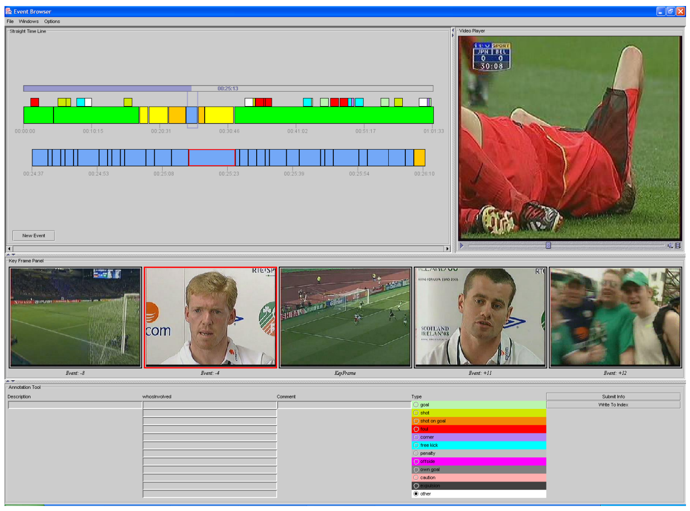


Figure C.3: Example of a keyframe browser

to classify descriptors found in a shot. A neural network classifier then determines the content found in a shot using descriptors such as object type (animal, rock or tree) and spatial information (indoors or outdoors). Motion information is also applied to detect and track moving blobs. This prior domain knowledge is gained by using wild life footage to train the neural networks. A further level of processing then determines which video sequences contain hunt events, using domain specific event inference. For hunt events domain specific event inference would be, event sequences that contain shots which exhibit smooth but fast motion. This motion would contain objects similar to an animal, followed by slower shots with little animal motion as a result of animal slowing down (e.g catching the prey). These rules are in the form of a state diagram, used to validate candidate shot sequences that might contain hunts. If the video sequence follows that of the state diagram, it is labelled as a hunt sequence. 7 wildlife hunts were used as part of a test collection, where 6 out of 7 hunts were detected with a high degree of precision.

Garg et al. (2003) use audio and visual information to recognise explosion events in film sequences, a useful indicator for action sequences. Automatic detection of these events can provide automatic highlights of film in a video on demand system. The audio and visual features for explosion events are modelled using a probabilistic framework called Multijects. A Multiject can be seen as a type of decision tree, where each node is a concept in a video sequence. Each concept is modelled by a set of low level features. For example an explosion can be recognised by a loud increase in volume, and bright yellow or red colours. Once these features are entered into the Multiject, a probability score is returned for potential concepts, where explosions are used as a simple example. Across 9 video sequences, over 85% of explosion events were correctly identified, outperforming a standard HMM. The framework is still at an early phase in development but it would be of interest to analyse how effective the Multiject framework is across a variety of more complex events, and across a more varied test collection. Also, the traditional HMM that was selected as comparison was generated in an ad hoc manner, with little focus on optimal parameter selection. It would be of interest to measure the performance of the Multiject framework in comparison with an optimally selected HMM or other statistical frameworks.

Adams et al. (2003) detect the event of Space rockets blasting off in video sequences provided by NASA[2], using both audio and visual information. The authors compared Support Vector Machine and GMM classifiers for classifying visual information, and GMM and HMM classifiers modelling audio information. Event detection was taken as a two class problem, where a specific event occurred or did not occur in a shot. Visual concepts such as the shape of the rocket object, occurrence of fire and smoke (from the engine) and the event taking place outdoors, were all used. Audio clues such as the rocket ship noise (explosion) were used to identify the event. The SVM classifier was found to be more accurate detecting visual concepts such as the Rocket shape and fire / smoke events, than the GMM. And for audio, the HMM classifier was found to be more accurate in detecting explosion sound than the GMM.

By integrating the different modalities, labelling of the Rocket launch event was improved, when compared to using classifiers from single modalities (Adams et al. (2003)). Where both SVM and a Bayesian network were compared for fusion. There was no significant difference overall between fusion strategies, with the SVM framework classifying events with 1% more accuracy, 63% of all events. When compared to using a HMM modelling the audio track, there was an improvement of 7%, with the HMM correctly classifying 56% of the events. The SVM using visual features was only 39% accurate. It is clear that on its own, using audio inside a HMM framework is a powerful tool for event detection, where integration of other modalities can significantly improve results further.

[2]This was a query by example task in TRECvid2002 (Smeaton & Over (2003*a*))

C.7 Summary

Video indexing is an important process for the automation of current labour intensive annotation methods. Recent advancements in identifying structure and content found in video automatically, has begun to help minimise the human workload for annotation. From reviewing the state of the art, it would appear that both audio information modelled statistically is a popular indexing method. Audio information is an important information source for indexing video structure that can be well modelled.

Even within genre, video can vary a lot. Defining a set of ad hoc heuristical rules were found to be too rigid for accurately identifying structure in video. Integrating prior knowledge into the indexing process has been shown to help map some structures. However, there is a need to integrate some form of leverage to counter act variance found in video. This is why statistical modelling has been successful.

Chapter 2 introduces the concept of applying domain knowledge to automatic annotation tools specific to the domain of sport.

Appendix D

Annotation

This section introduces the annotation labels that were used to generate the test collection data. Labelled samples were required for both model training and testing. Table D.1 was a high level annotation locating the boundaries between the main semantic structures. Both segment boundaries and the segment label were recorded. Table D.2 was a more in-depth annotation for the three main semantic structures Advert, Game and Studio. Sub-structures such as speech, music and silence were labelled.

Code	Content
BBC	Captured from BBC
ITV	Captured from ITV
1	Opening Titles (music only)
2	Closing Titles (music only)
3	Studio
4	Reports
5	Press Interviews
6	Musical Segues (these would normally include music and speech)
7	Studio Interviews
8	Game
9	Exciting Events
10	Stadium Interviews
11	Music Inside the Stadium
12	Advertisements

Table D.1: High-level annotation labels

Code	Class	Contents
Code	Class	Contents
A-SIL	Advert	Silence
A-S	Advert	Speech
A-MS	Advert	Music + Speech
A-M	Advert	Music
A-SE	Advert	Sound Effects
CN	Game	No Speech + Low Crowd Sounds
SN	Game	Speech + Low Crowd Sounds
SC	Game	Crowd Chanting / Singing + Speech/No-Speech
CC	Game	Crowd Cheering / Excited Speech
MS	Game	Music in Stadium
W	Game	High Pitched Sounds / Whistle
Event	Game	Reported Event
S-SIL	Studio	Silence
S-S	Studio	Speech (Male)
S-Sf	Studio	Female Speech
S-M	Studio	Music
S-MS	Studio	Music + Speech
S-SE	Studio	Sound Effects

Table D.2: Low-level annotation labels

Appendix E

TV Broadcast Channel Comparison Experiment

E.1 Introduction

A small experiment was performed before the major experiments in Chapters 6, 7, 8, 9 and 10. The experiment compared sequences from both the BBC and ITV channels to evaluate if there was any differences, particulary in the Game sequences. This experiment was to test the hypothesis that indexing algorithms could be developed that would generalise across broadcast channels. If any difference in the audio stream was found between content classes such as the Game and Studio sequences, then indexing algorithms would have to be developed separately for each channel. If the assumption was proved that there was no significant difference in recording quality between semantic classes from different channels, then the indexing algorithms could be generated independent of TV channel.

E.2 Experimental Results

A GMM modelling both the Studio and Game classes were generated for each channel. This was to test the hypothesis that there was no significant difference (except for advertisements) in audio recording between both channels, BBC and ITV. The GMM models representing each channel were generated using labelled training. Both models were then evaluated over a test collection, with labelled samples from both channels, where the maximum likelihood criterion was used for classification. Both models were measured using classification accuracy.

	BBC	ITV
BBC	**0.5188**	0.535
ITV	0.4812	**0.465**

Table E.1: TV channel comparison results

Table E.1 presents a summary of the results from the experiment in the form of a confusion matrix. The total percentage of samples classified correctly are in bold, and the remaining two entries represent the total percentage of falsely labelled samples.

Both GMM models classified approximately 50% of test samples correctly. This was evidence to suggest that there was little difference between audio recording across both TV channels. It was assumed that the GMM classifiers appeared to be modelling similar content, which resulted in low classification accuracy for both models. A similar result could be repeated by randomly assigning new test samples to one TV channel or another. This result satisfied the assumption that the similar semantic content from both TV channels could be represented by the same statistical model.

Appendix F

Content Analysis Results

This section provides extra results and findings from Chapter 7.

F.1 HMM Experiment Plots

Plots from the state selection experiments. Both figures contain extra information concerning the 15 runs for each experiment.

