Robust Video Content Analysis via Transductive Learning Methods

Dissertation

zur Erlangung des Doktorgrades
der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt dem
Fachbereich Mathematik und Informatik
der Philipps-Universität Marburg

von

Ralph Ewerth

aus Hanau

Marburg/Lahn, 2008

Erstgutachter:	Prof. Dr. Bernd Freisleben
Zweitgutachter:	Prof. Dr. Otthein Herzog

Tag der mündlichen Prüfung:

FÜR MEINE MUTTER

TO MY MOTHER

ABSTRACT

Several technological innovations, such as increased hard disk capacities, improved network bandwidth and mobile multimedia devices, have fostered an enormous increase of multimedia data in recent years. The growing amount of multimedia data raises the question of how to efficiently index, summarize and retrieve multimedia content. Up to now, the necessary automatic understanding of multimedia content is an unsolved problem in practice. In addition, the variability of multimedia sources and content is enormous, and obviously this is also true for video databases.

The research question addressed by this thesis is how to build robust video content analysis and indexing approaches that work reliably on arbitrary videos. Many video content analysis approaches are considered to be "robust" by their inventors. However, in most cases this means that an algorithm or system has proven to work well on one or more (hopefully large) test sets. Furthermore, often a single classification model or decision threshold is applied to all test videos in the same way, which might be a learned model using machine learning techniques or a set of pre-defined parameters that have been estimated empirically. Obviously, this is a problem as long as we do not restrict video databases in some respect, since videos can vary in many ways: in terms of recording devices, the recording circumstances, the used compression technology, editing layout, genre and, of course, in terms of content. Hence, there is a need for the development of algorithms that work reliably and independently of the factors mentioned above – a robust video content analysis and indexing algorithm should automatically adapt to a particular video with respect to the video content, editing layout and so on, and its indexing quality should not depend on compression artefacts that are present in a given video.

This thesis investigates how a high-quality analysis and indexing result can be obtained or improved for a particular given video by considering the context of content and compression appropriately. One of the major contributions of this thesis is to consider the analysis process for a particular video as a setting that is well suited for transductive learning. Transductive learning is not aimed at obtaining a general classification function for all possible test data points (as in inductive learning) but at obtaining a specific classification for the given test data only. The idea is that the desired classification function has to be optimal for the unlabeled test data only and not in general (as in the case of inductive learning). In this thesis, this idea is applied to achieve robust video content analysis: the unlabeled data of a particular, previously unseen video are incorporated into the learning and classification process. For this purpose, a self-/semi-supervised learning ensemble framework is presented that exploits an initial classification (or clustering) result to improve its quality for a particular video. The proposed framework is based on feature selection and ensemble classification; it is called *self-supervised* when the baseline approach relies on unsupervised learning, and it is called *semi-supervised* when the baseline approach relies on supervised learning. Within the scope of this thesis, solutions for several video content analysis and video indexing problems are presented. Apart from the solutions that are based on the proposed learning framework, some proposals in this thesis employ unsupervised learning or deal with compression artefacts adequately. Overall, the following tasks are considered: shot boundary detection, estimation of camera motion, face recognition, semantic concept detection and semantic indexing of computer game sequences. Several strategies are investigated to utilize the transductive setting in order to obtain better results for different video content analysis tasks: 1.) dealing with compression artefacts (video cut detection, camera motion estimation); 2.) estimating parameters automatically (video cut detection); 3.) applying self-supervised learning (video cut detection, face recognition/clustering); 4.) applying semi-supervised learning (semantic video retrieval, semantic video indexing); 5.) applying transductive support vector machines (SVM) (semantic video retrieval). Experimental results on large test sets (which are publicly available in most cases) demonstrate the very good performance of the proposed approaches. In particular, the ensemble version of the proposed framework works reliably for all considered video content analysis tasks, in contrast to the realization of the framework using a single self-/semi-supervised classifier and in contrast to the transductive SVM. Finally, the thesis is concluded with a summary of the contributions and some areas of future work are outlined.

ZUSAMMENFASSUNG

In den letzten Jahren ist die Menge der Multimediadaten im Bereich der Computeranwendungen und im Internet stark gewachsen. Dies ist eine Folge verschiedener technologischer Entwicklungen, die zu größeren Festplattenkapazitäten, besseren Netzwerkbandbreiten und effizienteren Kompressionsmethoden für Multimediadaten führten. Nicht zuletzt hat die Verbreitung von mobilen Geräten mit multimedialen Funktionen zugenommen (z. B. Mobiltelefone, digitale Kameras), diese Geräte können ihrerseits Multimediadaten generieren bzw. empfangen und versenden. Mit der stetig anwachsenden Menge multimedialer Daten wächst allerdings auch die Notwendigkeit, solche Daten anhand des Inhalts effizient zu indexieren, zusammenzufassen und durchsuchen zu können. Jedoch ist das hierzu notwendige rechnergestützte automatische Verstehen von beliebigen multimedialen Inhalten nach wie vor ein ungelöstes Problem, bedingt durch die große Variationsbreite von multimedialen Inhalten und Quellen. Die Variationsmöglichkeiten sind auch im Falle von Videoaufnahmen vielfältig; dies gilt sowohl für im Internet zum Download bereitgestellte Videos als auch für Aufnahmen in Film und Fernsehen. Videos können unterschiedlichen Genres angehören, die Aufnahme- und Kompressionsqualität kann sehr unterschiedlich sein, und nicht zuletzt können beliebige und sehr unterschiedliche Inhalte mittels eines Videos präsentiert werden. Dies führt unmittelbar zu der Fragestellung, ob und wie Algorithmen zur automatischen Videoanalyse und Videoindexierung entwickelt werden können, so dass ihre Annotationsqualität bezüglich eines einzelnen Videos mit beliebigem Inhalt bestmöglich ist. Viele in der Literatur vorgeschlagene Systeme werden von den jeweiligen Autoren als "robust" bezeichnet. Jedoch bedeutet dies in den meisten Fällen lediglich, dass ein solches System auf einer ausreichend großen Testmenge von Videos ein gutes Ergebnis erzielt hat. In der Regel wird ein mittels maschinellem Lernen erstelltes Klassifikationsmodell bzw. ein empirisch gefundener Schwellenwert für jedes Video einer solchen Testmenge in der gleichen Weise angewendet. Offensichtlich kann dies nicht immer zu bestmöglichen Ergebnissen führen, wenn etwa die analysierten Videos nicht die gleichen Eigenschaften teilen oder zum Beispiel nicht aus dem gleichen Genre stammen. Dies zeigt die Notwendigkeit für Algorithmen, die zuverlässig für jedes beliebige Video funktionieren, unabhängig davon, welchem Genre ein Video angehört oder wie es komprimiert wurde etc. – ein tatsächlich robuster Algorithmus zur Videoanalyse sollte sich automatisch an die jeweiligen Charakteristika eines Videos in Bezug auf Inhalt, Editing-Artefakte, Kompression usw. anpassen.

In dieser Dissertation wird untersucht, wie ein hochwertiges initiales Analyse- bzw. Indexierungsergebnis für ein Video beliebigen Inhalts durch die Berücksichtigung des Kontexts von Inhalt und Kompression erreicht oder verbessert werden kann. Eine der maßgeblichen Ideen dieser Arbeit ist, den Analyseprozess eines Videos als ein transduktives Lernszenario aufzufassen: Transduktives Lernen zielt auf die Erstellung eines Lern- und Klassifikationsmodell ab, das die gegebenen Testdaten – und nicht alle theoretisch möglichen Testdaten - bestmöglich klassifiziert. Im Gegensatz zum induktiven Lernen geht es also nicht darum, ein allgemein gültiges Lernmodell zu erstellen. In dieser Dissertation wird die Idee des transduktiven Lernens für die robuste inhaltliche Analyse von Videos verwendet: die Daten und Merkmale des zu analysierenden Videos werden in den Lern- und Klassifikationsprozess mit einbezogen. Zu diesem Zweck wird ein neues Framework mit transduktiven Klassifikatoren vorgestellt. Ein initiales Klassifikationsergebnis wird genutzt, um die Güte des Analyseergebnisses für ein Video zu verbessern. Das vorgeschlagene Framework basiert auf Methoden zur Merkmalsselektion und Ensembleklassifikation. Es wird *selbst-überwacht* genannt, wenn das initiale Analyseergebnis auf einem Ansatz des unüberwachten Lernens basiert; sofern das initiale Analyseergebnis auf einem Ansatz des überwachten Lernens basiert, wird das Framework *semi-überwacht* genannt. Im Rahmen dieser Arbeit werden für verschiedene Probleme der inhaltlichen Videoanalyse und der Videoindexierung Lösungen präsentiert, die nicht ausschließlich auf dem vorgeschlagenen Framework basieren, sondern in einzelnen Fällen auch Methoden des unüberwachten Lernens nutzen bzw. Kompressionsartefakte angemessen berücksichtigen. Die behandelten Problemfelder sind: Schnitterkennung, Bestimmung von Kamerabewegung, Gesichtserkennung, semantische Konzeptdetektion und semantische Indexierung von Computerspielsequenzen. Verschiedene Strategien werden verfolgt, um Analyseergebnisse eines jeweiligen Ansatzes in einem transduktiven Szenario zu verbessern: 1.) Berücksichtigung von Kompressionsartefakten (Schnitterkennung, Bestimmung von Kamerabewegung); 2.) automatische Parameterschätzung (Schnitterkennung); 3.) Anwendung des selbst-überwachten Lern-Frameworks (Schnitterkennung, Gesichtserkennung/-gruppierung); 4.) Anwendung des semi-überwachten Lern-Frameworks (semantisches Video Retrieval, semantische Videoindexierung); 5.) Anwendung von transduktiven Support Vector Machines (semantisches Video Retrieval). Die experimentellen Ergebnisse auf umfangreichen, öffentlich verfügbaren Testmengen von Videos demonstrieren, dass die vorgeschlagenen Ansätze für "robuste" inhaltliche Videoanalyse und –indexierung sehr gut funktionieren und die Annotationsqualität tatsächlich verbessern. Insbesondere wird gezeigt, dass die Ensemblevariante des Frameworks für alle betrachten Analyseaufgaben robust funktioniert, im Gegensatz zur Verwendung von einzelnen transduktiven Klassifikatoren. Im letzten Kapitel werden die Beiträge dieser Dissertation abschließend zusammengefasst und diskutiert, des Weiteren werden zukünftige Forschungsfelder skizziert.

Danksagung

Nachdem nun das Ende meiner Promotionsphase sehr nahe gerückt ist, möchte ich die Gelegenheit nutzen, mich bei den Menschen zu bedanken, die mich in beträchtlicher Weise unterstützt haben, im Wesentlichen während meiner Zeit als Doktorand.

Zunächst gilt mein besonderer Dank meinem Doktorvater Bernd Freisleben. Von Beginn an hat er mich inspiriert und motiviert, meine Forschungsvorhaben voranzutreiben und mich dabei stets in fachlicher und organisatorischer Hinsicht unterstützt. Inzwischen kann ich auch sagen, dass er dies bis zur letzten Minute getan hat. Seinen Stil als Leiter unseres Forschungsprojekts empfand ich als sehr angenehm, von Beginn hatte ich sehr viel Freiheit zum selbständigen Arbeiten, und alle Ideen waren willkommen. Andererseits hatte er jederzeit ein offenes Ohr für meine Anliegen, auch für jene, die nicht direkt mit dem Promotionsvorhaben zu tun hatten. Ich kann mir tatsächlich nur schwerlich einen besseren Doktorvater vorstellen.

Herzlich möchte ich mich bei Professor Otthein Herzog für seine rasche Zusage bedanken, diese Dissertation zu begutachten und als Prüfer zur Verfügung zu stehen.

An dieser Stelle sei noch drei Professoren des Fachbereichs Mathematik und Informatik gedankt. Sowohl Manfred Sommer als auch Wolfgang Hesse haben mich während meiner Diplomphase erheblich unterstützt, hierfür möchte ich mich nachträglich bedanken. Manfred Sommer habe ich auch den Kontakt zu Bernd Freisleben und hiermit zum Forschungsprojekt „Methoden und Werkzeuge für die rechnergestützte medienwissenschaftliche Analyse“ zu verdanken. Des Weiteren möchte ich Eyke Hüllermeier danken, der mir in einigen fachlichen Diskussionen wertvolle Hinweise und Anregungen gegeben hat und stets offen für meine Fragen war.

Bedanken möchte ich mich bei der Deutschen Forschungsgemeinschaft (DFG) für die finanzielle Unterstützung meines Promotionsvorhabens im Rahmen des obengenannten Projekts, das Teil des DFG-Sonderforschungsbereichs „Medienumbrüche“ ist (SFB/FK615, u. a. Universität Siegen und Philipps-Universität Marburg).

Mit meinen Kolleginnen und Kollegen im Projekt haben sich zahlreiche Kooperationen ergeben. Mein herzlicher Dank gilt zunächst Julinda Gllavata für unsere Zusammenarbeit in den ersten beiden Projektjahren. Für die ebenfalls sehr angenehmen und zahlreichen Kooperationen sei Markus Mühling besonders gedankt, der mich insbesondere in der Endphase meiner Doktorarbeit des Öfteren entlastet hat. Weiterhin gilt mein Dank für Kooperationen und diverse fachliche Diskussionen Thilo Stadelmann, Christian Schridde, Matthew Smith und Thomas Friese.

Danken möchte ich auch allen ehemaligen und derzeitigen Mitgliedern meiner Arbeitsgruppe für das angenehme Arbeitsklima. Besonderer Dank geht hier an Michael Engel, Ralph Jansen, Thomas Friese, Matthew Smith und Mechthild Keßler für die Unterstützung verschiedenster Art, etwa wenn es um Server, Netzwerk und Co oder um organisatorische Dinge ging. Vielen Dank auch an die 13h-Mensa-Fraktion für enspannte Mittagspausen, neben bereits erwähnten Mitstreitern denke ich hier vor allem an Eniana Mustafaraj, Julinda Gllavata und Ermir Qeli. Für die äußerst unterhaltsamen Gänge zum Getränkeautomaten danke ich Christian Schridde und Niels Fallenbeck. Erstgenanntem gilt noch mein besonderer Dank für die fürsorgliche ergonomische Feinjustierung meines Monitors und meines Bürostuhls. Niels gebührt noch Dank für die Durchsicht einiger Teile meiner Dissertation, ebenso wie Markus und Thilo.

Viele Projektarbeiten haben tatkräftige Unterstützung durch eine Reihe von sehr engagierten Studenten bzw. studentischen Hilfskräften erfahren und wurden teilweise nur durch diese möglich. Zunächst sei hier als „softwaretechnischer Ideengeber“ für die Software Videana Mathias Grube genannt, der gerade in der Anfangszeit des Projekts sehr viele eigene Ideen in die Entwicklung von Videana eingebracht hat. Danken möchte ich auch Martin Schwalb, der sowohl als studentische Hilfskraft im Projekt als auch in seinen Studienarbeiten hervorragende Arbeit geleistet hat. Dies hat sich auch in einer beachtlichen Zahl gemeinsamer Publikationen niedergeschlagen. Auch Christian Beringer, Tobias Kopp und Michael Niebergall haben sich durch unermüdlichen Einsatz und sehr gute Arbeit hervorgetan, ebenso wie Björn Agel und Dominik Seiler (Leider war uns der „Titel“ in tamilischer Handschrifterkennung nicht gegönnt). Auch die gegenwärtige „Generation“ studentischer Hilfskräfte in Person von Bing Shi und Christian Zöfel leistet ausgezeichnete Arbeit, so dass ich feststellen kann, dass wir immer viel Glück mit der Besetzung dieser studentischen Stellen gehabt haben. Insbesondere die Teilnahmen an den TRECVID Evaluationen in den Jahren 2005-2007 wären ohne den unermüdlichen Einsatz von Christian Beringer, Tobias Kopp, Björn Agel, Dominik Seiler, Bing Shi und Christian Zöfel nicht ohne Weiteres möglich gewesen. Vielen Dank nochmals!

All meinen Freunden sei gedankt für die Unterstützung, die ich durch sie während dieser Zeit erfahren konnte. An erster Stelle sei Thomas Hornisch genannt, der mir in den vergangenen Jahren ein sehr guter und verlässlicher Freund wurde. Ihm sei auch gedankt für die Ratschläge in jedweder Hinsicht meines Lebens, auch wenn ich diesen zu selten folge. Birte Jule Diehl alias Julie danke ich für all die lustigen Mensaessen auf den Lahnbergen und viele weitere lustige Stunden. Es freut mich auch sehr, dass mir Michael Rink als langjähriger Freund, nunmehr fast ein Viertel Jahrhundert, erhalten geblieben ist, ebenso wie Claudia Appel, und bei beiden möchte ich mich für die verlässliche Unterstützung über eine so lange Zeit hinweg bedanken. Ziemlich nahe dran an den

Höhen und Tiefen meiner Doktorandenzeit waren natürlich auch meine Mitbewohner in der Liebigstraße, die etwaige Stimmungsschwankungen meinerseits locker wegsteckten und deren moralischer Unterstützung ich mir immer sicher sein konnte: ein großes Dankeschön an Bastian Blum, Katja Langenbrinck, Rahel Jahre und Winnie Marcus. Einige meiner Freunde teilten wiederum mein Doktorandenschicksal, und zwar in geisteswissenschaftlichen Disziplinen, und so gab es, nicht nur in dieser Hinsicht, viele Gemeinsamkeiten zum Austausch: Zunächst sei Eva Platten gedankt, meiner langjährigen Zug-„Bekanntschaft"; des Weiteren Hinna Wolf, mit der ich schon während meiner Diplomprüfungsphase viele Feierabendgetränke teilte; zudem Katrin Scheffer, meiner Beinahemitbewohnerin, und Dr. Sabine "Hillary" Manke, die mich auf den letzten Promotionsmetern dann doch souverän überholte.

In den letzten 15 Jahren hat Franz Appel eine Menge Rollen in meinem Leben übernommen. Angefangen als vermeintlicher Schwiegervater in spe, dann als Chef während meines Studenten-„Neben"-Jobs bis hin zu einem väterlichen Freund und Ratgeber, hat er mich in all den Jahren mit Rat und Tat unterstützt und mich durch seine erfrischend geradlinig kontroverse Sicht auf allerlei Dinge, sei es Zwischenmenschliches, Berufliches oder Politisches, sehr bereichert.

Ganz besonders möchte ich meinen Brüdern Helmuth und Wolfgang danken, die mich seit jeher unterstützen, sei es in praktischer oder moralischer Hinsicht gewesen.

Danken möchte ich meinem Vater für alles Positive, das er mir mitgeben konnte.

Aus tiefstem Herzen danke ich meiner Mutter, deren Liebe und Gottvertrauen, und vor allem deren unglaubliche Güte ich niemals vergessen werde. Sie war mir jederzeit ein Vorbild, etwa wie es ihr in mitunter schwierigeren Zeiten bei der Erziehung von drei Kindern gelang, immer Ruhe auszustrahlen und nie den Humor zu verlieren, nach ihren Überzeugungen zu leben und nicht nur von ihnen zu sprechen. Es ist mir nur schwer möglich, meiner Bewunderung und meinem Respekt für ihre Lebensleistung mit Worten Ausdruck zu verleihen. Wie gerne hätte ich den Abschluss meiner Promotion mit Dir erlebt und gefeiert.

Und sicher nicht zuletzt gilt mein besonderer Dank Antje: für die unglaublich liebevolle Unterstützung, die sie mir während meiner gesamten Doktorandenzeit zuteil werden ließ, die immer für mich da war, auch wenn es manchmal nicht selbstverständlich war, für die unermessliche Geduld und nicht zuletzt für die große Portion Humor bei der Beobachtung meines Promotionsvorhabens.

CONTENTS

Abstract vii
Zusammenfassung ix
Danksagung xiii
1 Introduction 1
1.1 Motivation 1
1.2 Problem Statement 4
1.2.1 Robust Video Content Analysis of Compressed Video Data 5
1.2.2 Robust and Adaptive Shot Boundary Detection 5
1.2.3 Robust Estimation of Arbitrary Camera Motion in MPEG Videos 6
1.2.4 Robust Face Recognition and Indexing in Video 6
1.2.5 Robust and Adaptive Detection of Semantic Concepts in Videos 7
1.3 Contributions 7
1.3.1 Transductive Learning Methods for Robust Video Content Analysis 7
1.3.2 Robust and Adaptive Shot Boundary Detection in Videos 9
1.3.3 Estimation of Arbitrary Camera Motion in MPEG Videos 10
1.3.4 Self-Supervised Learning of Face Appearances in Videos and TV Casts 10
1.3.5 Adapting Concept Models to a Video via Semi-Supervised Learning 10
1.3.6 Semantic Analysis of Computer Games for Psychological Research 11
1.3.7 Robust Video Content Analysis in the Compressed Domain 12
1.4 Publications 12
1.5 Organization of this Thesis 15
2 Background 17
2.1 Introduction 17
2.2 Content-Based Video Retrieval 17
2.2.1 Different Levels of Multimedia Understanding 17
2.2.2 Sensory Gap and Semantic Gap 18
2.2.3 Image and Video Frame Features 18
2.2.4 Structure of Videos and Movies 18
2.2.5 The Process of Content-Based Video Retrieval 19
2.2.6 Performance Measures for Video Indexing and Retrieval 26
2.2.7 Compressed Video Domain 28
2.3 Scientific Movie Analysis 32
2.3.1 Shots 32
2.3.2 Cuts 32
2.3.3 Gradual Shot Transitions 32
2.3.4 Shot Sizes and Camera Distance 33
2.3.5 Camera Movement 34
2.3.6 Shot Protocols and Sequence Protocols 35
2.3.7 Visualizations for Movie Analysis 35
2.3.8 Videana: A Software Tool for Scientific Film Studies 36
2.4 Used Machine Learning and Optimization Methods 37
2.4.1 Unsupervised Learning 38
2.4.2 Supervised Learning 45
2.4.3 Semi-Supervised Learning 47
2.4.4 Transductive Learning 50
2.4.5 Meta-Learning Methods and Ensemble Learning 51
2.4.6 Optimization: Downhill-Simplex Algorithm 56
2.5 Summary 57

3 Transductive Learning Ensembles for Robust Video Content Analysis 59
3.1 Introduction 59
3.2 Motivation 59
3.3 Framework 60
3.3.1 Feature Selection 62
3.3.2 Ensemble Classification 63
3.4 Specific Instances of this Framework 64
3.5 Summary 64
4 Robust and Adaptive Shot Boundary Detection 65
4.1 Introduction 65
4.2 Related Work 66
4.2.1 Surveys on Shot Boundary Detection 66
4.2.2 Approaches for Shot Boundary Detection 68
4.2.3 Performance Analysis 76
4.2.4 Summary 78
4.3 Performance Comparison Study 79
4.3.1 Introduction 79
4.3.2 The Evaluated Algorithms 80
4.3.3 Cut Detection Experiments 82
4.3.4 Gradual Transition Detection Experiments 88
4.3.5 Summary 89
4.4 Video Cut Detection and Compression Artifacts 91
4.4.1 Introduction 91
4.4.2 Problem Analysis 92
4.4.3 FDN: Frame Difference Normalization 95
4.4.4 GFDN: GoP-Oriented Frame Difference Normalization 96
4.4.5 Experimental Results 97
4.4.6 Summary 99
4.5 Unsupervised Shot Boundary Detection 101
4.5.1 Introduction 101
4.5.2 Video Cut Detection without Thresholds 103
4.5.3 Unsupervised Gradual Transition Detection 105
4.5.4 Experimental Results: Cut Detection 109
4.5.5 Experimental Results: Gradual Transition Detection 112
4.5.6 Summary 118
4.6 Video Cut Detection Using an Ensemble of Classifiers 119
4.6.1 Introduction 119
4.6.2 Proposed Ensemble of Classifiers for Video Cut Detection 119
4.6.3 Experimental Results 121
4.6.4 Summary 122
4.7 Self-Supervised Learning for Robust Shot Boundary Detection 125
4.7.1 Introduction 125
4.7.2 Related Work 125
4.7.3 Self-Supervised Learning for Robust Video Cut Detection 127
4.7.4 Experimental Results 128
4.7.5 Summary 131
4.8 Automatic Performance Prediction for Video Cut Detection 133
4.8.1 Introduction 133
4.8.2 Related Work 133
4.8.3 Performance Prediction for Unsupervised Video Cut Detection 134
4.8.4 Experimental Results 134
4.8.5 Summary 136

4.9 Summary of Chapter 4 138

5 Robust Camera Motion Estimation in MPEG Videos 139

5.1 Introduction 139

5.2 Related Work 140

5.3 Estimating Camera Motion in MPEG Videos 141

5.4 Experimental Results 144

5.5 Summary 146

6 Self-Supervised Learning of Face Appearances in TV Casts and Movies 147

6.1 Introduction 147

6.2 Related Work 148

6.3 Self-Supervised Learning of Face Appearances in TV Casts and Movies 150

6.3.1 Face Detection 151

6.3.2 Face Sequence Generation 151

6.3.3 Verification of Skin Color Probability 153

6.3.4 Aggregating Face Sequences in Dialogs 154

6.3.5 Bunch Graph Matching 155

6.3.6 Correction of In-Plane Rotation 156

6.3.7 Gabor Wavelet Feature Extraction 157

6.3.8 Face Recogntion/Clustering 158

6.3.9 Self-Supervised Learning for Robust Face Clustering 160

6.4 Experimental Results 161

6.5 Summary 170

7 Semi-Supervised Learning for Semantic Video Retrieval 171

7.1 Introduction 171

7.2 Related Work 172

7.3 Semi-Supervised Learning for Semantic Video Retrieval 175

7.3.1 Learning the Initial Concept Model 176

7.3.2 Adapting a Concept Model to a Video 178

7.3.3 Experimental Results 180

7.4 Transductive SVM for Semantic Video Retrieval 187

7.4.1 Applying Transductive SVMs for Video Indexing and Retrieval 188

7.4.2 Experimental Results 190

7.5 Summary 193

8 Semantic Video Analysis for Psychological Research by Semi-Supervised Learning 199

8.1 Introduction 199

8.2 Related Work 200

8.3 Semantic Analysis of Computer Games by Semi-Supervised Learning 201

8.3.1 Semantic Classes for the Computer Game Experiment 202

8.3.2 Extraction of Audio Features 203

8.3.3 Extraction of Visual Features 205

8.3.4 Semantic Classification 207

8.4 Experimental Results 211

8.5 Summary 214

9 Conclusions 215

9.1 Summary 215

9.1.1 Transductive Learning Methods for Robust Video Content Analysis 216

9.1.2 Un- /Self-supervised Learning for Robust Shot Boundary Detection 218

9.1.3 Estimation of Arbitrary Camera Motion in MPEG Videos 219

9.1.4 Self-Supervised Learning for Person Recognition in Video 219

9.1.5 Semi-Supervised Learning for Semantic Video Retrieval 219

9.1.6 Semi-Supervised Learning for Semantic Annotation of Computer Games 220

9.1.7 Robust Video Indexing in the MPEG Compressed Domain 220

9.1.8 Summary of Contributions 221
9.2 Future Work 223
9.2.1 Transductive Learning Ensemble for Robust Video Content Analysis 223
9.2.2 Shot Boundary Detection in Videos 224
9.2.3 Estimation of Arbitrary Camera Motion in Videos 224
9.2.4 Person Recognition in Video 224
9.2.5 Detection of High-Level concepts 225
9.2.6 Semantic Indexing of Computer Games 225
Annex A. List of Figures 227
Annex B. List of Tables 231
Annex C. References 239
Annex D. Lebenslauf (CV) 259

Chapter One

1 INTRODUCTION

1.1 MOTIVATION

In recent years, several technological innovations have fostered an enormous increase of multimedia data: increased hard disk capacities, processor power, network bandwidth, the improvement of audio, image and video compression technologies and digital photo and video cameras. The growing amount of multimedia data is not restricted to content providers, because consumer products generate a significant amount of multimedia content, too. Along with the increase of multimedia data, several possible applications have emerged. In the field of music entertainment, the popularity of MP3-encoded music files and the possibilities to download music files via the Internet generated new requests to automatically organize and search music collections and databases. In the field of image retrieval, people wish to organize their digital music or photo collection in an intelligent way; for example they want to find all the pictures showing a certain person or a certain place, without the need for manual annotations. Commercial use cases of large image and photo collections also increase the demands for search and browsing facilities in such systems. The same is true for video databases. Smith et al. [143] discuss the business potential for the creation, distribution and management of media content, for consumers and for media enterprises. Television (TV) broadcasters produce storage-intensive video content, which raises issues such as how to efficiently index, summarize and retrieve video content. Markkula and Sormunen [108] investigate the TV program making process and find that produced and unproduced video is needed at different stages of a journalist's workflow. If video shots of interest could be easily found in a large database, the production of new material could be alleviated or even partially automated. In addition, the proliferation of video content and podcasts has emerged in the web (e.g., video.google.com; www.youtube.com), and the need for efficient video search and retrieval facilities is growing accordingly. In 2007, several Internet-TV applications have evolved, which are either provided by large existing communication service providers like Arcor (<http://www.arcor.de>) and Deutsche Telekom (<http://www.telekom.de>), or new start-up enterprises like Joost (<http://www.joost.com>) or the Global internetTV Portal (<http://global-itv.com>).

During the last years, the emergence of related media products and research events reflects the acknowledged importance of searching and retrieving multimedia content. One of the main research events that directly focuses on video retrieval is the TREC Video Retrieval Evaluation (TRECVID) workshop series. TRECVID emerged from the well-known TREC (Text Retrieval

Conference) series that is aimed at encouraging research in information retrieval from large text collections. TRECVID started in 2001 as a task of TREC and became an independent workshop in 2003. Recently, research efforts started in the context of the TRECVID conference series (<http://www-nlpir.nist.gov/projects/trecvid>) to investigate how to efficiently explore and exploit rushes material (raw video material, captured for production) for subsequent production purposes. Since 2002, the International Conference on Image and Video Retrieval focuses explicitly on the topic of image and video retrieval and has become an ACM conference in 2007.

This thesis focuses on the processing of video material to support efficient search in video databases. Most of the research efforts in the field of content-based image and video retrieval started in the beginning of the nineties of the last century (for example, see surveys for image search engines [59], for image and video retrieval [7, 190], and for multi-modal video indexing [146]). Many different disciplines contribute to this research field. Closely related fields are, for example, those of image and video processing and compression, signal processing, computer vision, pattern recognition, cognitive sciences, artificial intelligence and machine learning, multimedia databases and information retrieval. In the field of content based video retrieval, the terms video content analysis, video indexing, and video retrieval are distinguished. The process of video content analysis aims at recognizing objects and events in a video while the closely related process of video indexing aims at supplementing video shots and scenes with automatically extracted meta-information which support a user's search for specific content in a video database. Often, the terms video content analysis and video indexing are used interchangeably. The retrieval process answers a specific user query and returns documents that are considered as relevant. Since video content analysis is a time-consuming process, it is conducted prior to any user query, that is the retrieval process is based on the results of a preceding analysis and indexing stage. The algorithms proposed in this thesis fall into the category of video content analysis and indexing. The proposal for semi-supervised learning of high-level features (semantic concepts) in chapter 6 is also considered as an indexing approach though its quality is tested in an appropriate retrieval scenario.

Many algorithms have been proposed to structure video documents or to recognize certain objects and events in videos. The temporal segmentation of a video into particular shots is considered as a fundamental step. Several proposals exist for shot boundary detection. The detection of faces is another fundamental task since the semantics of a video is normally related to the appearance of humans. Many specialized detectors were suggested for objects and events in video, for example text detection and recognition, people detection, face detection and recognition, camera and object motion detection, object detection (cars), setting detection (indoor/outdoor, urban, vegetation, mountain, forest, beach etc.), audio segmentation into segments of speech, music and silence, speaker and speech recognition. Once such information has been extracted from video documents and metadata have been stored appropriately (e.g., using a metadata standard such as MPEG-7 [114]), they may serve for subsequent queries. It should be clear now that the time-consuming process of content analysis should be accomplished before a user query is formulated. Then, the retrieval system will evaluate the video documents in the database by matching the query and the metadata and return those shots that are considered as most relevant to the user. The quality of the system answer clearly depends on the quality of the underlying content analysis process and on the ability to match the query with concepts available in the metadata. In addition, the quality of content detectors may vary depending on the variability of content, the related genres and compression. This issue of robustness is one of the the main motivations of this thesis.

Apart from the generic application scenario mentioned above, automated video content analysis and indexing can also support scientific tasks. In media science, formal analysis of movies plays an important role in scientific movie analysis. Korte [88] describes systematic movie analysis (in German: “Systematische Filmanalyse”) as an approach to investigate the presentation of movies, the contextual conditions of their production in relation to the social and technological environment and the possibilities of movie perception. In some respects, this process aims at objecting the movie sensation which, however, is not possible in general. Syntactic movie analysis aims at collecting objective facts about a movie. Important aspects are the kind of shot composition, the types of transitions that were used to connect shots, cut (shot boundary) frequency, the use of camera movement, and the presentation of actors, i. e. which shot sizes were applied (close-up shot, wide shot, etc). These cinematic elements are important clues for the analysis process. Furthermore, scene and sequence boundaries are identified and protocols are created with semantic descriptions of shot and scene content.

The work presented in this thesis is also motivated by the project “Methods and Tools for Computer-Assisted Scientific Media Research” (MT), located at the Universities of Marburg and Siegen. It is part of the research center “Media Upheavals” (SFB/FK615) which consists of thirteen research projects, funded by the German Research Foundation (Deutsche Forschungsgemeinschaft); some of them conduct scientific film analysis. The research center aims to investigate the foundations and the structural aspects of the comprehensive media upheavals and their impact to media culture and media aesthetics at the beginning of the twentieth century (introduction of films and cinema) and in the transition to the 21st century (Internet and WWW). The goal of the MT project is to provide a high-performance video content analysis system to support other subprojects applying film analysis. The software workbench Mediana is under development to provide such support [38, 104]. In addition to database tools, Mediana includes the video content analysis tool Videana. Several content analysis algorithms are integrated into Videana, including cut and shot boundary detection [36, 37, 40, 43], text detection [60, 61, 62, 64, 66], text tracking [63], text segmentation (for "video OCR" [65, 67], camera motion estimation [39], face detection and face recognition [44, 47], and speaker recognition [152]. Although the processing power of today's computers is enormous, the processing time of such algorithms and the data-intensive multimedia file organization are still challenging issues. Therefore, a first prototype for distributed execution using Grid Services has been developed for the task of video cut detection [41], distributed versions of other algorithms are currently under development.

The digitized videos that are analyzed in the different projects span a very wide range from the beginning of film creation (in the period from 1895 until 1914) up to today's hybrid forms of computer games and movies. Hence, it is obviously difficult and time-consuming to adapt content analysers to a particular genre. This issue also motivates the research question of this thesis: Is it possible to design video analysis approaches that adapt themselves automatically to a particular video. This problem statement is discussed in more detail in the section below.

1.2 Problem Statement

As explained above, there is a strong need for methods that automatically explore, organize and index multimedia content. Up to date, the automatic understanding of multimedia content remains an unsolved problem in practice. For large multimedia databases, the variability of multimedia sources and content is enormous, and obviously this is also true for video databases. On the other hand, in the fields of video content retrieval and scientific movie analysis, computer-assisted methods promise a dramatic reduction of human annotation efforts required for these tasks. Unfortunately, even the best video content analysis approaches are not perfect, and the higher the challenge of a human-like scene understanding is, the more imperfect the approaches are. For scientific movie analysis, the need for accurate content analysis algorithms is evident: As the number of false annotations increases (though generated automatically), the application of computer-assisted approaches for media research purposes becomes questionable: scientific hypotheses based on imprecise experimental data are useless.

The question is how to build video content analysis and indexing approaches that work reliably on arbitrary videos. Many video content analysis approaches are considered "robust" by their inventors. However, in most cases this means that an algorithm or system has proven to work well on one or more (hopefully large) test sets. Furthermore, in most cases one classification model or decision threshold is applied by the system to all test videos in the same way: This might be a learned model using machine learning techniques or a set of pre-defined parameters that have been estimated empirically. Obviously, this is a problem as long as we do not restrict our video database in some respects. But videos can vary in many ways: the kind of recording device, the recording circumstances, the used compression technology, editing, genre and, of course, in terms of content. In particular, this is true for the media research center "Media Upheavals" in which the creation date of video material ranges from the beginning of film in 1895 up to the newest developments like hybrid forms of movies and computer games. Hence, there is a need for the development of algorithms that work reliably independent of the factors mentioned above. There is a need for algorithms that automatically adapt to a particular video and with respect to the content that is present in a given video.

The research question is how to build video analysis and indexing approaches that work robustly and reliably on any video, independent of compression and content. This thesis investigates solutions to obtain a high-quality analysis and indexing result for a particular video by considering the context of content and compression appropriately. Keeping these requirements in mind, the following specific problem statements can be formulated for different video analysis and indexing tasks.

1.2.1 Robust Video Content Analysis of Compressed Video Data

Video compression is a basic technology that facilitates the construction of video databases by a large data reduction. To reduce the amount of data needed for video coding, spatial and temporal redundancy in a video is exploited as well as the characteristics of the human visual system which is less sensitive to visual information of higher frequencies. In order to speed-up the processing time of video analysis and indexing approaches, it is reasonable to exploit information embedded in compressed videos. On the other hand, compression and the related quality deterioration lead to artefacts that hamper the video analysis process. Thus, video analysis approaches need to exploit compressed video information but also have to cope adequately with compression artefacts. The research question is how to utilize compressed information to efficiently analyze video data and to adequately cope with compression artefacts at the same time. In particular, shot boundary detection and camera motion estimation are investigated with respect to this question.

1.2.2 Robust and Adaptive Shot Boundary Detection

Shot boundary detection is a very important basic task in order to structure a video document for subsequent indexing purposes. For scientific movie analysis, the identification of shot changes and their type yields information about the producer's intention to express a certain meaning. For example, a high cut frequency might express a high tempo to supplement an action scene, a long dissolve might indicate a dream sequence, and fading effects are often used in movies to mark the beginning and the end of a scene (e.g., refer to Arijon [6], Hickethier [73] or Korte [88]). In addition, quantitative measures like cut frequency allow an objective comparison of different films with respect to media research questions. For media research, a shot boundary detection approach must be very accurate since false detection results would hamper or inhibit scientific conclusions. If manual corrections are required, this is a time-consuming task which would reduce the amount of video material that can be investigated. These considerations motivate our next research question: How can we build a robust shot boundary detection approach that achieves best possible performance on a particular video: a.) independent of compression artefacts; b.) independent of genre and content, and c.) without any parameter adjustment that requires user interaction and knowledge?

1.2.3 Robust Estimation of Arbitrary Camera Motion in MPEG Videos

In movie production, camera distance and camera motion are very important elements to express a certain atmosphere or a meaning through the way a scene is captured and presented to the viewer. Since a moving camera is a powerful stylistic device (see Arijon [6]), there is a strong interest of media researchers to investigate how the camera device was used by the producer. Technically, it can be distinguished between a travelling camera (translation along the x-, y- or z-axis) and a rotating camera (camera is rotated around one of the three axes). Several algorithms have been proposed to solve the problem of camera motion estimation in digital videos, for both compressed and uncompressed domain. However, the distinction between translation along the x-axis (y-axis) and rotation around the y-axis (x-axis) has only rarely been considered, and no approach of this kind is known for the MPEG domain. Thus, the research question is how to estimate arbitrary camera motion in MPEG videos.

1.2.4 Robust Face Recognition and Indexing in Video

Retrieving information about the occurrence of persons in a video is important for many video indexing and retrieval applications. Also, the recognition of a person's appearances in a video or in a movie is necessary in order to understand the video/movie content, since in most cases the plot is basically related to persons or actors. However, in general it is not known a priori who will appear in a video. Given an arbitrary video, pose and illumination of faces are uncontrollable, and it is not possible to make assumptions about the occurrences of faces. Keeping this in mind, the following questions should be answered: "In which shot Y and scene Z does person X appear, how often does person X appear in the entire video and, in which pose and size?", provided that only the shot and eventually the scene boundaries are given in advance for a video/movie, but no face models are available. In particular, the research question whether a person's face appearance in a particular video can be learned by a system itself in order to improve the recognition rate for that video is investigated.

1.2.5 Robust and Adaptive Detection of Semantic Concepts in Videos

The ultimate goal of research efforts in the domain of multimedia retrieval is the automatic understanding of audiovisual content. If it was feasible to automatically understand what is shown in a video shot or what is said in a news cast, then it would be much easier to answer user queries for multimedia databases. It is a very hard problem to automatically recognize objects and events in an image or in a video, although there has been some success for several object recognition tasks, for example regarding face detection and face recognition. The question is whether indexing or retrieval performance can be improved for some high-level concepts when their appearance is strongly related to a particular video source (e.g., the weather news in a news cast or a sequence category in a computer game session).

1.3 Contributions

This thesis focuses on the need for robust and adaptive algorithms for video content analysis. One of the major contributions of this thesis is to consider the analysis process for a particular video as a setting that is well suited for transductive learning. Transductive learning is not aimed at obtaining a general classification function for all possible test data points (as in inductive learning) but at obtaining a specific classification for the given test data only. In this thesis, this idea is applied to achieve robust video content analysis: the unlabeled data of a particular, previously unseen video are incorporated into the learning and classification process. The resulting, final classification function is expected to work well in particular for the given data of a particular video, but it is not necessarily expected to work well in general.

1.3.1 Transductive Learning Methods for Robust Video Content Analysis

Typically, a machine learning technique falls either into the class of s*upervised learning* or *unsupervised learning*. In supervised learning, a set of labeled training data is used to find a mapping between example data x and a target function y. The goal of unsupervised learning is to find an interesting structure in the given data. In case of unsupervised learning, labels are not available or used for learning.

Semi-supervised learning lies between the classes of supervised and unsupervised learning. Here, the training set consists of both labeled and unlabeled data: one part of the training data has labels, whereas no labels are available for the training samples of the other part. In most applications, the goal is to find a mapping between the training samples and the labels by exploiting the additional information available through the unlabeled data.

In this thesis, the following learning process is called *self-supervised*: First, an initial model is generated using unsupervised learning and this model is used to label the previously unseen, originally unlabeled (test) data. Then, only these labels are used to learn a new model in a supervised learning scheme which is finally utilized to re-classifiy the test data.

Transductive learning was introduced by Vapnik [166] and is closely related to semi-supervised learning. Given a (labeled) training set and an unlabeled test set, the idea of transductive learning that the desired classification function has to be optimal for the test data only and *not* to infer a general decision rule (as in the case of inductive learning).

As mentioned above, it is proposed in this thesis to consider the task of robust video content analysis as a setting which is well suited for transductive learning. For this purpose, a self-/semi-supervised ensemble learning framework is presented that exploits an initial classification (or clustering) result and the unlabeled video test data to improve its quality for a particular video. The proposed framework is based on feature selection and ensemble classification; it is called *self-supervised* when the baseline approach relies on unsupervised learning, and it is called *semi-supervised* when the baseline approach relies on supervised learning. Within the scope of this thesis, solutions for several video content analysis and video indexing problems are presented. Apart from the solutions that are based on the proposed learning framework, some proposals in this thesis employ unsupervised learning or deal with compression artefacts adequately. Overall, the following tasks are considered: shot boundary detection, estimation of camera motion, face recognition, semantic concept detection and semantic indexing of computer game sequences. Several strategies are investigated to utilize this transductive setting in order to obtain optimal results for different video content analysis tasks:

- dealing with compression artefacts (video cut detection, camera motion estimation),
- automatic parameter estimation (video cut detection),
- applying self-supervised learning (video cut detection, face recognition/clustering),
- applying semi-supervised learning (semantic video retrieval, semantic video indexing),
- applying transductive support vector machines (semantic video retrieval);

These strategies are applied separately to each video in order to improve a detection, recognition or indexing result for a particular video. Self-supervised and semi-supervised learning schemes are developed which are based on an initial clustering (self-supervised scheme) or classification (semi-supervised scheme) result. In these schemes, feature selection and ensembles classification are employed to improve an initial result. It is demonstrated for several video content analysis tasks that an initial result can be utilized as training data and further exploited to learn and to adapt a machine learning model to a particular video, despite the fact that these automatically generated training data contain errors. Apart from transductive, semi-supervised and self-supervised learning, some of the proposed approaches work in the compressed MPEG domain in order to save computation time. It is shown for the tasks of shot boundary detection and camera motion estimation how compression artefacts can be removed and compressed data can be exploited successfully.

The following robust and adaptive approaches are proposed to analyze video content, in particular with respect to compression artefacts and the characteristics of a particular video. Except for the camera motion estimation approach, all proposals consider the task of video content analysis as a transductive setting. First, a novel unsupervised shot boundary detection approach is developed which is focused on the avoidance of any parameter settings. This approach is further extended to a self-supervised learning ensemble, which is the first application of the transductive learning framework proposed in this thesis. Second, an approach to estimate camera motion in MPEG videos is presented. Third, a system for face recognition in arbitrary videos is presented that automatically copes with in-plane rotation and learns the face appearances in a certain video by itself. Finally, it is demonstrated that semi-supervised and transductive learning are also applicable to the task of concept detection, although the baseline performance is noticeably below the baseline performances of the tasks of cut detection, camera motion estimation and face recognition.

1.3.2 Robust and Adaptive Shot Boundary Detection in Videos

The main ideas for robust and adaptive video content analysis are introduced by examples and step by step for a video cut detection algorithm in Chapter 4. The proposed techniques enable our approach to improve the detection quality for a particular video. The main ideas are as follows. First, it is demonstrated how cut detection algorithms can suffer from compression artefacts, and a solution is presented to automatically cope with such artefacts. Second, an unsupervised approach is developed with the advantage that the time-consuming task of manual creation of training data is not required. Third, the impact of any parameter settings is (nearly) completely removed by the automatic estimation of an important parameter. Furthermiore, the unsupervised approach is extended to a self-supervised learning ensemble to employ the transductive setting. To adapt to a particular video to improve detection accuracy, the best features are selected for this video, split into different feature sets, and then several classifiers are trained directly on a video using these different feature sets. Finally, this ensemble of classifiers is used to re-classify the video under consideration. Since the algorithm labels the training data for a particular video by itself and then learns a model using these data, this process is called self-supervised learning.

1.3.3 Estimation of Arbitrary Camera Motion in MPEG Videos

The computation of optical flow is necessary to estimate camera motion in a video sequence. For MPEG videos, computation time can be saved if the motion vector data present in MPEG videos can be utilized. However, MPEG motion vectors do not necessarily represent the "true" motion for a particular frame region since the computation of a motion vector is targeted at compression efficiency. As a consequence, there are outliers in a motion vector field that do not represent motion well. In Chapter 5, an approach to enhance a motion vector field in terms of motion representation is employed. Finally, it is demonstrated that an enhanced motion vector field is more suited to compute camera motion. To estimate camera motion, a three-dimensional camera model is assumed and the problem of camera motion estimation is considered as an optimization problem where a best parameter fit in the 3D camera motion model must be estimated.

1.3.4 Self-Supervised Learning of Face Appearances in Videos and TV Casts

The appearance and behaviour of people is one of the most important aspects to gather the meaning of an image or a video sequence. Hence, one of the most important tasks in video indexing applications is to answer the question "In which shot or scene does person X appear?". Assuming that there is no knowledge about the fact who will appear in a particular video, face examples cannot be used for a supervised machine learning approach. In Chapter 6, an automatic person indexing system for videos is presented that basically relies on state-of-the-art techniques for face detection, tracking and recognition and unsupervised learning. The given task is considered as a transductive setting and the baseline system is extended to a self-supervised learning ensemble. The main idea is to use an initial clustering result to gather information about the faces of the appearing persons and then to select those features that are best suited for this particular video to discriminate between the different persons and their faces, respectively. The initial clusters and related features are then exploited to train face models directly on the video and to re-classify the faces in a video using the classifiers.