F.2 HMM Experiment Tables

In this section are two examples from the automatic content analysis investigation. Each table provides a summary of the content recognised automatically by the HMM for one video file. Each one second unit was assigned to a hidden state label. Then the audio track for each file was physically segmented into one seconds files that were grouped according to the state label. These audio clips were then inspected, recording the findings in a form similar to Tables F.2 and F.2.

Croatia versus Mexico (BBC1)		
State#	Classes	Summary of content found
1	Game	Crowd clapping, player shouts, incomplete commentary speech
2	Game	General crowd sounds such as chants, singing, drums, clapping, boo, whistling. Incomplete commentary speech
3	Game	Speech + crowd clapping
4	Game	Speech + crowd booing, Speech + crowd chants
5	Game	Speech + general crowd sound
6	Game	Excited Speech + clapping. Crowd Cheering + Excited speech. Events including Goal + Red Card
7	Game	General or low crowd sounds. Air horn
8	Game	National anthem
9	Game	Speech + low crowd sounds
10	Game	Incomplete speech + chants. Air horn
11	Studio, Studio Int	Speech (male and female)
12	Game	Low crowd sounds
13	Game	National Anthem + incomplete speech
14	Game	National Anthem, tannoy announcement. Speech + stadium annoucements
15	Game	Crowd clapping + low crowd sounds
16	Studio, Report	Speech, Speech + ambience sound
17	Game	Speech + crowd singing
18	Game	Stadium sounds (drums) + chants + incomplete speech
19	Studio	Speech
20	Game, Report	National anthem, Speech+Music
21	Press Int, Report	Speech +Music, Outside report plus shouting in report. Press Interview
22	Report, Game	Speech + Whistle, crowd booing + whistling, Referee whistle, Speech + music
23	Report, Music Seg'	Music
24	Report, Studio	Incomplete speech, silence, music
25	-	-

Table F.1: File: Croatia versus Mexico, BBC1

Portugal versus South Korea (ITV1)		
State#	Class	Summary of content found
1	Game	General crowd sounds, crowd chanting
2	Game	Crowd sounds, chants + incomplete speech
3	Game	Speech, tannoy announcement
4	Game	Speech + crowd chants
5	Game	Speech + general crowd sounds
6	Game	Excited speech + clapping, excited speech + crowd cheering. Events including Goals
7	Game	Speech + general crowd sounds
8	Game	General crowd sounds
9	Game, Studio	Speech, speech mixed with crowd sounds
10	Game	Speech + crowd chants
11	Studio	Speech
12	Game	Low crowd sounds (no speech or cheering)
13	-	-
14	Ads, Studio, Report	Speech. Speech + noise mixed in. Interviews (clean speech)
15	Game	National Anthems, crowd clapping + chanting
16	Studio, Studio Int, Press Int	Speech, incomplete speech plus laughter. Interview speech
17	Game Int, Studio, Report	Speech + noise, speech + game noise mixed in
18	Game	Crowd chants + clapping
19	Ads, Studio	Speech
20	Game	National anthems
21	Ads, Reports	Music + speech, Speech + sound effects
22	Ads, Game	Speech + tannoy, high pitch cheering + national anthem, crowd whistling, music, sound effects, Referee whistle
23	Ads, Reports	Music, music + sound effects
24	Ads, Studio	Music + speech, sound effects. silence, incomplete speech
25	Ads	Silence, incomplete speech

Table F.2: File: Portugal versus Korea, ITV1

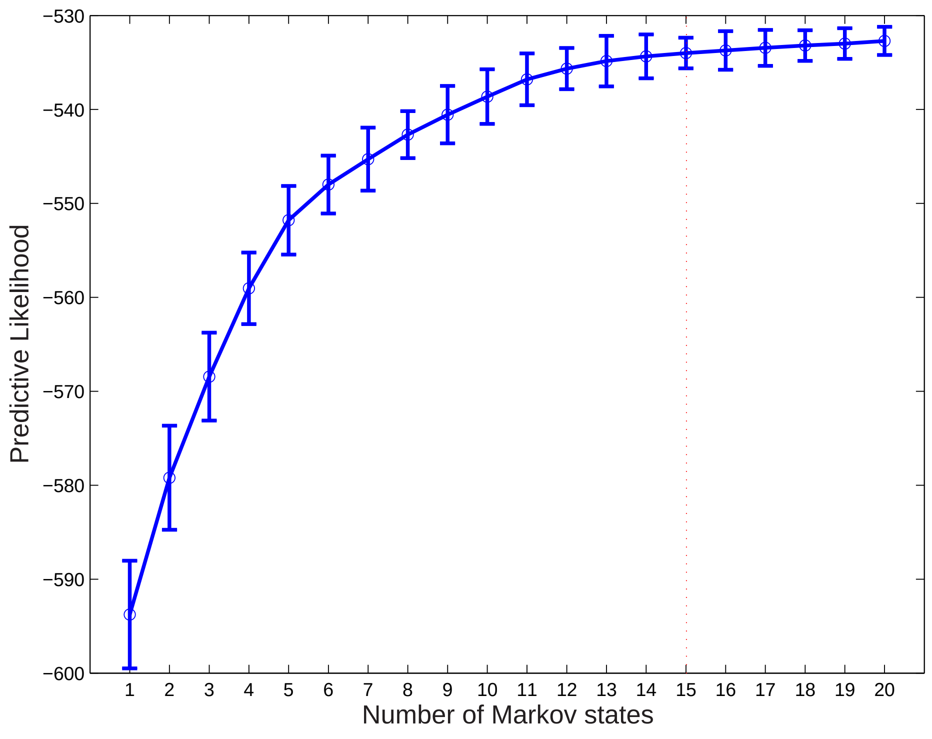


Figure F.1: Finding the optimal number of states on the synthetic data. The blue line is the mean. The red dotted line indicates the 15^{th} state added to the HMM. The error bars represent the standard deviation for each state across the 15 runs.

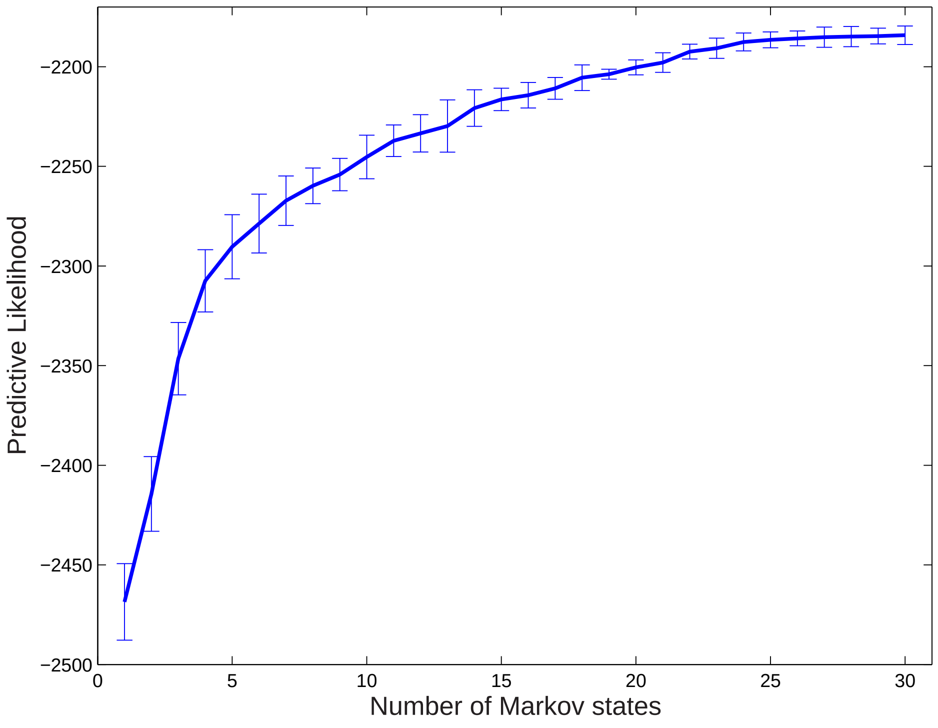


Figure F.2: Finding the optimal number of states on the real data. The blue line is the mean. The red dotted line indicates the 25^{th} state added to the HMM. The error bars represent the standard deviation for each state across the 15 runs.