1.3.5 Adapting Concept Models to a Video via Semi-Supervised Learning

Apart from shot boundary detection, camera motion detection, and person recognition, there are research efforts to bridge the semantic gap between the audiovisual data and the meaning related with a scene. These efforts are targeted at identifying so-called high-level features or semantic concepts in shots, for example concepts like: car, person, overlayed text, indoor, sports scene, beach, explosion, or popular persons like politicians or sportsmen. Machine learning is extensively used in this field to map low-level features to high-level concepts. The detection performance of state-of-the art approaches depends on the difficulty and frequency of a particular concept. In Chapter 7, it is demonstrated that semi-supervised learning can be also applied to high-level features (concepts) in a transductive setting, that is the appearance model of high-level features can be improved with respect to a particular video. Surprisingly, in some cases a baseline performance of about 40% average precision is sufficient as long as the appearance of a concept is related to a particular video recording. Usually, a model is applied to the shots of all videos in the same way. The novel idea is that a baseline model is adapted to each video individually. For a particular video, the top (bottom) ranked shots retrieved according to the baseline model are exploited to select relevant features specifically for a video and to train a new model (represented by an ensemble of classifiers) on this video. This classifier ensemble is then applied to obtain new probability scores for the shots of a video. It should be noted that the number of training samples is relatively small in this scenario: a news video from the TRECVID test set has about 250-400 shots which may serve as training data.

1.3.6 Semantic Analysis of Computer Games for Psychological Research

In addition, it will be demonstrated for an interdisciplinary semantic video indexing task of computer game sessions that it is beneficial to consider this task as a transductive setting and to apply semi-supervised learning. Computer games play a very important role in today's entertainment media and belong to the most popular entertainment products. Unfortunately, the number of computer games containing serious violent content increases. Weber et al. [172] present an experimental setting that is based on the definition of certain game states and capture a player's brain activity via fMRI (functional magnetic resonance imaging) while the player is playing a violent computer game. Several semantic game events are distinguished: 1.) inactive; 2.) preparation; 3.) search and explore; 4.) danger; 5.) under attack, and 6.) fighting and killing. Once the game recordings are annotated with these semantic categories, the interrelationship of violent game events and the underlying neurophysiologic basis (brain activity) of the player can be investigated. Normally, human annotators are required to index such game content according to the current game state, but this is a very time-consuming task. In this context, computer-based automatic video content analysis of computer game recordings promises several advantages: Human annotation efforts can be reduced noticeably, and the annotation process is speeded up and is based on reproducible and objective criteria only. At the same time, researchers are enabled to investigate a larger number of computer game videos to gather more experimental data. An automatic semantic video analysis system that supports the experimental design described above by automatically identifying the game states (i.e., categories) is presented in Chapter 8. The system is aimed at minimizing the human annotation effort and thus requires manual annotations for a single video only to facilitate the semi-supervised learning process. In our approach, only a single video sequence with a duration of 12 minutes is required to provide training data and hence, the human annotation effort is kept at a minimum. To achieve a more robust result, an automatic semi-supervised correction step is employed separately for each video: Based on the initial classification result, the system automatically labels the frames in a new video and adapts its concept models to this video by employing feature selection and adaptively building a specialized classifier for a particular game video.

1.3.7 Robust Video Content Analysis in the Compressed Domain

Finally, it is demonstrated for the tasks of video cut detection and camera motion estimation how to cope efficiently with compression artifacts in MPEG videos. Compression artifacts might lead to noisy representations of visual information and might degrade indexing quality. In this respect, a systematic bias in frame dissimilarity measurements in MPEG videos can be identified which potentially hinders video cut detection performance. Solutions to deal with these artefacts are presented in Chapter 4.4. Regarding the estimation of arbitrary camera motion, a solution that employs MPEG motion vectors is presented in Chapter 5. To deal with noisy motion vectors that do not represent “true” motion, an effective outlier removal algorithm is applied.

1.4 Publications

The following papers have been published in the context of this thesis:

1. Ewerth, R. and Freisleben, B. *Frame Difference Normalization: An Approach to Reduce Error Rates of Cut Detection Algorithms for MPEG Videos.* In Proceedings of the IEEE International Conference on Image Processing, Barcelona, Vol. 2, IEEE Press, 2003, 1009-1012.

2. Gllavata, J., Ewerth, R., and Freisleben, B. *Finding Text in Images via Local Thresholding.* In Proceedings of the IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), Darmstadt, IEEE Press, 2003, 539-542.

3. Gllavata, J., Ewerth, R., and Freisleben, B. *A Robust Algorithm for Text Detection in Images.* In Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis, Rome, IEEE Press, 2003, 611-616.

4. Ewerth, R., Gllavata, J., Gollnick, M., Mansouri, F., Papalilo, E., Sennert, R., Wagner, J., Freisleben, B., Grauer, M. *Methoden und Werkzeuge zur rechnergestützten medienwissenschaftlichen Analyse*. In Siegener Periodicum zur Internationalen Empirischen Literaturwissenschaft, 20, H. 2, 2003, 306-320.

5. Gllavata, J., Ewerth, R., Stefi, T., and Freisleben, B. *Unsupervised Text Segmentation Using Color and Wavelet Features*. In Proceedings of the 3rd International Conference on Image and Video Retrieval 2004, Lecture Notes on Computer Science LNCS 3115, Dublin, Springer-Verlag, 2004, 216-224.

6. Ewerth, R. and Freisleben, B. *Improving Cut Detection in MPEG Videos by GoP-Oriented Frame Difference Normalization*. In Proceedings of the 17th International Conference on Pattern Recognition 2004, Cambridge (UK), Vol. 2, 2004, 807-810.

7. Ewerth, R., Schwalb, M., Tessmann, P., and Freisleben, B. *Estimation of Arbitrary Camera Motion in MPEG Videos*. In Proceedings of the 17th International Conference on Pattern Recognition 2004, Vol. 1, Cambridge (UK), 2004, 512-515.

8. Gllavata, J., Ewerth, R., and Freisleben, B. *Text Detection in Images Based on Unsupervised Classification of High-Frequency Wavelet Coefficients*. In Proceedings of 17th International Conference on Pattern Recognition, Vol. 1, Cambridge (UK), 2004, 425-428.

9. Ewerth, R. and Freisleben, B. *Video Cut Detection without Thresholds*. In Proceedings of the 11th International Workshop on Signals, Systems and Image Processing, Poznan, Poland, 2004, 227-230.

10. Gllavata, J., Ewerth, R., and Freisleben, B. *Tracking Text in MPEG Videos*. In Proceedings of ACM Multimedia 2004, New York, ACM Press, 2004, 240-243.

11. Ewerth, R., Friese, T., Grube, M., and Freisleben, B. *Grid Services for Distributed Video Cut Detection*. In Proceedings of the 6th IEEE International Symposium on Multimedia Software Engineering, Miami (USA), IEEE Press, 2004, 164-168.

12. Gllavata, J., Ewerth, R., and Freisleben, B. *A Text Detection, Localization and Segmentation System for OCR in Images*. In Proceedings of the 6th IEEE International Symposium on Multimedia Software Engineering, Miami (USA), IEEE Press, 2004, 310-317.

13. Ewerth, R., Beringer, C., Kopp, T., Niebergall, M., Stadelmann, T., and Freisleben, B. *University of Marburg at TRECVID 2005: Shot Boundary Detection and Camera Motion Estimation Results*. In Online Proceedings of TRECVID Conference Series 2005: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, 2005.

14. Ewerth, R. and Freisleben, B. *Gesichtsdetektion und -erkennung in Bildern und Videos für die medienwissenschaftliche Analyse* (in German). In: Universi Verlag, W. Beilenhoff, W. Hülk, K. Kreimeier, und M. Erstic (eds.), 2006, 229-256.

15. Ewerth, R. and Freisleben, B. *Self-Supervised Learning for Robust Video Indexing*. In Proceedings of the IEEE Conference on Multimedia & Expo 2006, Toronto, 2006, 1749-1752.

16. Ewerth, R., Mühling, M., and Freisleben, B. *Self-Supervised Learning of Face Appearances in TV Casts and Movies*. In Proceedings of the IEEE Symposium on Multimedia, San Diego, CA, USA, 2006, 78-85.

17. Ewerth, R., Mühling, M., Stadelmann, T., Agel, B., Seiler, D., and Freisleben, B. *University of Marburg at TRECVID 2006: Shot Boundary Detection and Rushes Task Results*. In Online Proceedings of TRECVID Conference Series 2006: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, 2006.

18. Ewerth, R. and Freisleben, B. *Computerunterstützte Filmanalyse mit Videana* (in German). In Augen-Blick: Hefte zur Medienwissenschaft, Schüren-Verlag, Marburg, 2007, 54-66.

19. Mühling, M., Ewerth, R., Stadelmann, T., Shi, B., Zöfel, C., and Freisleben, B. *University of Marburg at TRECVID 2007: Shot Boundary Detection and High-Level Feature Extraction*. In Online Proceedings of TRECVID Conference Series 2007: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, 2007.

20. Ewerth, R., Schwalb, M., and Freisleben, B. *Using Depth Features to Retrieve Monocular Video Shots*. In Proceedings of ACM International Conference on Image and Video Retrieval, Amsterdam, ACM Press, 2007, 210-217.

21. Mühling, M., Ewerth, R., Stadelmann, T., Freisleben, B., Weber, R., Mathiak, K. *Semantic Video Analysis for Psychological Research on Violence in Computer Games*. In Proceedings of ACM International Conference on Image and Video Retrieval, Amsterdam, ACM Press, 2007, 611-618.

22. Ewerth, R., Freisleben, B. *Semi-Supervised Learning for Semantic Video Retrieval.* In Proceedings of ACM International Conference on Image and Video Retrieval, Amsterdam, ACM Press, 2007, 154-161.

23. Ewerth, R., Schwalb, M., Tessmann, P., and Freisleben, B. *Segmenting Moving Objects in MPEG Videos in the Presence of Camera Motion.* In Proceedings of 14th International Conference on Image Analysis and Processing, Modena, Italy, IEEE Press, 2007, 819-824.

24. Ewerth, R., Mühling, M., and Freisleben, B. *Self-Supervised Learning of Face Appearances in TV Casts and Movies.* Invited paper (Best papers from IEEE International Symposium on Multimedia 2006, ISM 2006): In International Journal on Semantic Computing, World Scientific, 2007, June, 185-204.

25. Ewerth, R. and Freisleben, B. *Adapting Appearance Models of Semantic Concepts to a Particular Video via Transductive Learning.* In Proceedings of 9th ACM SIGMM Int'l Workshop on Multimedia Information Retrieval, in conjunction with ACM Multimedia 2007, Augsburg, Germany, 2007, 187-196.

1.5 Organization of this Thesis

This first chapter gives an overview about the organization of this thesis: the broader scope and the main contributions are briefly introduced. The background of this thesis with respect to content-based video indexing and retrieval, video compression, scientific film studies, and different machine learning approaches is presented in Chapter 2. In Chapter 3, the principles of the transductive learning ensemble framework and its main components are introduced. Chapter 4 starts with a comprehensive literature survey for the issue of shot boundary detection in videos, followed by an experimental comparison study. Then, several proposals are presented in Chapter 4 for shot boundary detection: an approach that deals with compression artefacts, an unsupervised approach, an ensemble approach, an approach based on the self-supervised learning framework, and finally, an approach for performance prediction is presented. In Chapter 5, an approach for camera motion estimation is presented that utilizes compressed domain motion information and deals with compression artefacts. The second application of the proposed self-supervised framework is presented in Chapter 6 for face recognition in videos. A variant of the semi-supervised learning approach is introduced in Chapter 7 for detecting semantic concepts whose appearance is related to a particular video. An interdisciplinary application of the semi-supervised learning approach is presented in Chapter 8 which supports psychological research on violence in computer games. Finally, Chapter 9 provides a summarization of this thesis and envisages areas for future work.

2 Background

2.1 Introduction

In this chapter, the research fields of video retrieval and scientific movie analysis, and furthermore some technological fundamentals of video processing and machine learning are introduced since they are essential for the understanding of the subsequent chapters. A brief overview is given for content-based video retrieval, followed by some thesis-relevant video compression techniques. An introduction to the field of scientific film studies is presented in the subsequent subsection. Finally, some machine learning and optimization methods are briefly discussed in section 2.4 since they are applied in the subsequently proposed approaches: clustering, classification with Support Vector Machines (SVM), semi-supervised learning, meta-learning methods and ensemble learning, and a simplex-based optimization method.

2.2 Content-Based Video Retrieval

In this section, the domain of content-based video retrieval is viewed from a global perspective to introduce the broader scope of this thesis, whereas the related work for the particular proposed video analysis approaches is discussed in the corresponding chapters. There are research efforts in the field of content-based image and video retrieval for many years. For example, see surveys for image search engines [59], for image and video retrieval, and multimedia retrieval [7, 190], respectively, and for multi-modal video indexing [146].

2.2.1 Different Levels of Multimedia Understanding

Bashir et al. [9] distinguish three levels of abstraction to model multimedia content and multimedia understanding, respectively:

1. low-level physical modelling of raw image and video data (e.g., describing an image by color histograms and texture descriptors);

2. representation of derived or logical features (e.g., object segmentation); and

3. abstractions at the semantic level by intelligent modelling based on concepts.

Bashir et al. [9] also review some techniques for low-level feature based image retrieval (color, shape and texture), discuss compressed domain processing and image segmentation techniques, high-dimensionality reduction and relevance feedback techniques. For the field of video indexing, they review approaches for temporal segmentation into shots, video summarization, motion analysis, and high-level semantic modeling.

2.2.2 Sensory Gap and Semantic Gap

Automatic understanding of multimedia content is a very difficult and challenging task (at least for machines, of course not for the large majority of humans). Gevers [59] identifies two issues for machine-based multimedia understanding, the sensory gap and the semantic gap: According to Gevers [59], the *sensory gap* is defined as "... the gap between the object in the world and the information in a (computational) description derived from a recording of that scene", whereas the *semantic gap* is defined as "... the lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data have for a user in a given situation". A user searches for objects and events on a conceptual level, whereas the image description is given by low-level data, extracted automatically from an image, and obviously the user's search intention and the data description might be coupled only loosely, if at all. Gevers states [59] that the correct association of semantic descriptions to images requires automatic object recognition – which unfortunately is still unsolved in general, which is also true for object segmentation [157].

2.2.3 Image and Video Frame Features

Image retrieval techniques can be basically used for processing and retrieving video frames which, for example, has been demonstrated by Kreyß et al. [89]. Gevers gives an overview [59] of image search engines. Images are described commonly by a set of features with respect to color, shape and texture. These features should be invariant to illumination, object pose and camera viewpoint in order to achieve the goal that similar objects yield similar feature values. *(Local) Shape* is considered by Gevers as "all properties that capture conspicuous geometric details in the image", whereas *texture* "... is considered as all what is left after color and shape have been considered ...". A detailed description of such features is beyond the scope of this thesis but is given, for example, by Gevers and Smeulders [59], color and texture features as used in the MPEG-7 standard are described by Manjunath et al. [106], and MPEG-7 shape feature descriptors are proposed by Bober [14]. An example for an image retrieval system is the 'PictureFinder' which is presented by Schober et al. [136]. This system employs description logics for the purpose of semantic image retrieval.

2.2.4 Structure of Videos and Movies

Content-based video retrieval aims to support an efficient user search for certain objects, events, shots or video documents in a possibly large video database. A video is built up in a hierarchical manner. It consists of a number of single frames (images) which are displayed at a certain frame rate to convey the impression of moving pictures. The fundamental unit for video retrieval is that of a shot. Korte [88] defines a shot as the smallest continuously exposed cinematic unit that, in general, consists of several frames. Arijon [6] defines a shot as an amount of film that is exposed in the camera without reloading. A scene is a sequence of shots which are related in space and/or time and belong together semantically [163], whereas a sequence consists of several scenes and conveys an essential part of the plot.

2.2.5 The Process of Content-Based Video Retrieval

A database query itself can be textual or visual. To answer a query, it must be comparable to the database content and thus its modality should be compliant to the representation of the videos in the database. In case of a visual query by an example shot, an example frame or a frame region, the query modality is compliant with the representation of the videos in the database. However, it became evident (Naphade and Smith [117]) during the first TREC video retrieval task in 2001 that query by content using low-level features is inadequate to address the challenge of high-level queries for video databases. Also, the use of automatic speech recognition (ASR) was not sufficient to answer those high-level queries. As stated by Naphade and Smith [117], these facts led indirectly to the formation of the high-level concept detection task at TRECVID in order to support high-level queries: the high-level feature task is to detect high-level features (concepts) like car, building, indoor/outdoor, waterfront, sports with the aim of annotating video shots automatically.

In case of a textual query, a modality conversion is required: either the textual query has to be converted to a visual representation or the video database content has to be supplemented with textual annotations. Usually, the latter is realized in practice since it is done by a concept detection system, according to the high-level features mentioned above. Therefore, it is common to define a lexicon with a number of concepts. For example, in the TRECVID high-level feature task the following concepts are defined: sports, weather, court, office, meeting, studio, outdoor, building, desert, vegetation, mountain, road, sky, snow, urban, waterscape, crowd, person, face, police, military, prisoner, animal, screen, US flag, airplane, bus, car, truck, boat, walking/running, people marching, explosion, natural disaster, maps, charts. In the pre-processing step of video content analysis, the video shots in the database are analyzed with respect to these concepts (often also called high-level features, semantic concepts, or semantic features). In this context, the meaning and interaction of the processes of video content analysis, video indexing, and video retrieval become clearer. The process of *video content analysis* aims at recognizing objects and events in a video document. These objects and events might be high-level features as well as low or mid-level features such as shot boundaries or camera motion. The process of *video indexing* supplies video frames, shots and scenes with automatically extracted meta-information (labels) that support a subsequent user search for a specific content in a video database. Related to the process of video indexing is the use of appropriate data structures and index structures for the usually high-dimensional feature vectors in order to achieve short query response times. The automatic classification of video content and subsequent assignment of content-based labels to video documents is refered to as *video indexing* [146]. Here, the process of content analysis is considered to be part of the indexing process. From our point of view, the automatic classification, which is considered as video content analysis in this thesis, should be distinguished from the video indexing process for the purpose of clarification. Of course, both processes are strongly related in most practical solutions. Finally, the process of *video retrieval* answers a specific user query and returns documents that are considered as relevant. Normally, the processes of video content analysis and indexing precede the query and retrieval process in order to guarantee a reasonable processing time.

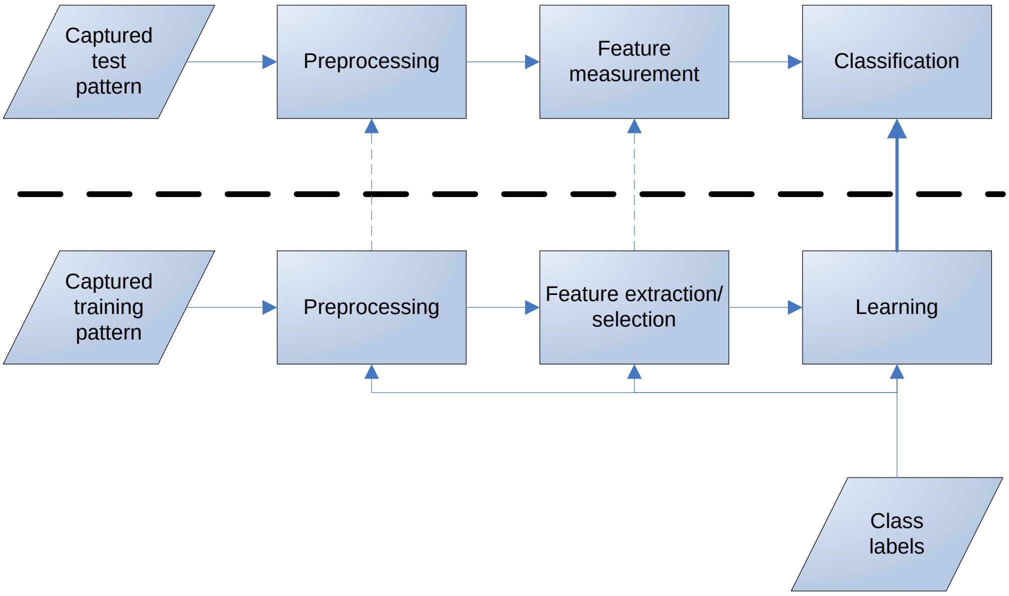


Figure 1: Typical model for statistical pattern recognition, (according to [79], slightly modified).

2.2.5.1 VIDEO CONTENT ANALYSIS

The process of video content analysis can be considered as a pattern recognition process since it deals with the recognition of events (shot boundaries, motion, dialogs etc.) and objects (superimposed text, faces etc). Watanabe [171] defines a *pattern* "as opposite of a chaos; it is an entity, vaguely defined, that could be given a name". According to Niemann [119], the set of patterns is defined as the set of objects (functions) that can be captured with an appropriate sensor device. A pattern can be a fingerprint image, characters and words in a document, a face, or a speech signal. Typically, a content analysis or pattern recognition system consists of two parts [79], a learning component and a classification component (see Figure 1). Training patterns are used to learn an appropriate model for the corresponding classes. Usually, such a training pattern has to be preprocessed, for example the pattern of interest must be segmented from the background, noise must be removed, it must be normalized, or any other operation may be applied that contributes to a compact and informative pattern representation. In the subsequent process, features are selected and extracted for the representation of this pattern. Then, a classifier is trained to separate the feature space according to the boundaries of different classes. A feedback path allows the system designer to modify the preprocessing as well as the feature extraction process based on the training results achieved so far. Finally, the trained classifier is enabled to automatically assign a class label to a newly presented input pattern based on the feature measurements.

2.2.5.2 VIDEO INDEXING

Snoek and Worring [146] comprehensively survey the field of multi-modal video indexing. They distinguish the indexing issues of granularity (what to index), the methods used (how to index), and the types that should be used to index (which indexes). Their framework is restricted to produced video, and it is suggested to view a video document from the author's perspective. According to Snoek and Worring [146], *multimodality* is defined as "The capacity of an author of the video document to express a predefined semantic idea, by combining a layout with a specific content, using at least two information channels.". Three information modalities are considered, visual modality, auditory modality and textual modality and thus a video can be viewed from at least three different perspectives. The intended semantic meaning can be expressed at several granularities, for the video as a whole, for so called logical units that consist of a set of named events or other logical units (e.g., a shot or a dialog scene in a movie), and finally for named events (e.g., a goal in a soccer match, weather news in a news broadcast). Content and layout are utilized by the author to express this meaning, and the following content elements are distinguished: setting, objects and people. Layout considers the syntactic structure, for which the fundamental units are frames (visual modality), audio samples (auditory modality) and characters (textual modality). Sensors (camera, microphone, and in a certain sense a title or caption editor) capture a continuous sequence of those fundamental units, resulting in camera shots, microphone shots and text shots which are not necessarily aligned temporally. The interrelationship of all these elements and the related multimodal video indexing framework is displayed in Figure 2. Several techniques are summarized in Snoek's and Worring's review, in particular for:

1. layout reconstruction (e.g., shot segmentation; music, silence and audio break detection),
2. content segmentation (e.g., uni-modal and multimodal approaches for people, object and setting detection),

3. modality conversion (e.g., from overlayed text to ASCII characters),

4. modality integration,

5. semantic indexing at the level of:

    - genre and sub-genre recognition,
    - logical units (dialog detection, violence detection, anchor shots in news casts, etc.),
    - named events (violent events in movies, score changes in sport videos, etc.).

During the analysis process, objects and events are recognized in a video and related metadata are generated automatically. These metadata are then associated with video segments at different levels of granularity. For example, the OCR result for a superimposed text might be stored for the related shot. An appropriate representation of media content descriptions and metadata is needed to store the extracted metadata in a reasonable fashion. A standardized description enables the exchange between different video database systems.

For example, MPEG-7 is such a standard [114] and provides a set of standardized descriptions for multimedia metadata. It was developed to support and facilitate a wide range of multimedia applications, for example media portals, media content retrieval, media broadcasting, ubiquitous multimedia etc. MPEG-7 media descriptions are XML documents conforming to schema definitions expressed with the XML schema variant MPEG-7 DDL, the Description Definition Language. The standard incorporates a set of normative elements, consisting of multimedia content and management descriptors (D), Description Schemes (DS), a Description Definition Language (DDL), and coding schemes. Descriptors describe features, attributes, or groups of attributes of multimedia content. They define the syntax and the semantics of a feature representation. Different levels of abstraction are addressed, at the lower level descriptors are defined for color, shape, texture, object motion and camera motion in videos, and harmonicity, timbre and energy in audio, whereas at the higher level descriptors might include events, abstract concepts, content, genres etc. [24]. Description schemes (DS) describe entities or relationships between multimedia objects or events and specify the structure and semantics of their components, which might be description schemes, descriptors or data types. In MPEG-7, data types are basic reusable data types that may be employed by descriptors and description schemes. The description definition language defines data types, descriptors, description schemes by specifying their syntax and allows their extension.

Overviews of the MPEG-7 standard are provided by Martinez [109] and by Chang et al. [24], respectively.

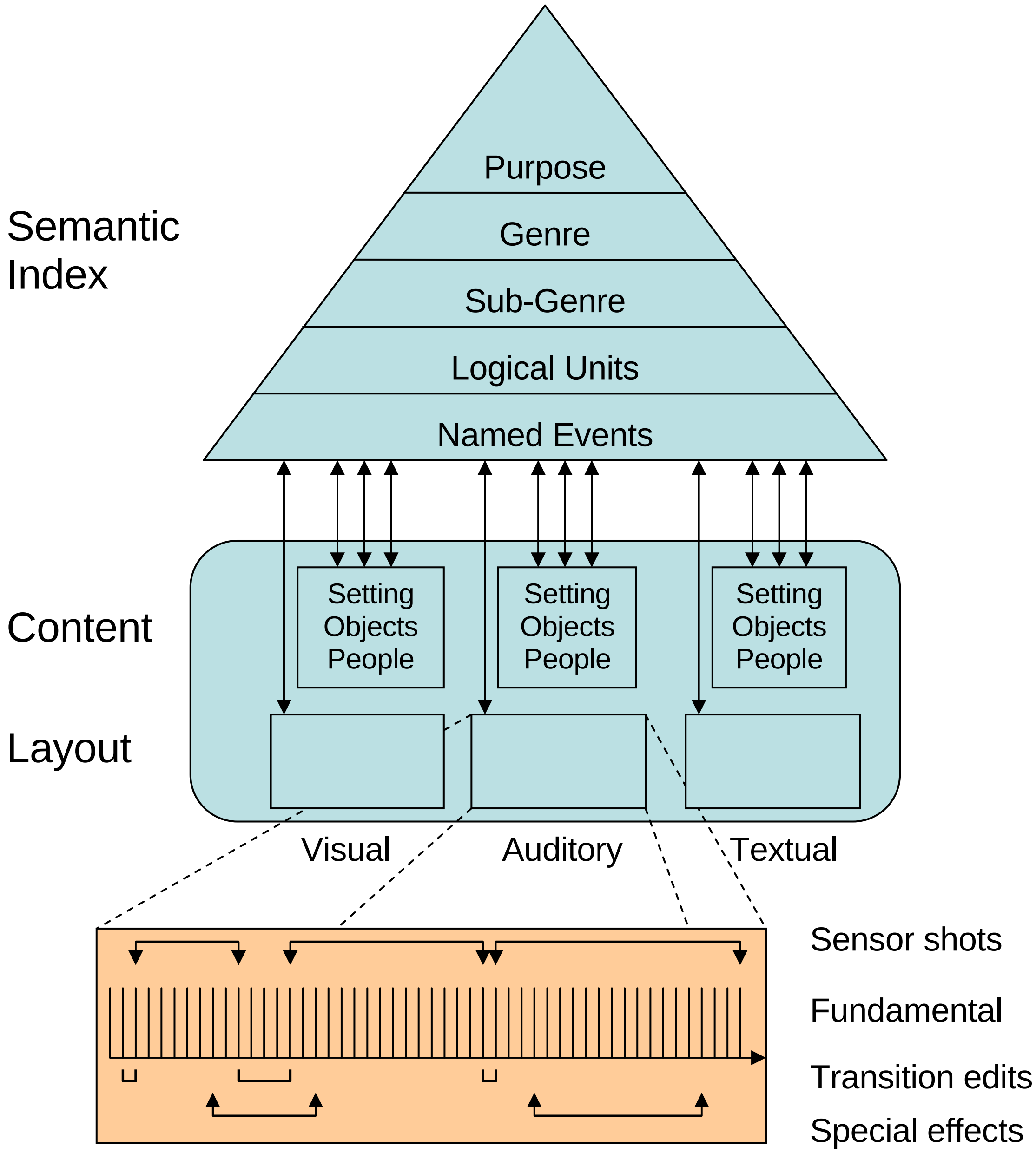


Figure 2: Framework for multimodal video indexing (according to Snoek and Worring [146]).

The MPEG-7 standard provides a tool set that supports the interchangeability of media description among different users. Apart from that, efficient index structures are needed for the high-dimensional feature data to support similarity search and nearest neighbour search. Westermann and Klas [173] analyze XML database solutions for their applicability of managing MPEG-7 media description and identify a number of shortcomings, for example with respect to the indexing of multi-dimensional numerical data. A survey of index structures for search in multimedia databases is presented by Böhm et al. [16].

2.2.5.3 VIDEO RETRIEVAL

Retrieval aims at representing relevant documents to a user according to a textual query at a conceptual level which represents his/her need. In case of video retrieval, such a document is normally a video shot in most systems. To offer a reasonable processing time, the processes of video content analysis and video indexing should have been conducted before a user submits a query to the system. Aslandogan and Yu [7] present an overview of techniques and systems for image and video retrieval. A very brief description of techniques is given for shot boundary detection, object detection and tracking, and text detection in video. Yoshitaka and Ichikawa's survey [190] is focused on retrieval methods for multimedia content. For video retrieval, some query-by-example approaches are presented which are based on object motion and spatio-temporal relations. However, these surveys do not address semantic video retrieval approaches of the current generation but rather discuss approaches which extract some low-level (color and texture) or mid-level features (text detection, object motion) for the purpose of retrieval. The principles of recent approaches to semantic video retrieval are briefly summarized below.

As explained in a previous section, current research efforts address the detection of high-level features (semantic features, concepts) which is also expressed by the high-level feature detection task at TRECVID. These high-level features are expected to serve as a basis to answer textual user queries at a conceptual level. For example, Snoek et al. [149] present an interactive lexicon-based video retrieval system which is based on learned concepts. It should become obvious at this point how closely the processes of analysis, indexing and retrieval are connected. It is expected that a certain number of high-level features is needed to achieve good retrieval results. To investigate this question, Hauptmann et al. [71] conducted a number of simulation experiments with respect to the number of annoted visual concepts assuming a detection accuracy of today's systems. They conclude that a number of 1000-3000 concepts is required to achieve retrieval results comparable to web search engines. Most recent research efforts address the development of generic video concept detection systems [5, 147, 148] since it might be very hard and in most cases impossible to build a number of 100 or 1000 different and specific concept detectors. A generic concept detector uses a carefully defined set of audio-visual and textual features and maps these low-level features to high-level features via supervised learning. Furthermore, such a generic concept detector should have a component to decide automatically on its own which features or classification models are best suited for a given concept. The concept detection system of IBM [5] and the MediaMill system [147, 148] are examples of such generic systems.

A query itself can be formulated as a text (concept or topic search) or the user submits a visual example to the retrieval system (query by example, query by sketch, etc.). Three examples for textual queries, part of the TRECVID search task 2006, are as follows:

1. "Find shots with one or more people leaving or entering a vehicle";
2. "Find shots of one or more people seated at a computer with display visible";
3. "Find shots of a natural scene with, for example, fields, trees, sky, lake, mountain, rocks, rivers, beach, ocean, grass, sunset, waterfall, animals, or people; but no buildings, no roads, no vehicles".

Once such a query is submitted to the system, the textual query can be matched with the textual annotations available in the video database. These annotations can be the result of automatic speech recognition, video OCR, automatic high-level feature (concept) detection, and manual annotations. These annotations are generated in the described video content analysis step.

In case of a visual query, the query example must be preprocessed and transformed to a feature vector representation which is comparable to the instances in the video database. Then, a similarity search can be conducted and the most similar video shots can be retrieved and presented to the user. In case of a textual query, the query terms must be analyzed and transformed to a form that can be processed by the system. Normally, the video database is not annotated with respect to any possible term or concept (high-level features). There are several possibilites to address this issue. One possibility is to employ an ontology or a lexicon like Wordnet (<http://wordnet.princeton.edu>) in order to find interrelationships between the query concepts and the concept annotations available for the video content. Another possibility is to provide an interactive system allowing the user to mark those shots which are relevant to him/her (relevance feedback). Then, the system might refine its internal model of the user query and return an improved result set of shots. Of course, this interactive process can be repeated until the user's need is satisfied (or the user gives up in case the results are not satisfactory). Finally, there are systems which allow the user to browse the video database via different browser interfaces. For example, several visualization techniques and browsing approaches (cross browser, grid browser) are employed in the MediaMill search engine [147] to enhance the user's search. Interactive retrieval is based on a set of pre-defined concepts as well.

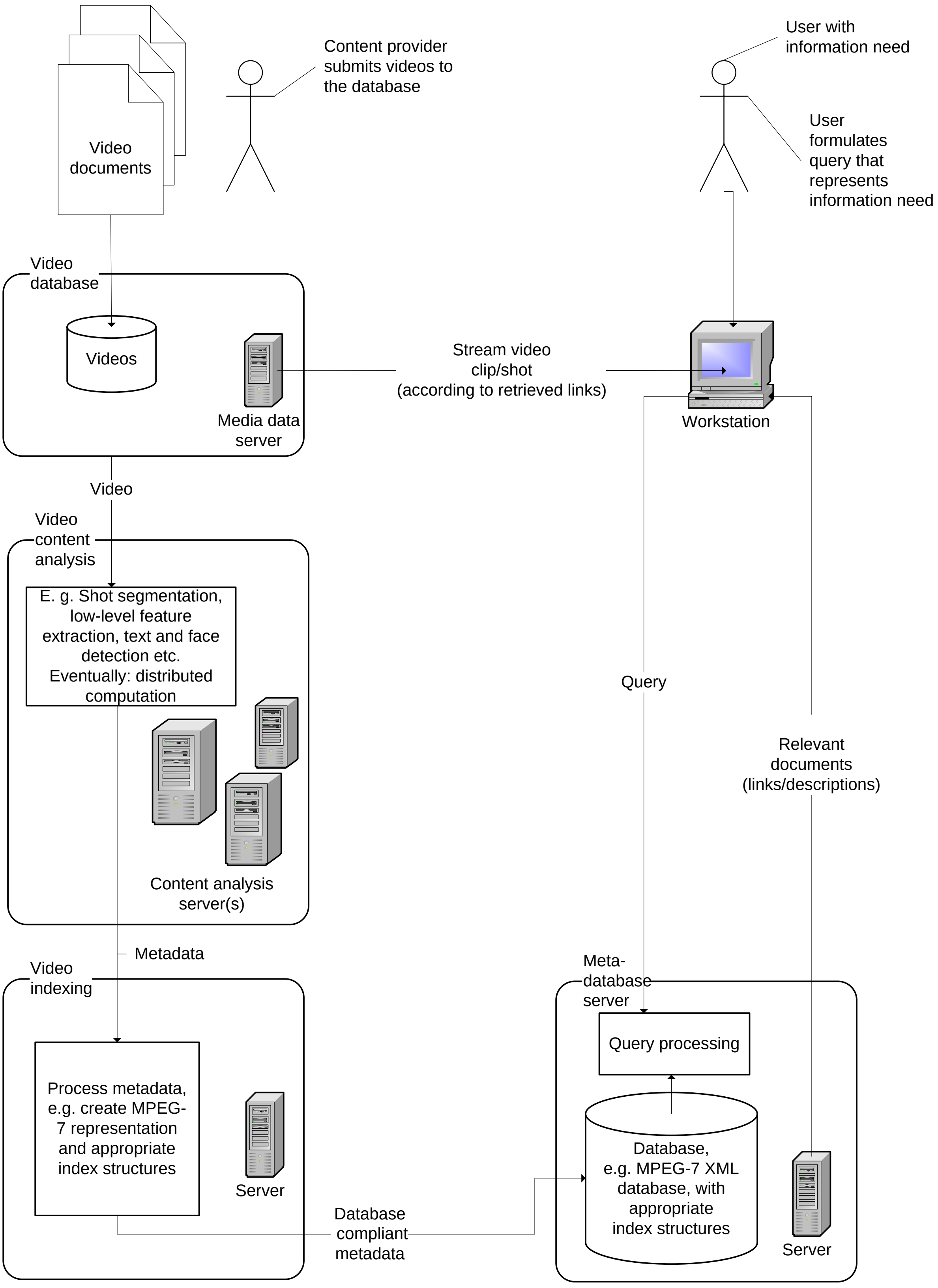

Video documents
Content provider submits videos to the database
User with information need
User formulates query that represents information need
Video database
Videos
Media data server
Stream video clip/shot (according to retrieved links)
Workstation
Video
Video content analysis
E. g. Shot segmentation, low-level feature extraction, text and face detection etc. Eventually: distributed computation
Content analysis server(s)
Query
Relevant documents (links/descriptions)
Metadata
Video indexing
Process metadata, e.g. create MPEG-7 representation and appropriate index structures
Server
Meta-database server
Query processing
Database, e.g. MPEG-7 XML database, with appropriate index structures
Server
Database compliant metadata

Figure 3: Schema for content-based video retrieval.

In Figure 3, the whole process of content-based video retrieval is displayed, including the processes of video content analysis and video indexing.

2.2.6 Performance Measures for Video Indexing and Retrieval

Performance measures indicate how well a video indexing or retrieval system works. In this thesis, the measures of recall, precision, f-measure and average precision will be used. For a detection task like video indexing, recall, precision, and f-measure are defined as follows.

Let A be the set of all observable events and let X be the set of events of a particular class of interest, $X \subseteq A$. Let D be set of events for which a detection system decided that they belong to class X, and let $C \subseteq D$ be the set of events for which this decision is correct, and let $F \subseteq D$ be the set of events for which this decision is false. Then, *recall*, *precision* and *f1* are computed as follows:

$$recall = \frac{|C|}{|X|} \tag{1}$$

$$precision = \frac{|C|}{|D|} \tag{2}$$

$$f1 = \frac{2 \cdot recall \cdot precision}{recall + precision} \tag{3}$$

In case of a retrieval task, a system usually returns a ranked list of documents (for example, a document might be a text file or a video shot), and the ranking reflects the system's confidence that the document is relevant for the user. Let then be A be the set of all documents in the database let R be the set of relevant documents, $R \subseteq A$. Then, for the whole ranking, *average precision* is defined as:

$$avg\_precision = \frac{1}{|R|} \sum_{k=1}^{|A|} \frac{|R \cap L^k|}{k} \psi(l_k)\,, \tag{4}$$

where $L^k = \{l_1, ..., l_k\}$is the subset of the k responses which are the most similar responses in A with respect to a confidence score, and $\psi(l_k)$ is a function which evaluates to 1, if $l_k \subseteq$ R, and otherwise to 0. Often, the average precision is not computed for the whole set but for the top-100 or top-2000 retrieved documents. In this case, $|A|$ must be replaced in formula (4) by the considered retrieval depth n.

The range of these measures is [0, 1]. Please note that, for reader's convenience, recall, precision and average precision are presented as percentage throughout this thesis, and f1 numbers are presented accordingly (i.e., always multiplied with 100, e.g., 72.9 instead of 0.729).

2.2.7 Compressed Video Domain

The data volume of uncompressed video is enormous. Assuming a rate of 25 frames per second, a resolution of 720*576 pixels, and an usual sampling such as 4:2:2 in YCbCr color space, 20 MB per second are required to capture a video and 1.2 GB for a minute, respectively. Obviously, it is reasonable to compress video data for storage and network transmission purposes. While it is necessary to decode the compressed bit stream to display the single frames, some video content analysis approaches work directly on the compressed data to save computation time. In this thesis, some proposed approaches work directly on compressed data as well. Hence, some basic concepts related to video compression are introduced now. These concepts are explained for the MPEG-1 standard since most of the basic compression ideas are incorporated here and most video analysis approaches use videos encoded with MPEG-1.

2.2.7.1 MPEG COMPRESSION AND METADATA STANDARDS

MPEG-1 is an ISO (International Organization for Standardization) standard (ISO standard 11172 [111]) for video compression and was finished in 1993. It consists of five parts, 11172-1 to 11172-5: System, Video, Audio, Compliance Test, and a technical report about software implementations. The development of the MPEG-1 standard is aimed at realizing playback of video from a CD-ROM.

The subsequent standard MPEG-2 (ISO standard 13818) [112] was finalized at the end of 1994 and consists of ten parts, 13818-1 to 13818-10: System, Video, Audio, Conformance, Software, Digital Storage Media - Command and Control, Non Backward Compatible Audio, 10-Bit Video, Real Time Interface und Digital Storage Media - Command and Control Conformance. Several profiles are defined in this standard that support applications like High-Definition Television (TV), satellite TV, non linear editing, and video on demand applications.

MPEG-4 (ISO standard 14996) [113] was originally targeted at low bit rate video coding, but during the development of this standard other application scenarios were targeted, for example for user interaction and mobile computing. In contrast to its predecessors, MPEG-4 follows an object-oriented approach and supports not only the encoding of video and audio but also allows the encoding of an arbitrary number of audiovisual objects. A scene description format supports the spatial and temporal composition of several audiovisual objects.

Finally, MPEG-7 [114] (finalized in 2001, ISO standard 15983) is not a standard for video compression but formalizes a set of metadata that are suited to describe multimedia content.

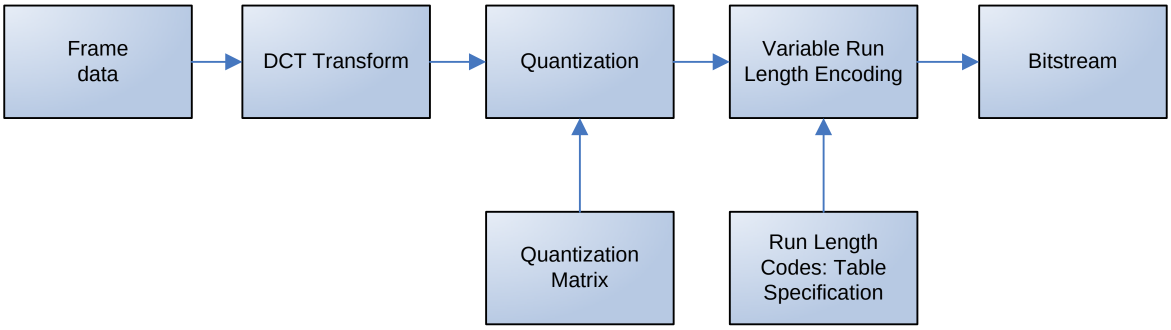

Frame data
DCT Transform
Quantization
Variable Run Length Encoding
Bitstream
Quantization Matrix
Run Length Codes: Table Specification

Figure 4: Scheme of the encoding process for I-frames according to the MPEG-1 standard. This coding scheme is similar for JPEG image encoding.

2.2.7.2 MPEG ENCODING

Since the compression principles of MPEG-1 are utilized in MPEG-2 and MPEG-4 as well and most of video content analysis test sets consist of MPEG-1 videos, the following description is based on the MPEG-1 standard. Several concepts are important in MPEG encoding terminology and are explained below.

Block: A block holds the data of one color channel and is of size 8*8 pixel.

Macroblock: A macroblock has a size of 16*16 pixels and consists of four blocks.

Slice: A slice holds a set of macroblocks.

Frame: A frame consists of slices.

Group of Pictures (GoP): A group of pictures consists of several consecutive frames.

Sequence: A number of GoPs forms an MPEG-1 video sequence.

An MPEG-1 sequence is encoded in a hierarchical manner, each of the bitstream elements (except for a block) explained above starts with a header that holds some information about the element. In MPEG-1, different frame types are defined that serve for several purposes: I-frames, P-frames and B-frames. Since current video standards use frame rates of about 25 up to 30 frames per second, it is reasonable to exploit the redundancy of temporally neighbored frames which are very similar in many cases. Motion estimation (encoding process) and compensation (decoding) try to exploit the camera or object movements in order to encode only the displacement of pixel blocks from frame to frame (and not the pixel block themselves). Several frame types are distinguished in MPEG video sequences:

I-frames (Intra Coding): These frames do not use any reference frames, they are encoded independently of other frames. The coding principle is similar to JPEG (Joint Picture Expert Group) encoding. To encode an I-frame, a video frame is divided into smaller pixel blocks of size 8 by 8 which are transformed using the Discrete Cosine Transform (DCT). The DCT coefficients are then quantized, and finally encoded using a variable run length coding. The basic process is displayed in Figure 4.

P-frames (Predictive Coding): The macroblocks of these frame type can be encoded using a previous I-frame or P-frame as a reference frame for motion estimation. The prediction of frame data in the reference frame is forwards and thus called also forward prediction. But macroblocks can also be intra-coded without motion estimation.

B-frames (Bidirectional Coding): The macroblocks of these frame type can use a previous *and* a subsequent I-frame or P-frame as reference frames for motion estimation. The prediction of frame data can be thus backwards as well. B-frames do not serve as reference frames. But macroblocks can also be intra-coded without motion estimation.

The motion estimation process itself is now described in more detail.

2.2.7.3 MOTION ESTIMATION

There is a lot of redundancy in a video sequence consisting of up to 30 frames per second. This redundancy is exploited in the compression task by motion estimation and compensation. Consecutive frames are normally very similar but, of course, content might change from frame to frame due to object and camera motion or editing effects. To exploit the similarity of consecutive frames, motion estimation is performed: for each macroblock of a frame to be encoded, the most similar macroblock is searched in one or two temporally neighbored reference frames. If such a best block match is identified, only the displacement vector and the difference between macroblock and best macroblock must be encoded. Coding frames with respect to reference frames is called inter-coding (P- and B-frames), if a frame is encoded independently of any other frames it is called intra-coding (I-frames).

2.2.7.4 GROUP OF PICTURES

An MPEG-1 video sequence consists of several GoPs. A group of pictures holds a number of I-frames, P-frames and B-frames which are ordered in a predefined manner. The first encoded frame in a GoP is an I-frame. If there are B-frames in a GoP, then the display order differs from the encoding order (see Figure 5) since for both encoding and decoding of a B-frame the subsequent reference frame (either I-frame or P-frame) must be encoded or decoded.

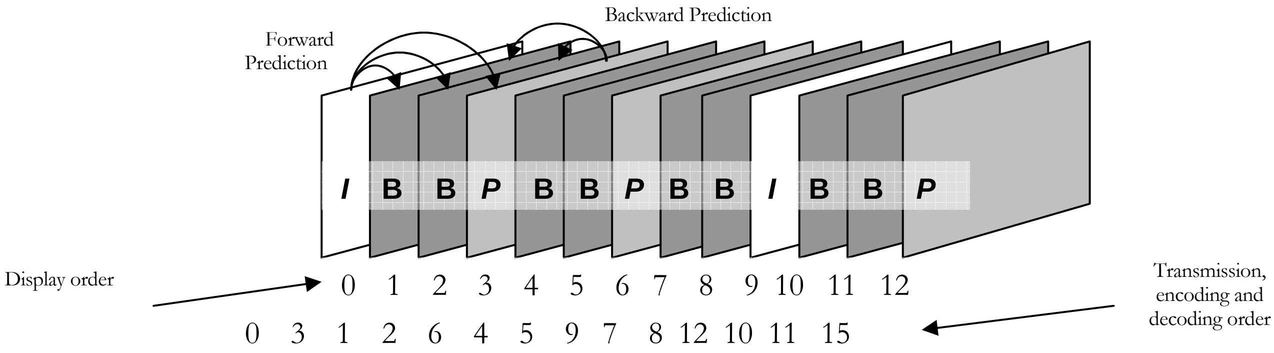


Figure 5: A typical frame ordering for a group of pictures in an MPEG-1 video. Below the encoding and the display order are displayed.

2.2.7.5 DC –FRAMES

For several processing or analysis purposes, a low-resolution version of a video is sufficient. Shen and Delp [139] as well as Yeo and Liu [188, 189] suggest approximated DC-frames for the analysis task of shot boundary detection in the compressed domain. The main idea is that the DC-coefficient of a DCT-transformed pixel block is equivalent to the average of this block, and a DC coefficient can be extracted easily from the compressed bitstream. A DC-frame is a subsampled frame that consists only of the DC-coefficients of its blocks. Obtaining a DC-frame from an I-frame is easy since the DC-coefficient is directly coded for each block. The DC term *c*(0,0) of a 8*8 pixel block is related to the pixel values *f*(*x*, *y*) at position (*x*, *y*) by:

$$c(0,0) = \frac{1}{8}\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y), \tag{5}$$

which is eight times the average intensity of the 8*8 block. A DC-frame is thus a kind of blockwise averaging of the original image. However, in predictive coded macroblocks in P-frames or B-frames, the DC-coefficients only represent the difference to the reference block in a reference frame which, however, is not aligned to a macroblock boundary in general (and hence it is not aligned to a block boundary, too). Figure 6 shows an example for a macroblock (grey) whose best block match in a reference frame is not aligned at a macroblock boundary. For such cases, motion information must be considered to obtain a DC-frame for a P-frame or a B-frame. Therefore, two versions of approximated DC are suggested by Yeo and Liu [188, 189]. The so-called zero-order approximation uses the DC coefficient of the block which has the most overlap with the current block. In the so-called first-order approximation, the contribution of the neighboring DC-values is weighted according to the overlapping of the corresponding blocks. Some properties for the first-order approximation are derived by Yeo and Liu [188, 189], e. g. that the maximum error can be ¾ of the maximum intensity value.

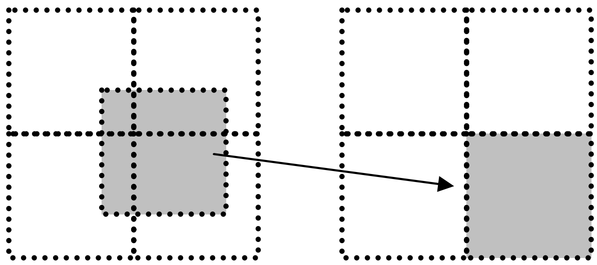


Figure 6: Example for a movement of a macroblock. Right: The macroblock that must be encoded. Left: Its best block match in a previous reference frame.

2.3 Scientific Movie Analysis

As mentioned in Chapter 1, the work presented in this thesis is also motivated by the research center "Media Upheavals" (SFB/FK615) which consists of thirteen media research projects. Some of them conduct scientific film analysis. Korte [88] describes systematic film analysis (in German: "Systematische Filmanalyse") as an approach to investigate the presentation of movies, the contextual conditions of their production in relation to the social and technological environment and the possibilities of movie perception. This process aims at objecting the movie sensation which, however, is not possible in general. In the following, some basic definitions for movie analysis are introduced. A movie is built in a hierarchical manner: a number of single frames (images) forms a shot, a scene consists of several shots that are related in space and time, and a sequence may consist of several scenes [73].

2.3.1 Shots

Korte [88] defines a shot as the smallest continuously exposed cinematic unit that, in general, consists of several frames. Arijon [6] defines a shot as an amount of film that is exposed in the camera without reloading. According to Korte [88], the following film elements (as described in the next subsections) are of special interest since they transport or express a certain meaning.

2.3.2 Cuts

A cut occurs when there is an abrupt change (abrupt) "transition" between two different subsequently shown shots. In contrast to gradual transitions, no transitional frames are involved [96].

2.3.3 Gradual Shot Transitions

The most popular transition types to connect shots are fade-in, fade-out, dissolve and wipe. A fade-in (fade-out) starts (ends) with a monochrome frame and the scene is gradually faded in (out), an example for a fade-in is given in Figure 7. A dissolve (see Figure 8) can be viewed as a combination of fade-in and fade-out without any monochrome starting (or ending) frames. The preceding shot gradually disappears while the subsequent shot gradually appears at the same time such that both shots are superimposed for several frames. If one shot enters from one side and pushes (wipes) the preceding shot out of the screen, or a thin line erases the preceding shot and reveals the new one, it is called a wipe.

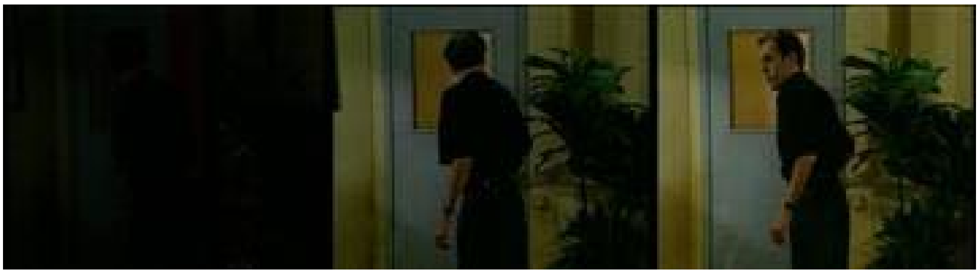


Figure 7: Example for a fade-in from a black monochrome frame. Three representative frames, taken from the MPEG-7 video riscos-sl.mpg.

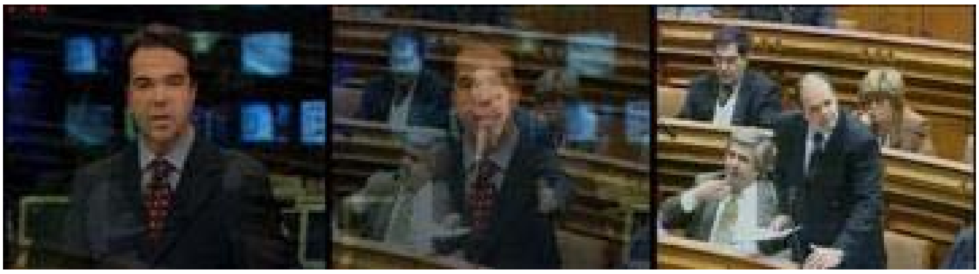


Figure 8: Example for a dissolve. Three representative frames, taken from the MPEG-7 video jornaldanoite.mpg.

2.3.4 SHOT SIZES AND CAMERA DISTANCE

Though it is not relevant for this thesis, the concepts of shot sizes and camera distance are introduced for the sake of completeness. The camera distance supplies the viewer with an impression of being far away or close to an object or a person of interest. It can be used to impose a certain atmosphere or a certain emotional reaction, for example a close-up might increase the viewer's identification with an actor and his/her emotions. The camera distance can be controlled by the choice of the lens as well. Arijon [6] distinguishes between five, Korte [88] differentiates between seven levels of camera distance: In practice, the distinction of different camera distances is not always very easy. The main camera distance types are:

1. Close-up or big close-up shot ("Groß"): Person's head or hand, facial expression or another detail is emphasized;

2. Close shot ("Nah"): Person's head and upper part of the body, facial expression is visible;

3. Medium shot ("Halbtotale"): Person's body down to the thigh;

4. Full shot ("Totale"): Person's body is completey visible;

5. Long shot ("Weit", "Super-Totale", or "Panorama): Person is in a landscape or in a big room. The background dominates the scene and details are rarely visible.

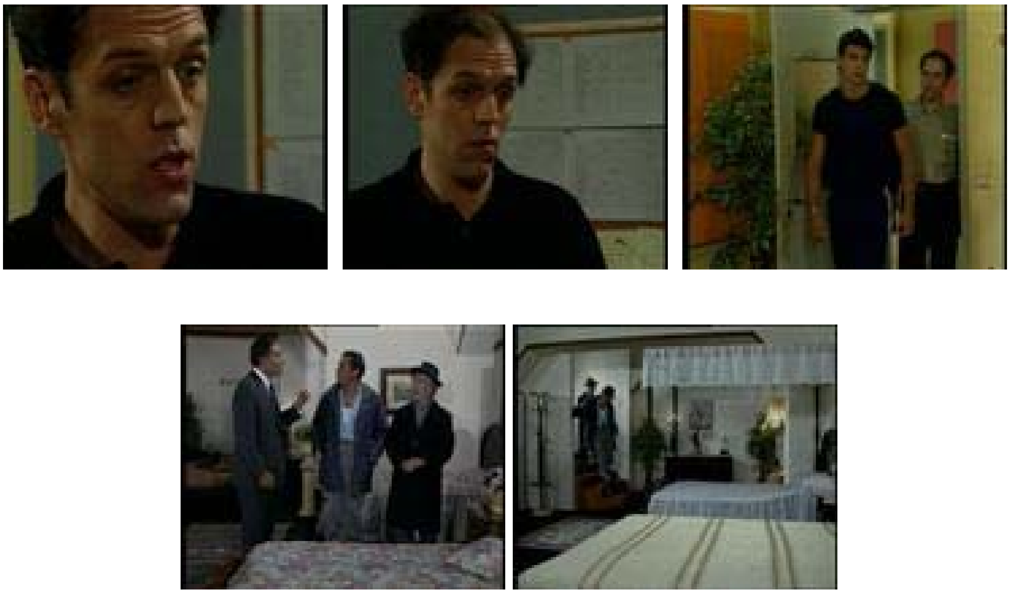


Figure 9: Examples for different shot sizes: from left to right and top to bottom: close-up shot, close shot, medium shot, full shot and long shot.

Examples for these shot sizes are displayed in Figure 9. Korte mentions two other camera distances:

1. American shot ("Amerikanische" or knee-shot): Person's body is framed from the knees up;

2. Extreme close-up ("Detail"): An extreme close-up figure singles out a detail, e.g. eyes, lips or a finger.

An approach to automatically recognize the camera distance is presented by Snoek [145].

2.3.5 Camera Movement

Two types of camera movement can be distinguished [73, 88]: Camera motion can be caused either by rotating the camera around one of the three axes or by a travelling camera which is realized via the translation along of one or more than one of the axes. Rotation around the y-axis (pan) results in a horizontal movement, rotation around the x-axis (tilt) results in vertical motion. Rotation shifts the cutting of the scene and extends the image space. A rotating camera may follow actors and their movement. A travelling camera is attached to a moving vehicle which allows the producer to control the speed of movement. The camera can move towards an object or move away from an object, this can be achieved also by zooming lens. The travelling camera is often used to keep moving characters in the frame [73]. A moving camera can be used to view the scene with the "eyes of an actor" as well. Furthermore, Hickethier [73] explains that the direction of movement is another important aspect. Recipients experience movements that are parallel to the image plane in a reserved manner, whereas movements along the viewing angle towards the viewer are experienced more aggressively and can be potentially interpreted as a threat. Apart from these possibilities to employ a moving camera many others exist (see [6, 73, 88] for further reading) but are not considered further here. Overall, it is obvious that camera movement is an important cinematic element which is utilized to express a certain meaning or to create a certain atmosphere.

2.3.6 SHOT PROTOCOLS AND SEQUENCE PROTOCOLS

Two types of transcriptions have become popular for scientific film studies [88]: shot protocols and sequence protocols. Both transcriptions transform the movie in a linear form according to its content and time structure. A shot protocol is a more detailed and comprehensive transcription since it is based on the smallest movie units, the shots. The elements of a shot protocol are as follows:

1. numbering of the shots;
2. duration of each shot;
3. camera activity: shot size, camera movement and perspective etc.;
4. description of visual shot content and plot;
5. description of audio track: dialogs, comments, sound, music, background noise etc.

A less detailed annotation form is that of a sequence protocol. Therefore, shots must be summarized to broader film units: sequences (or scenes). Then, for each sequence of interest, several content descriptions might be provided along with their time of appearance in the movie.

2.3.7 VISUALIZATIONS FOR MOVIE ANALYSIS

Korte [88] describes several visualization techniques for movie analysis: sequence diagrams, shot diagrams and cut frequency diagrams. A sequence diagram is a visualization of the sequence protocol where subsequent sequences are displayed horizontally or vertically. Sequences are visualized by boxes and their size is related to the sequence duration. Furthermore, sequences can be divided in sub-sequences and both sequences and sub-sequences are annotated with a title. Accordingly, the shot duration serves as the key element for a shot diagram where subsequent shots are displayed horizontally. Each cut is represented by a vertical line, and thus the structure of shot composition, the rhythm of shot composition and cut frequency are clearly recognizable. Often, tempo and suspense are correlated in a movie with cut frequency. Thus, the cut frequency diagram is a very popular instrument of analysis where the number of cuts per time unit is displayed for the whole movie. An example of a cut frequency diagram is displayed in Figure 10.

2.3.8 VIDEANA: A SOFTWARE TOOL FOR SCIENTIFIC FILM STUDIES

In the context of the research project "Methods and tools for media scientific analysis", the software *Videana* [45] has been developed. This software is aimed at relieving media scientists from the very time-consuming task of manually annotating videos and films. For this purpose, several automatic video content analysis algorithms are integrated in *Videana*: shot boundary detection, camera motion estimation, text detection and face detection. The integrated algorithms for shot boundary detection and camera motion are presented in this thesis. In addition, the graphical user interface of *Videana* allows a user to play videos, access particular frames, and the user is allowed to refine and correct automatic analysis results. Finally, it is possible to insert keywords for certain objects and events. For a more detailed description of *Videana* and its functionality, the reader is refered to [45].

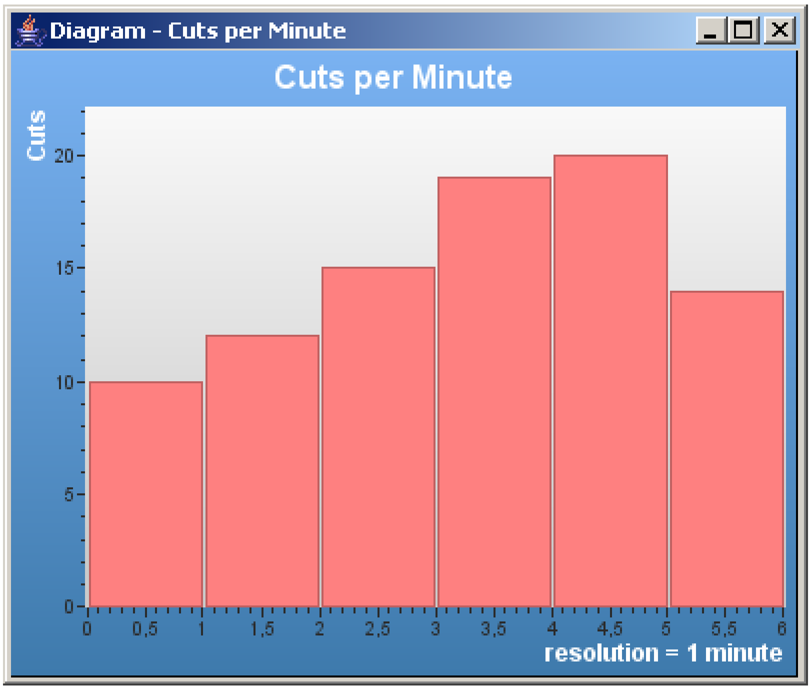


Figure 10: Example for a cut frequency diagram for a short music video clip, generated with *Videana*.

2.4 USED MACHINE LEARNING AND OPTIMIZATION METHODS

In the subsequent sections, some machine learning and optimization techniques are briefly introduced. These techniques are either used in approaches proposed in this thesis later on or they are closely related to them.

Machine learning techniques have been successfully applied to pattern recognition tasks for many years [79]. This is also true for the field of multimedia analysis, computer vision and multimedia indexing and retrieval. Possible concepts or definitions of the term "learning" in the context of machine learning are not discussed here (for example, a starting point of such a discussion is given by Witten et al. [175]). Typically, machine learning techniques fall either into the classes of supervised learning or unsupervised learning. In supervised learning, a set of labeled training data is used to find a mapping between example data x and a target y. Let $X = \{x_1, ..., x_n\}$be a set of n examples, where $x_i \in R^d$, $1 \leq i \leq n$. Furthermore, let $Y=\{y_1, ..., y_n\}$ be the labels of the examples, where either $y_i \in R^d$ (regression) or $y_i \in L$ (classification), where L is a finite set of (discrete) labels. Then, the goal of supervised learning is to find a mapping from x to y, given the training set of pairs (x_i, y_i). In case of unsupervised learning there are no labels y_i. One possible goal of unsupervised learning is to find an interesting structure in the data X.

Often, labeling of training data is a very time-consuming and expensive task. On the other hand, a lot of unlabeled data is often available which is not used for supervised learning due to an unfeasible labeling effort. In many cases, semi-supervised learning is aimed at these scenarios. This kind of machine learning approach lies between the classes of supervised and unsupervised learning. In a semi-supervised setting, the training set consists of both labeled and unlabeled data: one part of the training data $X_L=\{x_1, ..., x_l\}$has related labels $Y=\{y_1, ..., y_l\}$, whereas no labels are available for the training examples $X_U=\{x_{l+1}, ..., x_n\}$of the other part. This can be considered as the standard setting of semi-supervised learning. In most applications, the goal is to find a mapping between the training examples and the labels by exploiting the additional information available through the unlabeled data.

Transductive learning was introduced in the mid-1970s (according to Vapnik, [166]) and it is closely related to semi-supervised learning. Given a (labeled) training set and an unlabeled test set, the idea of transductive learning is to find predictions only for the test instances - and *not* to infer a general decision rule which might be optimal for the population of all possible instances – in particular, the predictions for the given test instances based on transduction are expected to be more precise than predictions based on a general rule. It follows Vapnik's principle [166]: "When trying to solve some problem, one should not solve a more difficult problem as an intermediate step".

Another approach to employ unlabeled data into the learning process is self-learning, also called self-training or self-labeling [136]. It starts with an initial classifier which is trained on the labeled data only. Then, this classifier is used to label a part of the unlabeled examples. In the next step, the additional labeled data points are also used to re-train the classifier. This process of labeling and re-training can be repeated for several learning rounds. For example, Nigam et al. [120] present an approach for text classification using a self-training approach based on a Naïve Bayes classifier and EM-clustering to employ unlabeled data.

These methods which are used in this thesis are described below.

2.4.1 Unsupervised Learning

Unsupervised learning or clustering methods are aimed at partitioning data objects into groups or clusters in a way that the members of one cluster are very similar to one another and objects of different clusters are dissimilar.

The class of clustering algorithms is divided into partitioning and hierarchical clustering methods. Partitioning methods divide a dataset into k clusters, where each cluster contains at least one object and each object belongs to exactly one cluster. To cluster data objects, a reasonable measure for the similarity of data objects is required: the distance function. The interpretation of distance values is usually that a low distance value indicates that two objects are similar, whereas a high distance value indicates that two objects are dissimilar. A reasonable distance function *d*(*o*, *p*) must fulfil at least three properties for all objects *o* and *p* of a set of objects *O*:

1. $d(o,p) \geq 0$ (6)

2. $d(o,p) = 0 \Leftrightarrow o = p$ (7)

3. $d(o,p) = d(p,o)$ (8)

A distance function is a metric, if, in addition, the triangle inequality holds, that is for all objects $o, p, q \in O$:

$$d(o,q) \leq d(o,p) + d(p,q) \tag{9}$$

For data objects with numerical attribute values, the Euclidean distance and the Manhattan distance are well-known distance functions and used frequently. Let $x=(x_1, ..., x_d)$ and $y=(y_1, ..., y_d) \in R^d$ be numerical data objects, then the *Euclidean distance* is defined as

$$d(x,y)=\sqrt{(x_1-y_1)^2+\ldots+(x_n-y_n)^2}\ , \tag{10}$$

and the *Manhattan distance* is defined as:

$$d(x,y)=|x_1-y_1|+\ldots+|x_n-y_n| \tag{11}$$

2.4.1.1 K-MEANS ALGORITHM

Let objects be points in a d-dimensional Euclidean vector space and let the Euclidean distance be the similarity measure. A centroid of a cluster *C* is the mean point of all data points in the cluster *C*:

$$\mu_C=(\overline{x_1}(C),\overline{x_2}(C),\ldots,\overline{x_d}(C))\text{, where: }\overline{x_j}(C)=\frac{1}{n_C}\cdot\sum_{p\in C}x_j^p \tag{12, 13}$$

is the mean value of the j^{th} dimension of all points in *C*, n_C is the number of elements of *C*.

K-means is a very popular clustering method suggested by MacQueen [105] for the first time. The clustering is aimed at minimizing the variance for all clusters, i. e. minimizing the term:

$$Var(Clustering)=\sum_{i=1}^{k}Var(C_i)\text{, where: }Var(C)=\sum_{p\in C}d(p,\mu_C)^2 \tag{14, 15}$$

Given a set of feature vectors and a similarity metric, the k-means algorithm assigns a feature vector to the cluster whose mean vector has the minimum distance to the feature vector. The pseudo code for k-means clustering is presented in Figure 11. The update of the cluster centroids due to re-assignment of a data object x_j^p from cluster C_1 to another cluster C_2 can be computed incrementally by:

$$\overline{x_j}(C_1^{'})=\frac{1}{n_C-1}\cdot(n_{C_1}\cdot\overline{x_j}(C_1)-x_j^p)\text{, and }\overline{x_j}(C_2^{'})=\frac{1}{n_C+1}\cdot(n_{C_2}\cdot\overline{x_j}(C_2)+x_j^p)\text{, respectively,} \tag{16}$$

where C_1'is the updated set $C_1'=C_1\setminus\{x^p\}$ and $C_2'=C_2\cup\{x^p\}$.

```
Input:  Set of feature vectors F;
        Number of clusters k;
Output: Set C consisting of k clusters;

Algorithm

K-means-Clustering(Feature Vector Set F, Integer k)
  Choose randomly k feature vectors $f_i$ from F as the centroid $\mu_i$ of
  cluster i from the set C={$C_1$, $C_2$,...,$C_k$};
  Remove these $f_i$ from F;

  while F is not empty
    Assign a feature vector f from F to the cluster $C_i$ with the
    nearest centroid;
    Remove f from F;
    Update the centroid for each cluster $C_i$;

  newAssignments = true;
  while (newAssignments==true)
    newAssignments = false;
    for each cluster $C_i$
      for each f $\in$ $C_i$
        if there is another cluster m with a nearer centroid than i then
          Assign f to the cluster m with the nearest centroid;
          Update incrementally centroids of cluster i and m;
          newAssignments = true;

  return C;
```

Figure 11: Pseudo code for k-means clustering.

According to [35], k-means has the following properties:

- it converges towards a (local) minimum;
- it has the complexity of O(*n*) for one iteration;
- result and runtime behavior depend strongly on the initial partitioning;
- minimization of the variance is sensitive to outliers.

2.4.1.2 CLUSTERING BY EXPECTATION MAXIMIZATION (EM)

Dempster et al. [32] suggest the EM clustering method in which a cluster is not described by representative points but with a probability distribution. Typically, a Gaussian distribution is used to describe a cluster in EM clustering. A set of *k* clusters is approximated by a mixture of *k* Gaussian distributions. A d-dimensional Gaussian distribution of a cluster C is given by:

- mean value of all cluster points: $\mu(C)$;

- *d*d* covariance matrix for the cluster points: *Σ(C)*.

The probability that a data point *x* is part of the data set *C* is given by:

$$P(x\,|\,C) = \frac{1}{\sqrt{(2\pi)^d\,|\,\Sigma(C)\,|}}\,e^{-\frac{1}{2}(x-\mu(C)^T(\Sigma(C))^{-1}(x-\mu(C))} \tag{17}$$

```
Input:  Set of feature vectors F;
        Number of clusters k;
Output: Set M, consisting of k Gaussian distributions for F;

Algorithm

EM-Clustering(Feature Vector Set F, Integer k)
  Create initial model M={$C_1$, $C_2$,...,$C_k$} of Gaussian distributions for F;

  do
    M':=M;

    // step 1 Expectation
    Compute the probabilities P(x|$C_i$), P(x), and P($C_i$|x) for each feature
    vector from F and each cluster;

    // step 2 Maximization
    Compute a new model M = {$C_1$, $C_2$,...,$C_k$} of Gaussians by estimating
    the parameters μ($C_i$)), Σ($C_i$), and $W_i$ for each i=1,...,k.
  while |E(M)-E(M')| > ε

  return M;
```

Figure 12: Pseudo code for EM clustering.

The combined effect of *k* Gaussians can be computed by:

$$P(x) = \sum_{i=1}^{k} W_i P(x\,|\,C_i) \tag{18}$$

The weight W_i is the ratio of the data set that belongs to cluster C_i, each Gaussian cluster C_i is described by the parameters $\mu(C_i)$ and $\Sigma(C_i)$. In contrast to k-means clustering, data objects can belong to several clusters at the same time with different probabilities. The probability for a data object to belong to the cluster C_i is:

$$P(C_i\,|\,x) = W_i\frac{P(x\,|\,C_i)}{P(x)} \tag{19}$$

The goodness of the clustering is measured by:

$$E(M) = \sum_{x \in D} \log(P(x)), \tag{20}$$

where M is the set of k Gaussian distributions. The higher $E(M)$ is, the more probable is that the data have been generated under the assumption of the k Gaussian distributions. During the iterations of expectation and maximization the term $E(M)$ is maximized by estimating the parameters of the k Gaussian distributions (please see also the pseudo code in Figure 11). The parameters $\mu(C_{i)})$, $\Sigma(C_i)$, and W_i are re-estimated by:

$$W_i = \frac{1}{n} \sum_{x \notin D} P(C_i \mid x), \tag{21}$$

$$\mu(i) = \frac{\sum_{x \in D} x \cdot P(C_i \mid x)}{\sum_{x \in D} P(C_i \mid x)}, \tag{22}$$

$$\Sigma(i) = \frac{\sum_{x \in D} P(C_i \mid x)(x - \mu(i))(x - \mu(i))^T}{\sum_{x \in D} P(C_i | x)}. \tag{23}$$

According to [35], EM clustering has the following properties:

- it converges against a (local) minimum;
- in general, the number of iterations is rather high;
- complexity of one iteration is O(n* | M |);
- result and runtime behavior depend strongly on the initial assignment.

2.4.1.3 HIERARCHICAL CLUSTERING

There are two kinds of hierarchical clustering approaches: agglomerative clustering and divisive clustering. Divisive clustering methods are top-down approaches which consider the whole set of data in the beginning as one cluster. Then, this cluster is divided into a number of clusters. Agglomerative approaches follow a bottom-up princicple. At first, *each object* is considered as one cluster. Then, the two most similar (object) clusters are merged iteratively and become a new cluster. This process can be repeated until there is only one cluster left. In contrast to partitioning clustering approaches, hierarchical approaches generate a dendrogram which represents the cluster structure of the data. A dendrogram is a tree and an inner node represents the cluster containing the objects of all its child nodes. A distance measure is needed to calculate the (dis-)similarity of two sets (*X* and *Y*) of objects. Common measures therefore are single-linkage, complete linkage, and average linkage [35]. Let *X* and *Y* be sets of objects, $x \in X$ and $y \in Y$, and let *dist*(*x*, *y*) be a distance measure for two objects *x* and *y*. Then, the following distance measures *dist_sl*, *dist_cl*, and *dist_al* can be defined for the three different linkage measures with respect to two sets of objects *X* and *Y*:

$$dist\_sl(X,Y) = \min_{x \in X, y \in Y} (dist(x,y)) \tag{24}$$

$$dist\_cl(X,Y) = \max_{x \in X, y \in Y} (dist(x,y)) \tag{25}$$

$$dist\_al = \frac{1}{|X||Y|} \cdot \sum_{x \in X, y \in Y} dist(x,y) \tag{26}$$

Based on one of these distance measures, the agglomerative algorithm can be formulated as depicted in Figure 13.

```
Input:  Set F of n feature vectors representing n the data objects;
Output: A dendrogram D, represented by its graph G = (V, E);

Algorithm

Agglomerative-Clustering(Feature Vector Set F)
  Compute distances between all pairs of feature vectors;
  Consider each feature vector $f_i$ as one cluster and assign it to $C_i$;
  V = {$C_1$, ..., $C_n$};
  E = ∅;
  k = n+1;
  while number of clusters > 1 do
    Build a new cluster (node) $C_k$ by merging the 2 most similar
    clusters nodes $C_i$ and $C_j$;
    V = V ∪ {$C_k$};
    E = E ∪ {($C_i$,$C_k$),($C_j$,$C_k$)};
    Compute the distances of the new cluster $C_k$ to all the other clusters;
    k = k + 1;
  return G;
```

Figure 13: Pseudo code for agglomerative clustering.

This algorithm for hierarchical clustering has the following properties:

- complexity is $O(n^2)$
- in case of single-linkage, the following effect might occur: it cannot be distinguished between two clusters if these clusters are connected via a “line” of objects whose distances are similar to distances within a cluster.

2.4.1.4 CLUSTERING VALIDITY

Measuring the clustering quality is important in order to obtain information about the success in clustering the data. Measures for cluster quality or cluster validity can be utilized to find the best k for k-means clustering which is often unknown in advance: K-means clustering is therefore run separately for several k in a reasonable range, and the clustering that yields the highest validity measure is considered as the final clustering result. To be applicable, such a measure must be independent of the number of clusters. Kaufmann and Rousseeuw [84] suggested the silhouette coefficient to measure the clustering validity. Let $C_M=\{C_1, C_2, ..., C_k\}$ be the set of clusters for a set of objects $o \in O$, $C \in C_M$, $o \in C$, then the average distance of an object o to an arbitrary cluster C_i is defined as:

$$dist(o,C_i)=\left(\sum_{p\in C_i} dist(o,p)\right)/|C_i| \tag{27}$$

Let $a(o) = dist(o, C)$ be the average distance of object o to its cluster and let $b(o)$ the average distance of object o to its neighbor cluster:

$$b(o)=\min_{C_i\in C_M, C_i\neq C} dist(o,C_i) \tag{28}$$

The silhouette $s(o)$ for an object $o \in C$ is then defined as:

$$s(o)=\left\{\frac{b(o)-a(o)}{\max\{a(o),b(o)\}}\right., \text{ if } |C|>1\text{; else } s(o)=0. \tag{29}$$

The silhouette coefficient $SC(C)$ of cluster C is the average of the silhouette coefficients $s(o)$ of all objects o in C.

2.4.2 SUPERVISED LEARNING

In contrast to unsupervised learning or clustering, the object classes of the training data must be known in advance for supervised learning or classification. The classification task is to assign a data object to a certain class based on its attribute (feature) values. The prerequisite is the existence of a training set for each class that allows us to learn an appropriate classification function. It is distinguished between generative algorithms and discriminative algorithms for supervised learning. Generative algorithms try to model the class-conditional density *p(x|y)* and to infer from that the predictive density *p(y|x)* by applying Bayes theorem. Discriminative algorithms do not estimate the class-conditional density but instead concentrate on estimating *p(y|x)*. Many classification methods have been suggested in the literature, for example Naïve Bayes-classification, k-nearest neighbour classification, decision trees, Support Vector Machine (SVM), Hidden Markov Models, and neural networks belong to the most popular supervised learning methods for pattern recognition tasks [79]. A brief introduction is given below for SVM since it is used later on in this thesis.

2.4.2.1 SUPPORT VECTOR MACHINE (SVM)

Given a set of m training data examples x_i, $i \in \{1, ..., m\}$, each of dimension n, that is $x_i = (x_{i1}, ..., x_{in})$, as well as a set of discrete labels y, for example where $y_i \in \{-1, 1\}$, then a SVM aims to find a maximum margin hyperplane that separates both classes, that is the hyperplane that maximizes the margin between the two classes of training examples. A hyperplane separating two classes in an n-dimensional space can be written as:

$$x = b + \sum_{j=1}^{n} w_j a_j \ , \tag{30}$$

where a_j are the attributes and b and w_j are the weights to be learned. The maximum margin hyperplane gives the maximum separation between the classes and it does not come closer to either than it has to (see Figure 14 for an example). The hyperplane depends only on the support vectors which are the data examples of each class with the minimum distance to the hyperplane. The other data objects are irrelevant for the construction of the hyperplane – it is uniquely defined by the set of support vectors. The maximum margin hyperplane can be written in another form based on the support vectors. Let y_i be the class value for a training instance x_i, then the maximum margin hyperplane x' is:

$$x' = b + \sum_{i} w_i y_i x_i \cdot x^* \ , \tag{31}$$

where *b* and w_i are numeric values which are determined by the learning algorithm, y_i is the class value for a training instance x_i, x^* is a test instance, x^* and x_i are vectors, and $x_i \cdot x^*$ represents the dot product. The determination of the hyperplane parameters *b* and w_i belongs to the class of constrained quadratic optimization techniques.

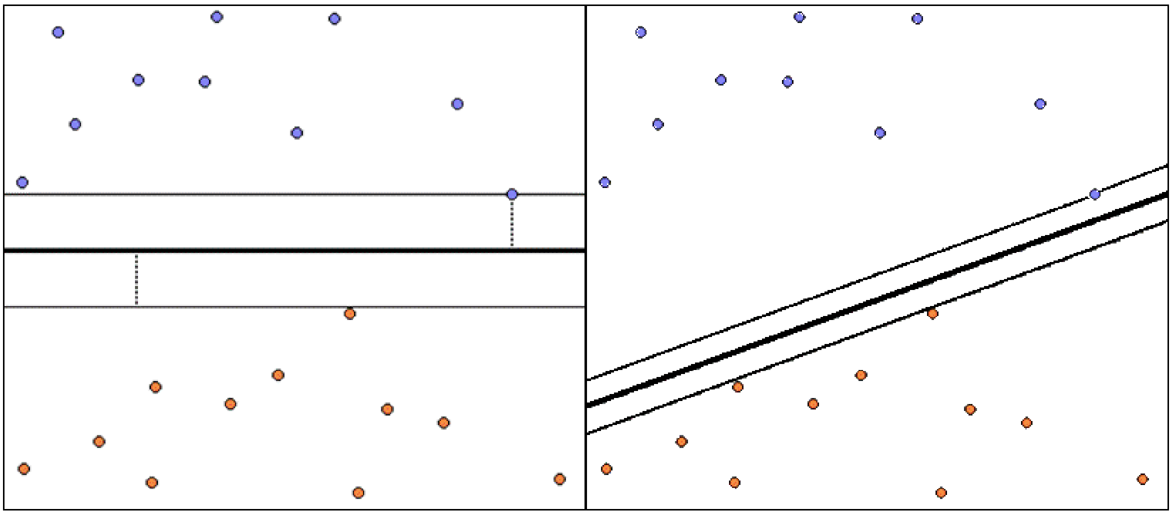


Figure 14: An example for a well class-separating hyperplane (left) and a non-optimal separating hyperplane (right).

The maximum margin hyperplane for the linearly separable case is computed by solving a quadratic optimization problem. The following term must be minimized over (*w*, *b*) [82]:

$$\frac{1}{2}\|w\|^2 \tag{32}$$

subject to:

$$\forall_{i=1}^{m} y_i(w \cdot x_i + b) \geq 1, \tag{33}$$

where *w* is a vector. However, in case when the classes are not linearly separable, a solution cannot be found for this optimization problem. To address this issue, Cortes and Vapnik [28] suggested the introduction of slack variables ξ_i for the non-separable linear case. Hence, the following term must be minimized over (**w**, *b*, ξ_1, ..., ξ_n) [21]:

$$\frac{1}{2}\|w\|^2 + C\sum_i \xi_i \tag{34}$$

subject to:

$$\forall_{i=1}^{m} : y_i(w \cdot x_i + b) \geq 1 - \xi_i \tag{35}$$

$$\forall_{i=1}^{m} : \xi_i \geq 0 \tag{36}$$

where *C* in (34) introduces a cost factor for errors. The larger *C* is, the higher is the penalty for errors. The variables ξ_i relax the constraint of formula (33) and allow errors in the classification of the training data.

Although SVMs belong to the class of linear separation techniques, they are also well suited to deal appropriately with non-linear class boundaries. A kernel trick is normally applied to deal with classes for which the decision function is non-linear. For example, the non-linear mapping can be accomplished in the formula for the maximum margin hyperplane by substituting the term $\boldsymbol{w{\cdot}x_i}$ by $(\boldsymbol{w{\cdot}x_i})^{n}$. This way, the dot product can be computed in the lower dimensional space before the non-linear mapping to the high-dimensional space is performed and hence, the computational burden is limited accordingly. The function $(\boldsymbol{x{\cdot}y})^{n}$ is called a polynomial kernel. Another kernel function that is often used with SVM is the radial basis function (RBF).

2.4.3 Semi-Supervised Learning

As mentioned above, semi-supervised learning approaches lie between the classes of supervised and unsupervised learning. In a semi-supervised setting, the training set consists of both labeled and unlabeled data. This can be considered as the standard setting of semi-supervised learning but other types of partial supervision are possible, for example, it might be known that some data points have the same class membership. In most applications, the goal is to find a mapping between the training examples and the labels by exploiting the additional information available through the unlabeled data. Of course, certain assumptions have to hold if the unlabeled data shall improve the classification process. One of these assumptions is the smoothness assumption: If two points x_1 and x_2 in a high-density region are close, then so should be the corresponding outputs y_1 and y_2 [25]. The *cluster assumption* claims that if points are in the same cluster, they are likely to be of the same class. This assumption can be reformulated: The decision boundary should lie in a low-density region. Some semi-supervised learning approaches are presented briefly below.

A pioneering work in the field is the co-training approach of Blum and Mitchell [13]. They suggest co-training to augment a small set of labeled data with unlabeled data. A prerequisite of co-training is the existence of two different views (i.e., two disjoint and independent feature sets) on the data. It is assumed that each of those views is sufficient to train a classifier in case that enough labeled data are available. First, each classifier is trained for its view on the labeled data. Then, class labels of a portion of unlabeled data are predicted by each classifier and the positive and negative predicted samples with the highest confidence are passed to the other learner to augment its set of training data. Blum and Mitchell report experimental results for the task of web site classification. Two independent types of features are considered here: page-based features and hyperlink-based features. The proposed co-training approach yields an error rate of 5%, whereas a combination of supervised classfiers achieves only an error rate to 11%. The pseudo code for co-training is presented in Figure 15.

Nigam et al. [120] present an approach for text classification that exploits unlabeled data using a Naïve Bayes classifier and EM clustering. An initial Naïve Bayes classifier is trained using only the labeled data. Then, using this classifier the unlabeled data are labeled probabilistically and finally EM is iterated until convergence on all document samples. Their experimental results demonstrate that this technique can significantly increase text classification accuracy when given limited amounts of labeled data and large mounts of unlabeled data. Nigam and Ghani show [121] that co-training can improve classification accuracy even when the assumption of two conditionally independent views on the data is relaxed: Co-training in conjunction with a randomly split feature set improved the baseline EM approach as well, but a self-trained approach and a co-EM algorithm outperformed baseline EM, too, and achieved similar results.

Wu [177] suggests the B-EM algorithm to incorporate a large set of unlabeled data in classifier training. Bootstrapping (forming a training set from a given data set by sampling with replacement) is used in order to obtain several training sets from unlabeled data. Experimental results are presented for synthetic test data and demonstrate that B-EM (employing unlabeled data to refine classification models) outperforms standard EM when both approaches were provided with the same number of labeled samples in the beginning (B-EM increases the number of examples by itself). In addition, experimental results are presented for a real-world image classification task based on the Corel database. Here, B-EM outperforms the baseline system as long as the number of initially labeled training samples is below 120, thus, it is reasonable to use B-EM when only a small number of training examples is available.

Rosenberg et al. [132] explore possibilities to reduce the amount of training data to train a supervised object detection system while achieving the same accuracy as a system trained with a large training set. Therefore, weakly labeled data are used: positive image examples were only labeled with the information that the object of interest is visible in the image but neither position nor size were annotated. In contrast to the approaches mentioned so far, EM is not considered to incorporate weakly labeled or unlabeled data but rather a self-training approach (comparable to Nigam and Ghani [121]). Rosenberg et al. modify the training for the object detection approach of Schneiderman and Kanade [135] which achieved top performance on standard face detection test sets [181, 187]. Weakly labeled training data for positive examples are included in their framework, that is images are labeled as positive examples when there is at least one frontal face observable but no details are given about position and size. Training data are divided into two groups: a set of fully labeled data and a larger set of weakly labeled data. First, an initial Bayesian classifier is built which is then used to assign scores to the weakly labeled samples. Two selection metrics to compute scores are investigated, the confidence selection metric and the MSE (mean squared error) selection metric. Weakly labeled data with highest scores are then added to the training set and a new face model is computed. The training set consists of 480 positive examples and 15.000 negative examples. In the experiments, they investigated how many fully positive examples are needed to achieve the same detection performance as in the case when the whole set of 480 fully labeled positive examples is used (baseline, full training). They found that performance was "saturated": self-training achieved a similar performance as in the full training setting, when the number of positive fully labeled examples was at least 120. If less than 35 positive examples were fully labeled, the performance was between 60% and 80% of the baseline setting. If more than 35 and less than 120 examples were fully labeled, the relative performance was between 80% and 95%. Finally, the MSE selection metric outperformed the confidence score metric.

```
Input:  Set L of labeled training examples;
        Set U of unlabeled examples;
Output: Two classifiers using two different views on the data;

Algorithm

Co-Training(Set L, Set U)
  Create a pool U' by choosing u examples at random from U
  for k iterations:
    Use L to train a classifier $h_1$ that considers only $x_1$ portion from x;
    Use L to train a classifier $h_2$ that considers only $x_2$ portion from x;
    Allow $h_1$ to label p positive and n negative examples from U';
    Allow $h_2$ to label p positive and n negative examples from U';
    Add these self-labeled examples to L;
    Randomly choose 2p+2n examples from U to replenish U';

  return $h_1$ and $h_2$;
```

Figure 15: Pseudo code for co-training according to [13].

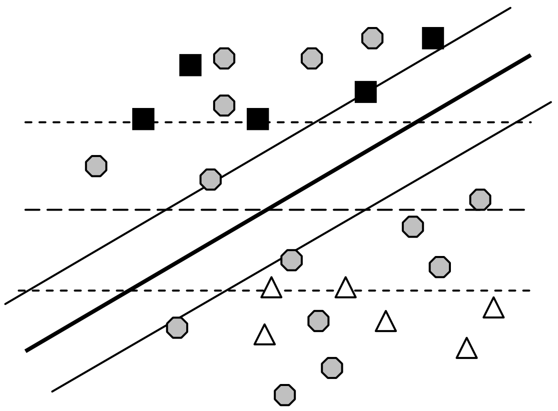


Figure 16: Example how a transductive SVM might refine and improve a maximum margin hyperplane achieved by an inductive SVM. Class 1 is represented by black rectangles, class 2 is represented by white triangles. Unlabeled data are displayed as grey circles. The SVM hyperplane is shown as a dashed line, whereas the TSVM hyperplane is displayed by a solid line.

2.4.4 Transductive Learning

In a transductive setting, there is a set of labeled training examples and a set of unlabeled test instances. So far, this is the standard supervised learning setting. But in contrast to the supervised learning setting, the unlabeled test instances are also employed in the transductive learning process and thus it belongs to the semi-supervied learning approaches as well. Transductive learning is aimed at obtaining an optimal classification for the given test instances only, whereas semi-supervised learning can be basically used for inductive classification model. Otherwise, any semi-supervised approach can be employed as a transductive learner, for example, in case when all used training instances are labeled and the employed unlabeled data are the test instances. In this section, the transductive SVM is presented which explicitly addresses a transductive setting.

A transductive SVM uses the unlabeled test samples and tries to achieve a model which is optimal for the particular test set. Therefore, the test samples are incorporated into the learning process (the process of estimating the maximum margin hyperplane). The test samples are represented by a set of k feature vectors u_j, where $j \in \{1, ..., k\}$, and the corresponding labels $y_j^* \in \{-1,+1\}$ are unknown. Although the labels of the test samples are not known, they are incorporated into the learning process in order to adjust the maximum hyperplane with respect to the test samples. An example for a maximum margin hyperplane that is derived by employing the unlabeled test samples and transductive learning is presented in Figure 16. For the linearly separable case, the following term must be minimized over (w, b, y_1^*, ..., y_k^*):

$$\frac{1}{2}\|w\|^2 \tag{37}$$

subject to:

$$\forall_{i=1}^{m} : y_i (w \cdot x_i + b) \geq 1 \tag{38}$$

$$\forall_{j=1}^{k} : y_j^* (w \cdot x_j + b) \geq 1, \tag{39}$$

whereas for the non-separable linear case slack variables are introduced, as for the inductive SVM. Hence, the following term must be minimized over (*w, b,* y_1^*, ..., y_k^*, ξ_1^*, ..., ξ_k^*) [82]:

$$\frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} \xi_i + C^* \sum_{j=1}^{k} \xi_j^* \tag{40}$$

subject to:

$$\forall_{i=1}^{m} : y_i (w \cdot x_i + b) \geq 1 \tag{41}$$

$$\forall_{j=1}^{k} : y_j^* (w \cdot x_j + b) \geq 1, \tag{42}$$

$$\forall_{i=1}^{m} : \xi_i > 0 \tag{43}$$

$$\forall_{j=1}^{k} : \xi_j^* > 0 \tag{44}$$

Training of a transductive support vector machine is achieved by solving the optimization problem of (37) or (40). Simply trying all possible assignments of y_1^*, ..., y_k^* to the two classes is only feasible for very small test sets. Joachims [82] suggests an algorithm to solve the optimization problem (40-44). It performs a local search starting from an initial labeling of the test samples computed by an inductive SVM. For this purpose, labels of test samples are swapped iteratively such that the objective function decreases. The ratio of positively and negatively labeled samples is maintained during the optimization process in order to prevent degenerated solutions in which all test samples are assigned to one class. For details regarding this algorithm, the reader is referred to [82].

2.4.5 Meta-Learning Methods and Ensemble Learning

The purpose of meta-classifiers is to build a classification model based on an ensemble of single classifiers and to combine their outputs in an optimal way. In this context, a classifier is often called an expert and its output is called a vote. The application of an ensemble scheme will only improve classification accuracy if the models are sufficiently different and each classifier achieves a minimum accuracy. An intuitive way to combine the experts is to apply a majority voting scheme [79, 89, 175]: The class label is assigned to a data object which has got the majority of all expert votes.

Adaboost (Adaptive Boosting, Freund and Schapire [55]) belongs to the class of meta-classifiers and is very popular. Other meta-classification methods besides boosting are bagging and stacking which are related to the majority voting scheme in some respect. The idea of bagging is to randomly generate a number of training sets from the original training set by sampling the training set with replacement. Then, a classification model is learned separately for each training set and finally these models are combined using majority voting (see Figure 18 and Figure 17).

Stacking or stacked generalization as suggested by Wolpert [176] is another way to combine different classification models. First, n different models are learned on different partitions of the training set, the individual classifiers. Then, on an unused part of the training set, the outputs of the n different model are utilized to learn a new (meta-)model, the stacked classifier which consists of a combination of the individual classifiers. The final decision is based on both the stacked classifier and the individual classifiers. The meta-learner can be any learning algorithm, usually a simple learning scheme like simple linear models or trees with linear models at the leaves work well [175]. The idea of boosting is based on a kind of resampling of the training set during a number of learning rounds. The resampling is controlled by assigning weights to the training samples. A higher weight is assigned to those training samples that were misclassified in a preceding training round. For the decision process, different weights are assigned to the individual classifiers dependent on their individual error in a training round. Freund and Schapire [55] showed that it is possible for a combination of weak classifiers, which are only slightly better than random guessing, to achieve an arbitrarily small error rate on the training set. In this thesis, the pseudo code presented in Figure 20 and Figure 19 is used in the implementation. Meanwhile, many variations of Adaboost have been suggested, such as Vector Adaboost [76] for the task of face detection, or Adaboost with totally corrective updates [150].

```
Input:  Number of m training instances;
        t: number of learning iterations;
        n: number of instances for sampling;
Output: Set F consisting of t classification models $f_t$;

Algorithm

Bagging-ModelGeneration()
  F=∅;
  for t iterations
    Sample n instances with replacement from the training data;
    Learn a classification model $f_t$;
    F=F∪$f_t$;
    Store the model;

  return F;
```

Figure 17: Pseudo code for bagging model generation.

```
Input:  Test instance instX;
        Set F consisting of t classification models $f_t$;
Output: Predicted class for instX;

Algorithm

Bagging-Classification()
  for each model $f_t$
    predictedClass = $f_t$(instX);

  return class that has been predicted most often;
```

Figure 18: Pseudo code for classification using bagging.

```
Input:  Set of labeled training feature vectors X;
Output: Set of classifier models F and set of related training errors E;

Algorithm

Adaboost-Model-Generation()
  F = ∅; E = ∅;
  for each instance instX ∈ X
    weight[instanceX] =  1/|X|;

  for each (iteration t)
    Apply learning algorithm to weighted training samples and save model
    $f_t$;
    Compute training error $e_t$ on weighted training set;
    if ($e_t$ == 0 || $e_t$ >= 0.5) terminate;
    F = F ∪ $f_t$; E = E ∪ $e_t$;
    for each (instance instX)
      if (instanceX classified correctly)
        weight[instX] *= $e_t$ / (1-$e_t$)
    Normalize all weights[i]; // i.e. sum is 1

return set;
```

Figure 19: Pseudo code for Adaboost model generation.

```
Input:  Set of classification models F;
        Set of training errors E;
        Instance instX;
Output: Classification label for instance;

Algorithm

Adaboost-Classification()
  for each class classX
    weight[classX] = 0;
  for each model $f_t$
    predictedClass = $f_t$(instX);
    weight[predictedClass] += -log($e_t$/(1-$e_t$));

return index of max(weight[classes]);
```

Figure 20: Pseudo code for Adaboost classification.