Appendix G

Model Selection Extra Results

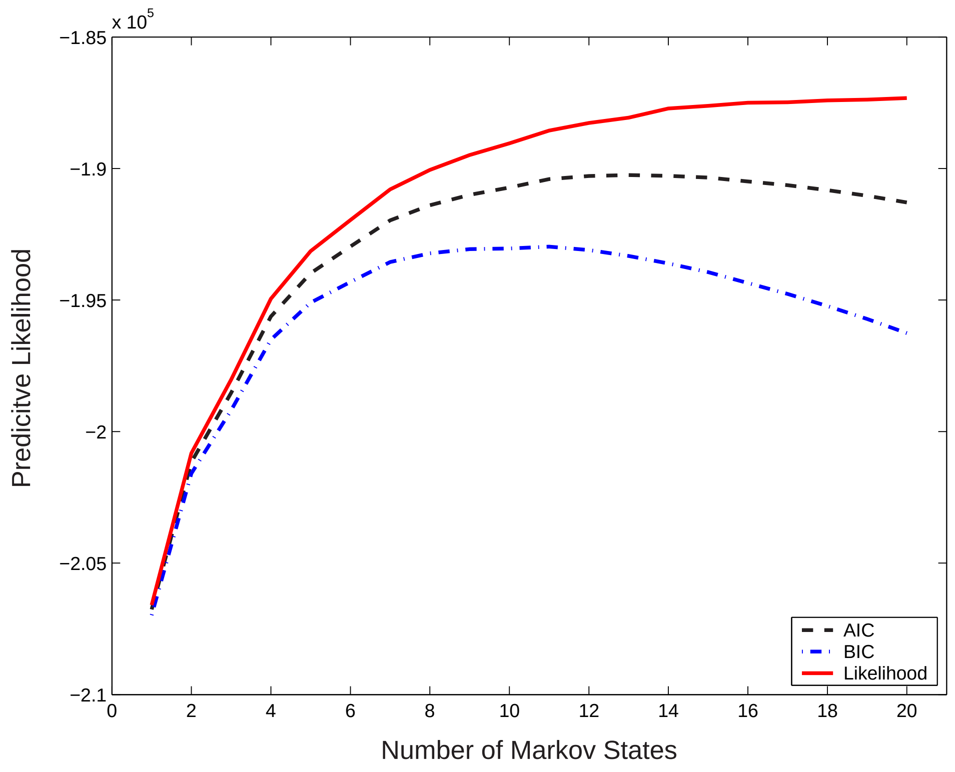


Figure G.1: For the Game class. A comparison of each strategy for hidden state selection. Notice, both the AIC and BIC scores create a peak, while the likelihood score continues to increase.

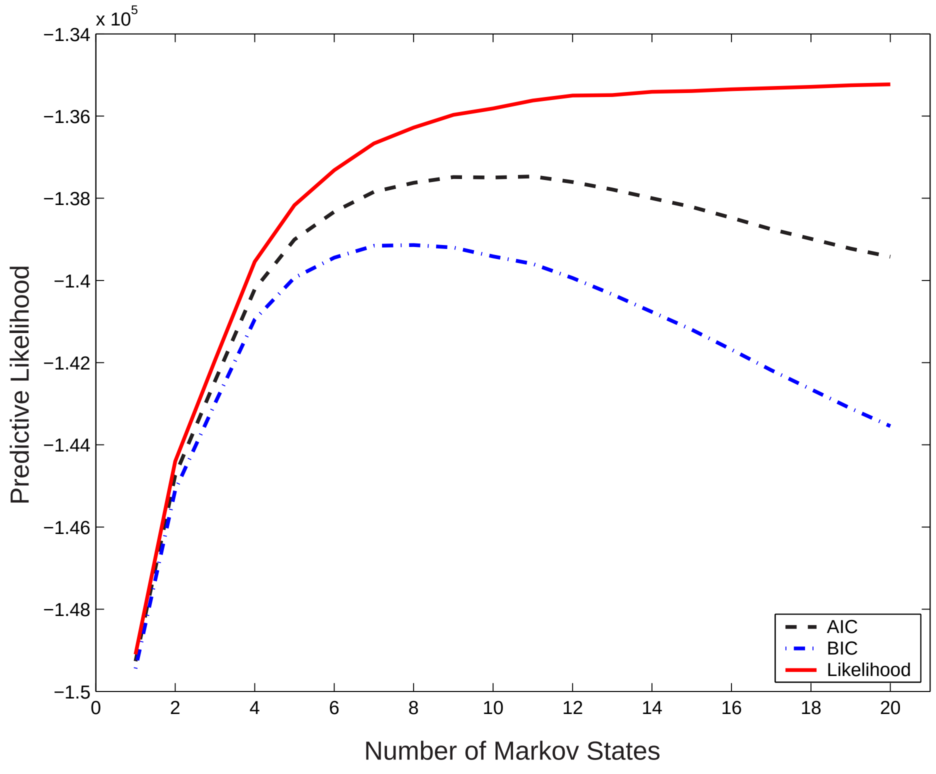


Figure G.2: A comparison of each strategy for hidden state selection for the Studio class.

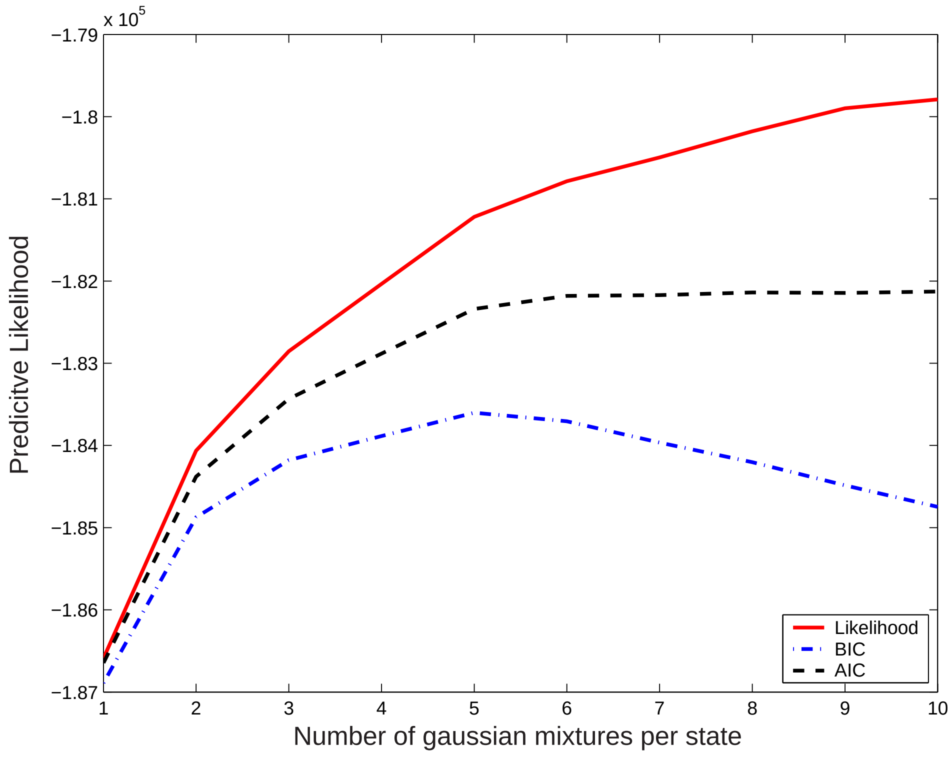


Figure G.3: Displays the three selection measures as the number of mixture components is increased, for the Advert class.

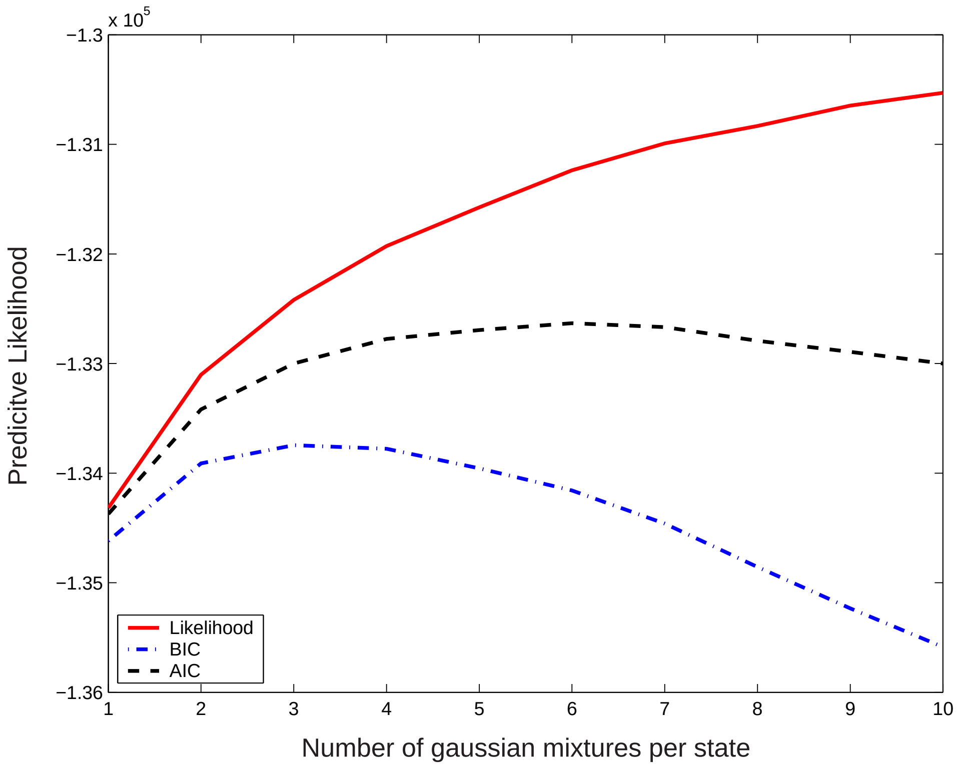


Figure G.4: Displays the three selection measures as the number of mixture components is increased for the Game class.