2.4.5.1 CREATION OF CLASSIFIER ENSEMBLES

In the previous section, three methods are presented to assemble a number of different classifiers to form an ensemble: Adaboost, bagging, and stacking. Another simple strategy is to combine a number of experts (classifiers) and to use a majority voting scheme [89] for decision making. While it is known that ensembles can outperform single classifiers on many tasks it is not known theoretically what is the best way to build an ensemble [20]. Two factors determine the accuracy of an ensemble: the accuracies of the single classifiers and the diversity of their outputs. While it has been shown for ensemble regression techniques how diversity can be quantified, there is no generally accepted measure for the diversity of classifier ensembles [20]. Kuncheva et al. [89] suggest the Q-statistics to measure the pairwise dependency of classifiers. Let f_i and f_j be two classifiers, let *a* be the the number of predictions where both classifiers predicted correctly, let *b* and *c* the number of cases in which exactly one of the classifiers misclassified the object, and let *d* be the cases when both classifiers misclassified an object. Then, $Q_{i,j}$ is defined as:

$$Q_{i,j} = \frac{ad - bc}{ad + bc} \qquad (45)$$

The Q-statistics for an ensemble is computed as the average of all possible pairings of the ensemble classifiers. It is shown empirically [89] that negative pairwise dependence is beneficial but not straightforwardly related to ensemble accuracy. Margineantu and Dietterich [107] investigated another measure for diversity: the kappa-statistics. Keeping the meaning of *a, b, c, d* from above, kappa-statistics is defined as:

$$\kappa = \frac{2(ac - bd)}{(a+b)(c+d)(a+c)(b+d)} \qquad (46)$$

The authors investigate several strategies to prune an Adaboost ensemble and showed that in some domains an ensemble can be pruned by about 60-80%. Applying the kappa-statistic resulted in one of the best pruning strategies. Furthermore, they produced kappa-error diagrams for Adaboost and bagging where each possible pair of classifiers is plotted in a 2D space (kappa and error rate). The diagrams showed that Adaboost produces a more diverse ensemble of classifiers than bagging.

Another approach is presented by Ho [74]: the training set is sampled in the feature space and models are trained on randomly chosen subspaces of the original input space. The outputs of the models can be combined by majority voting.

Brown et al. [20] survey the field of diversity creation methods for classifier combination. They categorize the approaches for ensemble diversity creation in three classes:

1. Starting point in hypothesis space: The aim is to reach different points in the solution space, including hopefully a point which is optimal in a global sense. However, Brown et al. [20] state that it is generally accepted that this type of diversity creation is the least effective method.

2. Manipulating the set of acceptable hypotheses: There are two ways of manipulating the set of hypotheses accessible to a learner. The first subclass of methods manipulates the training data and produces differing subsets of the training data which are passed to different learners so that they hopefully learn different things about the classes to separate. Training data can be resampled by either using a subset of features or a subset of training samples. The second subclass of methods aims for diversity creation by using different learning strategy in the ensemble.

3. Hypothesis space traversal: This type of methods alters the way the hypothesis space is traversed, leading different classifiers (function approximators) to different hypotheses.

2.4.6 Optimization: Downhill-Simplex Algorithm

The simplex algorithm of Nelder and Mead [118] belongs to the most popular methods to solve nonlinear minimization problems due to its simplicity [80]. It is easily applicable since first-order and second-order derivatives are not needed. Furthermore, it is also applicable when discontinuous functions appear in the problem statement. However, such a simple method comes along with some drawbacks. Normally, the method converges slowly, and it does not always converge towards a stationary point, even for strict convex problems with two variables. The method utilizes a simplex, which is a polyhedron of n+1 vertices in n dimensions. For example, in the two-dimensional case, this is a triangle on a plane. Given the unrestricted problem

$$\inf_{x \in R^n} f(x) , \tag{47}$$

then, the Nelder-Mead method considers n+1 points x_1, x_2, ..., x_{n+1} which form a simplex in R^n and can be ordered as follows according to their function values:

$$f(x_1) \leq f(x_2) \leq \ldots \leq f(x_{n+1}) \tag{48}$$

The points x_i with low values $f(x_i)$ should be considered as "good" points and those x_i with high values $f(x_i)$ as "bad" points. The main idea of the method is to move the simplex towards a good solution, that is the function value is minimized. Therefore, it is tried to iteratively find a better point for the worst point x_{n+1}. The pseudo code is depicted in Figure 21.

The algorithm consists of three main steps: reflection, expansion, and contraction. The better points x_1 to x_n define a hyperplane that splits R^n into two parts. A new point x can be found by reflecting x_{n+1} through the centroid of that hyperplane (reflection step). If this point is not again the worst point, then the process is repeated. In case that the reflected point is the worst point, then the simplex is shrunk (contraction step). If the reflected point is the best point, then the simplex is expanded: Since the new point is better than all the other points, this direction is searched to obtain a better solution.

2.5 Summary

In this chapter, the related research fields and technological fundamentals of this thesis were introduced. Mainly, this thesis contributes to the field of video content analysis, indexing and retrieval, and in this field machine learning techniques are extensively used. The research field of video content analysis, indexing and retrieval were briefly introduced in Chapter 2.2. The reader will find more detailed discussions for specific approaches in subsequent chapters where several approaches are proposed for this domain. Since the processing of compressed videos is relevant for video content analysis, the foundations of MPEG compression were presented in Chapter 2.2, too. Some principles of machine learning were described in Chapter 2.4. as well as those machine learning and optimization techniques which are used in the proposed approaches later on. Furthermore, since this thesis is also motivated by scientific film studies in the research center "Media Upheavals", some basic concepts of scientific film studies were introduced in section 2.3.

```
Input:  Function f;
Output: Point x that minimizes f;

Algorithm

Nelder-Mead-Simplex-Method()
  Choose n+1 affine independent points $x_1,…,x_{n+1} \in R^n$;
  Estimate function values $f(x_1)$, ..., $f(x_{n+1})$ and sort $x_i$ such that
  $f(x_1) \le f(x_2) \le ... \le f(x_{n+1})$;
  Choose $a > 0$, and $b > \max\{1,a\}$, and $d \in (0,1)$;

  for k = 1 to z
    Compute the centroid c of $\{x_1,...,x_n\}$;
    $x_r$= c + a $(c-x_{n+1})$; // reflected point

    // Distinguish the three cases of reflection, expansion, and
    // contraction

    if $f(x_1) \le f(x_r) \le ... \le f(x_n)$ // reflection step
      Replace $x_{n+1}$ with $x_r$ and sort n+1 $x_i$ points according to $f(x_i)$;

    else if $f(x_r) < f(x_1)$ // expansion step
      $x_e = c + b(x_r-c)$;
      if $f(x_e)< f(x_r)$
        Replace $x_r$ with $x_e$;
      Replace $x_{n+1}$ with $x_r$ and sort n+1 $x_i$ points according to $f(x_i)$;

    else if $f(x_r) > f(x_n)$ // contraction step
      if $f(x_r) < f(x_{n+1})$
        $x_c = c + d(x_r-c)$;
      else
        $x_c = c + d(x_{n+1}-c)$;

      if $f(x_c) < \min\{f(x_r), f(x_{n+1})\}$
        Replace $x_{n+1}$ with $x_c$;
      else // shrink polyhedron towards best point
        for i=2 to n+1
          $x_i=0.5(x_1+x_i)$;

      Sort n+1 $x_i$ points according to $f(x_i)$;
    end if;
  end for;

 return $x_1$;
```

Figure 21: Pseudo code for the simplex algorithm of Nelder and Mead.

Chapter Three

3 TRANSDUCTIVE LEARNING ENSEMBLES FOR ROBUST VIDEO CONTENT ANALYSIS

3.1 INTRODUCTION

As explained in the previous chapters, transductive learning is not aimed at obtaining a general classification function for all possible test data points (as in inductive learning) but at obtaining a specific classification for the given test data only. The idea is that the desired classification function has to be optimal for the unlabeled test data only and not in general (as in the case of inductive learning). One of the contributions of this thesis is to consider the analysis process for a particular video as a transductive learning setting: the unlabeled data of a particular, previously unseen video are incorporated into the learning and classification process. In this chapter, a framework of transductive learning ensembles is presented that exploits an initial classification (or clustering) result to improve its quality for a particular video. The proposed framework is based on feature selection and ensemble classification; it is called *self-supervised* when the baseline approach relies on unsupervised learning, and it is called *semi-supervised* when the baseline approach relies on supervised learning. In the following, the framework which is used in the proposed approaches for cut detection, face recognition, semantic video shot retrieval, and semantic video indexing (please refer to chapter 4, 6, 7, and 8 for details) is briefly introduced. The framework is presented in a general form and its adaptation to a video content analysis task is described as part of the corresponding approach in the related chapters. The foundations of the framework are based on some principles of feature selection, ensemble classification and on the idea of semi-supervised learning, in particular, semi-supervised learning in a transductive setting. The principles of these technologies are discussed in Chapter 2.

3.2 MOTIVATION

The idea behind our framework is that the appearance of certain objects, events or concepts varies noticeably from video to video. For example, for a particular video, the author's intention, its production, compression and of course its content typically differs significantly. An editor might use very short gradual transitions involving one or two frames instead of abrupt cuts. Such an effect might also be caused by a frame rate conversion. The appearance of an actor might significantly depend on the movie and the character he/she is playing in it. If we want to obtain an indexing result that yields information about the occurrences of actors in a movie, it might be beneficial to learn the characteristic features for only the people present in this video. In addition, if the system has to recognize a person in a video shot, it is not necessary to learn a classification model that discriminates the person's face from all possible faces in the world. It is absolutely sufficient if classification models exist that allow us to distinguish between the persons who appear in this video. Furthermore, the appearance of semantic concepts is often related to a particular video or TV cast. For example, weather news are typically related to the following concepts: A map usually shows the area of interest, some symbols indicate rain, clouds, wind and sun. Furthermore, displayed text gives information about locations, temperatures, all together explained and moderated by a human expert. For a particular instance of weather news, those general elements take a concrete shape, for example the map color, the font type and size of text and the style of the symbols are identical for a certain TV program. In addition, the spatial composition of the shot will be specific for this TV cast, for example the moderator's position and the camera distance will be specific as well. The same is true for the presentation of maps or charts in a particular news cast.

The goal of the proposed framework is to learn and exploit the specific appearance of an object, event or concept with respect to a particular video in order to finally improve the indexing or retrieval result. The basic components are described below in more detail.

3.3 FRAMEWORK

The goal of the proposed framework is to achieve a robust indexing or retrieval result for a given video based on an initial result. This initial result might have been obtained via unsupervised learning or supervised learning. These results can be obtained at different granularities: frames, sequences of frames, shots, scenes etc. Subsequently, these units are considered as samples and each sample is represented by a set of features. For example, in case of cut detection, the (dis-) similarity of consecutive frames is commonly used as a feature. The feature vectors of all samples are used to obtain an initial clustering or classification result for the given problem.

To adapt a model to its appearance in a particular test video *v*, the baseline model is used either to classifiy the frames or shots or to obtain probability scores for them. This initial result is used to generate a training set which consists only of samples of the test video *v* under consideration. In case that the initial result is a binary classification result, all samples are used for the subsequent training process. Otherwise, if probability or confidence scores are available, they can be used to select only the top (bottom) *p* percent of best (worst) samples as positive and negative training data. The automatically generated training data are used to select the best features via Adaboost for the classification task of this video. Then, the set of selected features is split into k disjoint sets in order to enable the training of different classifier views for the video *v*. The feature sets are used to train new classifiers directly on the video. Finally, the newly trained classifiers and the initial classifier form an ensemble and the the video samples are re-classified using this ensemble and majority voting. The initial classifier is incorporated in order to prevent performance degradation. The main components of the framework are displayed in Figure 22.

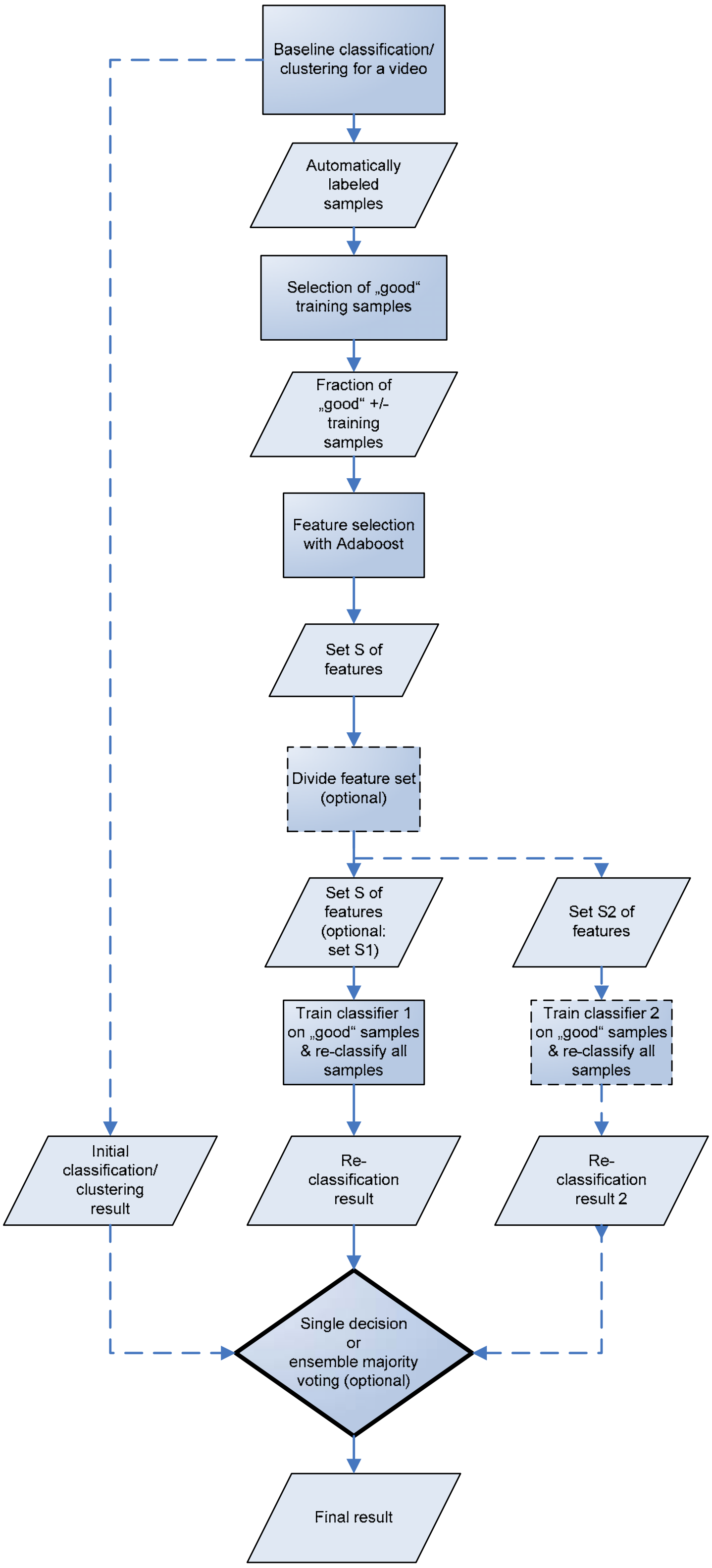

Baseline classification/ clustering for a video
Automatically labeled samples
Selection of „good“ training samples
Fraction of „good“ +/- training samples
Feature selection with Adaboost
Set S of features
Divide feature set (optional)
Set S of features (optional: set S1)
Set S2 of features
Train classifier 1 on „good“ samples & re-classify all samples
Train classifier 2 on „good“ samples & re-classify all samples
Initial classification/ clustering result
Re-classification result
Re-classification result 2
Single decision or ensemble majority voting (optional)
Final result

Figure 22: Framework for transductive learning ensembles for video content analysis. The process is applied to each video separately. Dashed lines indicate optional processing steps.

```
Input:  Set S of training samples for test video v consisting of positive
        sample set P and negative sample set N;
        // each sample shot s ∈ S is described via a set of features F
Output: List of ranked features;

Algorithm:

AdaboostFeatureSelection(P, N)
  for each training instance instX ∈ S:
    weight[instX] =  1/|S|;
  F'= ∅;
  for i=1 to N boosting rounds
    for each feature f
      // the next if-condition is normally not part
      // of  an Adaboost procedure but inserted
      // in order to avoid selecting a feature twice
      // (with possibly different thresholds)
      if f ∈ F' then
        continue loop with next feature;
      Find the threshold t that separates the classes A and B using
      feature f with the lowest error e, this error e is the sum of
      weights of the misclassified samples;
      Choose f  with minimum error $e_{min}$ as current feature f';
      if ($e_{min}$ == 0 || $e_{min}$ >= 0.5) then terminate;
      for each instance instX
        if  instX is classified correctly then
          weight[instX] *= $e_{min}$ / (1- $e_{min}$);
      Normalize all weights[i] (sum is 1);
      F' = F' ∪ {f'};
      Feature f' has rank i in the feature selection ranking;

  return list of ranked features;
```

Figure 23: Pseudo code for feaure selection based on Adaboost.

3.3.1 Feature Selection

This step is aimed at finding the features which charcaterize an object, event or concept in a particular video. Once the positive and negative training samples have been obtained automatically for a test video *v*, feature selection is conducted using a slightly modified version of the Adaboost procedure. As described in the previous chapter, it was shown by Freund and Schapire [55] that Adaboost minimizes the error on the training data as the number of training rounds (and hence the number of classification models) increases, as long as each selected classification model achieves an error rate below 0.5 ("weak classifiers"). In our framework, a weak classifier is built for each feature and a feature is evaluated based on the classification error in each training round. A classification error is estimated for each (weak) classifier by estimating the best threshold that separates positive and negative samples using the one-dimensional data of a single feature. This classification error depends on the current sample weights of the samples which might change from round to round. The training samples are weighted equally in the beginning. Training samples which are misclassified by the actually chosen weak classifier are re-weighted such that they have more impact in the next training round for the next "weak classifier". Thus, a newly selected feature has a higher probability to correctly classify those training samples that have been misclassified in preceding rounds. The modified version of Adaboost as shown in Figure 23 is employed to perform feature selection: In round k, the feature whose related weak classifier yields the lowest error on the current weighting of the training data is selected as the k-th feature.

3.3.2 Ensemble Classification

After the best features have been selected, they are used to train other classifiers on the automatically labeled training samples of the *test* video v under consideration. In our system, SVM is used since it has proven to work well for several video indexing tasks. However, training a new classifier only on the automatically labeled samples might is possibly to the initial result. In particular, it is not guaranteed that the samples are labeled correctly, and the quality of the new classifier depends in general on the accuracy of the initial result. Furthermore, the initial result might be a good one and should not be discarded totally. To address these issues, an ensemble approach is utilized in our framework.

Besides Adaboost, another possibility is to build an ensemble of classifiers using majority voting. In this case, the classifiers which form an ensemble should have a certain level of independence. As discussed in the previous chapter, there is some evidence [89] that a reasonable degree of independence of ensemble classifiers improves accuracy and guarantees at least the accuracy of the weakest classifier in the ensemble if the classifiers' accuracies exceed a certain value. In our framework, the boosting procedure of Adaboost is exploited to generate two disjoint feature sets which have a certain degree of independence. Features are assigned alternatingly to two different feature sets according to their odd and even rank, respectively, in the feature selection process. Subsequently, two new classifiers are trained using one of the feature sets. This approach is motivated by the fact that during the Adaboost process the weights of those samples which have been misclassified by the preceding weak classifier are increased. Thus, the next selected classifier should be partially independent of its direct predecessor. In case of n classifiers, the feature with rank k is assigned to classifier k modulo n. In subsequent SVM training, models are learned based only on the automatically labeled training examples taken from the current test video v, yielding two (or n) new SVM models. Finally, at least three SVM models are available for each video, the initial (global) model and two (or n) local models. These models were learned with the training data that have been labeled automatically for this particular test video v using the initial model.

3.4 Specific Instances of this Framework

The concrete realization of this framework depends on the task at hand. Several modifications are possible and some of them will become apparent in the subsequent chapters of this thesis. First, the initial result can be generated using either an unsupervised or a supervised approach. In case the initial result is based on unsupervised learning, the process of the whole framework is called self-supervised. Otherwise, it is called semi-supervised. The used features differ with respect to the given problem. While the feature selection process reduces the number of features, it is also possible to generate good initial results for certain tasks with a very small feature set, but to employ feature selection on an enlarged set of features. For example, this is the case for video cut detection, where two features are sufficient to obtain a rather good initial result. The fraction of selected training samples varies from task to task. Also, there are cases where the ensemble approach is not required to improve the final result. As described for the face recognition/clustering approach presented in Chapter 6, the feature selection process has much more impact on the quality of the final result than the number of classifiers in the final classification stage. Finally, it should be noted that in general, any classifier could be used in our framework in the initial stage as well as in the final stage.

3.5 Summary

In this chapter, a framework for transductive learning ensembles was introduced for the task of video indexing and retrieval. In particular, its basic components consisting of feature selection and ensemble classification were presented. Finally, the possibilities of a concrete realization of the framework were discussed.

4 ROBUST AND ADAPTIVE SHOT BOUNDARY DETECTION

4.1 INTRODUCTION

A video sequence typically consists of a large number of shots which have been put together during a production process either to tell a story or to communicate any kind of information (e.g., news or documentary). Video arts might have another intention but are not considered here. A shot is the fundamental processing unit in video retrieval applications and most indexing and retrieval algorithms rely on a correct temporal segmentation of a video into particular shots. The task of shot boundary detection (SBD) is to perform such a temporal segmentation of a given video into single shots and to recognize abrupt and gradual transitions. Synonymous terms in the literature for shot boundary detection are: shot detection, temporal video segmentation, video segmentation, shot segmentation, and video cut detection (excluding transition detection). Some authors also use the term scene in conjunction with temporal shot segmentation, but according to our definitions for shot and scene in Chapter 2, they actually mean shot segmentation.

TRECVID Test Set	#Frames	#Trans.	#Transitions per 100 frames	Cuts [%]	Gradual Trans. [%]	Dissolves [%]	Fades [%]	Others [%]
2003	596054	3734	0.63	70.7	29.3	20.2	3.1	5.9
2004	618409	4806	0.77	57.7	42.3	31.7	4.8	5.7
2005	744604	4535	0.61	60.8	39.2	30.5	1.8	6.9
2006	597043	3785	0.63	48.7	51.3	39.9	1.3	10.1
2007	637805	2317	0.36	90.8	9.2	5.4	0.1	3.7

Table 1: Composition of the test sets for the shot boundary detection task at TRECVID in the years from 2003-2007.

If a shot abruptly follows a preceding shot, such an abrupt transition is called a "cut". Alternatively, one can insert transitional frames between two consecutive shots which results in a smoother perception of a shot change ("gradual transition"). Lienhart [96] states that a cut is defined as the direct concatenation of two shots with no transitional frames involved: cuts lead to a perceptible temporal visual discontinuity. The most common gradual transitions are fades (in and out), dissolves, and wipes. Although most shot changes in videos are abrupt, the percentage of gradual transitions might be also large for some video collections and genres. The frequency of gradual transitions varies strongly from video to video. For example, in the test sets for the shot boundary detection task of the TRECVID evaluations from 2003 to 2007, the percentage of gradual transitions varies noticeably and lies between 10% and 51% (see Table 1). Hence, it is obvious that the correct detection of gradual shot boundaries is as important as cut detetction for some applications.

In this chapter, several improvements for shot boundary detection are proposed. Initially, the related work for shot boundary detection is comprehensively reviewed in Chapter 4.2. The first contribution is related to cut detection and compression artefacts. A problem associated with frame dissimilarity measurements in compressed video sequences is identified and two solutions are presented for this problem (Chapter 4.3). Then, an unsupervised shot boundary detection approach which reduces the need for parameter adjustments is presented (Chapter 4.4). Another advantage of the unsupervised approach is that there is no need for the time-comsuming task of creating a sufficiently large amount of training data (i.e., labeling the shot boundaries manually). Experimental results demonstrate that the detection quality depends on the sliding window size. It will be shown that this parameter can be estimated by evaluating the quality of our clustering result. This way, the need for manual parameter adjustment is removed for cut detection. The unsupervised approach to gradual transition detection is supplemented by a sophisticated false alarm removal. For this purpose, a high-quality motion estimation algorithm is employed (Chapter 4.4.2). Furthermore, it will be shown that cut detection can be improved further when an ensemble of classifiers is applied (Chapter 4.5). This is a reasonable approach to improve performance in case when training data are already available. A self-supervised approach for cut detection is presented in Chapter 4.6 which is the first application of our transductive learning ensemble framework as suggested in Chapter 3. In this approach, the initial clustering result is employed to learn the best features and to adaptively learn a classification model for a particular video. Finally, it will be demonstrated in Chapter 4.7 that the quality of a clustering result can be utilized for automatic performance prediction of video cut detection results. The work presented in this chapter has been partially published in [36, 37, 40, 42, 43, 45, 115].

4.2 RELATED WORK

4.2.1 SURVEYS ON SHOT BOUNDARY DETECTION

First, several related surveys and comparison studies are discussed. Koprinska and Carrato [87] present an overview of temporal video segmentation research and the related problem of camera motion estimation. The authors provide a taxonomy and discuss approaches for both the uncompressed and the compressed video domain. Several metrics for frame dissimilarity measurement are discussed, including pixel-wise, block-based, histogram-based, local histogram-based and block-based motion-compensated dissimilarity measurements, as well as feature-based approaches considering the number of edge pixels. Various methods are presented to classify frame dissimilarities, such as simple thresholding, local thresholding, χ^2-test, likelihood ratio test, and clustering approaches. In addition to these algorithms, which tackle the cut detection problem using a bottom-up and data driven view, there are top-down, model-driven approaches which model certain video transitions or use for example Hidden Markov Models for shot boundary detection and camera motion estimation. Such approaches are particularly suitable to detect more sophisticated video events, such as gradual shot transitions and camera motion. Finally, the authors present algorithms for compressed videos which directly use DCT coefficients, in particular DC frames, macroblock information and coded motion vectors.

Lefèvre et al. [92] provide an analysis of the computational complexity of several temporal video segmentation algorithms with respect to real-time requirements.

Bescos et al. [10] present a comparison of different frame disparity functions, these are functions which measure frame dissimilarities. They distinguish between deterministic (e.g., summation of absolute differences), statistic parametric (e.g., likelihood ratio test) and statistic non-parametric disparity functions.

A study comparing the cut detection performance has been conducted by Boreczky and Rowe [18] in 1996. Five algorithms are investigated, making use of: 1. frame histogram differences for a 64-bin gray-scale histogram and a global threshold; 2. histogram differences in regions, where the number of region differences exceeding a threshold must exceed another threshold; 3. the twin-comparison method; 4. motion-compensated differences and a global threshold; 5. DCT coefficient differences and a global threshold. The algorithms are tested on five videos having 419745 frames and 2507 cuts in total. The histogram-based and the region histogram method outperform the others in most test cases.

Lienhart [96] has conducted a comparison study in 1999, investigating four algorithms, of which two are able to detect cuts. The other two approaches are specialized for detecting fades and dissolves, respectively. The first cut detection algorithm is based on histogram differences, and a global threshold is used within a sliding window. This simple approach outperforms the other approach which is based on the change edge ratio in subsequent frames. The test set consists of four videos.

Two comparison studies have been reported by Gargi et al. in 1998 and 2000 [57, 58]. In the second study, the same algorithms are investigated as in the first one, but a larger test set has been used and thus, this study is considered here. They also have conducted an interesting study on human ground-truth performance on live videos and found that best results have been obtained when the volunteers previewed the video once at full speed and then detected shot boundaries when viewing the video at half speed. The detection delay of the nine volunteers due to "live" detection was between 7 and 20 frames. The video test set used in that study consists of four video sequences with a total duration of approximately 76 minutes including 959 cuts and 262 gradual transitions (MPEG-1 encoded, 30 frames per minute, 320*240 pixels resolution). Six algorithms for the MPEG domain are investigated, which make use of DCT coefficients, macro block information, motion prediction statistics and DC-frames. Various color spaces and frame difference metrics are evaluated for the histogram-based algorithms. It is shown that histogram intersection works best while the choice of color space and histogram dimensionality has less impact, as long as not only luminance information but also color information is considered. The approach of Yeo and Liu [189], using approximated DC-frames and a local thresholding technique within a sliding window, outperforms the other five approaches, achieving a recall of 79% and a precision of 88%. Furthermore, Gargi et al. have investigated the impact of source effects to shot detection algorithms, such as choice of encoder and bit rate. Although the algorithm of Yeo and Liu still performs best, this approach is more sensitive to source changes than the second best algorithm proposed by Shen and Delp [139].

4.2.2 Approaches for Shot Boundary Detection

Apart from the surveys presented above, a number of approaches for shot boundary detection are presented in the following. From the large set of proposals, the most popular approaches and the most recent advances in the field have been chosen.

To detect cuts, a straightforward approach is to measure the differences between the pixels of consecutive frames, but this approach is very sensitive to object motion, camera motion, brightness changes and noise [87]. Many approaches (e.g., [139] and [188]) propose the use of histograms since they are less sensitive to motion and the other events mentioned above. The estimated dissimilarity between consecutive frames is commonly used to decide whether there is a cut at frame *k*. Therefore, a rule is required to make a decision about the presence of a cut based on these measurements. An early popular method is the twin comparison method suggested by Zhang et al. [195]: Frame dissimilarity is measured by counting the pixels that changed by more than a threshold n. If the percentage of changed pixels exceeds another relatively high threshold *t*, a cut is declared. In addition, if a dissimilarity value exceeds a low threshold t_l, then a start of a gradual transition is assumed. Furthermore, dissimilarity values of the consecutive frames are added up as long as they exceed the lower threshold t_l. If this sum exceeds the high threshold t_h, a gradual transition is declared. In case when a dissimilarity measure falls below the low threshold, the start of the gradual transition is dropped and the search continues at the next frame. The advantage of this method is that it is able to detect cuts as well as gradual transitions. However, the application of global thresholds to an entire sequence results in many false alarms and missed cuts. Furthermore, the problem of determining an appropriate value *t* must be solved.

To address this issue, Yeo and Liu [188, 189] suggest a sliding window technique and use a local parameter to compute a threshold separately for each window position. This window consists of 2*m+1 frame dissimilarity measures, for a small m>0. To decide whether there is a cut at position k, the frame dissimilarities of the neighboring frames are taken into account. A peak at frame position k is considered as a cut only if it is the maximum value and if it is n times larger than the second largest peak in the window. This principle has been used in many variations (e. g. [11], [69], [139], [164]) and typically results in a robust detection performance. Furthermore, the authors show [188, 189] for the MPEG domain (which is also proposed by Shen and Delp [139]) that it is sufficient to use sub-images, so called DC-frames, to achieve a good cut detection performance. The main advantage of using DC-frames is that the processing time is reduced. Since DC values represent the average value for a DCT (discrete cosine transform) transformed pixel block of size 8*8, the use of these values represents a kind of inherent smoothing. In total, two parameters must be set in this approach. Yeo and Liu report the best results for the combination of a histogram and a pixel-based metric, with a recall and a precision of 99% (5 videos, 386 cuts, 70768 frames). Yeo and Liu suggest an approach to detect gradual transitions as well. Frame dissimilarities are measured with a temporal distance of k, where k must exceed the duration of the longest transition. Then, the idea is to detect plateaus with sloping sides in such a time series, which is achieved similar to cut detection using a local threshold. They report a recall of about 90% and and a precision of about 75% for 19 gradual transitions in five test videos.

Jacobs et al. [78] present an approach for shot boundary detection which is based on RGB color histograms and edge change ratio. They measure frame dissimilarities for a frame distance of 5 within a sliding window. To detect cuts, thresold parameters are used, whereas a finite state machine is used to detect gradual transitions. At TRECVID 2004 evaluation, their best submission achieved the following results: a recall of 89% and a precision of 65% for cuts, and a recall of 39% and a precision of 62% for gradual transitions.

Taskiran et al. [155, 156] use multiple features including the intersection of DC-frame histograms, the standard deviation of color components, the number of intra-coded, forward and backward predicted macroblocks, and the prediction type of macroblocks in MPEG videos. The authors consider only cut detection but claim that their approach can be easily extended to gradual transition detection. The features, which are called a generalized trace by the authors, are used to learn a binary regression tree for decision making. To make classification more robust, features are agglomerated in a sliding window of size $2m+1$ around the center frame which is to be classified. The training consists of two steps. First, using one part of the training set, a large tree is built that overfits this training data. Then, in the second step, the tree is pruned back to maximize a desirable criteria on the remaining training data. Taskiran et al. determine the probability for a cut by applying the binary regression tree to a multi-dimensional feature vector for each frame [155, 156]. If two cuts occur within ten frames, the candidate with the lower probability is discarded. A fixed threshold is then used for decision making. The authors report a recall of 92% and a precision of 94%.

Truong et al. [164] have proposed a local mean ratio filter (recall: 97.9%; precision: 97.5%) as an enhancement to cut detection algorithms in order to reduce noise in frame difference sequences. The sum of absolute bin-wise color histogram differences is used to measure frame dissimilarity. Furthermore, enhancements are suggested for existing fade and dissolve detectors: Thresholds are selected according to mathematical models of the transition types. Finally, transitions that have relatively similar start and end frames are considered as false alarms and are deleted.

Kuo and Chen [91] suggest an interesting approach to cut detection which is only based on the macroblock information encoded in MPEG videos. An analysis is provided on the ratio of forward and backward predicted macroblocks in inter-frames in the temporal neighbourhood of a cut. So called masks are defined to detect a cut in a particular frame type (namely I-, P-, and B-frame). It is argued that the number of macroblocks coded with a backward predicted motion vector is low in the B-frames preceding a cut, while the number of forward predicted macroblocks is low in the frame that represents a cut itself and the subsequent frames. Based on this observation, the corresponding ratios are considered to calculate the probability for a certain frame of being a cut. A threshold parameter is used for decision making. The approach is extended according to the twin-comparison method [195] to detect dissolves as well. The authors report a recall of 96% and a precision of 98% for a test set of five videos with 61 shot boundaries.

Tahaghoghi et al. [153] divide frames into 4*4 regions and disregard the frame center. Using a temporal sliding window, the similarity is computed pair-wise between *all* frames in this window using histograms of the HSV (Hue, Saturation, Value) color space. The sliding window is divided into pre-frames, the current frame and post-fames. The similarities are ranked with descending similarity, and the number of pre-frames in the top half represents the final similarity value. The sliding window size is a pre-defined parameter. To detect gradual transitions, similarity scores are computed in a slightly different way. The similarity to the current frame is computed for each frame in the sliding window. Then, the ratio of the average similarity of the subsequent frames (post-frames) and of the preceding frames (pre-frames) is calculated. Peaks in the ratio curve indicate a gradual transition, but the authors point out that there are still many false alarms left. These false alarms are removed by comparing the similarity of the transition start and end frame: A candidate is considered as a false alarm if these frames have a dissimilarity which is below 25% of the theoretical maximum dissimilarity. In the TRECVID evaluation in 2005, their best submission [170] achieved a recall of 92% and a precision of 93% for cuts and a recall of 73% and precision of 65% for gradual transitions.

In TRECVID's shot boundary detection evaluation task, IBM's submissions (Smith et al. [144], Adams et al. [2], Amir et al. [3,4,5]) belonged to the top submissions for several years. A detailed description is not publicly known, however, some general ideas are reported in their papers: Their baseline system uses three-dimensional histograms in RGB color space, frame dissimilarities are computed for frames with a temporal distance of 1, 3, 5, 7, and 13. A finite state machine (FSM) consisting of up several states (pre-cut, cut, after cut, fade-in start and end, in shot, etc.) employs the dissimilarity data to decide about state transitions. A sliding window of size 61 is moved along the frames where the center frame is subject to analysis. For each position, adaptive thresholds based on the data extracted in this window are computed which are then used to decide whether a state is kept or another state is reached. This baseline system was extended at TRECVID 2003 [3] with a flash detector, a better handling of fades, an improved estimation of gradual transition boundaries and detection of MPEG encoding errors. At TRECVID 2005, the system achieved the following results: a recall of 93% and precision of 89% for cut detection, and 84% of recall and 72% precision for gradual transition detection.

Liu et al. [101] present an approach for shot boundary detection which is based on different detectors for cuts, fades, dissolves, short dissolves and wipes. Each detector is realized via a finite state machine with either four or five different states. Overall, sixteen different functions are used to decide whether a state is kept or left, these functions are evaluated based on threshold parameters, except for some verification steps which are based on SVM. The video is analyzed frame by frame using the FSMs, and state changes are based on comparing one or more features with pre-defined or adaptive thresholds. For cut and dissolve detection, a SVM is applied to verify transition candidates. Eighty-eight intra- and inter-frame features are extracted, the inter-frame features are extracted for a frame distance of 1 and 6. The intra-frame features measure the histogram mean, variance, skewness, flateness, dynamic range and edge ratio in horizontal and vertical direction. The inter-frame features measure the difference for these features for two different frames, histogram distance in HSV space, but also include motion features and measures for matching error and matching ratio of two frames. For dissolve verification, another set of specialized features is used including predominantly variance (of brightness) and edge features. The features are extracted from a region of interest (i.e., the frame border is ignored). The authors report the following results for the TRECVID 2006 test data: a recall of 89.4% and precision of 90.4% for cut detection, and a recall of 77.5% and precision of 85.8% for gradual transition detection.

Each of the approaches mentioned so far depends on an adequate threshold parameter setting. The class of statistical approaches tries to avoid threshold setting in order to deal with this problem. For example, Hanjalic [69] estimates the distribution of frame dissimilarities for both cuts and non-cuts in a training stage and uses a likelihood ratio test for decision making to minimize the average detection error. The dissimilarity metric is based on motion-compensated pixel differences of subsequent DC frames. Due to the small size of DC-frames, the motion compensation is estimated for pixel blocks of size 4*4 in order to remove artefacts based on motion. Three features are considered: First, the likelihood for a frame dissimilarity value (not) being a cut depending on its height is considered. Second, the conditional probability for a boundary presence is introduced which considers the fact that a cut is represented by a sharp isolated peak in a sliding window of frame dissimilarity series of size $2m+1$. Third, a priori knowledge is included, namely that the probability for a shot boundary increases with the number of frames elapsed since the last shot boundary. Although there are not any thresholds in the decision stage, several parameters have to be estimated in the training stage, in particular for the distributions of intra-shot and inter-shot dissimilarity values. A Gaussian distribution is assumed for the inter-shot values. In total, eight parameters must be trained in advance: three parameters describe the distribution of intra-shot dissimilarities, two parameters describe the distribution of inter-shot dissimilarities, and one parameter is needed for both the conditional probability function and the Poisson distribution which is used to model the distribution of shot lengths. Finally, a parameter is needed to describe the sliding window size. Furthermore, a statistical dissolve detector is suggested by Hanjalic. Dissimilarities between frames are measured at a larger temporal distance (e.g., frame distance of 22). At those positions, where dissimilarity is the maximum in a sliding window, the variance characteristic is analyzed. The conditional probability for a dissolve is defined by the relative change of variance in the sliding window. A perfect detection performance of 100% recall and precision is reported for a small test set of five videos having 104 cuts, 79% recall and 83% precision are reported for dissolve detection (23 dissolves in the test set).

Bescós [10] analyzes several frame disparity functions, these are functions which measure frame dissimilarities. Deterministic (e.g., summation of absolute differences), statistic parametric (e.g., likelihood ratio test) as well as statistic non-parametric disparity functions are considered. The metrics are evaluated based on the measure of divergence, an index that describes the separability of two classes. To detect cuts in MPEG-2 videos, Bescós chooses two metrics which accomplish the best divergence between the classes "cuts" and "non-cuts" [11]: Addition of squared pixel differences (Y channel) from DC-frames and a likelihood ratio test for the mean and variance of color information (Cb and Cr channel). A third feature is computed which uses a small sliding window of size 1. Furthermore, a simple supervised parallelepipedic classifier is applied to learn a classification function. Regarding gradual transitions, Bescos shows that divergence between transition and non-transition class increases when the frame distance is increased from 1 up to at least the transition length, whereas the metric choice is not crucial for divergence. Several characteristics must be met by a gradual transition: First, in the time series with frame distance n ($n \geq L$, where L is transition length) the frame center of the sliding window should have a maximum value. Second, the peak should be reached gradually. For such frame positions, a three-dimensional feature vector is created that captures the properties of a typical triangular pattern caused by a gradual transition in a time series with frame distance $n>1$. Results are reported for a subset of the MPEG-7 test set (2074 cuts, 460 gradual transitions): 99% recall and 95% precision for cuts, and 87% recall and 66% precision for gradual transitions.

Chua et al. [26] propose a unified approach to detect cuts and gradual transitions by using a temporal multi-resolution approach. It is realized by applying a wavelet transform to frame histograms and other measures. The authors use histogram differences as well as a coarse representation of MPEG motion vectors. First, they detect candidates from the set of local maxima and then they apply an adaptive thresholding technique. Finally, they use support vector machines to find an optimal hyperplane to separate cuts and non-cuts. Their baseline approach [26] using histogram metrics achieved a recall of 98% and precision of 78% while the extended version [27] includes motion information as well as a refinement approach with support vector machines that achieved a recall of 93% and a precision of 96 % for cut detection, and a recall of 89% and a precision of 88% for gradual transitions (MPEG-7 video test set). However, results at the TRECVID 2002 evaluation are significantly worse: recall and precision of about 70-75% for cuts and 38% recall and 47% precision for gradual transitions [22].

Zheng et al. [197] propose a shot boundary detection framework which is based on a cut detector, a fade detector and a gradual transition detector. Several features are extracted from both the compressed and the uncompressed domain: mean and standard deviation of pixel intensities for fade detection, color histogram, pixel-wise difference measures for cut and gradual transition detection, and motion vector information which is used for gradual transition detection only. The fade detector is based on monochrome frame detection and tracking using several pre-defined thresholds. To detect cut candidates, a second-order difference measure is compared against a threshold. The second-order difference is computed by subtracting neighbored difference measures. Cut candidates are post-processed by a flashlight detector and a gradual transition filter. To detect gradual transitions, the authors distinguish between short and long transitions where the twin comparison method [195] is used to detect the short ones. To detect long gradual transitions, a finite state automata model is employed. Gradual transition candidates must first pass an adaptive threshold t_l that is based on motion measured by motion vector analysis. If three subsequent dissimilarity measures fall below this threshold, then the end of the transition is assumed. A gradual transition is declared if the sum of all dissimilarity measure is above a second threshold $t_{h.}$. Yuan et al. [191, 193] extend this method via a graph partition model for temporal video shot segmentation in which a graph is built based on multi-pair frame similarity measures. Within a sliding window, frame similarity is measured for all frame pairs. While each frame is considered as a node in a graph, these similarity values are considered as the related edge weights. The graph partition model tries to divide this graph in two subgraphs A and B by optimizing an objective function: the association between the two subgraphs has to be minimized whereas the association in a subgraph has to be maximized. Yuan et al. incorporate the graph partition model into the original framework and substitute some components of [192] by using support vector machine for cut detection and gradual transition detection. One SVM is trained for cut detection, three SVMs are trained for gradual transition detection for different temporal resolutions using the TRECVID test sets of 2003 and 2004. At TRECVID 2005, this approach achieved the top f1-measure for video cut detection and belonged to the best two approaches for gradual transition detection.

Lelescu and Schonfeld [93] suggest statistical sequential analysis for shot boundary detection. They extract DC-frames from I- and P-frames and reduce dimensionality of DC-frames to a small *M* by using principal component analysis. This way, a sequence of feature vectors is generated representing the P- and I-frames. The problem of shot change detection is viewed as a sequential change detection problem in the parameters of the probability density function associated with the feature vector random process. Two models are introduced for shot changes, the additive change and non-additive change in the parameter space. Experimental results show that the additive model outperforms the non-additive model in both recall and precision (for a test set having 206 shot changes, containing 93 special effects, i.e., gradual transitions): recall is increased from 81% up to 92% and precision is increased from 65% up to 72%. Both statistical sequential analysis methods outperform a standard histogram comparison approach.

Boccignone et al. [15] present an approach to shot boundary detection which is motivated by the human perception of visual scenes. Humans view a scene by moving their eyes across several fixation points (up to four times a second) and integrating the gathered information. The visual attention process naturally filters out unwanted information and brings relevant information to the observer's consciousness. The eye movements result in a visumotor trace of a world view, and the authors intend to model the visumotor process [15]. First, in a pre-attentive stage, salient points (focus of attention: FoA) are extracted from an image at different scales, and these features form a saliency map. Then, these locations of interest are traversed in the order of decreasing saliency which results in a scanpath (according to the visumotor trace). Finally, the higher perceptual level is modelled in which the observer infers about a shot change by analyzing his own visumotor trace. Therefore, a function *M*(*t*) is defined that represents the consistency between two scanpaths based on a pairwise comparison of FoA according to their corresponding scan path order. The function *M*(*t*) is based on three consistency types: local spatial consistency (position of FoA), local temporal (difference of viewing time) consistency and local visual consistency (based on histogram and texture). Histograms are created with respect to *M*(*t*) for shot boundaries and intra shot changes on training sequences and their ideal distributions are derived from the histograms. Then, the time series is analyzed using a likelihood ratio test based on *M*(*t*) to decide whether there is a cut or a dissolve at a certain frame position. The authors achieved an average recall of 97% (dissolves: 92%) for cuts and an average precision of 95% (dissolves: 89%) for a test set of 130 minutes, including 1304 cuts and 336 dissolves. Part of their own test set was a subset of the TRECVID 2001 shot boundary test set (about 20% of the whole test set), for which an overall performance of 92.5% is reported. This small video test set consists of 10 short sequences including 422 cuts and 185 dissolves. Unfortunately, due to partial selection of video sequences it is hard to compare to other approaches at TRECVID 2001.

The use of clustering algorithms is a possibility to overcome the need for a training stage that usually requires a lot of manually labeled data.

Günsel and Tekalp [68] use two dissimilarity features (absolute histogram difference and Chi^2-statistics) and propose the k-means algorithm for clustering. The dissimilarity values of subsequent frames are passed to the clustering process, the cluster with the larger mean vector is considered as the shot change cluster. If subsequent frames are in this cluster, these frames are considered as a gradual transition. The authors report a recall of 79% and a precision of 90% for a news video test set containing 167 shot boundaries (both cuts and gradual transitions).

Gao and Tang argue [56] that a clear distinction between the two classes cuts and non-cuts cannot be made and suggest a fuzzy k-means algorithm. Two features are used in their approach: frame dissimilarities based on a pixel-wise metric as well as on a histogram-based metric. The frame dissimilarities are processed in two stages at different temporal resolutions. In the first stage, the number of cut candidates is minimized by considering only the differences between each n-th frame. In this way, the authors intend to prevent a performance degradation of the k-means algorithm, since it usually works best when the (two) classes are of comparable size. The frame dissimilarities of the coarse temporal resolution are then used to obtain fuzzy membership values for each of both classes. Finally, the representatives of the "fuzzy" set must be assigned to one of the classes in case of cut detection. The candidates with a fuzzy membership of [0.4, 0.6] are defuzzified by introducing a new metric. This metric is the result of multiplying the values obtained from the histogram and the pixel-based metric. The local maximums using the new metric are considered as cuts. Now, the classification algorithm is repeated for the shot transitions found at the temporal resolution *n*, but the original temporal resolution (frame distance of 1) is considered now in order to find the exact cut positions. If subsequent frames are in this cluster, these frames are considered as a gradual transition. In total, five parameters must be set in this approach. Two parameters are used to define the range of the "weak" membership values which are used as input to the defuzzifying process. Furthermore, the estimation of local maximums must use at least one parameter and the fuzziness must be set. Finally, the sub-sampling of the coarse temporal resolution is defined by the parameter *n*. The detection performance reported for this approach is a precision of 96.5% and a recall of 98.1%, including gradual transitions, for a quite large test set of 14 news video sequences with 3893 shot boundaries in total (12 news videos, 3468 cuts, including detection performance of gradual transitions).

4.2.3 Performance Analysis

Despite the TRECVID evaluation efforts since 2001, it is quite difficult to answer the interesting question which shot boundary detection approaches work best, in particular for those proposals which have not been evaluated at TRECVID. Several authors have presented very good results for the shot boundary detection task in recent years but unfortunately some of these approaches have not been evaluated on a publicly available test set (e.g., TRECVID or MPEG-7). Nevertheless, we try to compare some of the top approaches discussed in the previous section and to make conclusions about their relative performance. Table 2 and Table 3 summarize the performance of approaches which were evaluated on (at least on a part of) one of the TRECVID test data sets or the MPEG video test set, respectively.

Some TRECVID results in terms of the f1-measure are presented in Table 2 for the following approaches: Amir et al. [3, 4, 5], Boccignone et al. [15]¸ Chua et al. [26, 27], Volkmar and Tahaghoghi [170], and Zheng et al. [197]. Since the results are obtained on TRECVID test sets from different years, it is difficult to compare them directly. However, the fact is employed that

IBM (Amir et al. [3, 4, 5]) participated in the shot boundary detection task annually since 2001 and according to their TRECVID reports, the used system did not change significantly over the years. In this way, it is possible to roughly compare also the TRECVID 2005 results with those of Boccignone et al. [15] and Chua et al. [26, 27] which were obtained on earlier TRECVID test sets.

For cut detection, Table 2 depicts that the graph partition approach of Zheng et al. [197] achieves the best result (f1: 93.5 on the TRECVID 2005 test set), followed by the approach of Volkmar and Tahaghoghi [170] (f1: 92.3 on the TRECVID 2005 test set) which uses a threshold-based approach based on pairwise frame similarities for different frame distances. Furthermore, it is observable in Table 2 that IBM's approach (Amir et al. [3, 4, 5]) outperforms the approaches of Boccignone et al. [15] and Chua et al. [26, 27] on the corresponding TRECVID test data sets of 2001 and 2002, respectively. Two research groups presented results on the MPEG-7 video test set: Bescos [11] and Chua et al. [26]. Although Chua et al. report also results for TRECVID 2002 test set, it is hard to conclude how the performance of Bescos' approach is related to the approaches summarized in Table 2.

Approach	Type	TRECVID Test Set	Cuts [F1]	Gradual Transitions [F1]
Amir et al. [3]	Threshold-based, Finite State Machine	2001	97.0	66.8
Amir et al. [4]	Threshold-based, Finite State Machine	2002	92.5	77.0
Amir et al. [5]	Threshold-based, Finite State Machine	2005	91.2	77.6
Boccignone et al. [15]	Statistical hypothesis test	2001 (Subset)	(92.5)	
Chandrashekhara/Feng/Chua [22] (Chua et al. [27])	Thresholding (+Supervised Learning/SVM)	2002	71.6 (+~10.0)	41.9 (+~10.0)
Volkmer and Tahaghoghi [170]	Thresholding	2005	92.3	68.3
Zheng et al. [197]	Graph partition model/SVM	2005	93.5	78.9

Table 2: Performance results in terms of F1-measure for selected approaches which have been evaluated on a TRECVID benchmark test set. Chua et al. report an increasement with respect to their baseline system of 10.0 in terms of f1-measure [27].

With respect to the detection of gradual transitions, Zheng et al. [197] and IBM's approach [5] achieved best performance on the TRECVID 2005 test set in terms of the f1-measure (78.9 and 77.6). Regarding the other approaches, it can be stated that they achieved worse results compared to both top approaches (except for Boccignone et al.'s approach [15] for which conclusions are not possible since the results are not separated for cuts and gradual transitions) but it is hard to make conclusions about the relative performance of the other approaches.

Approach	Type	MPEG-7 Test Set	Cuts [F1]	Gradual Transitions [F1]
Bescos [11]	Supervised Learning	Subset of MPEG-7	97.0	75.0
Chua et al. [27]	Supervised Learning/SVM	Subset of MPEG-7	94.0	88.0

Table 3: Performance results in terms of F1-measure for selected approaches which have been evaluated on parts of MPEG-7 video test set.

It should be noted that there are some methodological issues with respect to the experimental results of Boccignone et al. [15], Chua et al. [27], and Bescos [11] though they use parts of commonly available video test sets. Boccignone et al. [15] use only a small subset of the TRECVID 2001 SB test set and do not explain whether evaluation has been conducted according to the TRECVID standard evaluation procedure. Furthermore, the authors present a performance of 92.5 for this part of their test set, but their results are not reported separately for cuts and dissolves. Hence, it is not clear whether their approach is competitive to the top TRECVID 2005 approaches. Bescos [11] gives detailed information about which videos of the MPEG-7 test set have been used. However, for the targeted application, the videos have been transcoded to MPEG-2 which removes the useful original property of the videos that they had been encoded with different encoders. As it will become clear in the next chapter, different encoder strategies might lead to artefacts in frame dissimilarity measurements. Furthermore, Bescos trained a simple classifier on one half of the test set which is a critical issue since this classifier possibly overfits to this half of the test set. While the approaches of Chua et al. and Bescos achieve similar results for cut detection, Chua et al.'s approach probably outperforms Bescos' approach for gradual transition detection. However, this cannot be concluded based on the information given by the authors due to missing information about the experimental setting.

Chua et al. report [27] very good results for a part of the MPEG-7 test set and mention that their extension (ATMRA) with SVM improved the baseline system (TMRA), which has been evaluated at TRECVID 2002 [22], by about 10.0 in terms of the F1-measure for all transitions. Considering the results achieved at TRECVID 2002, it is concluded that ATMRA's performance is below those of the top 2005 TRECVID approaches.

4.2.4 SUMMARY

Several approaches have been suggested for the task of shot boundary detection in recent years. These approaches are divided into three classes: rule-based and parameter-dependent approaches,

approaches using supervised learning, and approaches using unsupervised learning. There are still many threshold-based approaches which work surprisingly well. However, empirically finding a reasonable threshold setting that works well on arbitrary videos remains an unsolved issue. Certainly, this is one of the reasons for the fact that statistical approaches and supervised learning dominate the field of shot boundary detection. Using supervised learning, the issue of parameter estimation is shifted to the applied machine learning method or to a statistical test that minimizes the probability of a detection error, respectively. There are relatively few approaches which use unsupervised learning to solve the problem although they can avoid the need for labeled training data which is time-consuming to create. Nevertheless, the successful cut detection approaches have at least one of the following properties:

- A sliding window technique is used;
- Frame dissimilarities are extracted for several frame distances;
- An ensemble of classifiers is applied.

Among the participants of the TRECVID shot boundary tasks in 2005 and 2006, the approaches of Amir et al. [3, 4, 5], Liu et al. [101], and Yuan et al. [192, 193, 194] belonged to the most successful sytems with respect to gradual transition detection. These approaches have the following properties in common. All of them use finite state automata, however, Yang et al. exchange it in their subsequent approach with a SVM [193, 194]. Also, several frame distances are used in the successful approaches. Interestingly, Liu et al. [101] use only one frame distance (length: 6) which is greater than 1. Amir et al. [3, 4, 5] and Liu et al. [101] used multiple pair-wise frame dissimilarities in the temporal neighborhood of the frame of interest. Only Yang et al. [193, 194] employ a temporal multiresolution approach. Liu et al. [101] use specialized transition detectors with respect to transition types, whereas Yang et al. [193, 194] apply several SVMs for different temporal resolutions.

4.3 Performance Comparison Study

4.3.1 Introduction

On the one hand, the TRECVID conference series provides valuable insights into the performance of shot boundary detection algorithms. On the other hand, this comparison is restricted to those approaches which are submitted by their inventors to TRECVID for evaluation. However, in recent years many algorithms for shot boundary detection have been proposed in the literature, but to the best of our best knowledge, the last performance comparison study (apart from the

TRECVID conference series) was published in 2000, based on approaches which were published between 1993 and 1995. Hence, this chapter basically tries to answer the question: "Which shot boundary detection approach currently performs best?" In addition, a second research question is addressed which has not been investigated extensively yet: "how robust are different detection principles", that is how does the quality of shot boundary detection approaches depend on:

- different test sets,
- different training sets,
- parameter tuning, and
- particular videos.

4.3.2 The Evaluated Algorithms

To answer the questions stated above, five representative high-quality approaches with different functional principles have been selected: the "winner" of the last comparison study [57], namely the popular and well known local thresholding approach proposed by Yeo and Liu [189], an approach relying on advanced statistical methods [69], a proposal using fuzzy unsupervised learning [56], an approach that is based on selecting metrics which achieved the best separability of classes "transition" and "non-transitions" on a training set (extended by us with a supervised learning scheme), and an approach based on motion information in compressed MPEG videos [91]. The five algorithms used in our performance comparison study were described in more detail in the previous chapter of this thesis. Three of these five approaches also include a method to detect gradual transitions.

All mentioned approaches were (re)implemented by us to run under the same conditions, according to the descriptions provided by the authors in their publications. Four different TRECVID test sets (from 2004 to 2007) and the comprehensive MPEG-7 video test set, all including more than eighty videos (>35 hours), are employed to draw a fair comparison of the detection performance.

It should be mentioned explicitly that the subsequently reported results are obtained with our implementation of the algorithms under investigation. There are many possibilities to introduce an implementation bias such that we might not have achieved the possibly best results for a particular algorithm. Of course, we have tried our best to achieve the optimal performance for each of the algorithms, but we certainly cannot guarantee the same implementation. However, the use of the

TRECVID and MPEG-7 test sets and the available ground truth data allow other researchers to test their algorithms on the same test set.

4.3.2.1 THE VIDEO TEST SET

Publicly available test sets have been used in our experiments to allow other researchers further comparisons: four different TRECVID SBD test sets (from the years 2004 to 2007) and the MPEG-7 test set. Our experimental MPEG-7 test set consists of 33 MPEG-1 videos, excluding the video hallo.mpg and the surveillance videos "speedwa*.mpg" and "etri*.mpg" because they contain no perceptible shot changes. Furthermore, the uncompressed videos in the MPEG-7 test set have been excluded. The ground truth data of the MPEG-7 test set are available from the authors of [26]. Some errors in the ground truth data were corrected. Finally, our test set consists of ca. 30 hours of video containing 13667 cuts.

4.3.2.2 EXPERIMENTAL SETUP

The MDC decoder (Li and Sethi [94]) has been selected for MPEG video decoding. In our experiments, approximated DC-frames have been used for feature extraction in any test case. The estimation of recall and precision is according to TRECVID's evaluation for the shot boundary detection task. Gradual transitions shorter than five frames are considered as cuts, detected cuts are only matched against ground truth cuts and detected gradual transitions are only matched against ground truth gradual transitions.

4.3.2.3 TRAINING STAGE

As already mentioned, the approaches of Hanjalic and the SVM approach need a training stage. In the following, it is described how the parameters of these approaches are estimated during training.

4.3.2.3.1 Parameter Estimation for Hanjalic's Approach

The parameters for the distribution of inter-shot dissimilarities are estimated using the TRECVID 2004 SBD test set. The parameters are estimated by analyzing the training videos and calculating the corresponding histograms for frame dissimilarities. The parameters for the distributions of intra-shot and inter-shot dissimilarities are estimated based on these histograms. The conditional probability function is simply approximated by a straight line, and the parameter for the shot length, which is modeled via a Poisson distribution, is taken from Hanjalic's paper.

4.3.2.3.2 Parameter Estimation for the SVM Approach

The videos of a particular test set are used to train the SVM. For the RDF kernel, the parameters gamma and the regulatory factor c are automatically estimated. A range of [-16, 3] is investigated for gamma, while for c the range [-4, 8] is checked with a step width of 1 for each. The best combination of both values is estimated in order to achieve the possibly best accuracy for the training videos using three-fold cross validation.

4.3.3 Cut Detection Experiments

4.3.3.1 FIRST EXPERIMENT: PERFORMANCE COMPARISON

Our first experiment has been conducted to investigate the performance of the algorithms, using the same parameters and frame dissimilarity metrics as described in the corresponding papers (as far as possible). The most important parameters are listed below for each approach separately.

Algorithm A (Yeo/Liu [189]): The sliding window parameter (window size is $2m+1$) has been set to $m=10$ since the video material used in their paper had a frame rate of 15 frames per second, in contrast to the majority of TRECVID videos which have a frame rate 29.97. The thresholding parameter is set to $n=2$; three-dimensional RGB histograms are used.

Algorithm B (Hanjalic [69]): All parameters are set according to [69]. The distribution of intra-shot dissimilarity values is approximated by the parameter set (1.33, 4, 2), and the distribution of inter-shot dissimilarity values by the expected mean of 42 and a variance of 10 (multiplied with 16 in our implementation to be compliant). The shot length parameter μ is set to 100. Our implementation uses a simplified linear conditional probability function. The sliding window parameter is set to $m=10$. The frame dissimilarities are computed using motion-compensated pixel differences.

Algorithm C (Gao/Tang [56]): The range of values for defuzzification is set to [0.4, 0.6], the estimation of the local maximum in the defuzzifying stage is done within a window of size $2m+1$, with $m=5$. The parameter n for the sub-sampling to achieve the coarse temporal resolution is set to 10. Better results were achieved with the fuzziness set to 1.5 (instead of a fuzziness of 2 as used by Gao and Tang [56]).

Algorithm D (Kuo/Chen, [91]): The threshold parameter for decision making (minimum probability to represent a cut) is set to 0.4.

Algorithm E ([11]): The three-dimensional feature vector as proposed in the paper is used. A sliding window size of 2*10+1 is used for selecting negative training samples and for cut candidate selection during the tests.

The results for the different approaches on the TRECVID 2005 test set are presented in Table 4: F1, recall and precision are reported with respect to the total number of cuts as well as the mean and standard deviation of these measures with respect to the results for the particular videos. The shot boundary detections of the tested approaches are evaluated according to the TRECVID evaluation procedure [162]. Table 4 shows that the SVM approach with pre-selected metrics (E) achieves the best detection performance in terms of the f1-measure (90.4) in case that the parameters of the approaches are set as described in the related papers. The fuzzy clustering approach C and the statistical approach B work sufficiently well. The local thresholding approach A achieves a less satisfactory result. The approach E which is based on MPEG motion information lacks on precision. Obviously, for all approaches the results are worse than reported by the authors in their papers. As a consequence, additional experiments have been conducted in order to improve the results of all approaches by either training new models or adjusting parameter settings.

Results TRECVID 2005 [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)
A: Local thresholding [189]	64.4	57.0	74.0	64.9 (±19.7)	63.5 (±22.1)	78.9 (±23.6)
B: Likelihood ratio test [69]	82.2	87.1	77.8	82.1 (±11.9)	88.5 (±8.2)	79.1 (±17.7)
C: Fuzzy Clustering [56]	85.0	90.8	79.9	84.2 (±07.3)	89.9 (±05.3)	80.0 (±11.3)
D: MPEG data [91]	19.1	56.0	11.5	40.5 (±26.5)	57.2 (±12.1)	47.8 (±35.4)
E: Optimal disparities (SVM) [11]	90.4	87.4	93.7	90.8 (±04.0)	88.0 (±06.0)	94.0 (±03.0)

Table 4: Experimental results for cut detection on the TRECVID 2005 test data for the different approaches using the parameter settings as described in the related papers.

4.3.3.2 SECOND EXPERIMENT: PARAMETER OPTIMIZATION

In addition to the baseline "paper" experiment, experiments have been conducted with respect to different test sets and different trainings sets (for the approaches of supervised learning). The parameters and classification models have been adjusted or trained using the TRECVID 2004 test set. The parameters of the different approaches are adjusted as follows.

Algorithm A (Yeo/Liu [189]): The sliding window parameter is set to *m*=16, the thresholding parameter *n* is set to 1.23. This parameter combination yields a well balanced detection performance on the TRECVID 2004 data in terms of recall and precision (f1 is approximately 84.0). Three dimensional RGB histograms of DC frames and histogram intersection are used to compare subsequent frames.

Algorithm B (Hanjalic [69]): The distribution of intra-shot dissimilarity values inter-shot dissimilarity values is estimated using the TRECVID 2004 test set. Except for this parameter, the same parameters are used as in the baseline approach. The estimated parameters are as follows: the parameters for intra-shot distribution are (0.02, 1.36, 0.06), while the estimated parameters for the inter-shot distribution are (374.5, 160). The sliding window size parameter *m* is set to 10.

Algorithm C (Gao/Tang [56]): Since changing parameters did not improve the performance on the training data, the baseline system is used for further experiments.

Algorithm D (Kuo/Chen, [91]): The threshold parameter for decision making (minimum probability to represent a cut) is set to 0.5, a sliding window size of 3 is used.

Algorithm E (Bescos, SVM extension of [11]): The parameters of the baseline model are not changed. The learned model is the same as in the baseline system. However, a sliding window size of size 2m+1 (m=10) is used.

The results for the TRECVID SBD test set 2005 using the optimized or learned parameters are presented in Table 5. The mean values for recall, precision, and f1 are displayed as well as their mean and standard deviation (with respect to the results for the particular videos) on this test set of videos. Also, for each approach the worst performance for a video in the test set is listed in terms of f1-value ("Min. F1"). The SVM-based approach E outperforms the other approaches clearly on the TRECVID 2005 SBD test set, its f1-measure of 90.4 is noticeably better than the second best f1-measure of 84.0 (C). The second best approaches A (Yeo and Liu [189]), B (Hanjalic [69]) and C (Gao and Tang [56]) achieve similar performance in terms of the f1-measure (Table 5). Again, approach D achieves an unsatisfactory result.

In addition, experimental results for the TRECVID test sets of 2006 and 2007 are presented in Table 6 and Table 7, and for the MPEG-7 test in Table 8. For the TRECVID 2006 test set, the SVM-based approach E again outperforms all other approaches in terms of the f1-measure, whereas approach B and C achieve a comparable performance on TRECVID 2007 test data. It is obvious that the performance of approach D strongly depends on the video test set: its f1-measure changes from 3.2 up to 50.2 depending on the 2006 and 2007 test set. A first analysis showed that the encoder of the NASA videos seemed to prefer a temporal direction for bi-directional coded inter-frames (B-frames). Such encoding effects are not considered by approach D which makes its detection performance rather unstable. Interestingly, the best approach fails completely for one video of the TRECVID 2007 test set. This video consists only of black-and-white shots but one of the features used by algorithm E [11] is based only on the chrominance color channels.

Results TRECVID 2005 [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)	Min. F1
A: Local thresholding [189]	82.8	84.8	81.0	82.4 (±07.4)	86.4 (±08.9)	79.4 (±08.7)	69.7
B: Likelihood ratio test [69]	82.3	72.8	94.8	84.0 (±07.7)	75.8 (±10.9)	94.9 (±03.0)	66.7
C: Fuzzy Clustering [56]	84.0	90.5	78.3	83.5 (±08.0)	89.8 (±05.1)	78.9 (±12.3)	69.5
D: MPEG data [91]	19.7	67.4	11.6	32.6 (±18.3)	68.3 (±16.7)	25.8 (±16.3)	3.7
E: Best disparities (SVM) [11]	90.4	87.4	93.7	90.8 (±04.0)	88.0 (±06.0)	94.0 (±03.0)	82.6

Table 5: Experimental results for cut detection on the TRECVID 2005 test data for the different approaches using optimized parameter settings.

Results TRECVID 2006 [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)	Min. F1
A: Local thresholding [189]	76.4	75.8	77.1	75.3 (±10.7)	74.1 (±13.7)	78.2 (±12.1)	57.1
B: Likelihood ratio test [69]	70.4	56.9	92.3	68.9 (±10.6)	56.0 (±12.4)	91.8 (±06.2)	52.5
C: Fuzzy Clustering [56]	76.2	83.2	70.9	75.8 (±07.5)	82.6 (±08.5)	70.3 (±08.1)	60.2
D: MPEG data [91]	50.2	66.8	40.2	49.8 (±06.1)	67.8 (±13.2)	40.4 (±06.4)	42.1
E: Best disparities (SVM) [11]	80.0	71.9	90.1	77.5 (±11.9)	70.0 (±16.1)	89.1 (±04.8)	50.3

Table 6: Experimental results for cut detection on the TRECVID 2006 test data for the different approaches using optimized parameter settings.

Results TRECVID 2007 [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)	Min . F1
A: Local thresholding [189]	87.6	93.9	82.0	86.0 (±09.9)	93.5 (±11.6)	80.6 (±11.5)	55.3
B: Likelihood ratio test [69]	92.2	87.2	97.8	92.1 (±07.0)	87.6 (±09.9)	97.4 (±03.7)	70.9
C: Fuzzy Clustering [56]	92.0	91.1	92.9	91.4 (±06.4)	90.4 (±07.5)	92.6 (±06.1)	71.1
D: MPEG data [91]	3.2	86.5	1.6	3.7 (±01.5)	85.9 (±06.6)	1.9 (±00.8)	1.3
E: Best disparities (SVM) [11]	92.9	89.9	96.7	89.1 (±23.2)	88.3 (±23.4)	90.1 (±23.2)	-

Table 7: Experimental results for cut detection on the TRECVID 2007 test data for the different approaches using optimized parameter settings.

Results MPEG-7 [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)	Min. F1
A: Local thresholding [189]	89.0	90.1	87.8	86.7 (±09.7)	90.3 (±11.2)	83.9 (±13.0)	55.6
B: Likelihood ratio test [69]	89.1	81.5	98.2	89.0 (±09.6)	83.7 (±14.0)	83.7 (±14.0)	60.0
C: Fuzzy Clustering [56]	76.1	90.7	65.6	78.8 (±21.2)	89.4 (±08.7)	76.3 (±27.0)	19.8
D: MPEG data [91]	7.5	79.5	3.9	25.6 (±31.2)	83.0 (±17.7)	24.2 (±33.8)	1.3
E: Best disparities (SVM) [11]	90.1	86.4	94.1	91.2 (±10.4)	87.9 (±14.2)	94.8 (±07.2)	58.6

Table 8: Experimental results for cut detection on the MPEG-7 video test set for the different approaches using optimized parameter settings.

ALL Test Sets [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)	Min. F1
A: Local thresholding [189]	84.7	86.5	83.0	83.9 (±10.3)	87.3 (±13.0)	81.4 (±11.7)	55.3
B: Likelihood ratio test [69]	84.6	75.3	96.4	85.4 (±11.9)	78.5 (±16.5)	95.7 (±04.9)	52.5
C: Fuzzy Clustering [56]	80.4	89.2	73.3	81.9 (±15.8)	89.5 (±08.3)	79.4 (±20.3)	19.8
D: MPEG data [91]	7.8	75.1	4.1	26.0 (±26.3)	78.6 (±16.5)	22.2 (±26.3)	1.3
E: Best disparities (SVM) [11]	88.7	84.3	93.6	87.9 (±14.6)	84.9 (±17.4)	92.6 (±12.5)	-

Table 9: Experimental results for cut detection on the MPEG-7 video test set for the different approaches using optimized parameter settings.

On the MPEG-7 test set, the statistical approach B outperforms the other approaches with an f1-value of 89.1. This is the only test set for which the fuzzy clustering approach is significantly worse than other approaches (f1: 76.1). Its decreased performance is caused by six videos for which only a precision of less than 50% could be achieved.

Results TRECVID 2005 "NEWS" [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)
A: Local thresholding [189]	80.9	79.4	82.6	83.3 (±7.3)	83.9 (±11.7)	83.7 (±07.0)
B: Likelihood ratio test [69]	78.1	67.0	93.5	78.1 (±6.8)	67.0 (±08.8)	93.5 (±02.6)
C: Fuzzy Clustering [56]	81.6	88.9	75.4	82.9 (±7.0)	88.8 (±05.3)	78.3 (±10.4)
D: MPEG data [91]	45.0	57.8	36.8	44.5 (±5.7)	58.2 (±08.7)	36.3 (±05.8)
E: Best disparities (SVM) [11]	88.6	85.1	92.3	90.0 (±4.3)	87.3 (±06.6)	93.0 (±02.9)

Table 10: Experimental results for cut detection on the news video part of the TRECVID 2005 test data for the different approaches using optimized parameter settings.

Results TRECVID 2005 "NASA" [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)
A: Local thresholding [189]	89.3	95.2	84.1	81.8 (±5.6)	92.7 (±19.6)	78.8 (±13.8)
B: Likelihood ratio test [69]	91.8	86.9	97.4	91.8 (±4.2)	86.7 (±03.5)	96.6 (±01.6)
C: Fuzzy Clustering [56]	90.1	94.5	86.1	84.6 (±4.9)	91.8 (±17.2)	80.0 (±10.8)
D: MPEG data [91]	10.5	91.1	5.6	9.0 (±5.3)	88.5 (±02.8)	4.8 (±05.0)
E: Best disparities (SVM) [11]	95.0	93.1	96.9	92.6 (±3.2)	89.5 (±05.1)	95.9 (±02.4)

Table 11: Experimental results for cut detection on the NASA video part of the TRECVID 2005 test data for the different approaches using optimized parameter settings.

The TRECVID 2005 test set is of special interest for another reason. It consists of two different video sources, namely news videos and NASA documentary videos. Due to this, the results for these two subsets are reported in Table 10 and Table 11. It is observable that the approaches perform differently on these two parts of the test set. The local thresholding appracoh A and the fuzzy clustering approach perform best on the NEWS data set, whereas the statistical approach B is better on the NASA data set.

To summarize, the statistical approach and the local thresholding approach outperform the other approaches. Although the statistical approach B has some deficiencies in terms of recall, its precision is outstanding: clearly above 90% for each test set. If one considers only the results for the TRECVID test (2005, 2006, and 2007), the fuzzy clustering approach C is slightly better than the approaches B and C. However, on the MPEG-7 test set, the performance is rather bad caused by six videos for which the approach produced a large number of false alarms. Except for the TRECVID 2006 test set, this approach achieves a recall of above 90% for all other test sets. Using the parameter adjustments of the TRECVID 2004 test set, the local thresholding approach works surprisingly well on the different test sets. It belongs to the top approaches on the test sets of TRECVID 2005, 2006 and MPEG-7 video test set. Another interesting question is how much the quality of an approach degrades on a particular video. Regarding the top approaches, it is observable that their worst result on a particular video is above 50.0 on the TRECVID test sets, for the fuzzy clustering approach it is even above 60.0 for all TRECVID videos. However, the worst performance for this approach is an f1-value of 19.8 on the MPEG-7 test set. In this respect, approach A and B are more robust: their worst f1- value on a particular video of all test sets is 52.5 and 55.3, respectively.

Finally, the impact of selecting a particular training set is investigated for the three best approaches which need training or parameter adjustment. Therefore, the performance on the TRECVID 2007 data was measured depending on three different training sets. The results for the approaches A, B and E are presented in Table 12, Table 13 and Table 14. The statistical approach is more robust with respect to the selection of training data, its f1-measure varies between 92.2 and 93.8, whereas the f1-measure for approach E lies between 90.0 and 92.9. Approach B is very stable with respect to precision but recall varies between 92.2% to 93.8%. Interestingly, also the approach of Yeo and Liu which has to rely on adequate parameter setting achieves very constant results. The estimated parameters seem to work well for all TRECVID sets from 2004 to 2007.

Alg. A: Local Thresholding [189] Results TRECVID 2007 [%]	F1	Recall	Prec.
Training Set 2004	87.6	93.9	82.0
Training Set 2005	88.6	93.8	84.0
Training Set 2006	87.2	94.1	81.3

Table 12: Experimental results on the TRECVID 2007 test data for the local thresholding approach depending on the used training set for learning.

Alg. B: Likelihood ratio test [69] Results TRECVID 2007 [%]	F1	Recall	Prec.
Training Set 2004	92.2	87.2	97.8
Training Set 2005	93.8	90.5	97.4
Training Set 2006	92.4	87.6	97.8

Table 13: Experimental results on the TRECVID 2007 test data for the statistical approach depending on the used training set for learning.

Alg. E: Optimal disparities [11] Results TRECVID 2007 [%]	F1	Recall	Prec.
Training Set 2004	92.9	89.9	96.1
Training Set 2005	91.7	87.4	96.5
Training Set 2006	90.0	84.7	95.9

Table 14: Experimental results on the TRECVID 2007 test data for the approach of Bescos [11] depending on the used training set for learning.

4.3.4 Gradual Transition Detection Experiments

Three of the investigated five approaches include a generic gradual transition detection algorithm, except Hanjalic's and Kuo and Chen's approach, which include only a dissolve detector. In addition to the cut detection experiments, the gradual transition detectors of the remaining three approaches have been tested on the TRECVID 2005 (1155 gradual transitions) and TRECVID 2007 (206 gradual transitions) test set. The parameters of Gao and Tang's approach are the same as for cut detection, the SVM models for Bescos' approach are learned for the following frame distances: 5, 10, 15, 20, 25, 30, 40, and 50. The parameters for the approach of Yeo and Liu are: frame distance is 20, the plateau width is 3, and the local thresholding parameter is set to 2.8. The results are presented in Table 15 and Table 16.

Results TRECVID 2005 [%]	F1	Recall	Prec.
A: Local thresholding [189]	40.4	34.9	47.9
C: Fuzzy Clustering [56]	2.0	1.0	66.7
E: Best disparities (SVM) [11]	22.5	36.7	16.2

Table 15: Experimental results for gradual transition detection on the TRECVID 2005 data, according to TRECVID evaluation.

Results TRECVID 2007 [%]	F1	Recall	Prec.
A: Local thresholding [189]	40.0	41.3	38.8
C: Fuzzy Clustering [56]	-	0.0	-
E: Best disparities (SVM) [11]	9.3	34.0	5.4

Table 16: Experimental results for gradual transition detection on the TRECVID 2007 data, according to TRECVID evaluation.

It is obvious that the results for gradual transition detection are much worse than for cut detection. The parameter approach of Yeo and Liu [189] achieves the best results for both test sets. It is aimed at detecting plateaus in a time series with a higher temporal distance. The fuzzy clustering approach C is practically unable to detect gradual transitions: its recall is 0% or 1%, respectively. Bescos' approach achieves the best result on the TRECVID 2005 data in terms of recall but is less precise than approach A on both test sets. However, it has been observed that the results change significantly in case when also short transitions are considered as gradual transition. This is in contrast to the TRECVID evaluation but nevertheless these results are reported in Table 17 and Table 18 since they give additional insights into the capabilities of the approaches. It becomes clear that all approaches A and C obtain better results for detecting short gradual transitions, except for Bescos' approach E. The fuzzy clustering approach seems to detect *only* short gradual transitions as reflected by the 1% recall in the TRECVID compliant experiment. Furthermore, Yeo and Liu's approach (A) detects short gradual transition more often and more precisely than long transitions. Interestingly, our implementation of A achieves a much better result for gradual transition than reported in the last comparison study of Gargi et al. [58]. Gargi et al. report a recall of 31% and precision of 8% for their implementation of A and their test set (265 gradual transitions).

Results TRECVID 2005 [%]	F1	Recall	Prec.
A: Local thresholding [189]	51.9	42.5	66.7
C: Fuzzy Clustering [56]	37.0	25.5	67.3
E: Best disparities (SVM) [11]	16.6	34.5	10.9

Table 17: Experimental results for gradual transition detection on the TRECVID 2005 data. In addition, short gradual transitions (with at least one transitional frame involved) are considered, in contrast to the results in Table 15 and Table 16.

Results TRECVID 2007 [%]	F1	Recall	Prec.
A: Local thresholding [189]	47.5	52.0	43.7
C: Fuzzy Clustering [56]	5.5	4.0	8.9
E: Best disparities (SVM) [11]	4.9	43.6	2.6

Table 18: Experimental results for gradual transition detection on the TRECVID 2005 data. Also short gradual transitions (with at least one transitional frame involved) are considered, in contrast to the results in Table 15 and Table 16.

4.3.5 Summary

In this section, several shot boundary detection algorithms with different functional principles have been comprehensively compared on four different test sets. Overall, for cut detection, our SVM realization of Bescos' approach achieves the best cut detection performance (f1: 88.7), whereas the statistical approach of Hanjalic [69] and the local thresholding approach of Yeo and Liu [189] show the second best performance (f1: 84.7 and 84.6). The fuzzy clustering approach has the advantage that no training data are needed. However, its performance for gradual transition detection is rather weak, the best approach is the one of Yeo and Liu which obtains a best f1-value of 51.1 on the TRECVID 2007 data.

The experiments demonstrate that all approaches lack robustness and their performance depends more or less on the test data or training data. This is also true for the best approach. Interestingly, the best approach E does not detect any cuts in an old grayscale video of the TRECVID 2007 test set. Furthermore, the impact of training set selection has been shown empirically for the best supervised approaches. While the choice of the training set had only little impact for Hanjalic's approach, the f1-measure varied for the SVM approach between 90.0 and 92.9 on the same test set.

4.4 Video Cut Detection and Compression Artifacts

4.4.1 Introduction

Several research efforts have been made in recent years to address the problem of detecting shot boundaries in digital videos, also in the compressed video domain. This chapter focuses on the analysis of frame dissimilarity measurements for MPEG encoded videos. Two techniques called "Frame Difference Normalization" (FDN) and "GOP-oriented frame difference normalization" (GOP: group of pictures), respectively, to enhance the performance of cut detection algorithms are presented. Furthermore, the proposed methods are not limited to a particular algorithm but are applicable to an entire class of cut detection algorithms.

One could argue that the problem of detecting cuts has been solved satisfactorily since many researchers reported very good recall and precision rates ranging up to 100% and up to 97% on common standard test sets (refer to performance analysis in the previous section). However, for example in the MPEG-7 test set which is used in our experiments there are many MPEG videos for which a well known high-quality cut detection algorithm [189] produces a systematic detection error. This kind of error has not been investigated previously, though related effects are mentioned in a few papers. For example, Smith et al. mention [144] that shot detection errors of their system were partially caused by a periodic noise pattern in dissimilarity time series at a period of 15 frames (one GoP), but no further analysis of this problem is provided. Gargi et al. [58] investigate the impact of source effects, the use of different encoders and bit rates. They show that the performance of all investigated shot detection algorithms varies depending on these factors, in particular for those approaches which use MPEG macroblock information.

In this chapter, it will be shown that there is often a bias in frame differences depending on MPEG specific frame types and propose the "frame difference normalization" (FDN) technique to reduce such "noise" patterns. Furthermore, it will be shown that frame difference measurements can vary systematically even for identical frame type transitions depending on their relative position within a GOP. Another method called "GOP-oriented frame difference normalization" (GFDN) to handle such errors is proposed. The basic idea of our approach is to adequately normalize frame dissimilarities with respect to certain compression artefacts in order to increase the detection quality. The performance of our proposal will be demonstrated by presenting experimental results for the MPEG-7 test set which is well suited for evaluation of this kind since it includes videos from several encoding sources. It is worth mentioning that the approach is general in the sense that it can be used in conjunction with an entire class of cut detection algorithms, namely those based on pixel-wise or histogram-based frame difference metrics and operating on MPEG videos. The work presented in this chapter has been partially published in [36, 37].

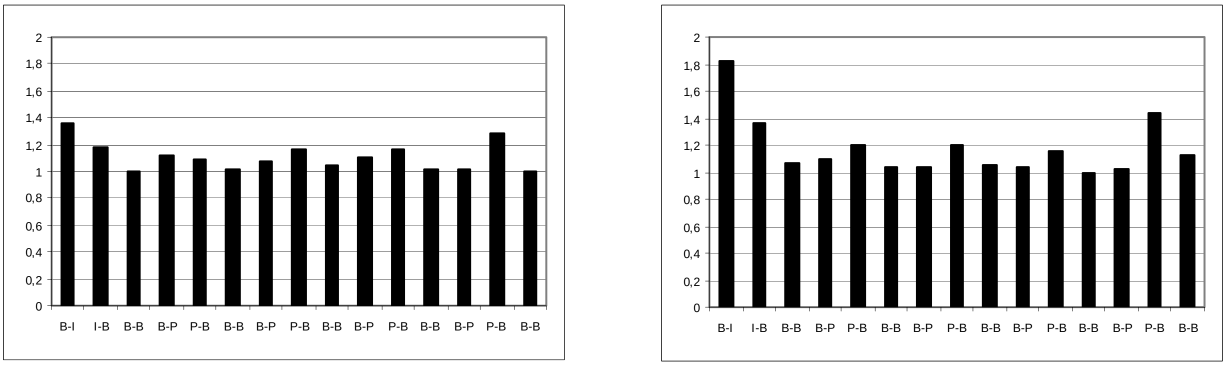


Figure 24, left: Average difference values for different GOP positions normalized in relation to the minimum difference for “riscos-sl.mpg”, using decoded full frames. Right: Average difference values for the same video, normalized in relation to the minimum difference as well, but approximated DC-frames were used.

4.4.2 Problem Analysis

MPEG encoders try to achieve a certain bit rate so there is not an arbitrary number of bits an encoder could use to encode a frame. Since an I-frame is encoded without any reference frame, the number of required bits is proportionally high compared to P- or B-frames (inter-coded frames). Otherwise, the degree of accuracy can vary depending on the frame type due to effects of motion compensation (e.g., inaccuracy or quantization of macroblock differences). For example, information loss might be large in I-frames since no reference frame can be used to encode this current frame. On the other hand, the number of available bits to encode this frame is restricted. Thus, the only way to control this restriction is the quantization process because variable run length encoding can not arbitrarily reduce the number of coding bits. Due to quantization, pixel information is lost and image quality usually decreases. But the image quality of the next inter-coded frame usually benefits from motion compensation and usually approximates the original frame content better than intra-coded frames. Furthermore, the number of bits usable for P-frame and B-frame encoding are restricted, too, such that an error propagation is possible for these frame types as well.

Thus, often a bias can be found in the estimated frame differences in MPEG videos. This bias depends on the properties of the frames that were involved in the calculation. The difference values can be estimated e.g. either with histogram or pixel based metrics. For a commonly used IPB-pattern like “IBBPBBPBBPBBIB...” there are at least five different frame type transitions: I to B, B to B, B to P, P to B and B to I. For example, for the video sequence “riscos-sl.mpg” from the mentioned MPEG-7 test set the average difference values for specific frame transitions were calculated. These average values varied from 140 (B to B) up to 240 (B to I). The difference values result from comparing histograms with 512 bins (YUV-combined) for approximated DC-frames. However, deeper analysis reveals that not only the involved frame types play an important role but also the relative frame positions within a GOP (group of pictures). For example, in Figure 24 the average frame differences can be seen for each relative GOP position calculated for the video “riscos-sl.mpg”. These values are normalized in the way that each value is divided by the minimum average. The following properties can be observed in Figure 24:

1. Two of the three largest peaks can be found for the transitions involving an I-frame.

2. The difference between two consecutive B-frames is consistently small.

3. The differences for the more frequent transition types vary depending on their GOP position.

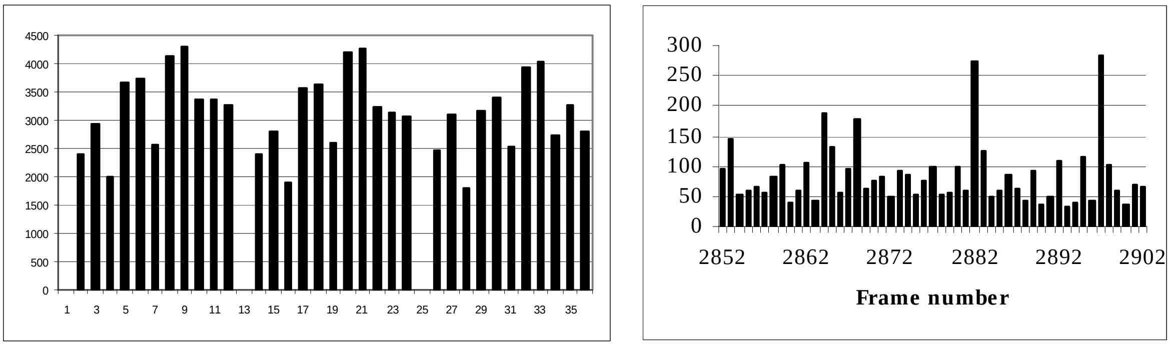


Figure 25, left: Difference for approximated DC-frame and DC-frames that are reconstructed based on the fully decoded frame. Right: Example for a possible false alarm detection at frame 2882: there are nearly no changes between frame 2881 and 2882 but frame 2882 is an I-frame.

The first property is not surprising since an I-frame is encoded without any reference frame. Thus, it is obvious that there are noticeable differences to the neighboring B-frames which were rebuilt based on prediction and motion compensation. Also, for the second property it is understandable that there are no noticeable differences between frames having the same frame type.

The third property is mainly observable for the transition type P-B in this example. The last P-B transition has the largest difference value for this kind of transition type. This can be explained by the subsequent I-frame because the B-frame can refer to this I-frame for motion compensation purposes as well as to the previous P-frame.

The effects described so far are potentially present in any MPEG video but their strength may vary strongly depending on the encoder used, bit rate and so on. Another reason for this is that an encoder’s motion estimation method is not defined but only the bitstream syntax is defined. Obviously, for a given bitrate, the encoding quality depends on the quality of the motion estimation process. Also, there is an error propagation within a GOP from P to P frame, and of course, also from P to B frame when a B-frame refers to a P-frame. Furthermore, although prediction error data will be encoded too, these data can not balance the errors because they are usually quantized.

The effects described above are enforced or overlaid when approximated DC-frames are used (see Figure 24 and Figure 25). It can be easily understood by considering the way approximated DC-frames are reconstructed. In general, motion vectors do not point exactly to DCT block boundaries where each block has size 8*8 pixels in MPEG format and is represented by DC coefficients in DC frames. If a DC value has to be reconstructed using a motion vector, in general this vector points to a block position including parts of the four neighboring DCT blocks. The first-order approximation suggested by Shen and Delp [139] and Yeo and Liu [188], respectively, uses a weighted sum of the DC values of the four neighboring blocks, and thus produces an (tolerable) error. In Figure 25 this effect is demonstrated by comparing the pixel differences between approximated DC-frames and the corresponding exactly reconstructed DC-frames belonging to the same video. It is obvious that there are no differences between two I-frames but an increasing error from P to P frame within a GOP (the GOP size is 15 here) until the next I-frame is involved.

The consequences for histogram and pixel based cut detection algorithms can be pointed out more clearly now. Let us assume that the cut detection algorithm, which has been proposed by Yeo and Liu [189], is used with parameter m=5 and the threshold value n=2 that in general gives very good detection results. The frame difference position with the maximum value within a sliding window is accepted as a cut if it is n times larger than the second largest value inside this window. In Figure 4, the histogram differences are shown for a nearly completely uneventful scene without any cut. Since the maximum difference between frame 2881 and 2882 (B->I transition) is more than n times larger than the second largest peak, our detector concludes that there is a cut and thus produces a false alarm. Many false alarms of this kind have been found in different videos. Instead of just increasing the parameter n which would result in a lower recall rate, a method is needed to handle such false alarms without reducing the recall rate significantly.

The issue is that frame similarity sometimes does not only measure the similarity of frame content but also frame changes due to compression artefacts described above. Thus, it can be assumed that the performance of most of the cut detection algorithms mentioned in the previous section is likely to decrease if they are applied to the kind of MPEG videos exhibiting the noise pattern as described above. They are potentially affected because these algorithms use either histogram or pixel based metrics for frame differences.

In the following two sections, two methods are presented to eliminate the kind of errors described above: "Frame Difference Normalization" (FDN) and GoP-Oriented Frame Difference Normalization (GFDN).

4.4.3 FDN: Frame Difference Normalization

For FDN, a set *S* of tuples of possible frame transition types is defined where I, P and B match the MPEG frame types:

$$S = \{(I, I), (I, P), (I, B), (P, I), (P, P), (P, B), (B, I), (B, P), (B, B)\} \tag{49}$$

Let *s* be in *S*, and $d_{i,s}$ the frame difference between frame *i* and *i*+1 with the frame type transition *s* at this position. Now, the average difference value for each specific frame type transition *s* is calculated. Let *f* be the number of frames in a given video, and let f_s be the number of frame transitions of type *s*, then the average value for a frame transition type s is estimated as shown in the following formula, where *p* is a number between 0 and 1.

$$avg_s = \begin{cases} \infty, & if\ \dfrac{f_s}{f} < p, \\ \dfrac{\sum_{i=1}^{f-1} d_{i,t_i},\ t_i = s}{f_s}, & else. \end{cases} \tag{50}$$

The purpose of *p* is as follows: Sometimes, untypical transitions occur for a specific IPB-pattern because an MPEG-1 encoder can insert an I-frame at an arbitrary position which would potentially cause new transitions like P to I. If the encoder inserts an I-frame and potentially produces an untypical frame type transition with a high difference value, this could cause a very high average value for this transition type. Thus, it is suggested to treat only those transitions that occur with a minimum frequency, such as 5% of all frame transitions (*p*=0.05). When the set of averages S_{avg} has been estimated in this way, the maximum value in S_{avg} is chosen as the baseline value *max*. Now, in order to obtain the normalizing factor for each transition type *s*, *max* is divided by avg_s:

$$normfactor_s = \begin{cases} 1, & if\ \#d_s < p, \\ \dfrac{\max(S_{avg})}{avg_s}, & else. \end{cases} \tag{51}$$

where *s* is a certain transition type and S_{avg} is the set of average differences for the different transition types. Then, each difference value is normalized by multiplying it by the appropriate normalization factor for its transition type (FDN). Finally, each $d_{i,s}$ at absolute frame position *i* of the difference sequence has to be multiplied with the appropriate $normfactor_s$ to obtain the normalized sequence *d**:

$$d^{*}_{i,s} = d_{i,s} * normfactor_{s} \quad (52)$$

The sequence *d** is then used in a histogram or pixel based cut detection algorithm to decide whether there is a cut or not.