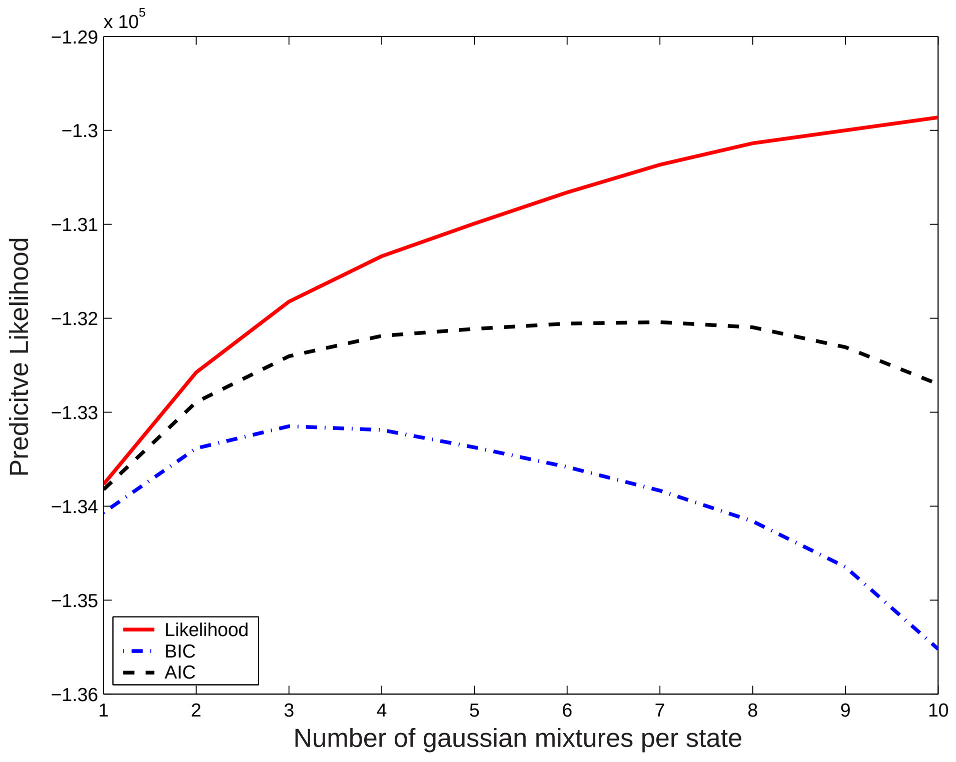


Figure G.5: Displays the three selection measures as the number of mixture components is increased, for the Studio class.

Appendix H

DP Search Algorithm

To introduce the Viterbi DP algorithm for finding the best path through a sequence of model likelihood scores, the notation by Rabiner & Juang (1993) is used.

Given an acoustic observation sequence $O = (o_1 o_2 \ldots o_T)$, where at each time point t, the observation vector o_t has been classified by a set of models $\{\lambda_i : i = \lambda_1, \ldots, \lambda_C\}$. The problem is to find the best class path $\omega = (\omega_1 \omega_2 \ldots \omega_T)$ from the sequence of returned model likelihood scores $l_j(o_t)$, where $(1 \leq j \leq C)$ and $(1 \leq t \leq T)$.

To do so, 4 variables are defined.

- $\delta_t(i)$: This is the accumulative likelihood for class i that ends at time t,
- $\psi_t(i)$: This is the class backtrack pointer for class i ending at time t,
- a_{ij}: The transition penalty for moving from class i to class j,
- $l_i(o_t)$: the model likelihood score for class i at time t.

To find the best class path, the following steps are taken.

0. Preprocessing

The first step is to estimate the parameters required in the algorithm. It is assumed that there is an equal probability that a sequence can start at any class, so for each entry in the initial probability matrix Π, will be $\frac{1}{C}$, hence these terms are omitted.

However, the following variables are required to be estimated:

- $l_i(o_t), \qquad 1 \le i \le C, 1 \le t \le T$
- $a_{ij}, \qquad 1 \le i, j \le C$

For estimating these variables, 12 video sequences set aside for system development were used. The segment changes and semantic structure were manually labelled for all of these video sequences. The likelihood scores calculated at each time point in the observation sequence $O = (o_1 o_2 \ldots o_T)$ were then calculated. The transition penalties a_{ij} were also estimated by calculating the condition probability:

$$a_{ij} = P(\omega_{t+1} = j | \omega_t = i) \tag{H.1}$$

$$a_{ij} = \frac{\text{expected number of transitions from class i to class } j}{\text{expected number of transitions from class } i} \tag{H.2}$$

This preprocessing step is only required to be preformed once and then saved for further use.

1. Initialisation

$$\begin{aligned} \delta_1(i) &= l_i(o_1), \qquad 1 \le i \le C \\ \psi_1(i) &= 0, \qquad 1 \le i \le C \end{aligned}$$

2. Recursion

$$\begin{aligned} \delta_t(j) &= \max_{1 \le i \le C} \left[\delta_{t-1}(i) a_{ij}\right] l_j o_t \\ \psi_t(j) &= \arg\max_{1 \le i \le C} \left[\delta_{t-1}(i) a_{ij}\right], \qquad 2 \le t \le T, 1 \le j \le C \end{aligned}$$

3. Termination

$$\begin{aligned} \delta^* &= \max_{1 \leq i \leq C} \left[\delta_T(i)\right] \\ \omega_t^* &= \arg\max_{1 \leq i \leq C} \left[\delta_T(i)\right] \end{aligned}$$

4. Best Class Path (Backtracking)

$$\omega_t^* = \psi_{t+1}(\omega_{t+1}^*), \qquad t = T-1, T-2, \ldots, 1$$

The DP algorithm forms a trellis that can be used to find the best class path through the observation sequence. Figure H.1 is an illustration of the DP algorithm for a three class problem, with an observation sequence of length $T = 3$. Essentially what is created by applying a DP algorithm for searching the best class path, is a first order Markov process. The transition probabilities a_{ij} define the behaviour and topology between classes. Unlike the ML decision process, the interaction between content classes is also considered during the final classification process.

To illustrate how the DP algorithm works and its assumed advantage over ML, a simple example is presented for segmenting and classifying an observation sequence.

H.1 DP Example

Given a video sequence with two possible classes, the model likelihood scores $l_i(o_t)$ were calculated,

$$L = \begin{bmatrix} 20 & 30 & 40 & 30 & 20 \\ 10 & 10 & 30 & 31 & 10 \end{bmatrix}$$

It is known that the correct path through the likelihood sequence L is $\{1,1,1,1,1\}$. However, if ML was applied to find the class path through L, the result would be $\omega = \{1,1,1,2,1\}$. So, the DP algorithm is applied instead to find the class path.

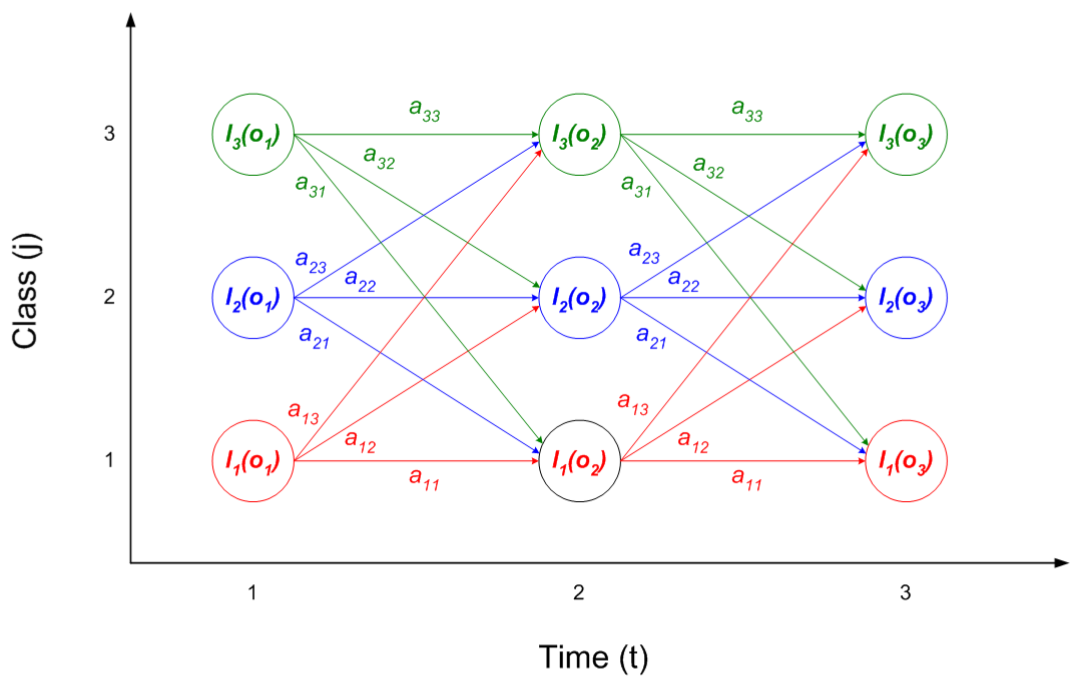


Figure H.1: DP algorithm

The first step in the DP algorithm is to calculate the transition penalties a_{ij}. This is achieved in a preprocessing step using a training sample of labelled observation sequences. Using these sequences, the equation H.2 is applied returning the results,