4.4.4 GFDN: GoP-Oriented Frame Difference Normalization

The analysis of frame difference characteristic in MPEG videos in the previous section shows that there are significant differences for the same transition type within a GOP depending on its relative GOP position. Thus, it is adequate to consider the relative position of a frame type transition within a GOP. A new method to handle the MPEG-specific bias called "GOP-oriented frame difference normalization" (GFDN) is proposed in this section. GFDN works as follows. First, the average difference value is estimated for each difference position within a GOP. Therefore, the first I-frame in the video sequence is located to have a good starting point and a variable index is set to 0. This variable represents the relative GOP position of the first frame involved in frame difference measurement. The frame differences for each relative GOP position (index) are added up to calculate the average difference for each relative position. The average differences are then used to estimate the normalization factors for each relative GOP position but only if the frequency of a frame transition type exceeds a threshold *p* (e.g., let f be 3-4 % of the total number of frames). The normalization factor for each relative position *s* is the maximum average divided by the average value of *s*, according to (52) but *s* now refers to the position within the GoP and does not refer to a certain frame transition type, as in the FDN method. The procedure is presented in Figure 26 in pseudo code. FDN and GFDN can be applied to both MPEG-1 and MPEG-2 videos.

```
Input:  Array of frame dissimilarity values
Output: Array of normalized frame dissimilarity values

Algorithm

doGOP-OrientedNormalization (double [] diff)
{
 // create arrays sum, avg and count of size GOPsize
 // find first I-x transition in video, save
 // difference position in iStartDiff
 // set sum[j] == 0 and for all j

 iIndex = 0;
 for each diff[i] with i >= iStartDiff
  String diffType = difftype(i);
  // to handle arbitrarily occurring I-frames

  if (secondChar(diffType) == "I" && iIndex != GOPsize-1) then
   iIndex = 0;
  else
   sum [index] += diff[i];
   count[index]+= 1;
   if (secondChar(diffType) == "I") then
    iIndex = 0;
   else
    iIndex++;

  i++;

 // calculate avg value for aech GOP position
 for each GOP-position index
   avg[index]= sum[index] / count[index];

 // use avg[] to calculate normalizing
 // factors array normfactors []
 // that will be used to normalize diff values

 float [] normfactors = getNrmFactors(avg[]);

 return normalize(diff[], normfactors []);
}
```

Figure 26: Pseudo code for GoP-oriented frame difference normalization.

4.4.5 Experimental Results

The MPEG-7 test set is well suited to test the proposed methods since the videos are from different providers and encoded with different encoders. Our experimental test set consists of 33 MPEG-1 videos, excluding the surveillance videos "speedwa*.mpg" and "etri*.mpg" because they do not contain perceptible shot changes. Also, the uncompressed videos have been excluded because the proposed technique is only relevant for the compressed video domain. Finally, our test set consists of about 12 hours of video containing over 5100 cuts.

Unfortunately, some errors in the ground truth data (kindly provided by Chua et al. [26]) had to be corrected for abrupt changes due to the following reason. The three decoders (MDC decoder, Java Media Framework decoder and the freeware VirtualDub) that have been used showed time codes for cuts that differed for more than tens of frames from the ground truth data. Furthermore, our estimation of recall and precision accepts a cut as detected correctly only for a deviation of one frame.

In our experiments, the MDC decoder [94] has been used for MPEG decoding. While this is trivial for I-frames, an approximation for P- and B-frames as described by Yeo and Liu [188] has been used. Furthermore, the cut detection algorithm of Yeo and Liu [189] has been implemented. The implementation works as follows: For two consecutive DC frames, the frame differences are estimated using a histogram metrics. Best results were obtained with a color histogram with 512 bins (9 bits) where the Y, U and V components of each pixel have been quantized from eight to three bits. Each bin represents a combination of three bits for Y, U and V. A frame transition is accepted as a cut if the transition considered has the maximum difference value within the sliding window of size 2**m* and is n times larger than the second largest value within the window. From now on, this implementation is called "Yeo/Liu implementation" (Y/L). This algorithm has been chosen due to its popularity and its superior performance in a comparison study [58].

The Yeo/Liu implementation has been extended both with the proposed FDN and the GFDN techniques. All implementations were tested with different threshold values, since in general there is no possibility to estimate an optimum threshold.

The results for the different tests are displayed in Table 19, where the number of really existing cuts, the number of detected cuts, the number of false alarms, the total number of errors (missed cuts plus false alarms), and recall and precision in % are listed.

GFDN leads to a significant error reduction and is superior to the FDN noise reduction technique and the baseline implementation. Table 19 demonstrates that there is a large reduction of false alarms in case of *n*=2 while keeping a high number of correctly detected cuts is kept. This is an important aspect for those cases where a high rate of correct detections is desired. For example, if a supplementary manual correction is possible with an appropriate software tool, it is easier to delete false alarms than detecting missed cuts. As shown in Table 19, the precision is improved by over 6% while recall only decreases by 0.3%. In the case of *n*=2.5, the total number of errors is reduced noticeably, too, where for *n*=2 the reduction of errors is significant. The GFDN method leads to lower error rates than the baseline implementation. Thus, for MPEG videos GFDN is the most effective way to systematically reduce noise in order to enhance precision.

Implementation Threshold	Y/L n=2	FDN n=2	GFDN n=2	Y/L n=2.5	FDN n=2.5	GFDN n=2.5
No. of cuts	5164	5164	5164	5164	5164	5164
Detected	4866	4854	4811	4735	4719	4724
False pos.	1312	1179	890	431	476	360
Errors	1610	1489	1163	860	921	800
Recall %	94.2	94.0	93.9	91.7	91.4	91.5
Precision %	78.8	80.5	85.1	91.7	90.8	92.9

Table 19: Results are presented for the MPEG-7 test set for both methods: FDN and GFDN. GFDN outperforms the baseline implementation and the FDN method.

4.4.6 Summary

In this section, it was shown that pixel or histogram based frame difference measurements in MPEG videos can vary systematically for identical frame transition types and even for identical frame transition types depending on their relative position within a group of pictures. The consequences for cut detection were explained, and two new methods called "Frame Difference Normalization" (FDN) and "GOP-oriented frame difference normalization (GFDN)" to handle such errors were proposed. GFDN is a refinement of the FDN technique and both have been tested on the large MPEG-7 video test set based on 33 videos. In the experiments, the new GFDN technique led to significant improvements in precision of a "classical" cut detection algorithm. Overall, the systematic noise reduction in MPEG videos with GFDN was superior to FDN due to a more adequate denoising method. The advantage of GFDN for MPEG videos is that it reduces a specific noise pattern systematically whereas a general noise filter would smooth all difference values in the same way. Furthermore, GFDN can be added to a number of different cut detection algorithms, such as [11] , [27] , [69], [139], [155], and [189].

4.5 UNSUPERVISED SHOT BOUNDARY DETECTION

4.5.1 INTRODUCTION

At least one of the following drawbacks can be found in many cut detection approaches proposed in the literature. First, many of them have to be tuned for a given test set: either by estimating the best set of parameters and thresholds, or an approach has to undergo a supervised training process to achieve the optimal detection performance. Second, in case when supervised learning is applied, labeled training data must be created, and this is a time consuming task. Third, if a clustering algorithm is used, either the feature vectors are too simple or additional thresholds are involved. Additionally, often only small or unavailable video test sets have been used which makes comparisons difficult.

Detecting gradual transitions is a surprisingly difficult problem in practice. For example, recall and precision of the best approaches evaluated at TRECVID 2006 [141] are about 10-20% lower than for cut detection. There are three main reasons for that. First, many different types of gradual transitions exist: dissolve, fade-in and fade-out, wipe, and many other effects which eventually use motion or 3D-effects. Second, the length of a gradual transition can be arbitrary, it ranges from one frame up to dozens of frames. Third, often the shot content changes slightly due to camera or object motion which causes many false alarms for state-of-the-art gradual transition detectors. In recent years, two kinds of approaches have emerged. The first type of approaches designs different detectors for different transition types. There are proposals for specialized dissolve detectors (e.g., [69, 97, 98]), fade detecors (e.g., [164]), and wipe detectors (e.g., [126]). The other class of approaches relies on the application of a general gradual transition detector (e.g., [11, 22, 27, 197]). While the use of specialized detectors promises to find corresponding effects with a higher accuracy, the development, adaptation and training efforts increase for this kind of approach. In addition, it is not clear how to deal with transition types which do not fall in one class of the available detector types. As shown in Table 1, other transition types might be up to 10% of all transitions in some applications. Thus, it is difficult to design a generic approach that works for any transition type.

Furthermore, it is known that camera motion causes a lot of detection failures for gradual transition detectors [93]. Despite this, many approaches do not incorporate motion information or an appropriate false alarm removal [11, 69, 189, 195]. Other approaches use motion vector information present in compressed videos. Zheng et al. [197] check a gradual transition candidate with respect to camera motion based on motion vector analysis. Chua et al. [27] consider motion information in their feature vector based on compressed motion vector information, too.

In this section, a novel unsupervised shot boundary detection approach is presented. It is a unified approach that applies similar principles to detect cuts and gradual transitions. The proposed cut detection approach avoids any parameter or threshold setting. To achieve this aim, the cut detection approach consists of the following processing steps: First, a sliding window technique is applied to dissimilarity values of subsequent frames. Then, each sliding window is represented by an adequate feature vector which is normalized individually for each video. A sliding window is only a cut candidate if it has the maximum difference in the middle, and is thus used in the next step as input to a k-means clustering algorithm. In case of cut detection, the number of classes is known: cuts and non-cuts. Then, the corresponding classes are initialized with the candidates which are most similar to an ideal cut or to an ideal non-cut representative, respectively. The clusters are filled and optimized using the classical k-means algorithm. These steps are repeated for a reasonable range of sliding window sizes *m*. Finally, the quality of each clustering is evaluated using the silhouette coefficient, and the best "cuts" cluster is chosen as the detection result. In this way, the only remaining parameter *m* is estimated automatically. The main novelty is that the proposed algorithm works without any parameters and thresholds by measuring the clustering quality. The very good performance of the proposal will be demonstrated by comparing the experimental results on a number of test sets with re-implementations of alternative high-quality approaches.

The proposed approach for gradual transition detection consists of three main components: a fade detector, an unsupervised gradual transition detector, and false alarm removal based on camera motion estimation. The general gradual transition detection process is preceded by fade detection [164] since fades can be detected more reliably than arbitrary gradual transitions. Afterwards, frame dissimilarities are measured at several frame distances. The main idea of the gradual transition detection approach is to view a gradual shot change as an abrupt shot change at a lower temporal resolution. Therefore, in contrast to other approaches [11, 69, 189, 195], subsampled time series of frame dissimilarities are considered for frame distances greater than 1. Although those previous approaches for gradual transition detection have considered frame dissimilarity measures at different frame distances, they retained the original temporal resolution for these measurements. Chua et al. [26] obtain a moderate subsampling of factor 4 using the wavelet transform which is applied to the original time series of dissimilarity values. It is typical for gradual transitions that they cause a plateau pattern in such measurements, which was first observed by Yeo and Liu [189]. However, such plateau patterns can be caused by motion as well and in practice they seldomly take the ideal shape which is assumed in theory. Thus, these plateau patterns are much harder to detect than isolated peaks. This is the main motivation for the subsampling applied in our approach.

Given a time series with a frame distance m which is subsampled accordingly by factor m, a gradual transition of length $n \leq m/2$ should be represented by an isolated peak in the time series - as it is the case for a cut at the highest temporal resolution (the original resolution of the video). The creation of feature vectors, the use of a sliding window and the clustering process take basically place in the same manner as in the cut detection approach. Finally, the frame positions of the beginning and the end of a transition are refined optionally. Since fades are recognizable well due the apprearance of monochrome frames, the proposed approach to gradual transition detection is supplemented by a fade detector according to the approach presented by Truong et al. [164].

To the best of our knowledge, sophisticated algorithms for detecting sequences of camera motion reliably have not been applied with the aim of removing the false alarms of a gradual transition detector. In this chapter, a novel false alarm removal scheme based on our high-quality camera motion estimation approach (which is described in detail in Chapter 5) is proposed.

The details of the cut detection and the gradual transition approach are presented in the next two sections. The work presented in this chapter has been partially published in [40, 42, 45, 46, 115].

4.5.2 Video Cut Detection without Thresholds

As discussed in the related work section, avoiding thresholds and other parameters in cut detection algorithms is a difficult problem. In this section, an algorithm is presented that does not incorporate thresholds and that is nearly parameter-free. In contrast to other cut detection approaches using clustering [56, 59], not only absolute frame dissimilarity measurements are considered but also a sliding window technique to enrich feature vector representation is adopted. No fuzzy variation for clustering is used, because this just shifts the problem of decision making into the defuzzifying stage in our opinion. First, the basic algorithmic steps are described before it is shown how the sliding window size can be estimated automatically, thus eliminating the only needed parameter. The algorithm works as follows:

1.) Measure Frame Dissimilarities. The frame dissimilarity is measured for subsequent frames with number i and i+1 are computed, resulting in a time series for the whole video. Motion compensated DC frames [69] are used, the motion estimation is conducted via a full search in a 4*4 search area. Let d be the sequence of frame differences where d_i is the frame difference between frame number i and i+1, then a sliding window t_i of size 2m+1 consists of the following dissimilarity values:

$$t_i = (d_i\text{-}m, d_i\text{-}m+1, ..., d_i\text{-}1, d_i, d_i+1, ...\ d_i+m) \qquad (53)$$

***2.) Find Cut Candidates*.** To reduce the number of cut candidates, a feature vector can only be a cut candidate between frame position *i* and *i*+1, if d_i is the maximum within the sliding window, called max from now on.

3.) Normalize Feature Vectors of Candidates. Several feature vector representations have been investigated and the *max* value and the second largest value within the sliding window, *sec*, are propsosed for feature vector representation. A feature vector is normalized adaptively depending on the given input video in the following way: The largest value d_{max} is estimated from the time series for the whole video. The normalization for the time series t_i representing a cut candidate between frame position *i* and *i*+1 is then performed as follows:

$$featureVec\ (t_i) = (max/d_{max}, 1\text{-}sec/max) = (max', sec'), \tag{54}$$

where *max*>0 and d_{max}>0. Theoretically, if *max*=0, then *sec'* is set to 0 as well. Thus, the range for both *max'* and *sec'* is [0, 1]. The Euclidean distance is used as a distance metric, and the set of candidates is then partitioned by the k-means algorithm in the subsequent steps.

***4.) Initialize the Cluster Classes "Cuts" and "Non-Cuts"*.** The cluster class "cuts" is initialized with the candidate that has the minimum distance to an ideal cut which is represented by the feature vector (1, 1), while the "non-cuts" class is initialized with the candidate with the minimum distance to an ideal non-cut which would be represented by the feature vector (0,0).

***5.) Assign Candidates to Classes*.** Now, the classical k-means algorithm is applied. First, the remaining cut candidates are assigned to the cluster class whose mean feature vector has the smaller Euclidean distance to the feature vector of the candidates.

6.) Optimize Class Memberships. Finally, as long as there are members in one class with a smaller distance to the mean feature vector of the other class, they are assigned to the other class. Then, the cluster with a mean feature vector nearer to the "ideal cut" feature vector is considered as the "cuts"-cluster.

However, experimental results presented in the next section show that the sliding window size affects detection performance significantly. Consequently, it is shown how the algorithm can be extended to estimate a suitable sliding window size. The silhouette coefficient (*SC*) for a clustering is usually computed in order to find a good estimate of *k*, if the number of clusters (*k*) is not known in advance (see also Chapter 2.4.1.4). The *SC is* utilized to evaluate the quality of the "cuts cluster" *C* which was obtained for a certain sliding window size. The *SC* for a feature vector *v* in *C* can be computed by formula (53) [35]:

$$SC(v) = \frac{b(v) - a(v)}{\max\{b(v), a(v)\}} \tag{55}$$

where *a*(*v*) is the average distance of *v* to the members in the same cluster *C*, whereas *b*(*v*) is the average distance of *v* to the members of nearest other cluster, in our case it is the "non-cuts" cluster. The *SC*(*C*) of the cuts cluster *C* is the average of the silhouette coefficients for all feature vectors in *C*. The value *SC*(*C*) is now exploited to measure the clustering quality of the cuts cluster for a reasonable range of sliding window sizes, and the proposed algorithm is modified as shown in Figure 27.

```
Input:  Minimum and maximum sliding window size minSize and maxSize;
Output: Estimated best sliding window size;

Algorithm

Find_Best_SlidingWindowSize()
  SCmax = 0; maxIndex = 1;
  for each window size m = minSize To maxSize
    Compute the cuts and the non-cuts cluster for sliding window size m;
    Compute the quality SC(C) of the cuts cluster C;
    if SC(C) > SCmax then
      SCmax = SC(C);
      maxIndex=m;

  return window size maxIndex;
```

Figure 27: Pseudo code for the algorithm to estimate the sliding window size.

Let us explain why it is justified to call this approach parameter-free. The sliding window size *m* is equivalent to the number of frames which can be between two cuts so that it is desirable to have the value *m* as low as possible. On the other hand, a very low *m* results in many candidates and many false alarms. Setting *maxSize* to an unreasonably high value, the range of investigated values for *m* includes at least the reasonable values and even some more inadequately high values (e.g., *m*>12 or *m*>15, depending on common frame rates of 25 or 30). For example, let be *m*=12, then only cuts with a distance of at least 0.52 seconds could be detected, assuming a frame rate of 25 frames per second.

4.5.3 Unsupervised Gradual Transition Detection

The proposed approach for gradual transition detection consists of three main components: a fade detector, an unsupervised gradual transition detector, and false alarm removal based on camera motion estimation. The general gradual transistion detection process is preceded by fade detection [164]. The main idea of the proposed approach for gradual transition detection is to view a gradual shot change as an abrupt shot change at a lower temporal resolution. For this purpose, subsampled frame dissimilarity time series are used, in contrast to related approaches [11, 69, 189, 195]. Given a time series with a frame distance *m* which is subsampled by factor *m*, a gradual transition of length $n \leq m/2$ should be represented by an isolated peak in the time series - as it is the case for a cut at the highest temporal resolution. Feature vector creation and the clustering process take place in a similar way as in the cut detection approach. Finally, false alarms are removed based on the results of a high-quality motion estimation algorithm. The main components are now described in more detail.

1.) ***Fade Detection.*** The fade detector is realized according to the proposal of Truong et al. [164]. First-order and second-order differences are computed for mean and variance of frame luminance. A smooting filter is applied to reduce the impact of noise. First, monochrome frames are detected by checking whether the variance of a frame is below a threshold. If so, then the subsequent (preceding) frames are analyzed in order to find a fade-in (fade-out). The first assumption is that the sign of the smoothed first-order luminance mean does not change during the fading process. As long as this is the case, the second-order differences of the variance curve are analyzed: It has been observed that there is a large negative spike near the start of a fade-out or the end of a fade-in in the second-order difference luminance variances. To detect such spikes, the subsequent (preceding) values of the smoothed second-order variance differences are checked for such spikes (indicating the start or end frame of the fade transition) until the sign of the smoothed first-order luminance mean changes.

2.) Gradual Transition Detection. After fade detection, the detection of gradual transition of any type takes place. First, frame dissimilarities are extracted for several frame distances. Feature vectors are created which describe the change in the frames via two features, as for cut detection. These feature vectors are then passed to the clustering process which produces two clusters: a "transition" cluster and a "non-transition" cluster.

2.1) Measure Frame Dissimilarities. First, frame dissimilarities are computed based on histograms of approximated DC-frames. Those dissimilarities are computed for certain temporal frame distances Δt. To detect gradual transitions, frames are compared at a higher temporal distance, for example up to 50 frames. Due to this, a histogram based metric is more suited to compute frame dissimilarities than a motion compensated pixel-based comparison, which is more sensitive to object and camera motion. A subsampled set of frame dissimilarity values is obtained for each of some temporal resolutions Δt:

$$\mathrm{D}(\Delta t\,, \mathit{offset})=\{d_{0,\,\Delta t},\, d_{1,\Delta t},\, \ldots,\, d_{i,\,\Delta t},\, \ldots,\, d_{n/\Delta t,\,\Delta t}\,\}, \tag{56}$$

where $\Delta t \in \mathrm{N}\backslash\{0\}$, and $\mathrm{d}_{\mathrm{i},\;\Delta \mathrm{t}}$ is the dissimilarity value for the frames $i*\Delta t+\mathit{offset}$ and $(i+1)*\Delta t+\mathit{offset}$, for all frames with $(i+1)*\Delta t \leq \mathit{maxFrameNumber}$. As mentioned above, the idea is to view a gradual shot change as a cut at a lower temporal resolution. Therefore, each time series of dissimilarity values with frame distance Δt is subsampled by the factor Δt. If a gradual transition of length k (this might be a cut of "length" 0 as well) starts at position n, this transition should be represented in all time series with $k<\Delta t/2$ by a peak in the dissimilarity measurements. The variable offset allows one to shift a subsampled time series. This can be useful to capture also transitions with a length of nearly Δt which start in the middle of two measurement timepoints: Such transitions will not yield an isolated peak in the time series but will produce two similar neighbored peaks of lower height.

2.2.) Create Feature Vectors. The feature vectors are now created similarly to the task of cut detection and consist of the same two components: *max'* and *sec'*. Different time series are obtained for the combinations of Δt and *offset*. The value *max'* is normalized for each time series using the corresponding maximum of the series. For each temporal resolution Δt, the basic sliding window size of $2*m+1$ is set separately based on the parameter x: $m=\max(x/\Delta t, c)$, where c is a constant and controls the minimum size for m (e.g. $c=2$). The parameter x represents the length of the sliding window at the finest temporal frame resolution, that is x represents the sliding window duration which is (nearly, except for rounding errors and the minimum window size c) equal for all temporal resolutions. By computing m separately for each Δt, fewer dissimilarity values are taken into account at lower temporal resolutions due to the preceding subsampling. The sliding window is not extended since at lower temporal resolutions the probability increases that neighbored cuts and transitions could be within the sliding window and affect the usefulness of the parameter *sec'*.

2.3.) Cluster Gradual Transition Candidates. The feature vectors are clustered using k-means (again with 2 clusters). At least three strategies are possible to cluster the time series. First, clustering can be conducted separately for each time series Δt including potentially time series for different offsets. Alternatively, clustering can be conducted for each time series separately for each combination of the parameters Δt and *offset*. A third possibility is to process the feature vectors of all time series in one clustering process. The latter requires a reasonable normalization across the several temporal resolutions. In our approach, the feature vectors are normalized according to the maximum value of the corresponding time series. In the first two cases, the clustering result must be merged afterwards.

2.4) Postprocessing of Gradual Transition Candidates. After k-means clustering, the members of the gradual transition cluster(s) (there are several clusters in case of clustering has been applied at least for each Δt separately) must be processed further. First, all transitions are removed whose frame interval includes a cut according to the cut detection results. Second, there might be feature vectors in one "transition" cluster or in the "transition" clusters of different clusterings which have a frame overlap. Two possibilities are considered to merge the frame interval of the transitions: union and intersection.

False alarms are removed if the start frame and the end frame are too similar, ehich is the case when dissimilarity value between start and end frame is below a threshold. Therefore, the average and standard deviation are computed for the cuts cluster with respect to histogram dissimilarity. Then, the threshold for false alarm removal is defined as:

$$thresh = dissim_{avg}(cuts) - dissim_{stddev}(cuts) \qquad (57)$$

A transition of length l is considered as a false alarm if the dissimilarity value at the start position in the time series $D(\Delta t, 0)$ *with* $\Delta t = l$ is below *thresh*.

Furthermore, a transition interval can be refined optionally. Therefore, the beginning of the transition is shifted forward frame by frame: If the dissimilarity of the new (start) frame and the end frame is equal or above to the dissimilarity value of the original start frame and end frame, the current frame is considered as the new transition start and the resizing process stops.

3.) False Alarm Removal via Camera Motion Analysis.

To enhance the gradual transition detection, camera motion estimation is applied to a video, too. For this purpose, the camera motion estimation approach is employed as proposed in this thesis and described in more detail in Chapter 5. The camera motion estimation algorithm returns a number of frame intervals where camera motion has been detected, separated for the following motion types: pan (horizontal camera movement), tilt (horizontal camera movement), and zoom in/out. Again, several strategies are possible to employ these results for false alarm removal. First, each combination of these motion types or a single motion type can be considered for false alarm removal. Furthermore, several cases are possible when a gradual transition has to be considered as a false alarm:

1. The frame interval of the gradual transition is completely covered by a motion interval.

2. The frame interval of the gradual transition and the motion interval intersect.

3. False alarm removal is applied on a frame basis: The frames that also exhibited motion are removed from the gradual transition.

Empirical results have shown that the first strategy works best in practice.

4.5.4 Experimental Results: Cut Detection

The proposed cut detection approach has been tested on four different test sets to obtain comparable results: the MPEG-7 video test (as suggested by Chua et al. [27]), and the TRECVID shot boundary test sets of the years 2005, 2006 and 2007. Our MPEG-7 test set consists of 32 MPEG-1 videos (about 12 hours, 5164 cuts), excluding the surveillance videos "speedwa*" and "etri* because they contain no perceptible shot changes. The MDC library [94] is used for MPEG decoding. Approximated DC-frames are created as described by Yeo and Liu [188]. The DC-frame dissimilarities are estimated using motion compensated pixel differences as suggested by Hanjalic [69]. A frame transition is accepted as a cut if its feature vector is in the cluster class "cuts" and the evaluation procedure is according to TRECVID evaluation.

The proposed approach is compared with two recently published state-of-the-art cut detection approaches (Hanjalic [69] and Bescos [11]) which reported very good detection results. Since these approaches require a training stage for parameter estimation and for classifier training, respectively, the TRECVID 2004 SBD test set has been used for training and parameter estimation. To obtain the best possible results, the implementation of Bescos' approach [11] (called Alg. A from now on) has been changed as follows: The simple parallelepipedic classifier was substituted by a support vector machine (SVM) based on a radial basis function kernel. SVMs have proven to be a valuable tool in pattern recognition and computer vision. Furthermore, the d3 metric, as described by Bescos [11], is normalized to have the same range of [0, 1] for all features which is reasonable for the SVM. Our implementation of Hanjalic's approach [69], called Alg. B, differs slightly from the original proposal, since a linear probability function has been used.

First, the impact of the sliding window size *m* is investigated for each algorithm using the TRECVID 2005 and 2006 SBD test sets. Therefore, the different approaches are tested with different sliding window sizes and the automatic estimation of parameter *m* has been removed from our approach in this setting. The results of the first experiment are displayed in Table 21 - Table 23. It is obvious that the cut detection results depend significantly on the sliding window size, for all approaches. For Hanjalic's approach [69], best results were achieved if the sliding window size of *m*=15 was used. The unsupervised approach is tested on two different test sets. Here, it becomes clear that best results are obtained for different sliding window sizes on the SBD test set 2005 and 2006. Hence, it is not optimal to use the same sliding window size for any video. In a second experiment, the automatic sliding window size estimation has been tested for the SBD test sets TRECVID 2005 and TRECVID 2006. The results are shown in Table 24 and Table 25. In one case when the range *m*=1..10 is investigated, precision decreases for the TRECVID 2005 test set. However, the automatic window size estimation works well for the other nine test cases. It is either superior (in five cases) to the best fixed sliding window size from the first experiment or it is only slightly worse (in four cases). This reinforces the statement that our approach is parameter-free since a very good estimator for *m* has been developed.

Sliding Window Size Parameter	F1	Recall	Prec.
M=1	89.9	90.5	89.4
M=10	90.4	87.4	93.7

Table 20: Cut detection results for the SVM extension of Bescos' approach [11] for different sliding window sizes on the TRECVID SBD test set 2005.

Sliding Window Size Parameter	F1	Recall	Prec.
M=5	80.6	88.9	73.8
M=10	82.2	87.1	77.8
M=15	88.1	85.1	91.3

Table 21: Cut detection results for Hanjalic's approach [69] for different sliding window sizes on the TRECVID SBD test set 2005.

Sliding Window Size Parameter	F1	Recall	Prec.
M=5	83.1	88.7	78.1
M=10	86.4	86.8	86.1
M=15	87.5	84.9	90.2

Table 22: Cut detection results for the proposed approach for different sliding window sizes on the TRECVID SBD test set 2005.

Sliding Window Size Parameter	F1	Recall	Prec.
M=5	82.9	81.7	84.1
M=10	83.1	78.5	88.2
M=15	81.9	75.8	89.1

Table 23: Cut detection results for the proposed approach for different sliding window sizes on the TRECVID SBD test set 2006.

Range (maxSize)	F1 [%]	Recall [%]	Precision [%]
m=1..10	80.6	87.7	74.6
m=1..15	88.0	86.6	89.5
m=5..15	87.2	86.7	87.7
m=5..20	87.3	84.5	90.2
m=1..20	87.1	84.3	90.0

Table 24: Results for different ranges for automatic sliding window size estimation on TRECVID 2005 test set.

Range (maxSize)	F1 [%]	Recall [%]	Precision [%]
m=1..10	83.5	80.0	87.3
m=1..15	83.5	79.3	88.1
m=5..15	83.4	79.1	88.3
m=5..20	82.6	77.1	88.9
m=1..20	83.3	78.8	88.3

Table 25: Results for different ranges for automatic sliding window size estimation on TRECVID 2006 test set.

In a further experiment, the unsupervised approach is compared with the approach of Bescos which we extended with a SVM classifier. The results for the different TRECVID test sets 2005, 2006 and 2007 and the MPEG-7 test set are displayed in Table 26 - Table 29. The results show that the proposed unsupervised approach outperforms the SVM approach on three of four test sets. Table 30 summarizes the results for all test sets. The overall performance on all test sets is slightly better, too. In addition, the performance of the unsupervised approach is more stable than the SVM approach, in particular with respect to precision but also in terms of recall and f1-measure. As a final surplus, the approach obtains satisfactory results even in the worst case, whereas the SVM approach fails completely for one video of the TRECVID 2007 test set.

TRECVID Test Set 2005 [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)	Min. F1
Prop. Unsupervised Approach	87.4	84.4	90.5	88.4 (±6.7)	86.0 (±9.4)	91.4 (±4.8)	73.1
Best disparities (SVM) [11]	90.4	87.4	93.7	90.8 (±4.3)	88.0 (±6.0)	94.0 (±3.0)	82.6

Table 26: Cut detection results on TRECVID data 2005. Comparison of the proposed unsupervised approach with the approach of [11] extended with SVM.

TRECVID Test Set 2006 [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)	Min. F1
Prop. Unsupervised Approach	82.6	77.1	88.9	81.6 (±06.2)	77.4 (±13.7)	88.8 (±6.1)	58.5
Best disparities (SVM) [11]	80.0	71.9	90.1	77.5 (±11.9)	70.0 (±16.1)	89.1 (±4.8)	50.3

Table 27: Cut detection results on TRECVID data 2006. Comparison of the proposed unsupervised approach with the approach of [11] extended with SVM.

TRECVID Test Set 2007 [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)	Min. F1
Prop. Unsupervised Approach	94.5	94.7	94.2	94.2 (±05.8)	94.3 (±06.5)	94.2 (±06.4)	79.8
Best disparities (SVM) [11]	92.9	89.9	96.7	89.1 (±23.2)	88.3 (±23.4)	90.1 (±23.2)	-

Table 28: Cut detection results on TRECVID data 2007. Comparison of the proposed unsupervised approach with the approach of [11] extended with SVM.

MPEG-7 Test Set [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)	Min. F1
Prop. Unsupervised Approach	92.0	88.4	96.0	91.3 (±08.8)	90.0 (±13.5)	94.3 (±5.7)	60.8
Best disparities (SVM) [11]	90.1	86.4	94.1	91.2 (±10.4)	87.9 (±14.2)	94.8 (±7.2)	58.6

Table 29: Cut detection results on MPEG-7 video data set. Comparison of the proposed unsupervised approach with the approach of [11] extended with SVM.

ALL Test Sets [%]	F1	Recall	Prec.	Mean F1 (±Std. Dev.)	Mean Recall (±Std. Dev.)	Mean Prec. (±Std. Dev.)	Min. F1
Prop. Unsupervised Approach	89.4	86.2	92.9	89.8 (±08.7)	88.1 (±12.8)	92.8 (±06.1)	58.5
Best disparities (SVM) [11]	88.7	84.3	93.6	87.9 (±14.6)	84.9 (±17.4)	92.6 (±12.5)	-

Table 30: Cut detection results with respect to all data sets. Comparison of the proposed unsupervised approach with the approach of [11] extended with SVM.

4.5.5 Experimental Results: Gradual Transition Detection

Finally, results for the unsupervised gradual transition detection approach are reported on the test sets of TRECVID 2005 and 2007 and compare it to the best approach of our comparison study [189]. Several experimemts are conducted on TRECVID 2005 SBD test set to investigate the impact of different parameters: frame distances, feature vector representation, sliding window size, clustering type, and false alarm removal. Table 31 - Table 37 show results for different frame distances. In each table, results are presented for several sliding window sizes using the feature vector representation (*max'*, *sec'*). In addition, for the frame distances 6, 25, and 50, results are listed in the corresponding tables using only *max'* as a feature for two sliding window sizes. Several observations can be made. First, when using only a single frame distance, the frame distances between 10 and 50 are able to achieve an f1-value of more than 0.6, best results are obtained using the frame distance of 25 and 30. Increasing the sliding window increases precision and decreases recall, as expected. Sliding window sizes greater than 2 reduce f1 measure in most cases noticeably. Using only the *max* feature (and not also the feature *sec)* yields a higher recall at the cost of precision (please see Table 31, Table 34, and Table 37). Hence, the sliding window size and the feature *sec* are useful parameters to increase precision.

Frame distance 6: window size parameter M	F1	Recall	Precision
M=6	54.5	43.2	73.9
M=3	55.2	46.2	68.7
M=2 (only *max* feature)	55.3	60.8	50.7
M=2	55.5	47.6	66.6
M=1 (only *max* feature)	47.9	70.7	36.2
M=1	53.6	47.5	61.5

Table 31: Results for a frame distance of 6 for different sliding window sizes.

Frame distance 10: window size parameter M	F1	Recall	Precision
M=6	58.2	47.0	76.5
M=3	62.6	55.0	72.7
M=2	62.2	56.0	70.0
M=1	60.9	58.0	64.2

Table 32: Results for a frame distance of 10 for different sliding window sizes.

Frame distance 20: window size parameter M	F1	Recall	Precision
M=6	49.8	36.4	78.7
M=3	61.7	51.5	76.8
M=2	64.7	56.9	75.0
M=1	66.1	61.6	71.3

Table 33: Results for a frame distance of 20 for different sliding window sizes.

First, different sets of frame distances are tested. Second, it is investigated which clustering strategy works best: clustering the feature vectors of all frame distances together or separately. Third, different sliding windows sizes are subject to analysis. Finally, the impact of false alarm removal based on camera motion estimation is anaylzed. The results in Table 46 show that the proposed approach outperforms the method of Yeo and Liu clearly.

Frame distance 25: window size parameter M	F1	Recall	Precision
M=6	44.0	30.3	80.6
M=3	59.0	46.8	79.8
M=2	64.1	54.0	78.7
M=2, only *max* feature	66.1	61.0	72.2
M=1	67.3	61.0	75.1
M=1, only *max* feature	64.8	69.1	61.0

Table 34: Results for a frame distance of 25 for different sliding window sizes.

Frame distance 30: window size parameter M	F1	Recall	Precision
M=6	39.1	25.9	79.5
M=3	55.9	43.3	79.0
M=2	61.6	50.6	78.7
M=1	65.7	58.1	75.6

Table 35: Results for a frame distance of 30 for different sliding window sizes.

Frame distance 40: window size parameter M	F1	Recall	Precision
M=6	29.7	18.2	80.0
M=3	49.0	35.2	80.8
M=2	57.4	44.8	79.8
M=1	64.0	55.0	76.6

Table 36: Results for a frame distance of 40 for different sliding window sizes.

Frame distance 50, window size parameter M	F1	Recall	Precision
M=6	26.5	15.9	79.0
M=3	42.0	28.4	80.6
M=2, *max* only	54.2	42.3	75.3
M=2	51.1	37.8	78.9
M=1, *max* only	62.1	55.6	70.2
M=1	61.3	50.6	77.6

Table 37: Results for a frame distance of 50 for different sliding window sizes.

In Table 38, results are presented for some combinations of different frame distances. Two clustering types are distinguished: using the feature vectors for different frame distances in one clustering process (“all”), or in separate clustering processes (“separate”). The feature vector consists of the features *max* and *sec* and the sliding window size parameter was set to 1. In all cases, the results are better than using only one frame distance. The clustering strategy does not seem to have much impact on the final result.

Frame distances (clustering type)	F1	Recall	Precision
6, 25 (all)	66.3	66.8	65.8
6, 25 (separate)	66.5	66.3	66.7
20, 40 (all)	67.7	64.7	71.1
20, 40 (separate)	68.0	65.1	71.1
25, 30 (all)	69.6	66.0	73.7
25, 30 (separate)	69.5	65.8	73.6
6, 50 (all)	63.8	61.5	66.3
6, 50 (separate)	63.8	61.4	66.5
25, 50 (all)	67.2	62.0	73.7
25, 50 (separate)	67.8	62.5	74.1
6, 25, 50 (all)	66.6	66.1	67.1
6, 25, 50 (separate)	65.0	64.7	65.4

Table 38: Results for thr combinations of different frame distances. Two clustering types are distinguished: using the feature vectors for different frame distances in one clustering process (“all), or in separate clustering processes (“separate”).

In Table 39, results are presented for different strategies to remove false alarms. The first strategy relies on the histogram-based comparison of the first and the last transition frame. The second strategy employs camera motion estimation. The best result in terms of f1-measure (69.8) is achieved when only pan is considered for false alarm removal, whereas using all camera motion types is less effective and yields the worst result.

Frame distances (strategy of false alarm removal)	F1	Recall	Precision
25, 30 (no false alarm removal)	67.6	77.8	59.7
25, 30 (histogram-based false alarm removal: avg-1.0*stdev)	69.6	66.0	73.7
25, 30 (histogram-based false alarm removal: avg-1.5*stdev)	69.5	68.0	71.0
25, 30 (histogram-based false alarm removal: avg-2.0*stdev)	67.6	68.7	66.5
25, 30 (false alarm removal based on all camera motion types)	59.6	58.7	60.5
25, 30 (false alarm removal based on pan)	69.8	76.7	64.1

Table 39: Results for different strategies of false alarm removal.

Table 40 and Table 41 show results for gradual transition detection in case when either the feature *max* or the the feature *sec* is used. Interestingly, the results are rather good for the feature *max*, in best case an f1-measure of 69.4 is obtained in conjunction with a false alarm removal strategy. As in the preceding experiment, the incorporation of a second frame distance yields only a slight improvement in terms of f1-measure.

Frame distances (clustering type, features, false alarm removal)	F1	Recall	Precision
25 (*max,* no false alarm removal)	61.1	78.7	49.9
25 (*max*, histogram-based false alarm removal: avg-1*stdev)	69.4	66.7	72.4
25 (*max*, histogram-based false alarm removal: avg-1.5*stdev)	68.6	68.5	68.7
25 (*max*, histogram-based false alarm removal: avg-2*stdev)	65.0	69.6	61.0
25 (*sec*, no false alarm removal)	50.0	54.1	46.5
25 (*sec,* histogram-based false alarm removal: avg-1*stdev)	57.0	45.5	76.2
25 (*sec,* histogram-based false alarm removal: avg-1.5*stdev)	56.3	47.1	70.1

Table 40: Performance for gradual transition detection when only one feature is used. Tested in conjunction with different false alarm removal strategies.

Frame distances (features, false alarm removal strategy)	F1	Recall	Precision
25, 30 (*max*, no false alarm removal)	59.9	83.4	46.7
25, 30 (*max* & *sec*, no false alarm removal)	67.5	77.8	59.6
25, 30 (*max*, false alarm removal: pan)	62.6	82.2	50.5
25, 30 (*max*, histogram-based false alarm removal: avg-1*stdev)	70.0	69.0	71.0
25, 30 (*max*, histogram-based false alarm removal: avg-1.5*stdev)	69.0	71.6	66.5
25, 30 (*max*, histogram-based false alarm removal: avg-2*stdev)	64.1	72.8	57.2
25, 30 (*max*, histogram-based false alarm removal: avg-2*stdev, pan)	66.5	71.9	61.9

Table 41: Performance for gradual transition detection when only one feature is used but two frame distances. Tested in conjunction with different false alarm removal strategies.

Frame distances (clustering type, false alarm removal)	F1	Recall	Precision
20, 25, 30 (all, no false alarm removal)	66.3	82.9	55.2
20, 25, 30 (all, histogram-based false alarm removal)	69.6	68.8	70.4
20, 25, 30 (all, histogram-based false alarm removal, pan)	72.5	67.9	77.8
20, 25, 30 (all, histogram-based false alarm removal, avg-2*stdev)	66.4	72.5	61.2
20, 25, 30 (all, false alarm removal:pan)	69.7	81.6	60.8
20, 25, 30 (all, histogram-based false alarm removal: avg-2*stdev, pan)	69.3	71.5	67.3
20, 25, 30 (all, histogram-based false alarm removal: avg-2.5*stdev, pan)	69.2	71.8	66.8
20, 25, 30 (all, *max*, no false alarm removal)	54.8	87.0	40.0
20, 25, 30 (all, *max*, histogram-based false alarm removal)	69.2	70.4	68.1
20, 25, 30 (all, *max*, histogram-based false alarm removal, pan)	72.3	69.6	75.3

Table 42: Performance for gradual transition detection when thrre frame distances are used. Tested in conjunction with different false alarm removal strategies and features.

Table 42 shows results in case three frame distances are used. Those frame distances are selected which achieved the best results when used as the only frame distance. The best result is achieved in case when both false alarm removal strategies are applied.

Frame distances, (clustering type, false alarm removal)	F1	Recall	Precision
6, 10, 20, 30, 40, 50 (all, no false alarm removal)	26.1	85.0	15.4
6, 10, 20, 30, 40, 50 (all, histogram-based false alarm removal, avg-0.5*stdev)	64.3	59.6	69.8
6, 10, 20, 30, 40, 50 (all, histogram-based false alarm removal)	66.2	68.0	64.4
6, 10, 20, 30, 40, 50 (all, histogram-based false alarm removal, avg-2*stdev)	54.4	72.1	43.7
6, 10, 20, 30, 40, 50 (all, histogram-based false alarm removal, pan, avg-0.5*stdev)	67.1	59.0	77.7
6, 10, 20, 30, 40, 50 (all, histogram-based false alarm removal, pan)	70.3	67.4	73.5
6, 10, 20, 30, 40, 50 (all, histogram-based false alarm removal, avg-2*stdev)	58.7	71.5	49.8

Table 43: Results for six or seven different frame distances, combined with different parameter settings.

Frame distances, (clustering type, false alarm removal)	F1	Recall	Precision
6, 10, 20, 30, 40, 50 (all, *max*, no false alarm removal)	12.5	89.2	6.7
6, 10, 20, 30, 40, 50 (all, *max*, histogram-based false alarm removal)	63.8	67.8	60.2
6, 10, 20, 30, 40, 50 (all, *max*, histogram-based false alarm removal, pan)	68.0	67.1	69.0
6, 10, 20, 30, 40, 50 (all, *max*, pan)	13.3	88.4	7.2

Table 44: Results for six or seven different frame distances used in conjunction with the feature *max* only, combined with different false alarm removal strategies.

Frame distances, (clustering type, false alarm removal)	F1	Recall	Precision
6, 10, 20, 30, 40, 50 (all, win=24, no false alarm removal)	66.8	81.2	56.8
6, 10, 20, 30, 40, 50 (all, win=24, histogram-based false alarm removal)	68.2	67.7	68.8
6, 10, 20, 30, 40, 50 (all, win=24, false alarm removal: pan)	70.6	80.2	63.1
6, 10, 20, 30, 40, 50 (all, win=24, histogram-based false alarm removal, pan)	71.0	66.8	75.8

Table 45: Results for six or seven different frame distances used in conjunction with sliding window size parameter 24, combined with different false alarm removal strategies

Finally, the combination of more than three frame distances has been tested, using six or seven different frame distances: 6, 10, 20, 30, 40, and 50. The results are displayed in Table 43, Table 44, and Table 45. It is obvious that the use of more than one frame distance does not improve detection quality, the combination of three frame distances yields best results.

Results for gradual transition detection on TRECVID 2005 [%]	F1	Recall	Prec.
Unsupervised Approach	70.3	67.4	73.5
Local Thresholding (Yeo and Liu [189])	40.4	34.9	47.9

Table 46: Results for gradual transition detection on the TRECVID 2005 test set for the proposed approach and the local thresholding approach of Yeo and Liu [189].

Overall, some conclusions can be drawn from these experiments. The use of three frame distances seems to be sufficient for gradual transition detection. The clustering strategy does not have much impact on the final result. The tradeoff between recall and precision can be controlled by the sliding window size applied in the feature selection process, the use of the *sec* feature acts in a similar way. Careful use of the false alarm removal strategy allows one to increase precision while recall remains quite stable. In particular, considering pan seems to be most reliable for false alarm removal based on camera motion. Regarding the false alarm removal based on histogram comparison, the threshold (*t=avg-c*stddev*) achieves best results for *c*=1 or *c*=1.5.

4.5.6 SUMMARY

A novel unsupervised approach approach for shot boundary detection in videos has been presented. The basic idea of the approach is to classify time series of frame dissimilarity measurements into cuts and non-cuts by using the well known k-means clustering algorithm. For cut detection, the impact of parameter and threshold settings is completely removed. The critical estimation of the sliding window size is achieved by measuring the clustering quality using the silhouette coefficient. Our proposal for cut detection was evaluated using several video test sets and outperformed our re-implementation of two other high-quality approaches. The unsupervised gradual transition approach works similar to the cut detection approach but is supplemented with fade detector and a false alarm removal process. The latter is based on high-quality camera motion estimation approach. The unsupervised shot boundary detection approach was evaluated at TRECVID 2005 and achieved very good results. For cut detection, an f1-measure 90.0 was achieved (recall: 93.6%, precision: 86.4%), for gradual transition detection an f1-measure of 70.0 was obtained (recall: 71.5%, precision: 68.4%) which is competitive to the best approaches. In contrast to the best systems at TRECVID, our approach requires neither labeled training data [197] nor a particular parameter tuning [5].

4.6 VIDEO CUT DETECTION USING AN ENSEMBLE OF CLASSIFIERS

4.6.1 INTRODUCTION

The proposed unsupervised approach to shot boundary detection has the advantage that labeled training data are not required. Also, the unsupervised approach works for arbitrary video data, independent of recording circumstances, compression artefacts and genre. However, in cases when some of those properties of the video data are known in advance and labeled data are available, supervised machine learning approaches are possibly a good choice to improve detection performance. In particular, the use of an ensemble of classifiers promises a benefit since it has been shown that an ensemble of classifiers can improve accuracy in recognition tasks [89]. In this section, the question is investigated whether the cut detection performance can be improved by combining multiple "experts". Since most transitions in a video are abrupt (without any transitional frames between the different shots), the ensemble supplements the unsupervised approach for cut detection. Experimental results demonstrate that an ensemble consisting of the unsupervised approach, an Adaboost classifier and a SVM can outperform a single classifier for the task of cut detection. The work presented in this chapter has been partially published in [42].

4.6.2 PROPOSED ENSEMBLE OF CLASSIFIERS FOR VIDEO CUT DETECTION

Two factors determine the accuracy of an ensemble: the accuracies of the single classifiers and the diversity of their outputs [20]. To achieve diversity, several strategies have been discussed in Chapter 2.4.5. In this chapter, the combination of different classifier types is tested. Many supervised machine learning techniques are applicable for the given task, for example naive Bayes classifier, k-nearest-neighbor, neural networks, Hidden Markov Models, Support Vector Machine. On the other hand, there are meta-learning methods like bagging, boosting and stacking. Among the boosting methods, Adaboost is a very popular method. Since SVM has proven to yield very good results in many pattern recognition tasks, SVM is chosen as the first classifier for the ensemble. As a second classifier Adaboost based on weak classifiers has been chosen since a feature selection process can be incorporated (in the case when each weak classifier is based on one feature). A feature selection process is useful in order to reduce the SVM training effort.

The ensemble is built in the following way. First, a set of features is defined which represents the dissimilarity of two frames with respect to brightness, color, edges and motion, and with respect to different frame distances. Then, an Adaboost classifier is trained on a training set. Thereby, some features are selected in the Adaboost training process. Only these features are used to train a SVM on the training set. Finally, the original unsupervised approach and the two additional classifiers form an ensemble. A frame position is considered as a cut if at least two of the three classifiers vote for "cut". The features are described in more detail in the next subsection.