$$A = \begin{bmatrix} 0.8 & 0.2 \\ 0.2 & 0.8 \end{bmatrix}$$

This step is only required once, for calculating the transition probabilities in A. These probabilities are stored and used for future classification of new observation sequences.

The DP algorithm can then initialised and executed.

1. Initialisation

$$\delta_1(1) = 20, \qquad \delta_1(2) = 10$$

2. Recursion

$$\begin{array}{llll}
\delta_2(1) = 480, & \psi_2(1) = 1, & \delta_2(2) = 80, & \psi_2(2) = 2 \\
\delta_3(1) = 15360, & \psi_3(1) = 1, & \delta_3(2) = 2880, & \psi_3(2) = 1 \\
\delta_4(1) = 368640, & \psi_4(1) = 1, & \delta_4(2) = 95232, & \psi_4(2) = 1 \\
\delta_5(1) = 5898240, & \psi_5(1) = 1, & \delta_5(2) = 761856, & \psi_5(2) = 2
\end{array}$$

3. Termination

$$\delta^* = 5898240, \qquad \omega_T^* = 1$$

4. Backtracking

$$\begin{array}{l}
\omega_4^* = 1 \\
\omega_2^* = 1 \\
\omega_3^* = 1 \\
\omega_1^* = 1
\end{array}$$

The Result

The class path found by the DP algorithm was $\omega = \{1,1,1,1,1\}$, which is the correct path. By taking into account the relationship between each class given the observation sequence, the DP algorithm avoided the potential error that was not picked by ML[1].

Note: It is clear even from this example that after just a few iterations the $\delta_t(i)$ values can become huge. To prevent this problem the $\delta_t(i)$ values were normalised. To ease calculation of the parameters during implementation of the DP algorithm, the log of both a_{ij} and $l_i(o_t)$ can be taken before initialisation (Rabiner & Juang (1993)), thus avoiding multiplications during the Recursion phase. However, to avoid the problem of

[1]This is an assumed advantage of applying the DP decision process over a ML approach (Huang & Wang (2000), Xie et al. (2002)). However, an ML segmentation algorithm implementing a median filter would also correct the classification error in this example.

working with potential values of $\log(0)$, the above algorithm version was implemented, normalising $\delta_t(i)$ at each step.

Appendix I

SuperHMM DP Search Algorithm

For finding both the best state and class path sequence through the SuperHMM, the One-Pass dynamic programming search algorithm defined in Ney & Ortmanns (1999) was applied. Other, more complex, search algorithms could be employed to find the best class path (Ney & Ortmanns (2000)), however the same algorithm was applied by Huang & Wang (2000) for segmenting video into genre. For comparison purposes the algorithm described below was thus selected.

Instead of finding the best word sequence through a sequence of phones in a large vocabulary continuous speech problem, the algorithm was implemented to locate the best class path sequence for structural segmentation and classification of football audio.

The search problem can be defined as follows. The aim of the algorithm is to find the best class sequence $C = c_1, \ldots, c_L$, where there are L transitions from one class to another class. e.g. change from one semantic structure to another as defined in Figure 3.2. The aim of the search algorithm is to assign an index pair (state,class) to each observation vector o_t at time t in a football audio sequence $O = \{o_t : t = 1, \ldots, T\}$ of length T. There are C known classes found in a typical football sequence. Each class is represented by a single HMM model λ_c, where the model for class c has $N(c)$ hidden states. This problem can be viewed as finding a time alignment path of (state,class) index pairs through the sequence O such that,

$$(s_1, c_1), \ldots, (s_t, c_t), \ldots, (s_T, c_T)$$

Figure I.1 is an example of such a time alignment path, searching the connected states

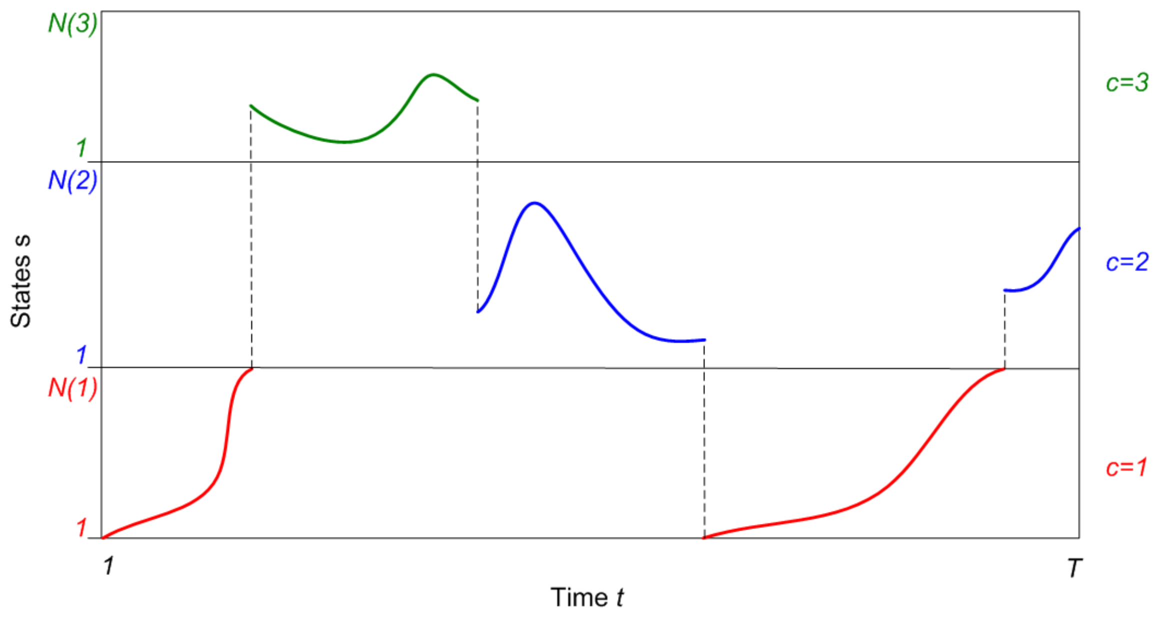


Figure I.1: Example of the search space for a SuperHMM. At each time step t, the observation sequence is assigned a state label s_t, each belonging to a class c_l.

in the SuperHMM. Within each class of the SuperHMM, the transition behaviour is that of the original HMM λ_c. e.g. the transition probabilities (see Chapter 4).

At the class boundaries, the transitions that link the terminal state S_b of any predecessor class b to the states s in any class c are followed. A dynamic programming algorithm is required to search for the optimal class path sequence. The search algorithm requires the definition of the following two quantities:

$Q_c(t,s)$	score of the best path up to time t that ends in state s of class c
$B_c(t,s)$	start time of the best path up to time t that ends in state s of class c

Figure I.1 highlights the two types of transition rules required. One rule for finding the path sequence within a class, and a rule at class boundaries. The SuperHMM algorithm uses these rules to decompose the path into two parts and formulate recurrence relations that can be solved by filling in table $Q_c(t,s)$.