4.6.2.1 FEATURE EXTRACTION

For the task of cut detection, 42 features for a certain frame distance were defined which describe the dissimilarity of DC-frames with respect to global histograms, pixel blocks, edges, and local histograms. Haar-Wavelet like features have been used to capture frame changes over time, as they are used in the face detection approach of Viola and Jones [167] to describe spatial changes in image regions. Let *A* and *B* be intervals, each of the size of a pre-defined frame distance *t*>0, *d* a time series of values (e.g., histogram-based frame dissimilarity values or luminance mean for each frame), then the feature types $f_{AB}(d, i)$ and $f_{A1A}(d, i)$ for frame position *i* are defined as:

$$f_{AB}(d,i) = abs\left(\sum_{k=i}^{i+t-1} d(k) - \sum_{k=i-t}^{i-1} d(k) \right) \tag{58}$$

$$f_{A1A}(d,i) = abs\left(d(i) - \sum_{k=i+1}^{i+t} d(k) - \sum_{k=i-t}^{i-1} d(k) \right) \tag{59}$$

Overall, the following set of features is computed for the ensemble cut detection approach.

- Motion compensated pixel-based dissimilarities (3 features): Motion compensation is conducted by estimating the best block matches for 4*4 sized pixel blocks in the DC frame. Pixel differences are then measured based on the best block matches. Also, the feature representations f_{AB} and f_{A1A} are calculated for these frame dissimilarity measures.

- Sliding window size features (15 features): ratio of the second largest dissimilarity value divided by the local maximum for several sliding window sizes, for the motion compensated pixel-based dissimilarities.

- Histogram-based dissimilarities (3 feature): histograms are used for the YUV color space, the most significant 3 bits of each color channel are combined, yielding a histogram size of 512 bins. Also, the feature representations f_{AB} and f_{A1A} are calculated for these frame dissimilarity measures.

- Frame differences of mean and variance of Y-luminance channel (6 features): Both measures are computed for each frame, the difference of two consecutive frames is considered as the dissimilarity value. Also, the feature representations f_{AB} and f_{A1A} are calculated for the mean and variance measures.

- Edge histograms of Sobel-filtered (vertically and horizontally) DC-frames (6 features): A histogram is computed for each filtered image. Dissimilarity values are then computed in the same way as for the previously described features (2 features). The feature representations f_{AB} and f_{A1A} are calculated for these frame dissimilarity measures (4 features). The following Sobel masks are used to filter a DC-frame vertically or horizontally:

$$sobelMask_{vertical} = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \qquad sobelMask_{horizontal} = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \qquad (60, 61)$$

- Local histogram dissimilarity features (9 features): Each DC-frame is divided into nine non-overlapping regions of equal size, histograms are computed locally for each region and dissimilarity of two frames is calculated for each region separately.

Adaboost is utilized for feature selection where the best $m \leq n$ features are used to train a SVM on the same test set of 2004. Two frame distances (1 and 2) were investigated resulting in a total feature number of 84. Thus, finally there are three classifiers evaluating each frame (considering the unsupervised approach as a kind of classifier as well). A majority vote is implemented in our approach: a cut is detected if at least two "experts" vote that a frame belongs to a new shot.

4.6.3 Experimental Results

The ensemble approach has been tested in conjunction with our TRECVID 2005 experiments. The frame distance for the second cut detection metric was set to 2. The frame distances for the gradual transition detection were set to: 6, 10, 20, 30, 40, 50. The MDC decoder has been used for MPEG decoding [94]. Feature selection using Adaboost was performed on the TRECVID [161] 2004 shot boundary test set. Eleven features were selected from the whole feature set to build an Adaboost classifier. The best 7 features were used to train a SVM (using the library "LibSVM" [23]) on eight of the twelve videos from the last year's test set. Only a subset of features and videos was chosen, since SVM training is a very time-consuming task.

The experimental settings and the results for the different runs are shown in Table 47. The parameters for the sliding window sizes had very little impact on cut detection, as discussed in the previous sections. Increasing the maximum sliding window size led to slightly better precision values whereas recall decreased very slightly. Using the classifier ensemble improved cut detection performance: Comparing the f1-measure for cut detection (see f1-measures for all submitted runs and all shot boundary tasks in Table 3), the f1-measures for the ensemble runs are between 90.3 and 91.2 while the results of the unsupervised baseline approach are between 89.3 and 89.9. In all cases, the ensemble approach led to better results in terms of both recall and precision. The contribution of the classifiers to the ensemble's performance was analyzed as well. Table 48 shows the cut detection results (defining the maximum a cut "length" as equal 0 or smaller/equal than 5) for only the Adaboost classifier and the SVM classifier, respectively, on the TRECVID 2005 test set. The SVM achieves a very high precision for the cut "transition" with a length≤5, whereas the Adaboost classifier is superior in recall. The low precision of 57.7% for the Adaboost classifier is mainly caused by one video ("NASA-Connect-AO.mpg"), whereas the precision for the other videos is about 83%.

Run	Recall	Prec.	F1	Cuts: Max WinSize1	Cuts: Max WinSize2	Ensemble	False Alarm Removal
marburg0	0.928	0.880	0.903	15	5	Yes	No
marburg1	0.932	0.888	0.909	15	5	Yes	Yes
marburg2	0.936	0.864	0.899	15	5	No	No
marburg3	0.922	0.891	0.906	15	10	Yes	Yes
marburg4	0.920	0.867	0.893	15	10	No	No
marburg5	0.925	0.895	0.910	15	15	Yes	No
marburg8	0.924	0.900	0.909	15	15	Yes	Yes
marburg6	0.926	0.892	0.912	18	9	Yes	Yes
marburg7	0.924	0.868	0.895	18	9	No	No

Table 47: Recall and precision for different maximum sliding window sizes, using optionally an ensemble of classifiers or false alarm removal.

Cut Detection Performance	Recall	Prec.	F1
SVM (cut transitions with length ≤5)	0.838	0.947	0.889
SVM (cut transitions with length ≤0)	0.871	0.939	0.904
Adaboost (cut transitions with length ≤5)	0.957	0.577	0.720
Adaboost (cut transitions with length ≤0)	0.978	0.484	0.648

Table 48: Experimental results for the classifiers used in the ensemble approach.

4.6.4 SUMMARY

In this section, it has been investigated whether an ensemble of classifiers can improve cut detection. Therefore, the unsupervised baseline approach was extended with two additional classifiers, a SVM classifier and an Adaboost classifier. Adaboost was further employed for feature selection. The classifiers were trained on the TRECVID 2004 shot boundary test set and the ensemble was evaluated at TRECVID 2005. Indeed, the ensemble of classifiers improved cut detection performance. For several parameter settings of the unsupervised baseline approach, the detection performance was above the detection performance of the single classifiers in either case, in terms of f1-measure. At the shot boundary detection evaluation at TRECVID 2005, this ensemble approach was among the small number of top approaches which achieved an f1-measure of above 90.0 (91.2) for video cut detection.

4.7 Self-Supervised Learning for Robust Shot Boundary Detection

4.7.1 Introduction

The performance of video analysis and indexing algorithms strongly depends on the type, content and recording characteristics of the analyzed video. Current video indexing approaches often make use of thresholding techniques or supervised learning which requires labeling of possibly large training sets. Furthermore, the application of the same training model or parameters might lead to a sub-optimal indexing accuracy for a given video *v*. In this chapter, the task of shot boundary detection is considered as a transductive learning setting in order to overcome these drawbacks. For this purpose, the transductive learning ensemble framework as proposed in Chapter 3 is applied the task of cut detection. Experimental results on the TRECVID 2005 test set show that the proposed approach improves the results of a high quality state-of-the-art video cut detection approach. The work presented in this chapter has been partially published in [43].

4.7.2 Related Work

First, several state-of-the-art video indexing approaches (shot boundary detection, camera motion estimation and face detection) are reviewed with respect to their robustness. Then, some approaches using transductive learning, self-supervised learning or co-training are discussed.

Smeaton and Over [140] show that shot detection results which were evaluated at TRECVID 2005 vary in terms of recall and precision depending on the different video sources ("NASA" and "News", see Figure 28).

A closer look at successful video indexing approaches reveals that even top approaches are not designed to adapt to a particular video source. Typically, pre-defined thresholds or parameters are used, as exemplified by two of the best performing shot boundary detection approaches at TRECVID 2005 which have been discussed in chapter 4.2.2 (Yuan et al. [192], Tahaghoghi et al. [153]).

Considering the best TRECVID results 2005 for the task of camera motion estimation yields a similar picture. Yuan et al. [192] estimate the motion parameters of a two-dimensional affine model and finally apply thresholding rules to decide about the presence of motion. They achieved the best results for pan and tilt detection. Ewerth et al. [39, 42] obtained best results for zoom detection by computing the parameters of a 3D-camera model and applying thresholds for decision making.

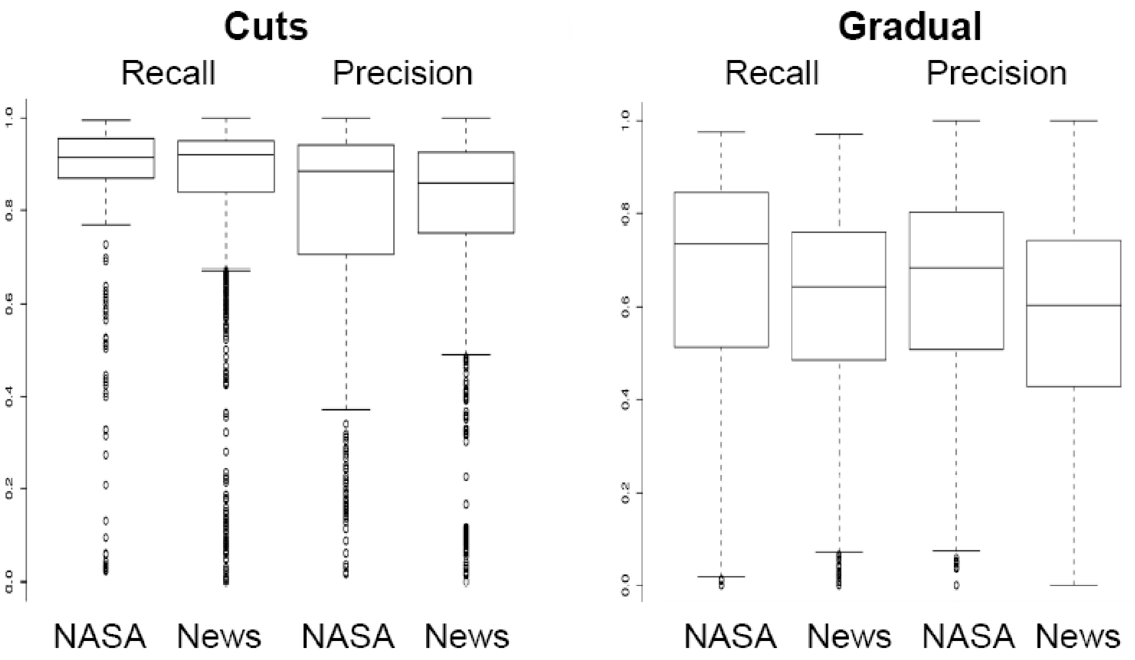


Figure 28: Distribution of recall/precision for shot detection at TRECVID 2005 depending of the video sources [140].

Two of the most recent and successful face detection approaches employ machine learning and need a large training set. Schneiderman and Kanade [135] apply the wavelet transform and use wavelet features from various frequencies of different spatial resolutions to train a Naive Bayes classifier. Viola and Jones [167] train a cascade of Adaboost classifiers and mainly focus on real-time processing of video frames.

Up till now, there are only few applications of transductive learning, self-supervised learning or co-training in the field of video content analysis. For example, Lieb et al. [95] propose a self-supervised approach for adaptive road following for driving vehicles to reduce the need that a road must be represented by unique identifying features. Wu and Huang [179] suggest self-supervised learning using labeled and unlabeled training data for object recognition in order to overcome the tedious and expensive task of labeling large training data sets. They extend a linear Discriminant-EM with a non-linear kernel. The experimental results show that their novel learning technique is competitive to SVMs and outperforms various approaches for hand-gesture recognition and fingertip tracking tasks. Oudot et al. [123] present a self-supervised method for writer adaptation in an online-text recognition system. In the self-supervised method, lexical results are compared with the classification hypothesis to find errors which are then used to re-estimate classifier parameters. Co-training (e.g., [178]) is a semi-supervised and multi-view learning approach which can be used if no sufficient amount of training data is available. The idea is to incorporate unlabeled data into the training and to make use of different feature sets (views) to train two classifiers. Wu et al. [178] suggest co-training for text detection in images. They train two SVMs on color and edge features and incorporate optical character recognition (OCR) into the training scheme.

4.7.3 Self-Supervised Learning for Robust Video Cut Detection

For the task of video cut detection, the transductive learning ensemble is realized in a self-supervised manner since the used baseline system relies on unsupervised learning. The key idea of the proposed approach is to use an initial result for a given video V using an initial set A of features. In this way, training data is generated automatically from the video itself. The unsupervised clustering approach (according to Chapter 4.5.2) is used as the baseline system to initially classify frames as cuts or non-cuts. Only two features are used (motion compensated pixel differences, and the ratio of the second largest dissimilarity value divided by the local maximum within a sliding window of size $2m+1$) in this approach in which an appropriate sliding window size is estimated automatically. Then, this initial result, including the classification errors, is used to select the best features for this video from a possibly large set B of features (where $A \subseteq B$, $|A| \leq |B|$). 42 features, as described in detail in section 4.6.2.1, have been defined for the feature set B for a certain frame distance describing frame dissimilarity with respect to:

- motion compensated pixel differences,
- histogram differences,
- luminance mean and variance,
- edge histograms of Sobel-filtered (vertically and horizontally) DC-frames,
- local histogram differences, and
- ratio of the second largest dissimilarity value divided by the local maximum for several sliding window sizes.

Two frame distances (1 and 2) are investigated, resulting in a total number of 84 features. An Adaboost approach [167] is applied to obtain a ranking of the best features for the particular test video V. This set of best features for the video V is split afterwards according to odd and even ranks. The feature split is conducted to subsequently train different classifiers with a reasonable degree of independence on the video V using only the training data generated from the video itself. Kuncheva et al. [89] show that the independence of classifiers is advantageous to increase accuracy of an ensemble of classifiers. According to the ranking order, thes selected features are split and passed to the self-supervised learning stage to train two different SVMs directly on the video *V* using the different feature sets. Together with the unsupervised system, they form an ensemble of three classifiers: a cut is detected if at least two of them vote that a frame is a cut.

4.7.4 Experimental Results

The proposed self-supervised framework has been tested on the TRECVID [161] 2005 shot boundary test set. The "MDC" library [94] was used for MPEG decoding and the "libSVM" library [23] for SVM implementation.

Several strategies of the proposed system have been implemented and compared with our unsupervised baseline approach on the TRECVID 2005 shot boundary test data. There are 3372 abrupt transitions in the video test set. Systematic experiments have been conducted in order to find the best way to exploit the automatically labeled data and to improve an original detection result. First, the unsupervised baseline system was used to automatically label data for a given video and all these data are then used to train a classifier. All positive For this purpose, SVM (45 features selected via Adaboost) and Adaboost (11, 45 and 180 training rounds) have been chosen as classifiers. In the first experiment, all *p* positive labeled training samples are used, the factor of negative training samples was set to 9**p*. The experimental results of this first experiment on the TRECVID 2005 video test set are presented in Table 50. The results of the classifiers which have been trained with data automatically labeled by the unsupervised baseline system achieve worse results in terms of F1-measure. Recall is increased for the classifiers, but precision is significantly decreased, too. Overall, simply using the first detection result to train a new classifier is *not* a possibility to improve detection performance for a given video. In a second experiment, only the best 90% of the positive labeled training data are used to train an additional classifier. A sample is considered as a "good" sample in case it is as near as possible to the feature vector of an "ideal" cut which would have the feature vector (1, 1) in the proposed unsupervised approach. The experimental results for this experiment are presented in Table 50. The results are clearly improved compared with the first experimental setup. It turns out that the selection of good training samples from the automatically labeled training data is an important step.

	K-means baseline system	Adaboost (11)	Adaboost (45)	SVM (45)	Adaboost (180)
Recall [%]	86.6	93.2	93.6	92.5	93.8
Precision [%]	89.0	77.6	78.5	87.8	77.7
F1	87.8	84.7	85.4	90.1	85.0

Table 49: Experimental results on the TRECVID 2005 test set for the unsupervised baseline system, for a SVM classifier and for some Adaboost classifiers using a different number of features. Each of them was trained separately on each video based all automatically positive labeled training samples. The number of negative training samples depends on the number of positive samples.

	K-means baseline system	Adaboost (11)	Adaboost (45)	SVM (45)	Adaboost (180)
Recall [%]	86.6	91.7	92.9	91.7	92.7
Precision [%]	89.0	88.1	88.3	89.1	88.7
F1	87.8	89.9	90.5	90.4	90.7

Table 50: Experimental results on the TRECVID 2005 test set for the unsupervised baseline system, for a SVM classifier and for some Adaboost classifiers using a different number of features. Each of them was trained separately on each video based only on the "best" 90% positive training samples. The factor of negative training samples remained unchanged.

The next experimental setup investigates whether it is useful to form an ensemble of classifiers that is built adaptively for a given video. Several strategies are possible. First, the unsupervised baseline system can be extended to an ensemble with the two classifiers (Adaboost and SVM) which were trained on the video. Furthermore, it is investigated whether it is beneficial to divide the selected feature into two disjoint feature sets where each of them is used to train a classifier. Here, two different strategies are considered. First, the feature set is split alternating depending on each feature's rank during the Adaboost selection process. Second, the feature set is partitioned into two subsets using k-means clustering. The experimental results for these methods are presented in Table 51 and Table 52. In Table 52, the proposed approach is also compared with the approach of Bescos [11] which we have extended with a SVM classifier. The results demonstrate that all ensembles improve the detection performance of the original approach. The ensembles consisting of the baseline system and two SVMs with different feature sets achieved slightly better results than the ensemble with different classifiers which were trained on the same features. In practice, the simpler alternating split of the features should be preferred since it avoids the clustering process of the features.

	K-means	Ensemble with K-means, and 2 SVMs, clustered feature sets, 11features (improvement)	Ensemble with K-means, and 2 SVMs, alternated features, 11 features (improvement)
Recall [%]	86.6	86.8 (+0.2)	87.2 (+0.6)
Precision [%]	89.0	90.5 (+1.5)	90.4 (+1.4)
F1	87.8	88.6 (+0.8)	88.8 (+1.0)

Table 51: Experimental results on the TRECVID 2005 test set for several classifier ensemble schemes. Supervised experts in these ensembles are trained only on the automatically labeled data for the given video.

	K-means	Bescos' approach [11] extended with SVM	Ensemble with K-means, Adaboost and SVMs. 45 features (improvement)	Ensemble with K-means, and 2 SVMs, clustered feature sets, 45 features (improvement)	Ensemble with K-means, and 2 SVMs, alternated features, 45 features (improvement)
Recall [%]	86.6	87.4	91.9 (+5.3)	92.2 (+5.6)	92.6 (+6.0)
Precision [%]	89.0	93.7	91.0 (+2.0)	90.9 (+1.9)	91.5 (+2.5)
F1	87.8	90.4	91.4 (+3.6)	91.5 (+3.7)	92.0 (+4.2)

Table 52: Experimental results on the TRECVID 2005 test set for three different self-supervised variations.

Furthermore, the self-supervised ensemble approach has been tested on the TRECVID test sets of 2006 and 2007. The results are shown in Table 53 and Table 54 and demonstrate that the self-supervised approach improves the baseline system on each test set. In particular, the results for the most difficult test set of 2006 are noticeably improved, recall is increased by 7.4% while precision remains at 88.2%. If the number of features is sufficiently large, the proposed approach outperforms Bescos' approach [11], too.

	K-means	Bescos' approach [11] extended with SVM	Ensemble with K-means, and 2 SVMs (alternated 45 features)
Recall [%]	79.1	71.9	86.5 (+7.4)
Precision [%]	88.3	90.1	88.2 (-0.1)
F1	83.4	80.0	87.3 (+3.9)

Table 53: Experimental results on the TRECVID 2006 test set for the ensemble scheme with two SVMs using alternating features. Supervised experts in these ensembles are trained only on the automatically labeled data for the given video.

	K-means	Bescos' approach [11] extended with SVM	Ensemble with K-means, and 2 SVMs (alternated features)
Recall [%]	95.3	89.9	95.3 (+0.0)
Precision [%]	93.5	96.7	95.6 (+2.1)
F1	94.4	92.9	95.4 (+1.0)

Table 54: Experimental results on the TRECVID 2007 test set for the ensemble scheme with two SVMs using alternating features. Supervised experts in these ensembles are trained only on the automatically labeled data for the given video.

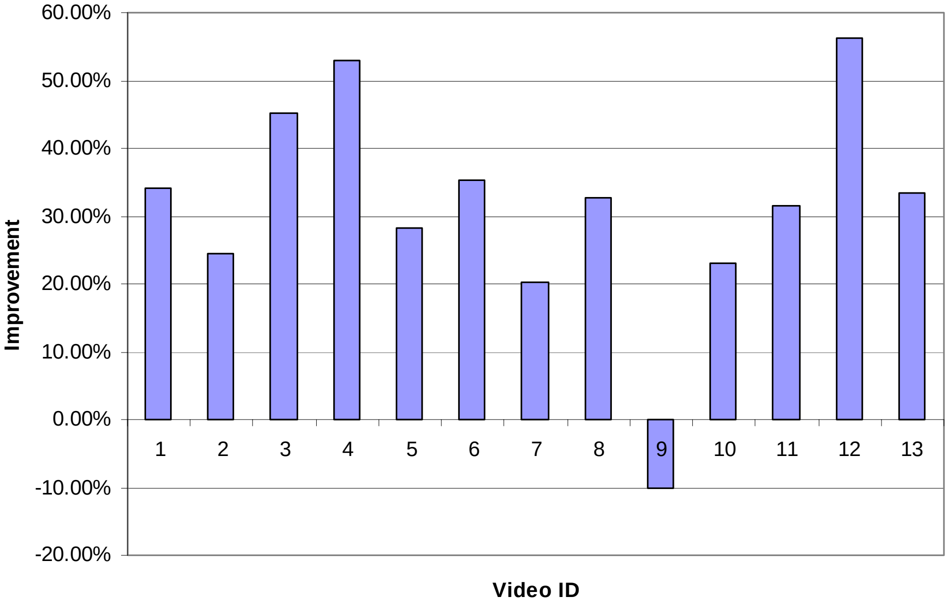


Figure 29: Reduction of error rate for each video of the TRECVID 2005 shot boundary test set. Error rate reduction approach when using the self-supervised approach is given in relation to the original unsupervised approach. Error rate is given separately for each video from the TRECVID 2005 shot boundary test set.

The results are presented in more detail in Figure 29. The self-supervised approach reduces the number of errors by more than 20% (and up to 56%) for eleven videos out of 12. The number of errors increases slightly for only one video. It is concluded that the self-supervised approach is able to learn and automatically improve a model for a given video by itself without any pre-labeled training data.

4.7.5 SUMMARY

In this chapter, the novel transductive learning ensemble has been applied to the first task of video indexing: cut detection. The ensemble is realized in a self-supervised manner. The approach is motivated by the analysis of video indexing approaches and issues related to the specifics and uniqueness of video source and content. Based on an initial classification result for a given video, the suggested approach utilizes Adaboost to select an optimal subset of features for this video. Then, this feature set is split into two complementary feature sets in order to train two SVMs on the given video. Two strategies to split a feature set are presented where both achieved similar results. A prototype of the learning framework applied to video cut detection has been implemented and tested on the test sets of TRECVID 2005 and 2007. Experimental results indicate that the self-supervised system is indeed able to learn automatically by itself without any pre-labeled training data: The f1-measure is significantly higher than that of the baseline system and achieves similar detection results as an ensemble using supervised classifiers. Dividing the feature set and assigning the features to two separate classifiers yielded slightly better results than using two different classifiers (Adaboost and SVM) which were trained on the same feature set. The splitting of the feature set allowed us to increase the independence of the classifiers, which is exploited in the subsequent execution of an ensemble of classifiers using majority voting.

4.8 AUTOMATIC PERFORMANCE PREDICTION FOR VIDEO CUT DETECTION

4.8.1 INTRODUCTION

Performance prediction is a powerful method in order to achieve an optimal result for content analysis and retrieval tasks. Recently, some research efforts evolved in the field of information retrieval addressing the issue of performance prediction [29, 30, 72]. In this field, performance prediction is utilized to identify difficult queries which often lead to bad retrieval results. He and Ounis [72] state that reliable prediction of query performance is a way to determine the best retrieval strategy for a given query. In this chapter, it is shown how performance prediction can be utilized in the context of video analysis. Here, performance prediction helps to identify the number of errors and can thus aid post-processing. An approach to automatically predict the precision and recall of a video cut detection result is proposed. The prerequisite is the application of unsupervised clustering for video cut detection which enables evaluation of clustering quality using the silhouette coefficient. It is shown that this clustering validity measure is highly correlated with the precision of a video cut detection result. A formula to estimate the precision of a detection result is derived and validated experimentally.

4.8.2 RELATED WORK

Recently, some research efforts have been made in the field of information retrieval concerning performance prediction. Cronen-Townsend et al. [30] argue that the coherence of a returned ranked list is positively related to the number of relevant documents in the list. They introduce a statistical measure for list coherence: clarity score. They estimate the correlation between the precision and the average precision for a variety of TREC collections, where correlation is between 0.49 and 0.62. Among others, it is suggested to use the clarity score to decide whether the user should be encouraged to reformulate his/her query. In contrast to the clarity score, which is based on a retrieval process and result, He and Ounis [72] study predictors that can be computed before a retrieval process takes place. For several predictors, statistically significant correlations with average precision are obtained, in the range of 0.21 up to 0.45.

To the best of our knowledge, there are only very few approaches considering the issue of automatic performance characterization in the field of content-based image and video retrieval. Vogel and Schiele [168, 169] present a probabilistic framework for content-based image retrieval that enables automatic performance characterization and optimization. Images are divided into a number of patches, and the concept detector is assumed to decide independently on each patch. Furthermore, it is assumed that the detector's probability for a correct decision and the concept distribution are known in advance. Based on these assumptions, closed-form expressions are derived for the probability of recall and precision of a retrieval result. These formulas are then further used to adjust an internal parameter to optimize retrieval performance. The framework is extended [169] by Vogel and Schiele with an approach to improve the accuracy of the detectors as well.

4.8.3 Performance Prediction for Unsupervised Video Cut Detection

In section 4.5, an unsupervised approach for video cut detection has been presented. Feature vectors describe a frame position with respect to the possibility that a cut occurred at this position. These feature vectors are clustered using the k-means algorithm to separate cut from non-cuts. Since k-means is utilized to solve a classification problem, the number of clusters is known in advance. Normally, the silhouette coefficient *SC* is used measure the clustering validity for different k to find the best suited *k* in case it is not known in advance. In our approach, *SC* is exploited to estimate another parameter automatically, the sliding window size *m*, by comparing the quality of the "cuts"-cluster ($SC(C_{cuts})$) for several clustering results for different m. It can be observed that $SC(C_{cuts})$ is highly correlated with the precision, and little with recall, of a clustering result. Thus, it is proposed to estimate the interrelation between $SC(C_{cuts})$) and precision and recall, respectively, via linear regression on a training set of videos. Therefore, the following linear approximations are used:

$$precision = p \cdot SC(C_{cuts}) + q \tag{62}$$

$$recall = r \cdot SC(C_{cuts}) + s \tag{63}$$

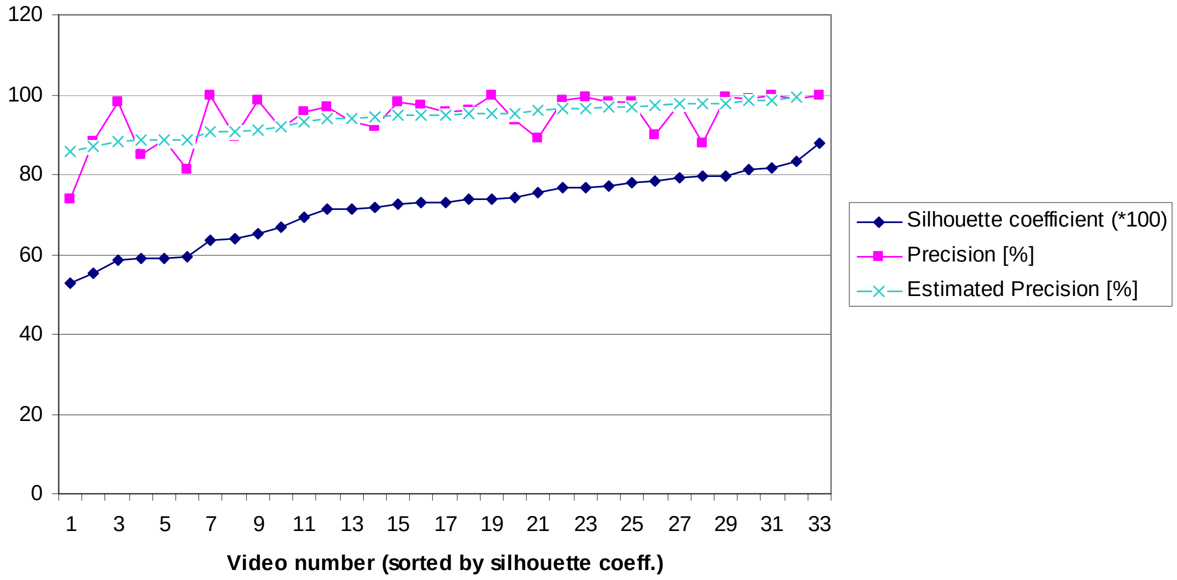
The parameters *p* and *q*, and *r* and *s*, respectively, are estimated by linear regression on a training set for which ground truth data are available and thus the precision outcome is known.

4.8.4 Experimental Results

The proposed estimation has been tested using two different test sets: the MPEG-7 video test set and the shot boundary test set of TRECVID 2005. The unsupervised video cut detection algorithm was run on the MPEG-7 test set and the precision for each video and the best Silhouette coefficient (for a number of sliding window sizes) were used to estimate the parameters *p*, *q*, *r* and *s* of Formulas (62) and (63) using linear regression. For the MPEG-7 video test set, the correlation between silhouette coefficient and precision is 0.63, the correlation between the silhouette coefficient and recall is 0.26. Using this test set, the parameters *p*, *q*, *r* and *s* are estimated as follows: p=0.620 and p=0.449; r = 0.186 and s=0.811. The video cut detection algorithm was then run on the TRECVID shot boundary test videos and precision and recall were computed for each video using Formula (62) and (63) with the previously estimated parameters. The experimental results are presented in Table 55 where the estimated precision, real precision, estimated recall, real recall, and the corresponding differences are presented. Assuming, that the precision outcome for each video had been estimated with the average precision obtained for the MPEG-7 test set (95.3%), the average estimation error would have been about 6.4% (standard deviation 4.0). Using the proposed precision estimation yields an average (absolute) estimation error of only 3.2% (standard deviation of 2.91). This estimate is twice as precise as if the precision achieved for the MPEG-7 test set (95.3%) would be used to estimate precision. The recall estimate is not as precise as for precision, as expected due to the lower degree of correlation. It is slightly better (0.9% more precise) than an estimate which is based on the recall for the MPEG-7 test set (recall on MPEG-7 test set: 94.5%).

Video ID	Silhouette Coefficient	Estimated Precision [%]	Precision[%]	Error Precision Estimation	Estimated Recall[%]	Recall [%]	Error Recall Estimation
1	0.564	87.38	80.37	7.01	0.916	0.719	19.7
2	0.677	92.46	90.94	1.52	0.937	0.868	6.9
3	0.635	90.57	87.23	3.34	0.929	0.865	6.4
4	0.723	94.52	88.49	6.03	0.945	0.961	-1.6
5	0.649	91.20	91.96	-0.76	0.931	0.888	4.3
6	0.634	90.52	88.55	1.97	0.929	0.939	-1.0
7	0.615	89.67	89.04	0.63	0.925	0.875	5.0
8	0.600	89.00	89.44	-0.44	0.922	0.794	12.8
9	0.673	92.28	83.11	9.17	0.936	0.914	2.2
10	0.774	96.81	96.60	0.21	0.955	0.893	6.2
11	0.717	94.25	89.84	4.41	0.944	0.833	11.1
12	0.834	99.51	96.72	2.79	0.966	0.980	-1.4
Average (of absolute values)		92.35	89.36	3.19 (stdev.: 2.91)	0.936	0.877	6.53 (stdev.: 5.56)

Table 55: Results for the TRECVID 2005 shot boundary test set: The silhouette coefficient for each cuts cluster which was obtained after k-means clustering, the estimated precision and the real precision, and the estimation error



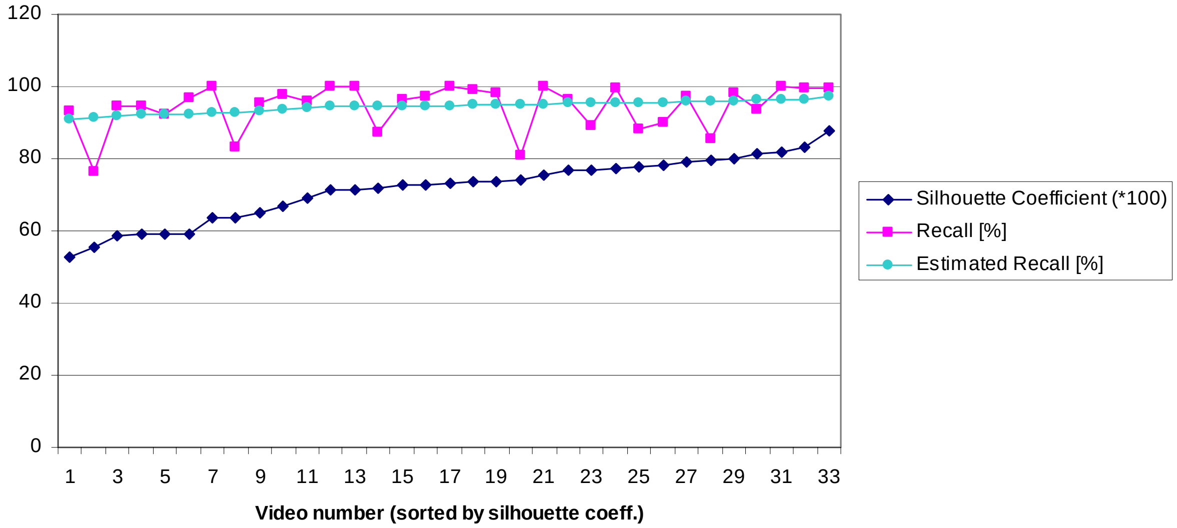


Figure 30, top: The silhouette coefficient (SC, scaled by 100), precision and estimated precision based on SC (both in %) for 33 videos from the MPEG-7 test set. Bottom: The data for recall.

4.8.5 Summary

In this section, an approach to automatically predict the performance of an unsupervised video cut detection task in terms of recall of precision has been presented. It is based on the observation that cluster validity, in our case measured with the silhouette coefficient, is strongly correlated with precision and with recall, though to a lower degree. Exploiting this fact, it is suggested to learn a linear interrelationship between the silhouette coefficient and precision and recall, respectively, hich is easily achieved by linear regression. In the experiments, the comprehensive MPEG-7 video test set was used to learn the interrelationship and the TRECVID 2005 shot boundary detection test set was used to test the prediction performance. Experimental results demonstrated that the prediction yields an average error of only 3.2% for precision and 6.5% for recall. In both cases, the estimates are more precise than a prediction that would be only based on the overall performance on the training set. In case of precision, the prediction is even twice as precise than standard approximation.

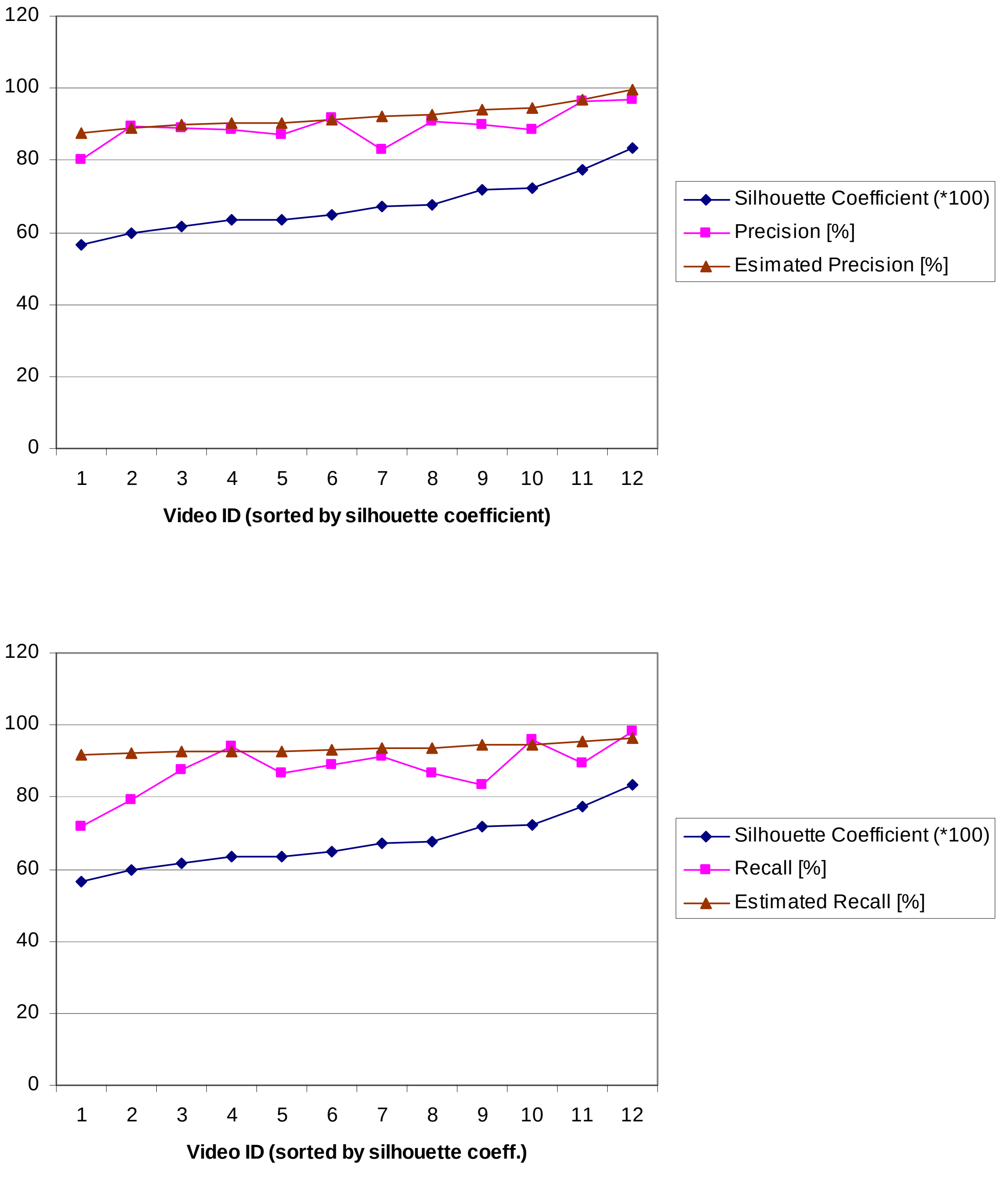


Figure 31, top: The silhouette coefficient (SC, scaled by 100), precision and estimated precision based on SC (both in %) for 12 videos of the TRECVID 2005 test set. Bottom: The data for recall.

4.9 Summary of Chapter 4

In this chapter, the problem of shot boundary detection in videos was studied extensively in order to introduce the principles suggested for robust and adaptive video content analysis. Several contributions have been presented in this context. First, recent research in this field was comprehensively surveyed and compared, including the related TRECVID evaluation as well as other approaches. Second, a systematic bias in frame dissimilarity measurements caused by MPEG frame types that may hinder cut detection performance in the compressed MPEG domain was identified for the first time. Two solutions to remove these artifacts were presented. Third, an unsupervised shot boundary detection approach was developed aimed at the reduction of user defined parameter settings. A measure for cluster validity, the silhouette coefficient, was employed to estimate an important parameter for cut detection, the size of the sliding window which is moved over the temporal frame dissimilarity measures. Furthermore, a self-supervised approach was suggested that exploits the transductive setting of the given task. An initial clustering result for a given video is used to estimate the best features for this video. These features are then divided into two disjoint sets which are used to train two classifiers, again only on the video under consideration. The unsupervised detector and the adaptively learned classifiers form an ensemble that analyzes the video and produces the final result using majority voting. Finally, it was observed that the silhouette coefficient is strongly correlated with precision and weakly correlated with recall. Formulas for precision and recall depending on a clustering were derived. Experimental results demonstrated that performance prediction can be improved this way, in particular for precision. All proposed approaches have demonstrated their effectiveness in experiments using the comprehensive MPEG-7 video test set and the TRECVID shot boundary test sets of 2005 and 2007, all publicly available.

Chapter Five

5 Robust Camera Motion Estimation in MPEG Videos

5.1 Introduction

In movie production, the use of camera distance and camera motion are quite important to express a certain atmosphere or a meaning through the way a scene is captured and presented to the viewer. Arijon [6] considers the moving camera as a powerful stylistic device and, accordingly, there is strong interest of media researchers to investigate in which way the camera device was used by the producer.

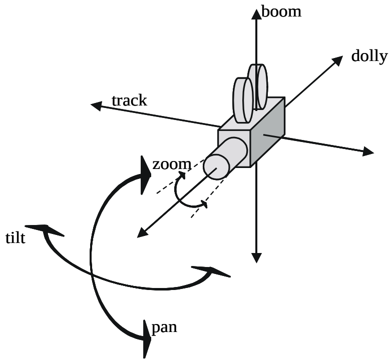


Figure 32: Various types of camera motion.

The estimation of camera motion is important for several video for scientific film analysis and for indexing and retrieval purposes and. From an aesthetical point of view, camera motion is often used as an expressive element in film production. Recently, some researchers propagated the term "computational media aesthetics" [2], which aims at bridging the semantic gap between the low-level feature extraction algorithms and the high-level queries raised by users.

There are different types of camera motion: rotation around one of the three axes and translation along the x- and y-axis. Furthermore, zoom in and out can be considered as equivalent to translation along the z-axis. The various types of camera motion are presented in Figure 32. Many algorithms have been proposed in the past to solve the problem of camera motion estimation, but in the overwhelming majority of approaches translation along the x-axis (y-axis) and rotation around the y-axis (x-axis) is considered as one type of motion.

In this chapter, a solution is presented for the problem of camera motion estimation in MPEG videos. The proposed approach is able to distinguish translation and rotation and works directly on motion data available from the compressed video stream. To the best of our knowledge, this is the first approach with these properties. Although the use of MPEG motion vectors improves runtime performance, they often do not model real motion adequately. Such "unreliable" motion vectors are called outliers. To obtain a motion vector field that represents the real optical flow in the best way, an effective algorithm for outlier removal is incorporated into our approach. Furthermore, the minimum number of motion vectors that is sufficient to obtain reliable estimation results is investigated. Experimental results demonstrate the very good performance of the proposed approach. The work presented in this chapter has been partially published in [39]

5.2 RELATED WORK

The related work can be distinguished according to the camera model used, which in turn determines the motion types that are possibly detected. Furthermore, there are some approaches to camera motion estimation [86, 124, 154] for the MPEG domain of compressed videos. Most approaches [86, 124, 151, 154] estimate the model parameters that yield the best fit to the optical flow measured, and thus, a typical minimization problem must be solved.

For example, Kim et al. [86] suggest to process motion vectors of MPEG videos. They assume a two-dimensional affine camera model and state to detect six types of motion: zoom, rotation, pan, tilt, object motion and stationary. Thus, results for translation along the x-axis (y-axis) and rotation around the y-axis (x-axis) are merged. Beforehand, motion vector outliers are filtered out by a simple smoothing filter. The parameter vector for the model is estimated by using the method of least squares.

Tan et al. [154] assume that there is only rotation and zoom and use a corresponding 3D camera model. Appropriate motion parameters are found by solving the minimization problem for the given 3D model. Their approach is aimed at detecting camera motion and related events like close-ups in MPEG sports videos. They use several heuristics to remove unreliable vectors from the motion vector field, for example if the shot is not stationary, then the zero vectors are removed. For certain thresholds, they achieve very good classification results for basketball videos.

Park et al. [125] compare two least square methods applied to MPEG videos: the Iterated Extended Kalman Filter and the Levenberg-Marquardt method. They assume a camera system with rotation and zoom but without translation. Those motion vectors are removed as outliers from the motion vector field which show the worst fit to the estimated model parameters. They have checked empirically the effectiveness of the approaches by investigating a generated mosaic image.

Srinivasan et al. [151] present an approach which can distinguish between camera rotation and translation. Their approach is based on an appropriate 3D camera model that includes rotation, translation (except along the z-axis) and zoom in and out. They compute an optical flow field for two consecutive uncompressed frames and solve a minimization problem by using the Nelder-Meade algorithm. To detect translation, the optical flow resulting from the estimated rotational camera parameters is set in relation to the original optical flow field. If the residual vectors are noticeably larger than zero and parallel, it is concluded that there is also translational motion.

Another approach is followed by Ngo et al. [122]. They compute spatiotemporal slices considering (x,t)-space and (y,t)-space from a video sequence of 2D-frames with spatial dimension (x, y) at time points t. Tensor histograms are utilized to analyze the visual slice patterns which occur according to camera and object motion. Their approach can distinguish stationary, pan, tilt and zoom.

Joly and Kim [83] choose a similar approach but apply the Hough transform to spatiotemporal images in order to estimate camera motion. Their approach is able to distinguish the camera actions stationary, pan, tilt and zoom.

5.3 Estimating Camera Motion in MPEG Videos

The approaches for the compressed domain presented in section 5.2 do not make the distinction between rotation and translation. An algorithm to solve this problem is proposed in this section. The algorithm consists of three steps.

1.) Extraction of motion vectors;

2.) Computation of a reliable motion vector field;

3.) Estimation of camera motion parameters.

The algorithmic steps are described in more detail below.

1.) Extraction of motion vectors.

The motion vectors are extracted directly from the compressed MPEG stream. In MPEG, the encoding of a P-frame is based on a previous reference frame, while the encoding of a B-frame can be based on two reference frames, a previous as well as a subsequent reference frame. Only the motion vectors from P-frames are processed in our approach, for two reasons. First, usually each third until fifth frame in a MPEG video is a P-frame, and thus, the temporal resolution is sufficient for most applications. Second, both the prediction direction and the temporal distance of motion vectors are not unique in B-frames, resulting in additional computational complexity. For each macroblock, a motion vector is estimated which points to a similar block in a reference frame. Motion estimation algorithms try to find the best block match in terms of compression efficiency.

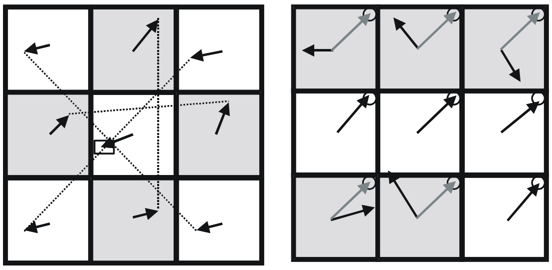


Figure 33: Two examples for outlier criteria showing a motion vector and its neighbors. Left: "smoothness"; Right: "neighborhood".

This can lead to motion vectors that do not represent the camera motion or object motion at all, which for example is possibly the case for homogenous areas in images due to noise and low image quality.

2.) Computation of a reliable motion vector field.

To deal with these noisy motion vectors, an outlier removal algorithm is applied. In the proposed approach, no heuristics to remove noise in a motion vector field are used (e.g., applied by Tan et al. [154]) since such heuristics typically handle special cases of failures and not the general case. In other approaches, a smoothing filter is applied to the motion vector field (e.g., suggested by Kim et al. [86]) but the drawback is that the erroneous outliers remain in the field and affect their possibly correct neighbors. Consequently, the outlier removal algorithm which was suggested by Dante and Brooks [31] is employed in the proposed approach. There are two main steps in the algorithm. A motion vector MV is declared as an outlier if both of the following criteria are not met (see the examples shown in Figure 33):

2.1) Smooth Change. MV is compared to each average of four pairs of opposite neighbors – if the number of averages that are close to the central MV is below a threshold, then the criteria of smoothness is not met.