The recurrence equation is then used for searching each class:

$$
\begin{aligned}
Q_c(t,s) &= \max_{1 \le s' \le N(c)} \{p_c(o_t, s|s') \cdot Q_c(t-1, s')\} && \text{(I.1)} \\
B_c(t,s) &= B_c(t-1, s_{max}(t,s;c))), 1 \le s \le N(c) && \text{(I.2)}
\end{aligned}
$$

where $N(c)$ is the number of states in class c, $s_c^{max}(t,s)$ is the optimum predecessor state for hypothesis (t,s) and predecessor class c, where

$$
s_c^{max}(t,s) = \arg\max_{s'} \{p_c(o_t, s|s') \cdot Q_c(t-1, s')\} \qquad \text{(I.3)}
$$

The back pointers $B_c(t,s)$ report the start time for each class end hypothesis. To hypothesis the change in class (e.g. class b to class c), a termination quantity ($H(c;t)$) is required, a class trace back pointer ($R(c;t)$), and a time trace back pointer ($F(c;t)$). When encountering a potential class boundary, the recombination over the predecessor classes is performed,

$$
\begin{aligned}
H(c;t) &= \max_{1 \le b \le C, b \ne c} \{p(c|b) \cdot Q_b(t, S_b)\} && \text{(I.4)} \\
R(c;t) &= \arg\max_{1 \le b \le C, b \ne c} \{p(c|b) \cdot Q_b(t, S_b)\} && \text{(I.5)} \\
F(c;t) &= B_{R(c;t)}(t, S_b) && \text{(I.6)}
\end{aligned}
$$

where

$$
S_b = \arg\max_{1 \le s \le N(c)} Q_b(t,s) \qquad \text{(I.7)}
$$

$p(c|b)$ is the class transition probability of class b to class c.

To allow for successor classes to be started, a special state $s = 0$ is introduced, passing on both the score and time index:

$$
\begin{aligned}
Q_c(t-1, s=0) &= H(c; t-1) && \text{(I.8)} \\
B_c(t-1, s=0) &= t-1 && \text{(I.9)}
\end{aligned}
$$

This equation assumes that first the normal states $s = 1, \ldots, S_c$ are evaluated for each class c before the start up states $s = 0$ are evaluated.

The algorithm starts at $t = 1$ and proceeds sequentially until T is reached. When T is reached the optimum class sequence C_l^* and class transition T_l^* are found by tracing back $R(c;t)$ and $F(c;t)$. This can be achieved by the following equation,

$$C_L^* = \arg\max_c Q_c(T, S_c) \tag{I.10}$$

where

$$S_c = \arg\max_s Q_c(T, s) \tag{I.11}$$

$$\begin{aligned} T_L^* &= F(T, C_L^*) && \text{(I.12)} \\ C_l^* &= R(T_{l+1}^*, C_{l+1}^*), l = L-1, \dots, 1 && \text{(I.13)} \\ T_l^* &= F(T_{l+1}^*, C_{l+1}^*), l = L-1, \dots, 1 && \text{(I.14)} \end{aligned}$$

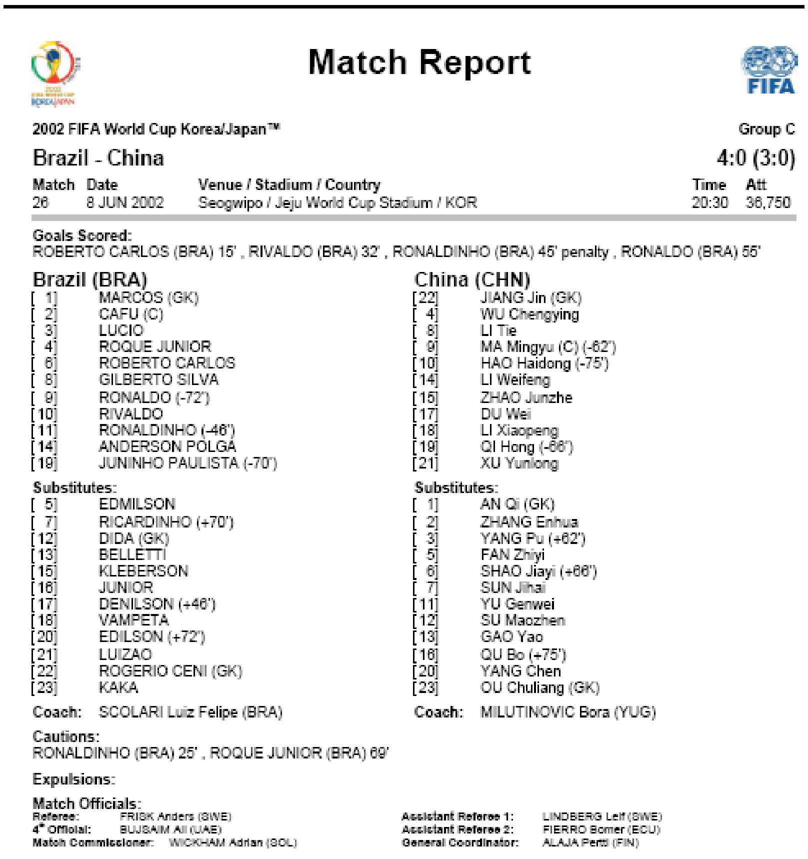
Table I summarises the main actions in the search algorithm.

Proceed over t from left to right			
	State Level: process (class,state)-hypothesis		
		-initialisation	$Q_c(t-1, s=0) = H(c;t-1)$ $B_c(t-1, s=0) = t-1$
		-time alignment -propagate back pointers	$Q_c(t,s)$ using DP search $B_c(t,s)$
		-prune unlike hypothesis -purge book keeping lists	
	Class Pair Level: process class end hypothesis		
		for each class c do	
			$H(c;t) = \max_{1 \le b \le C, b \ne c}\{p(c\|b) \cdot Q_b(t, S_b)\}$ $R(c;t) = \arg\max_{1 \le b \le C, b \ne c}\{p(c\|b) \cdot Q_b(t, S_b)\}$
			store best predecessor $R(c;t)$ store best boundary $F(c;t) = B_{R(c;t)}(t, S_b)$

Figure I.1 is an illustration of the time alignment algorithm through a SuperHMM, returning the ‘optimal’ class sequence. For further details about this DP search algorithm please refer to Ney & Ortmanns (1999).

Appendix J

Example of a FIFA Match Report



Match Report

2002 FIFA World Cup Korea/Japan™ — Group C

Brazil - China — **4:0 (3:0)**

Match	Date	Venue / Stadium / Country	Time	Att
26	8 JUN 2002	Seogwipo / Jeju World Cup Stadium / KOR	20:30	36,750

Goals Scored:
ROBERTO CARLOS (BRA) 15' , RIVALDO (BRA) 32' , RONALDINHO (BRA) 45' penalty , RONALDO (BRA) 55'

Brazil (BRA)		China (CHN)	
[1]	MARCOS (GK)	[22]	JIANG Jin (GK)
[2]	CAFU (C)	[4]	WU Chengying
[3]	LUCIO	[8]	LI Tie
[4]	ROQUE JUNIOR	[9]	MA Mingyu (C) (-62')
[6]	ROBERTO CARLOS	[10]	HAO Haidong (-75')
[8]	GILBERTO SILVA	[14]	LI Weifeng
[9]	RONALDO (-72')	[15]	ZHAO Junzhe
[10]	RIVALDO	[17]	DU Wei
[11]	RONALDINHO (-46')	[18]	LI Xiaopeng
[14]	ANDERSON POLGA	[19]	QI Hong (-66')
[19]	JUNINHO PAULISTA (-70')	[21]	XU Yunlong
Substitutes:		**Substitutes:**	
[5]	EDMILSON	[1]	AN Qi (GK)
[7]	RICARDINHO (+70')	[2]	ZHANG Enhua
[12]	DIDA (GK)	[3]	YANG Pu (+62')
[13]	BELLETTI	[5]	FAN Zhiyi
[15]	KLEBERSON	[6]	SHAO Jiayi (+66')
[16]	JUNIOR	[7]	SUN Jihai
[17]	DENILSON (+46')	[11]	YU Genwei
[18]	VAMPETA	[12]	SU Maozhen
[20]	EDILSON (+72')	[13]	GAO Yao
[21]	LUIZAO	[16]	QU Bo (+75')
[22]	ROGERIO CENI (GK)	[20]	YANG Chen
[23]	KAKA	[23]	OU Chuliang (GK)
Coach:	SCOLARI Luiz Felipe (BRA)	**Coach:**	MILUTINOVIC Bora (YUG)

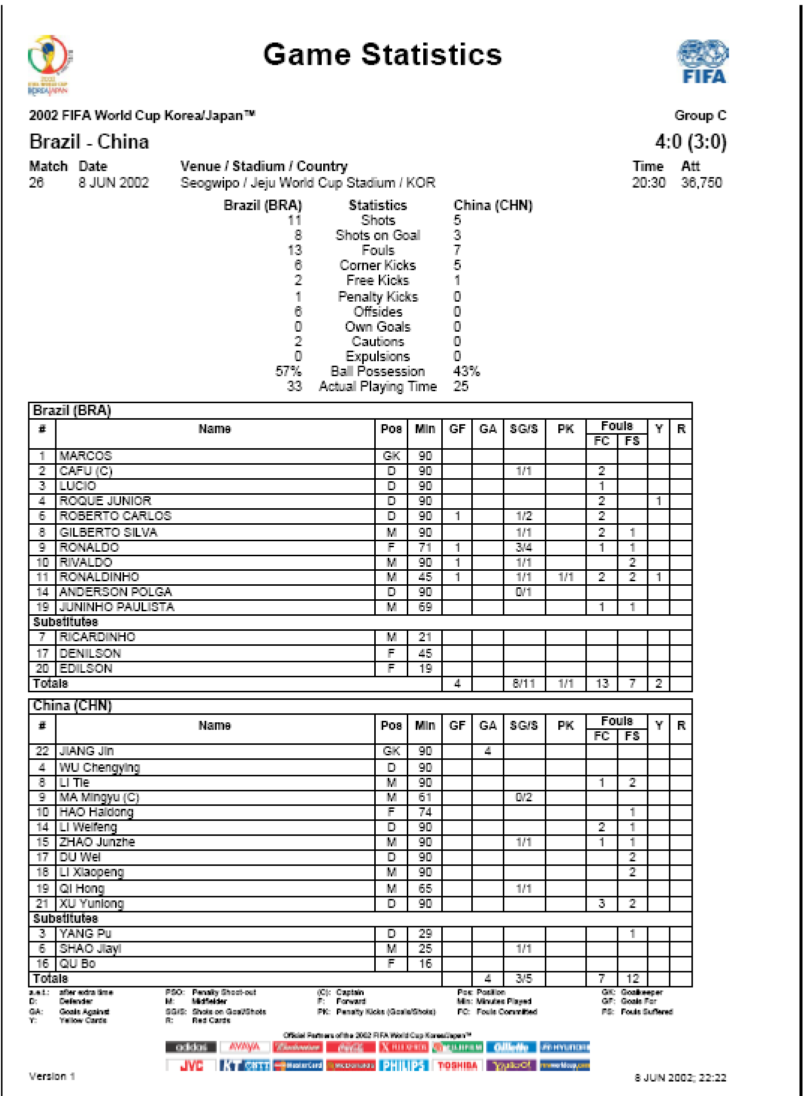
Cautions:
RONALDINHO (BRA) 25' , ROQUE JUNIOR (BRA) 69'

Expulsions:

Match Officials:

Referee:	FRISK Anders (SWE)	**Assistant Referee 1:**	LINDBERG Leif (SWE)
4th Official:	BUJSAIM Ali (UAE)	**Assistant Referee 2:**	FIERRO Bomer (ECU)
Match Commissioner:	WICKHAM Adrian (SOL)	**General Coordinator:**	ALAJA Pertti (FIN)

Figure J.1: An example of an official match report, page1.



Game Statistics

2002 FIFA World Cup Korea/Japan™ **Group C**

Brazil - China 4:0 (3:0)

Match	Date	Venue / Stadium / Country	Time	Att
26	8 JUN 2002	Seogwipo / Jeju World Cup Stadium / KOR	20:30	36,750

Brazil (BRA)	Statistics	China (CHN)
11	Shots	5
8	Shots on Goal	3
13	Fouls	7
6	Corner Kicks	5
2	Free Kicks	1
1	Penalty Kicks	0
6	Offsides	0
0	Own Goals	0
2	Cautions	0
0	Expulsions	0
57%	Ball Possession	43%
33	Actual Playing Time	25

Brazil (BRA)

#	Name	Pos	Min	GF	GA	SG/S	PK	Fouls FC	Fouls FS	Y	R
1	MARCOS	GK	90								
2	CAFU (C)	D	90			1/1		2			
3	LUCIO	D	90					1			
4	ROQUE JUNIOR	D	90					2		1	
6	ROBERTO CARLOS	D	90	1		1/2		2			
8	GILBERTO SILVA	M	90			1/1		2	1		
9	RONALDO	F	71	1		3/4		1	1		
10	RIVALDO	M	90	1		1/1			2		
11	RONALDINHO	M	45	1		1/1	1/1	2	2	1	
14	ANDERSON POLGA	D	90			0/1					
19	JUNINHO PAULISTA	M	69					1	1		
Substitutes											
7	RICARDINHO	M	21								
17	DENILSON	F	45								
20	EDILSON	F	19								
Totals				4		8/11	1/1	13	7	2	

China (CHN)

#	Name	Pos	Min	GF	GA	SG/S	PK	Fouls FC	Fouls FS	Y	R
22	JIANG Jin	GK	90		4						
4	WU Chengying	D	90								
8	LI Tie	M	90					1	2		
9	MA Mingyu (C)	M	61			0/2					
10	HAO Haidong	F	74						1		
14	LI Weifeng	D	90					2	1		
15	ZHAO Junzhe	M	90			1/1		1	1		
17	DU Wei	D	90						2		
18	LI Xiaopeng	M	90						2		
19	QI Hong	M	65			1/1					
21	XU Yunlong	D	90					3	2		
Substitutes											
3	YANG Pu	D	29						1		
6	SHAO Jiayi	M	25			1/1					
16	QU Bo	F	16								
Totals					4	3/5		7	12		

a.e.t.:	after extra time	PSO:	Penalty Shoot-out	(C):	Captain	Pos:	Position	GK:	Goalkeeper
D:	Defender	M:	Midfielder	F:	Forward	Min:	Minutes Played	GF:	Goals For
GA:	Goals Against	SG/S:	Shots on Goal/Shots	PK:	Penalty Kicks (Goals/Shots)	FC:	Fouls Committed	FS:	Fouls Suffered
Y:	Yellow Cards	R:	Red Cards						

Official Partners of the 2002 FIFA World Cup Korea/Japan™

Version 1

8 JUN 2002; 22:22

Figure J.2: An example of an official match report, page2.

Appendix K

A Sports Video Browsing and Annotation System

K.1 Introduction

In this chapter, an application is outlined that utilises the indexing algorithms described in this thesis. The system is an ongoing project now at the prototype stage, where at present, there has been no system evaluation. Two MSc students, Andrew Grimble and Craig Hutchinson have helped with the coding and ongoing development of the system. For a full explanation of the current implementation please refer to Grimble (2004).

The algorithm can be used as a browsing tool, viewing the games in the collection for entertainment, and also for analysis. The annotation functionality was also integrated, to further index the collection for future studies.

K.2 Off-line Indexing

The system integrates a number of indexing algorithms for the purpose of browsing football video. Football video files are indexed off-line applying 5 algorithms:

1. A standard shot segmentation algorithm using histogram differences was used,

for parsing each video file into shots.

2. For each shot, the middle key was extracted[1].

3. The audio track for each video file was then analysed. First the structure segmentation and classification algorithm using BICseg and a set of GMM models was applied (see Chapter 9 for more details). This process grouped each shot into the correct content class, such as Advert, Game and Studio.

4. Key events in the segmented Game sequences, were then detected using the BICseg and HMM algorithm, described in Chapter 10.

5. After event detection, 5 keyframes surrounding the location of each key event were extracted. One, at the exact time point of the event, and two, either side. This was to provide a visual summary of each event.

The results of each algorithm were then integrated into an XML index file for the video. The keyframes extracted from the video were stored in the same directory as the XML index file and the video. When a video file was opened, by the Browsing and Annotation system, the XML file was used to load the indexing information and keyframes into the system. The tool also contained an annotation facility for both editing and updating the XML index file. Further details such as event detection type, player names and shot content, could be included in the index file. Thus allowing for a more complete description of the video, by allowing human intervention.

K.3 The System

The system is divided into 4 panels. The top left panel is the browsing pane, the top right panel is the video player, the middle panel is the keyframe summary, and the bottom panel is for manual annotation. The system has two browsing modes; a standard linear timeline browsing functionality (Figure K.1), and a new circular designed event browser (Figure K.3). The browsing panel can be switched between the timeline and circular browser, from the menu bar.

The test collection is stored on a server that the application links to. A user can open

[1]For further details on both shot segmentation and keyframe selection, see Appendix C.

any indexed football file stored on the server using the menu bar. Once a video file is selected, the XML index and the keyframes are also loaded into the system. The viewer can then browse the content of the video file using either browsing functionality. It is believed that both browsers, and the keyframe summary panel, provide an in-depth overview of the video content to the user.

The three main functionalities in the system are now discussed, the two browsing panels and the annotation system.

K.3.1 Timeline Browser

The timeline browser is a variation on standard video browsers. There are four main components in this browser: two timelines, the event labels, and a progress bar (see Figure K.2). Both timeline allows the user to scan the content of each video shot by moving the mouse over either timeline. By doing so, a keyframe summary of the current shot and the two neighbouring shots, either side of the focused shot, are updated accordingly. The keyframe summary provides an overview of the current video location.

The variation on other implementations of the timeline browser, is the two timeline layers, designed to minimise clutter when presenting information. The top layer is a complete overview of the video, displaying a table of contents for the video. This timeline integrates information from both the *structure segmentation and classification*, and *event detection* algorithms. For example, to assist browsing, the different segments are colour coded in correspondence with the semantic structure each segment was labelled as. In Figure K.2, the green segments correspond to the Game sequences, the yellow segments are Studio sequences, orange labels are Advert sequences, and the blue labels correspond to a Press interview in the football video. These labels supply the viewer with a guide to the structure of the video, and allow for locating specific areas of interest. To illustrate this point, Advert sequences can be skipped by clicking directly after an orange label.

The bottom timeline (layer 2), zooms in on the current section that is being examined. For example, in Figure K.2 is currently browsing over the 25th minute of the video. In the bottom timeline, the section of video from the 24th to 26th minute is focused upon,

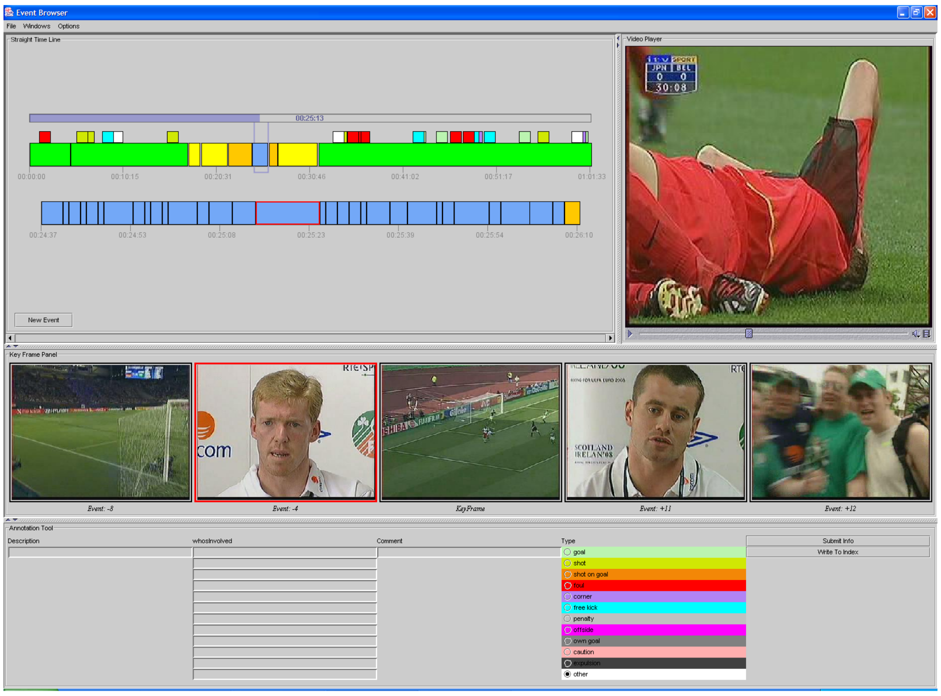


Figure K.1: 2-layered linear timeline browser

in more detail. In this timeline, the shot information is also displayed, using the shot segmentation output. Moving the mouse across each shot will update the keyframe summary in the panel below. By clicking on this, timeline or on the keyframe panel, will start the video at that exact location.

Also displayed, above the top timeline, is the location of all events, represented by the square labels. Again, by moving over these labels, the keyframe summary is updated to summarise the content of the event. Clicking on either the event label, or keyframe summary, will start the video at the event location.

Finally, above the top timeline, is a progress bar that displays the current position of the video file.

K.3.2 Circular Event Browser

The circular event browser, Figure K.3, has similar functionality to the timeline browser, but concentrates solely on presenting the event detection results to the user. By doing so, allows the user to browse the key events identified in the football game automatically. An event can be browsed by moving the mouse over an event label, which is one

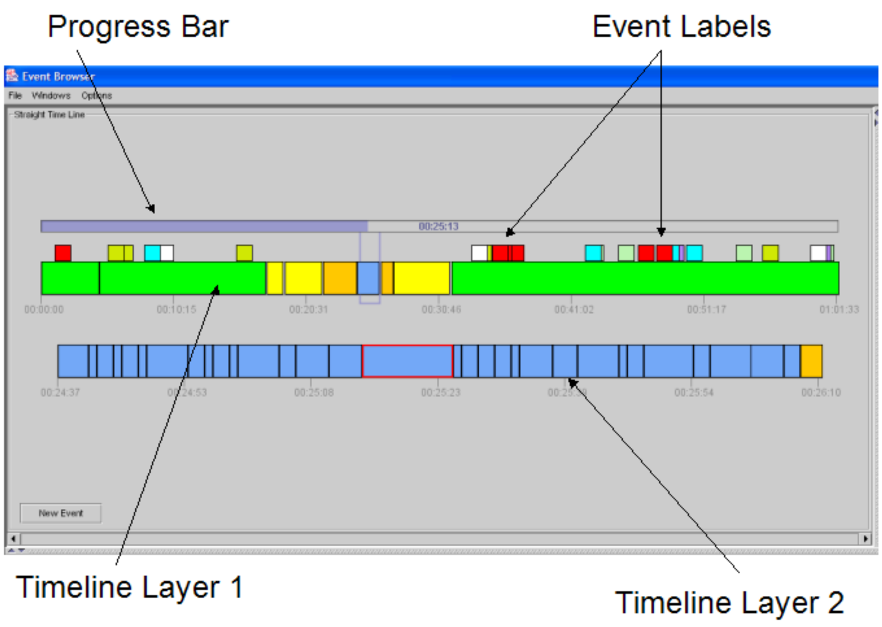


Figure K.2: The 4 components in the linear timeline browser

of the square labels in the circular panel, in the top left corner of Figure K.3. Moving the mouse over one of these square labels updates a visual summary of the event, in the keyframe panel.

The circular timeline was designed to minimise clutter in the browser, and present a complete overview of the football match to the user. The different circular layers in the browser correspond to the importance of the event. For example, the inner layer corresponds to important events such as goals. However, the outer layer corresponds to less important events such as fouls.

Clicking on either an event label or a keyframe, will start the video at that exact point, allowing the user to jump to specific points of interest.

K.3.3 Annotation Functionality

There is also an annotation functionality in the browser that allows the user to update the XML index file. This is found at the bottom of the system in Figures K.1 and K.3. The annotation facility allows events to be updated or corrected. For example, an event type list is provided for classifying each event. The list is a comprehensive guide to

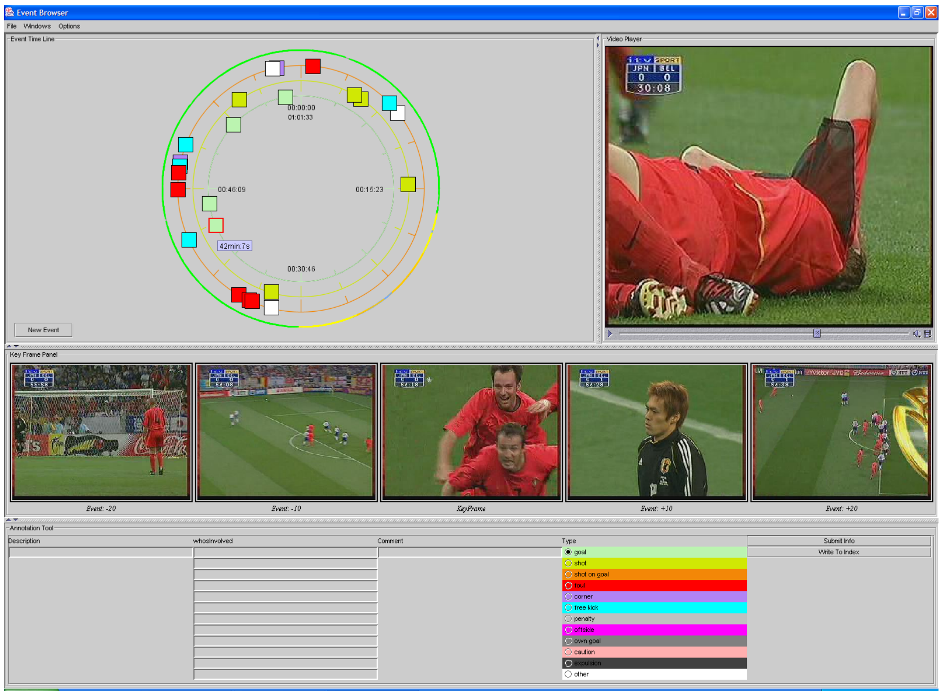


Figure K.3: The circular event browser

possible events. An event is classified by selecting an option on this list. Further details can also be entered such as player names, team names and other descriptive information about the event. Falsely detected events can also be removed from the index file by using this facility. It was believed that this tool could be useful to assist user browsing but also be applied for annotating the test collection in more depth for future analysis and algorithm development.