2.2) Neighborhood. A neighborhood motion vector supports the central MV if it lies within a tolerance circle. If the number of supporting vectors is below a threshold, then the criteria of neighborhood is not met.

Additionally, the number of motion vectors can be reduced for example by considering only each second motion vector in both horizontal and vertical direction.

3.) Estimation of camera motion parameters.

To potentially distinguish translational camera movements and corresponding rotation, an appropriate 3D camera model (see Figure 34) has been chosen as described by Srinivasan et al.

[151]. Formulas (64) and (65) describe the translational components u_x and u_y in the image plane depending on the focal length *f*, the translational movement t_x and ty along the x-axis and y-axis, the rotational components r_x, r_y and r_z around all three axes, and the zoom factor r_{zoom}. Consider an external point at (*X*, *Y*, *Z*) which is projected onto the image plane at point (*x*,*y*), where *x*=*f**(*X*/*Z*) and *y*=*f**(*Y*/*Z*). Motion estimation is done as follows.

$$u_x(x,y) = -\frac{f}{z} * t_x - \frac{x*y}{f} * r_x + f\left(1+\frac{x^2}{f^2}\right) * r_y - y*r_z + \\ f\left[\tan^{-1}\left(\frac{x}{f}\right)\right]\left(1+\frac{x^2}{f^2}\right) * r_{zoom} \tag{64}$$

$$u_y(x,y) = -\frac{f}{z} * t_y - \frac{x*y}{f} * r_y + f\left(1+\frac{y^2}{f^2}\right) * r_x - x*r_z + \\ f\left[\tan^{-1}\left(\frac{y}{f}\right)\right]\left(1+\frac{y^2}{f^2}\right) * r_{zoom} \tag{65}$$

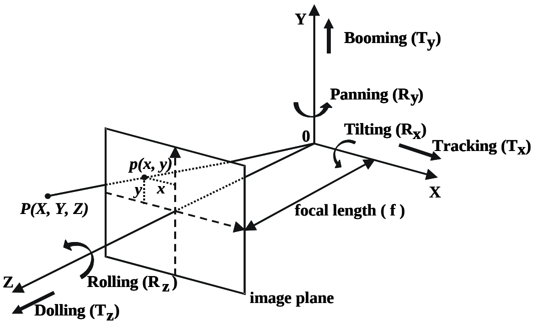


Figure 34: The 3D camera model that is used in the proposed approach.

Let v_i be the difference vector between a motion vector at macroblock position (*x*,*y*) in the original and the estimated motion vector field, let *V* be the sum of all v_i, and let θ_i be the absolute angle between a vector v_i and *V*. As suggested by Srinivasan et al. [151], the parameter values r_x, r_y, r_z and r_{zoom} are estimated by minimizing the term $P=\Sigma v_i^2 * \theta_i$ using the Nelder-Meade algorithm. This results in an estimated motion vector field *VF* where the difference vectors v_i are mostly parallel. In contrast to the proposal of Srinivasan et al. [151], the mean vector of *VF* is directly considered as the translational camera motion parameters t_x and t_y in our approach, since it improved the experimental results described below.

5.4 Experimental Results

Our video test set consists of 32 MPEG-1 video sequences (352*288 pixels resolution) based on well textured Pov-Ray scenes including all kinds of camera motion and many combinations of motion types. The advantage is that the camera motion parameters can be entirely controlled which allows us to verify the estimation quality in a reliable way.

The Nelder-Meade algorithm has been implemented to estimate the parameters of the camera model as described in the previous section. The MPEG decoding is realized using the MDC library [94]. Several combinations have been investigated in our experimental setting: 1) Baseline implementation; 2) Extension with outlier removal; 3) Reduction of motion vector amount; 4) Assuming an incorrect focal length.

A camera motion is detected if the corresponding parameter is above a certain threshold. The experimental results can be summarized as follows: The extension of the baseline implementation with outlier removal led to a noticeable increase in recall and precision as shown in Table 56, where recall is the number of correctly detected motions divided by the number of motions present in the videos, and precision is the number of correct detections divided by the total number of detections. Furthermore, the required minimal number of motion vectors has been investigated in order to check the possibility to speed up algorithm runtime. The detection results remained quite stable, even if only 1/16 of the motion vectors (after outlier removal) was used (Table 57). Very good detection results were achieved if translation along the x-axis (y-axis) and rotation around the y-axis (x-axis) were considered as a single motion type (Table 58). Furthermore, the robustness of the approach was tested for different assumptions of the focal length f. In the Pov-Ray scenes, f was set to 2. Setting the focal length to an incorrect value of 1.5 or 2.5 did not significantly change the estimation performance.

Recall/Precision [%]	Baseline implementation	Outlier Removal (Change to basis implementation)
Rotation(x)	71 / 71	78 / 78 (+7/ +7)
Rotation(y)	77 / 43	75 / 41 (-2/ -2)
Rotation(z)	86 / 75	95 / 89 (+9/+14)
Translation(x)	68 / 48	74 /56 (+6/ +8)
Transaltion(y)	55 / 35	75 / 58 (+20/+23)
Zoom	98 / 94	99 / 99 (+1/ +5)

Table 56: The detection results listed separately for each motion type for the baseline implementation and its extension with outlier removal.

Recall /Precis. [%]	1/1 # MV	1/4 # MV	1/8 # MV	1/16 # MV
Rotat. (x)	78 / 78	70 / 74	77 / 77	70 / 74
Rotat. (y)	75 / 41	73 / 41	75 / 40	71 / 40
Rotat. (z)	95 /89	90 / 86	85 / 84	88 / 85
Transl. (x)	74 / 56	75 / 52	76 / 55	74 / 58
Transl. (y)	75 / 58	68 / 56	70 / 53	58 / 51
Zoom I/O	99 / 99	96 / 97	98 / 99	96 / 97

Table 57: The results for each motion type if the number of motion vectors used in the optimization algorithm is reduced after outlier removal.

Recall / Precis. [%]	*Baseline Implem.*	*Outlier Removal*	*1/8 # MV*
R(x) and T(y)	89 / 88	96 / 93	96 / 93
R(y) and T(x)	92 / 76	100 / 79	99 / 79
R(z)	86 / 75	95 / 89	85 / 84
Zoom I/O	98 / 94	99 / 99	98 / 99

Table 58: The results if translation and the similar rotation type are considered as a single motion type.

Furthermore, we have participated in the low-level feature task at TRECVID 2005. The task was to identify these three types of camera motion, pan, tilt, and zoom, in a very large video collection of 140 videos. Hence, the distinction of camera translation and rotation was not necessary in this task, The system parameters were estimated using some videos from the TRECVID training set for which partially ground truth data exists. These ground truth data were created jointly by us, KDDI Labs and the Joanneum Research University of Graz [8]. The following parameters were estimated for our system using the available training data: *rot_x* =0.001, *rot_y*=0.0015, *rot_zoom*:=0.00075.

Submission	Pan		Tilt		Zoom		Total Average	
	Recall	Prec.	Recall	Prec.	Recall	Prec.	Recall	Prec.
UMarburg	76.1%	92.4%	72.4%	96.2%	89.4%	93.1%	79.3%	93.9%

Table 59: Experimental results in terms of recall and precision for our submission for the TRECVID 2005 low-level feature task.

Submission	Pan (F1)	Tilt (F1)	Zoom (F1)	Overall (F1)
UMarburg	83.5	82.6	91.2	86.0

Table 60: Experimental results in terms of F1- measure for our submission for the TRECVID 2005 low-level feature task.

These threshold parameters were used for our system. In addition to our above reported experiments, the motion vectors of frame border macroblocks were not considered to compute a motion vector field. The TRECVID organizers selected a number of shots from the very large video test set. Shots were considered for evaluation when they either showed clearly one of the motion types of interest or none of them. This procedure yielded an evaluation set of shots that consisted of 587 shots with pan, 210 shots with tilt, and 511 shots with zoom, 1159 shot with none of these motion types. Finally, 2226 shots were considered for evaluation in total. Each institute was allowed to submit up to seven experimental runs. The experimental results for the University of Marburg are presented in Table 59 and Table 60. As displayed there, precision is above 90% for all tasks, whereas recall is between 72% and 76% for pan and tilt detection. For tilt detection, only the submissions of Tsinghua University [124] achieved better results than our proposed approach in terms of the f1-measure. The detection quality for pan achieved an average performance compared to the submissions of other institutes. In terms of the f1-measure, the Marburg submission achieved the top performance among all participating groups for the zoom detection task. The experimental results demonstrate the very good performance of the proposed approach. In particular, they show that the use of MPEG motion vectors is well suited to compute camera motion parameters as long as noisy motion vectors are removed appropriately from the vector field.

5.5 SUMMARY

In this chapter, an approach for the estimation of camera motion was proposed for the MPEG domain. In contrast to other approaches for this domain, its main advantage is the fact that it potentially distinguishes between rotation around the y-axis (x-axis) and translation along the x-axis (y-axis). Furthermore, the incorporation of an outlier removal algorithm reduced the noise in the motion vector fields and increased the detection performance. Our comprehensive video test set includes all kinds of camera motion and allows to entirely control the camera motion parameters in these videos. However, most detection errors are due to the difficult distinction of translational and rotational camera movements. The proposed approach achieved the best result for zoom detection and the second best result for tilt detection at TRECVID 2005 in terms of f1-measure.

Chapter Six

6 Self-Supervised Learning of Face Appearances in TV Casts and Movies

6.1 Introduction

The recognition of persons in videos and movies is an important task in order to understand the video content, since in most cases the plot is related to persons or actors. The following questions are sought to be answered: "In which shot Y and scene Z does person X appear, how often does person X appear in a video and, in which pose and size?".

While frontal face detection in images/videos and face recognition in constrained environments has reached a certain level of maturity, the recognition of persons in videos without any constraints and a priori knowledge (e.g., training of statistical models) remains a challenging and yet unsolved problem. This is due to the low quality and resolution of video recordings in contrast to high resolution photo imaging, and the high degree of freedom in face/person appearances in videos in terms of illumination and pose. Many approaches have been suggested for the tasks of face detection and face recognition in images and videos, but only a few proposals have been made for the task that is addressed in this chapter: Automatic person recognition/clustering in arbitrary videos without any a priori knowledge. In addition, some of the related approaches require a training stage to learn a model of the person which needs to be recognized. Many proposals for automatic cast/actor/person recognition in videos seem to simply ignore the state-of-the-art approaches in the field of face detection and recognition. Sometimes, particular face detection and face recognition procedures have become part of systems but neither their use has been rectified empirically nor their individual contribution to the overall system performance has been investigated.

The contributions of this chapter are as follows. First, an automatic video annotation system with respect to a person's occurrence is presented based on state-of-the-art building blocks for face detection, tracking and recognition. Second, it is demonstrated how the face detection results can be exploited to estimate the eye positions precisely to automatically cope with in-plane rotated faces. Third, the main contribution is to consider the task of indexing videos with person information as a transductive learning setting. To exploit this setting, the transductive learning ensemble framework, as presented in Chapter 3, is realized via a self-supervised learning approach. It is proposed to adaptively learn face appearances in a video by estimating relevant features for a person's face. Therefore, it is investigated in which way an initial face clustering result can be further improved by self-supervised learning, in particular with feature selection and appropriate re-classification using only the training face samples present in the given video. Several possibilities to train Adaboost and Support Vector Machine (SVM) classifiers or to train an ensemble of these classifiers, respectively, on a video are discussed and compared. Finally, experiments that investigate the contribution of the main components are presented. The work presented in this chapter has been partially published in [44, 47].

6.2 Related Work

The work in the field of face detection has been comprehensively surveyed by Yang et al. [187], and in the field of face recognition by Zhao et al. [196]. There are more than hundreds of approaches suggested for each of the two tasks, but only relatively few approaches address the problem of automatic actor/person recognition in TV casts and movies.

Eickeler et al. [34] apply a neural network for face detection and a pseudo 2-dimensional Hidden Markov Model (HMM) for face recognition. The recognition task is integrated into a k-means clustering procedure. Initally, a face is assigned to each cluster at random. A HMM is trained for each cluster and then the faces are assigned to the clusters according to the HMM classification result. This process is repeated until the clusters remain unchanged. The authors present experiments for a short news video where the faces of three news speakers and an interviewed person could be clustered correctly. Since k-means is used, the number of persons must be known in advance.

Raytchev and Murase [129, 130, 131] present three different algorithms for clustering face sequences captured in constrained environments. The similarity between two face sequences is measured by the minimum distance of a face of the first sequence and a face of the second sequence. The videos are recorded under the assumption that a person moves towards a fixed camera, such that motion information can be utilized to detect the face. Raytchev and Murase compute a distance matrix that holds the dissimilarity values for face sequence pairs [129]. Based on this matrix, a clustering process is applied. In their second paper [130], a clustering process is applied based on measures of attraction and repulsion. The values of attraction and repulsion are a function of the distance of two face sequences. For each face sequence in a cluster, it can be measured whether the other face sequences of this cluster attract (i.e., they are similar) or repulse (they are dissimilar) the given face sequence. In their third paper, so called VQ-faces are presented [131]. In a first step, all faces are clustered in an agglomerative clustering process in order to cluster the faces according to their pose. In the next step, the face clusters are connected by exploiting temporal interrelations. A recognition rate of 89.3 % is reported for 17 persons each having between 7 and 40 face sequences.

Fitzgibbon and Zisserman [52] present a distance metric that is invariant with respect to affine transforms. An arbitrary affine transform can be described by six parameters. The authors formulate a distance measure that estimates the square error between an image and an aligned image that was computed using the affine transform estimate. The face detection approach of Schneiderman and Kanade [135] is used to detect frontal faces. K-medoid clustering is applied to the face images so that the number of face classes (clusters) must be known in advance. This system was extended to an automatic video indexing system for person information [53]. Face appearances in consecutive frames are used to track a face using the distance metric described above. Then, a subspace is computed for a face sequence using Principal Component Analysis (PCA). The distance between two subspaces is measured using the joint manifold distance. Agglomerative clustering is applied to these subspaces until the distance between two clusters exceeds a predefined threshold. The authors present a cast face list for a test video with many duplicates, but present no recognition measures in terms of recall or precision.

There are several training-based approaches to person recognition in videos. For example, Everingham and Zisserman [50, 51] present a system for person identification in videos in which one training image is required for each person. An ellipsoid was chosen to represent a coarse 3D face model that is utilized to solve the problem of pose variation. The main idea is to compare only face appearances showing the same pose. Therefore, for each pose (described by a six-dimensional vector) of a person a view-dependent texture is saved. Very similar textures of the same pose are only saved if they possibly represent different facial expressions. For face detection, a color-based approach is applied that assigns a probability for each pixel to belong to a face region. Image pyramids are used to decide which regions have to be processed further. Then, the pose is estimated and the related views of the person are rendered with variations in facial expression. Edge-based features are now extracted from the detected face and the rendered views which are compared to find the best match to classify the face. If it is a new pose, the face model is extended accordingly. Experimental results are presented for three characters of a thirty minute sitcom: the ROC curves show recognition rates of 75-95% at a false alarm rate of about 10%.

Satoh [134] presents a complete system for person recognition in a TV soap. A neural network approach is applied for face detection, and tracking of faces is realized using a skin color model. The recognition procedure requires a number of training face sequences which should cover most of the possible variations in pose and expression. Several holistic recognition technologies are compared by Satoh, namely Eigenfaces, Fisherfaces, a subspace-based method and a kernel function subspace method. Experimental results are presented for an episode of a Japanese soap. Another episode of the same soap was used to train the recognition system. The best recognition results were achieved by the kernel function subspace method.

Acosta et al. [1] present a face recognition system for video indexing that utilizes a skin color based face detector and self-eigenfaces (an extension of eigenfaces described by Torres et al. [160]) for recognition. In the self-eigenface approach, a subspace is created separately for each person using five training images. MPEG-7 test sequences were used in the experiments and a recognition rate of 90% is reported. However, no details are given about the kind and the number of frames, shots or sequences.

6.3 Self-Supervised Learning of Face Appearances in TV Casts and Movies

An automatic system for cast/actor/person recognition is presented in this section. Given the shot and optionally the scene segmentation for an arbitrary video, the system outputs the number of persons present in a video and assigns person IDs to the shots.

The main assumption is that a person's face is shown frontally at least for a few frames in a professionally produced video. This assumption typically holds for the large majority of produced videos like movies and TV casts where the intention is to show the actors and persons in a recognizable manner. By utilizing a tracking procedure, our system can basically cope with issues of discontinuity and occlusion. Once a frontal face is detected, it is tracked until the next shot boundary. For example, if a person faces the camera, then turns around, and then turns back again to the camera, the face sequence is not interrupted and one coherent face sequence is generated by the system.

The system consists of the following main components (see also Figure 35): face detection, face sequence generation via feature detection and feature tracking, optional verification of skin color probability, selection of representatives for a face sequence, optional face sequence aggregation, a bunch graph matching procedure, in-plane rotation handling, and the face recognition module. The skin color verification module can be used for color videos in order to reduce the number of false alarms. It should be applied since any false alarm potentially degrades the clustering/recognition performance. The face sequence aggregation module should be used for all videos where the shots are structured and organized in scenes, which is the case for movies, feature films and soaps. In addition, to achieve robustness for a certain video, a self-supervised re-classification procedure is incorporated that utilizes Adaboost [55] for feature selection, and SVM, Adaboost and classifier ensembles to improve the learned face models. The main components are explained in more detail below.

6.3.1 Face Detection

The face detection approach of Viola and Jones [167] with Lienhart's extensions [99] is used in our system. While at least two other approaches [135, 186] achieve slightly better results in an experimental comparison on standard test sets [187], the Adaboost-based approach of Viola and Jones was chosen since it is a very fast approach that nearly operates in real-time on today's computers and thus can even be applied to every single frame of a sequence. Since this approach usually reports many detections of slightly different sizes and positions, an average rectangle is computed based on the reported detections, in case that the number of detections exceeds a threshold. The number of detection hits is later used to select the best faces within a face sequence with respect to frontal appearance.

6.3.2 Face Sequence Generation

A tracking procedure is used to assemble face appearances of the same person in subsequent frames. Once a face has been detected, the tracking procedure enables our system to conflate face appearances of a person which, for example, lowers and raises again the head or whose face is partly occluded for a short time. First, in the detected region of a frame, a feature detector is applied to find points of interest that are suitable for tracking. For this purpose, a feature detector has been selected that chooses the pixels with the highest eigenvalues. The estimated feature points are then tracked in the next frame using the optical flow computation of Bouquet [19]. This approach is an extension of the Lukas-Kanade [103] approach to compute the optical flow. The extension processes image pyramids to enable the estimation of fast movements as well. For this purpose, optical flow is estimated in the pyramid image with the lowest resolution. This first result is then refined in the iterations at higher resolutions.

Only feature points in the inner area of the detected face region are selected in order to reduce the influence of the background (see Figure 36). A face is tracked successfully if the ratio of tracked feature points within the detected face region of a subsequent frame is above a predefined percentage value. The feature points of a detected face are tracked as long as either an associated face appearance is found, or the end of the shot is reached. If a face is tracked successfully, new feature points are calculated and the tracking procedure starts again.

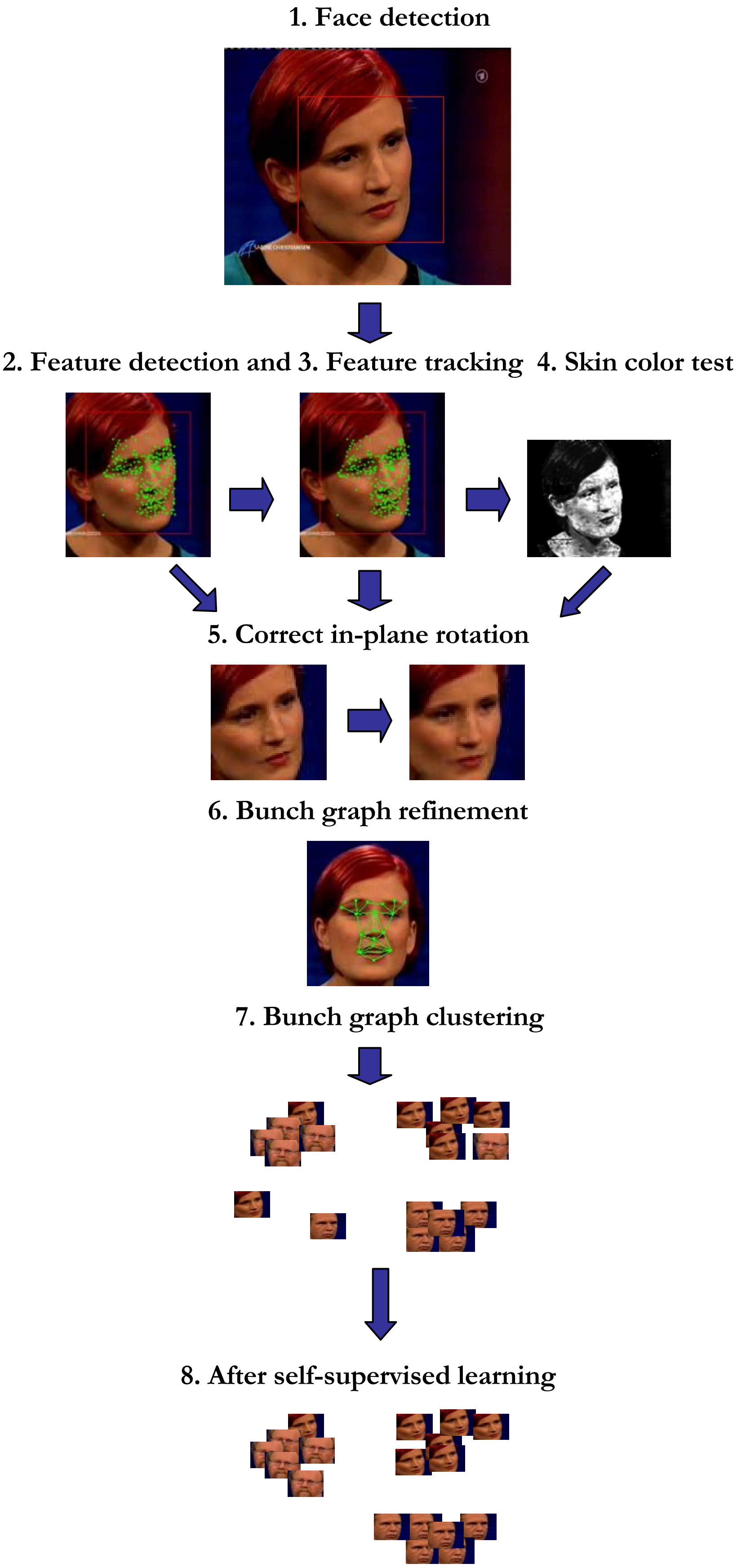

1. Face detection
2. Feature detection and 3. Feature tracking
4. Skin color test
5. Correct in-plane rotation
6. Bunch graph refinement
7. Bunch graph clustering
8. After self-supervised learning

Figure 35: The main processing steps of the self-supervised system are displayed here. Optionally, aggregation of face sequences appearing in dialogs can be applied.

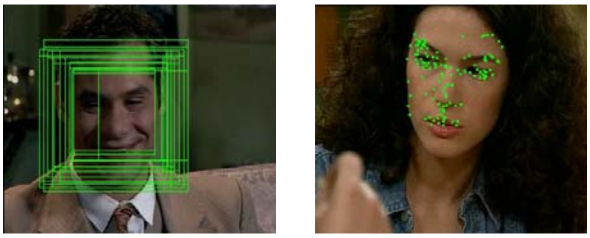


Figure 36, left: Example for face detections at slightly different scales and position. positions. Right: Only features in the inner face area are used for tracking.

6.3.3 Verification of Skin Color Probability

To reduce the number of false alarms, a verification step with respect to skin color probability is used optionally. Terrilon et al. [158] have compared nine different color spaces in terms of their applicability for skin color modeling. They conclude that the TSL (tint, saturation, luminance) color space, which was developed explicitly for skin color modeling, is most suitable for this task. Since skin color differs mainly in the luminance values, the T and S channels can be utilized to detect skin regions in images. Based on a training set of more than 4000 skin color vectors, the probability that a color pixel vector c=(s, t) is a "skin color pixel" can be computed via:

$$p(c \mid skin) = \frac{1}{2\pi\sqrt{\mid \sum_S \mid}} \cdot e^{-\frac{1}{2}(c-\mu_s)^T \sum_S^{-1}(c-\mu_s)} \tag{66}$$

where μ_S and the covariance matrix Σ_S are the model parameters of en elliptical Gaussian distribution; c is a vector that consists of the T and S components, and T, S and L in this color space are computed by the following formulas:

$$r = \frac{R}{R+G+B} \quad g = \frac{G}{R+G+B} \quad b = \frac{B}{R+G+B} \tag{67}$$

$$S = \sqrt{\frac{9}{5 \cdot (r'^2 + g'^2)}} \tag{68}$$

$$L = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \tag{69}$$

$$T = \begin{cases} \arctan\left(\frac{r'}{g'}\right) / 2\pi + \frac{1}{4}, & if \quad g' > 0 \\ \arctan\left(\frac{r'}{g'}\right) / 2\pi + \frac{3}{4}, & if \quad g' < 0 \\ 0, \quad if \quad g' = 0 & \end{cases} \tag{70}$$

where r'= r-1/3 and g'= g-1/3.

6.3.4 Aggregating Face Sequences in Dialogs

In this part, another optional processing step for movies can be applied that aims at connecting face sequences of temporally neighbored shots. Therefore, production rules for dialogs in movies are exploited. Arijon [6] describes typical patterns of film and movie production. One of these patterns is related to the usually applied techniques to show a dialog. Typically, several cameras are used to capture the action within a scene (a scene consists of several shots that belong together semantically in terms of space and time). Assuming that a scene segmentation is available (e.g., computed by the approach suggested by Truong et al. [163]), this movie production pattern for dialogs can be exploited to merge subsequent face sequences of one person. To aggregate face sequences, it is necessary to recognize which shots have been taken by the same camera. Therefore, position and size of the detected face regions are considered, as well as the histograms of the corresponding frames. The distance (dissimilarity) of two frame histograms (HSV color space) Q and V is computed using the Chi-square metric:

$$\chi^2 = \sum_i \frac{(q_i - v_i)^2}{(q_i + v_i)^2} \tag{71}$$

Two face sequences A and B (A preceding B) are merged if the following conditions are met:

- both face sequences belong to the same scene;
- the color histogram of the last frame of sequence A must be similar to the first frame of sequence B;
- the detected face region in the last frame of sequence A must significantly overlap the detected face region in the last frame of sequence B.

Face sequences within a scene are merged until there are no face sequences left that meet these conditions. Face sequence aggregation makes sense if many dialogs take place in the corresponding video, for example in feature films, movies or soaps.

.

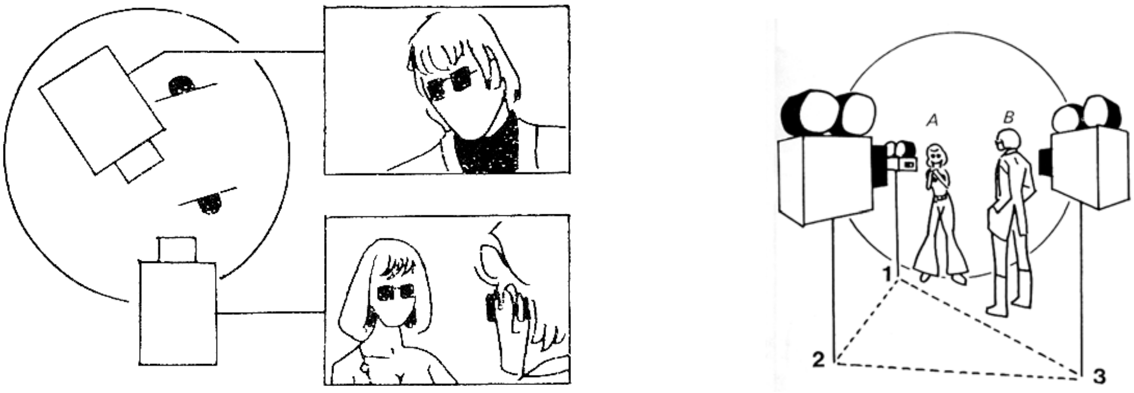


Figure 37: Settings to capture a dialog of two persons with two and three cameras (taken from [6]).

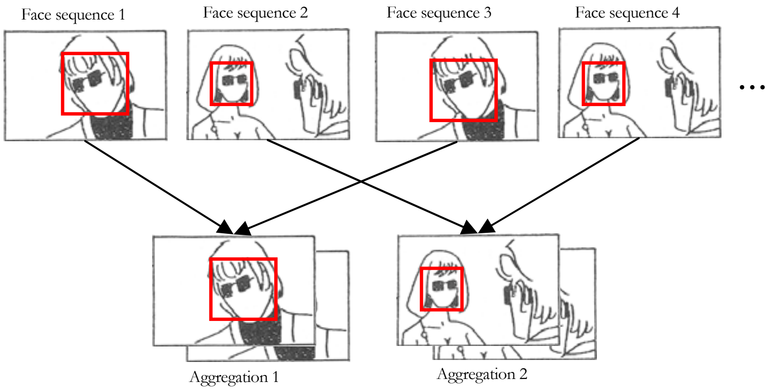


Figure 38: The principle of aggregating two face sequences A and B of two persons in a dialog scene (taken from [6]).

6.3.5 Bunch Graph Matching

The face recognition procedure used in the proposed system is based on the Elastic Bunch Graph Matching approach suggested by Wiskott et al. [174]. This approach was selected due to the following reasons. The used wavelet coefficients are (to a certain degree) independent of illumination changes and also, the bunch graph approach tolerates slight derivations in pose. Otherwise, different bunch graphs can be created for very different poses (e.g., frontal and profile). However, the bunch graph matching procedure is more difficult in low-resolution video. Therefore, only graph positions in the inner area of the face recognition are used for the frontal face graph in our system (see Figure 1). Since the nodes are not placed at the face border, face background has nearly no impact on the extracted features. Furthermore, it could be observed that the eye positions are detected constantly by the face detector. This can be explained if one considers the two best features reported in the paper of Viola and Jones [167]: the best two features clearly cover the eye region and thus represent luminance changes there. This is exploited by the bunch graph matching procedure (explained below) which is started at the eye positions for this reason.

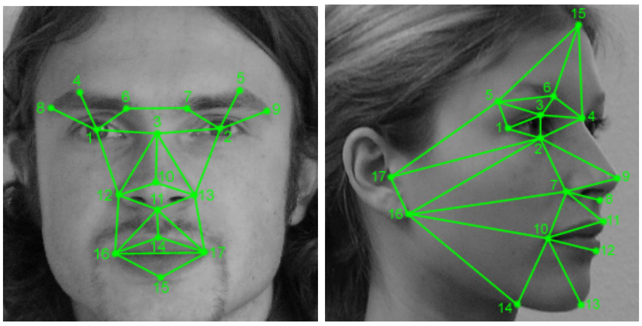


Figure 39: The face graph structure as used in the proposed system. Left: Frontal face graph. Right: Profile face graph.

The following procedure, derived from the localization procedure described by Bolme [17], is applied to find the bunch graph nodes. Node localization is divided into two steps: initial estimation and refinement. First, the previously estimated node positions are used to estimate the current node's position using the following formula:

$$\vec{p}_n = \frac{1}{M}\sum_{i=1}^{M}\vec{p}_i + \vec{v}_{in} \tag{72}$$

M is the number of previously localized nodes and its positions p_i. v_{in} is the translation vector between the i-th and the n-th node of the bunch graph. Each node provides an estimation of the current node's position. To increase the impact of closer nodes, weights are used additionally for each node:

$$\vec{p}_n = \frac{\sum_{i=1}^{M} w_{in}(\vec{p}_i + \vec{v}_{in})}{\sum_{i=1}^{M} w_{in}}, \text{ with } w_{in} = e^{-|\vec{v}_{in}|} \tag{73}$$

In the second step, the initial estimates are refined by a displacement estimation. The displacement estimation is based on the extracted wavelet features at each potential node position which are then compared with the wavelet features of the average bunch graph. The average bunch graph used in our system was created with 149 frontal faces of the FERET [127] database, normalized to a size of 64*64 pixels.

6.3.6 Correction of In-Plane Rotation

To estimate the in-plane rotation of the face, the rotation angle of the face is estimated using the eye positions which can be well localized based on the face detection results. In the optimal case, the eyes should be located on the same pixel row. Therefore, two iterations of the bunch graph matching procedure as described above are executed, but only for the nodes representing the eyes (and not for all nodes of the face graph). The estimated eye positions are used to compute a rotation matrix for the rotation angle a:

$$R = \begin{pmatrix} \cos(\alpha) & \sin(\alpha) & mid(X) \\ -\sin(\alpha) & \cos(\alpha) & \frac{dY}{2} + mid(Y) \end{pmatrix} \tag{74}$$

In Figure 40, examples for successfully rotated images are presented.

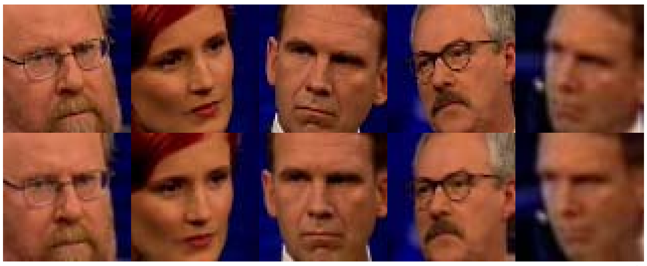


Figure 40: Examples of automatically corrected in-plane rotated faces. The original faces as extracted from the video are displayed in the top row. Below are the results after correcting in-plane rotation.

6.3.7 Gabor Wavelet Feature Extraction

Once the node positions of the face graph are estimated, the Gabor wavelet features are extracted from these positions. The functions to compute the wavelet coefficients can be expressed as follows

$$\begin{aligned} g_{\theta,\lambda,\varphi,\sigma,\gamma}(x,y) &= e^{-\frac{x'^2+\gamma^2 y'^2}{2\sigma^2}} \cos(2\pi \frac{x'}{\lambda} + \varphi) \\ x' &= x\cos\theta + y\cos\theta \\ y' &= -x\sin\theta + y\cos\theta \end{aligned} \tag{75}$$

A Gabor wavelet is controlled by five parameters: orientation θ, wave length λ, phase φ, radius σ of the Gaussian function, and the aspect ratio γ. The wavelet coefficients in the proposed system are computed according to the wavelet specification suggested by Bolme [17], as displayed in Table 61. Thus, at each node 40 wavelet coefficients are extracted, for five orientations and eight wave lengths. Such a set of wavelet coefficients is considered as a jet. A face bunch graph consists of all the jets for all nodes extracted from several frames, where the term bunch indicates that more than one jet (a bunch of jets) was assigned from two or more frames.

Wavelet Parameter	Values (according to [17])
Orientation	$\left\{0,\frac{\pi}{8},\frac{\pi}{4},\frac{3\pi}{8},\frac{\pi}{2},\frac{5\pi}{8},\frac{3\pi}{4},\frac{7\pi}{8}\right\}$
Wave length	$\left\{4,4\sqrt{2},8,8\sqrt{2},16\right\}$
Phase	$\left\{-\frac{\pi}{4},\frac{\pi}{4}\right\}$
Radius	$\sigma=\frac{3\lambda}{4}$
Aspect ratio	1

Table 61: The Gabor wavelet specification as used in our system.

6.3.8 FACE RECOGNTION/CLUSTERING

In the addressed application scenario, no classical recognition task is incorporated since the identities are not known in advance. Thus, an unsupervised clustering approach is employed in the beginning that assigns similar face sequences to the same cluster. Since the number of characters in a movie is not known in advance as well, an agglomerative clustering method is applied. For this purpose, a distance metric that describes the (dis-)similarity of two face sequences is needed. Three different possibilities to compare face sequences are integrated: Single Linkage Clustering, Centroid-based Clustering, and an Identity Bunch Graph method. Distance metrics are needed to compare jets, face graphs and face bunch graphs. The similarity of two jets J and J' is computed by

$$S_a(J,J')=\frac{\sum_j a_j a'_j}{\sqrt{\sum_j a_j^2 \sum_j a'^2_j}} \tag{76}$$

where a_j is the amplitude of the wavelet coefficient with index j, its phase is ignored since it changes rapidly depending on the position (but this property can be utilized for displacement estimation). The similarity of two face graphs F_1 and F_2 is computed by:

$$S_F(F_1,F_2)=\frac{1}{n}\sum_{i=1}^{n} S_a(J_i^1,J_i^2) \tag{77}$$

where n is the number of feature positions (nodes and edge positions). This similarity function (interval: [-1, 1]) can be easily transformed to a dissimilarity function:

$$d_F(F_1,F_2)=0.5\cdot\left(1-S_F(F_1,F_2)\right) \tag{78}$$

The dissimilarity of two clusters (of face sequences) X and Y with face graphs F and G according to Single Linkage clustering is then:

$$d_{SLC}(X,Y) = \min_{F \in X, G \in Y} d_F(F,G) \tag{79}$$

The dissimilarity of two clusters (of face sequences) X and Y according to centroid-based clustering requires the computation of a centroid face graph, which consists of average jet amplitudes:

$$a_{i,j}^{F_X} = \frac{1}{N}\sum_{\mathrm{k=1}}^{\mathrm{N}} a_{\mathrm{i,j}}^{\mathrm{F_k}} \tag{80}$$

where N is the number of face graphs in cluster X, m is the number of jets of a graph (i = 1, ..., m), n is the number of amplitude values of a jet (j=1, ..., n)., and F_X is the centroid face graph. The dissimilarity of two clusters X and Y with centroid face graphs F_X and F_Y, respectively, is then defined as:

$$d_C(X,Y) = d_F(F_X, F_Y) \tag{81}$$

Similarity measurement based on identity bunch graphs allows comparing unknown views of persons by combining a person's jets of different face appearances. A face bunch graph holds the jets for several face appearances of a person, such that a node holds not only the jet of one face but a whole bunch of jets extracted from several faces. The similarity of a face graph F and an identity bunch graph is defined as:

$$S_{FtoBG}(F,B) = \tfrac{1}{N}\sum_{i=1}^{N} \max_j (S_a(J_i^F, J_{i,j}^B)) \tag{82}$$

,where $J^B_{i,j}$ is the j-th jet of the i-th bunch of the bunch graph. A jet from the face graph F is compared with all jets of the corresponding bunch and the most similar jet is selected. The similarity of two clusters X and Y is then:

$$S_{IBG}(X,Y) = \max(\max_{F \in X}(S_{FtoBG}(F,B_Y)), \max_{F \in Y}(S_{FtoBG}(F,B_X))) \tag{83}$$

The related distance function is:

$$d_C(X,Y) = d_F(F_X, F_Y) \tag{84}$$

With any of these three distance measures, the clustering works as follows. Initially, each face sequence represents one cluster, except when an aggregation process has taken place: Then, a cluster may contain several aggregated face sequences. After initialization, the cluster distances are computed pair-wise. If the minimum distance does not exceed a pre-defined threshold, the two most similar clusters are merged into one cluster and its distance to all other clusters must be re-computed. This process is repeated until no pair of clusters is sufficiently similar, which is the case when all dissimilarity values exceed a threshold.

6.3.9 Self-Supervised Learning for Robust Face Clustering

The clustering process outputs a number of clustered face sequences as its result. However, in this clustering process all the wavelet features have contributed equally to the similarity measure of two faces. In this way, the characteristic features of a person (e.g., beard, eyes) might be covered by many other features which are not suitable to distinguish between two persons. Furthermore, it is observable that the clustering process often also results in a number of clusters with only one face sequence, often related to a less frequent pose or facial expression of a person. Here, the idea is to stop the clustering process conservatively (i.e., to use a low threshold to terminate the clustering process early) in order to prevent the merging of face sequences of different persons.

Each cluster that has a minimum number of face sequences is further considered as a training set for an identity. The members of this cluster are the positive training samples for this identity, whereas the face sequences of the other clusters are considered as negative training samples (they should also exceed a minimum number of face sequences, in order to reduce the number of false training samples). At this point, the training sets may contain some face samples which have the wrong labels. The idea here is that as long as the correctly labeled samples dominate the training set for an identity, the classifier should generate a generalized model that mainly represents the "good" samples of the training data. In the training stage, not only the amplitudes of the Gabor wavelet features are used, but also the phase information. Adaboost [55, 167] is utilized to select the features that distinguish best between a considered cluster X and all the other clusters. These features are then used to train a classifier. Several possibilities have been investigated: First, since Adaboost is a meta-classifier itself, it can be used for subsequent classification. Second, the selected features can be used to train any other classifier: in our experiments, a SVM (support vector machine) is used. Another possibility is to build an ensemble of classifiers using majority voting. In this case, the classifiers forming an ensemble should have a certain level of independence, as it has been discussed in Chapter 2. In the self-supervised approach for cut detection proposed in Chapter 4.7, the boosting procedure of Adaboost was successfully exploited and features were assigned in an alternating fashion to two different classifiers according to their rank in the Adaboost feature selection process.

The idea of the Adaboost approach is to combine a number of n "weak classifiers" to build a strong classifier within n rounds of training. For each weak classifier (based only on one feature in our case), a minimum classification error is estimated. This classification error is computed based on the weights of the training samples that are weighted equally in the beginning. Misclassified training samples are re-weighted such that they have more impact in the next training round for the next "weak classifier". Thus, a selected feature has a higher probability to correctly classify those training samples that have been misclassified in preceding rounds. This property is the motivation to split the feature set depending on their rank for subsequent training. In case of n classifiers, the feature with rank k is assigned to classifier k modulo n. If Adaboost is chosen as the classifier, this procedure can be viewed as a reorganization of the ensemble.

Then, each classifier is trained with its own feature set using the clustering result as training data. The n classifiers are combined to form an ensemble of classifiers using the number of votes for each person. Finally, each face sequence is classified with k classifiers (where k is the number of clusters from the clustering process with a minimum number of face sequences). A face is assigned to cluster i (with classifier i) if the number of votes of classifier i exceeds all other ensemble votings and is at least 1. The overall process is called self-supervised, because the system labels the training data all by itself.

6.4 Experimental Results

In this section, experimental results for a TV news video (about 4 minutes duration, 4 persons), a talk show (about 8 minutes duration, 6 persons) and for a TV soap (about 38 minutes duration, 4 characters) from the MPEG-7 [14] video test set are presented. Some system components are realized using Open Source Software, namely: Intel's OpenCV library [www.intel.com/technology/computing/opencv] is used for face detection, feature detection and optical flow estimation (feature tracking), the libSVM library is used as the SVM implementation [http://www.csie.ntu.edu.tw/~cjlin/libsvm], the Elastic Bunch Graph Matching code for face recognition/comparison is provided at [www.cs.colostate.edu/evalfacerec/algorithms5.html].

The impact of several parameter settings on the overall recognition/clustering performance. has been tested systematically. Selecting faces according to the number of hits of the face detector ended up with better face clustering results than using the similarity to the trained face bunch graph. The impact of different parameter settings, in-plane rotation removal and self-supervised training is investigated in our experiments as described below. In all reported experiments, Single Linkage clustering was applied.

The experiments were evaluated as follows. The ground truth data were created on a per shot basis per frontal face appearance. Given a clustering result, a particular cluster should contain all the shot indices in which a certain person appears. A cluster C is assumed to represent the person who dominates the cluster. If two or more clusters are dominated by the same person, only the cluster C_x which contains the maximum number of face sequences for this person X is considered for the calculation of recall. Let c_x be the number of shot indices that are correctly assigned to cluster C_x for a person X, and let T_x be total number of appearances of this person, then recall and precision for person X are defined as:

$$recall_X = \frac{c_x}{T_X}; \quad precision_X = \frac{c_x}{|C_X|} \tag{85}$$

The overall recall and precision are defined as:

$$RECALL = \frac{\sum_X c_X}{\sum_X T_X}; \quad PRECISION = \frac{\sum_X c_X + \sum_Y c_Y}{\sum_X |C_X| + \sum_Y |C_Y|} \tag{86}$$

where $|C_x| = c_x + f_x$; f_x is the number of falsely assigned face sequences. The clusters C_y are those clusters which have not been assigned to a person X. Let c_y be the number of shot indices of the person who dominates cluster C_Y. Recognition is measured on the basis of the detected face sequences; the F1-measure is computed by:

$$F1 = \frac{2 \cdot RECALL \cdot PRECISION}{RECALL + PRECISION} \tag{87}$$

First, the baseline system, without removal of in-plane rotation and self-supervised training, was tested on the news and talk-show video. In the news video, the baseline system achieves a perfect recognition result without self-supervised learning. Several experiments for the more demanding talk show video have been conducted to investigate the sensitivity of the overall system performance with respect to different parameter settings. The second series of experiments investigates the contribution of in-plane rotation removal and self-supervised learning for the talk show sequence. Finally, experimental results are presented for a TV soap with a duration of 38 minutes.

In the first series of experiments, the impact of the following parameters was investigated, the default values are given in brackets:

- minimum number of detection hits to accept a face detection; (5)
- minimum number of detected faces needed to form a face sequence; (11)
- threshold for skin color probability to verify a face occurrence; (0.1)
- maximum number of best faces representing a face sequence; (9)
- threshold to terminate the clustering process; (0.03)
- number of selected features in the self-supervised learning process. (30)

In the first experiment, the number of minimum detection hits was varied: 1, 3, 5, 7, 9. The experimental results are presented in Table 62. It is observable that precision degrades (due to an increasing number of false alarms) when only one detection hit is sufficient to verify face detection. Best results are achieved for a minimum of 5 or 7 detection hits.

In our system, a minimum number of detected faces must be exceeded within a tracked sequence in order to consider it for the subsequent recognition process. The parameter defining this minimum number was tested with the following values: 5, 8, 11, 14, 17. The corresponding experimental results are displayed in Table 63.

The threshold for skin color probability to verify a face detection is set very conservatively in our system to identical experimental results.

A face sequence is represented in the clustering and in the self-supervised learning process, respectively, by the *n* "best" faces. The best faces were selected according to the number of reported detection hits by the face detector. Gabor wavelet features were extracted only from these faces (each face is scaled to size 64*64). The number of best faces representing a face sequence has been varied in the next experiment and the values 1, 3, 6, 9, 12 have been tested. Experimental results are presented in Table 64. The results are quite stable for all tested values, however, the best performance was achieved using a medium number of 6 or 9 representative best faces.

The threshold for skin color probability to verify a face detection is set very conservatively in our system to prevent obvious false alarms. All experiments, using the probabilities 0.05, 0.1, 0.15, 0.2, and 0.25, yielded identical experimental results.

Finally, the impact of the clustering termination threshold was tested. The results are presented in Table 65. The results in this experiment are quite stable as well, except for the threshold of 0.036. In this case, too many clusters are merged and the performance degrades.

Detected Faces	1	3	5	7	9
Recall	72.0	73.3	70.7	62.7	60.0
Precision	73.6	88.3	94.0	95.0	96.0
F1-measure	72.8	80.1	81.0	76.0	73.8
#Detected Persons	38	17	13	13	9

Table 62: The experimental results for the talk show sequence when the number of detection hits is varied in the process of face sequence generation.

Minimum Number of Faces per Sequence	5	8	11	14	17
Recall	70.7	70.7	70.7	70.7	69.3
Precision	91.8	94.3	94.0	94.0	93.8
F1-measure	79.9	80.8	80.7	80.7	79.8
#Detected Persons	17	16	13	13	12

Table 63: The experimental results for the talk show sequence when the minimum number of faces forming a face sequence is varied in the face sequence generation process.

Maximum Number of Best Faces per Sequence	1	3	6	9	12
Recall	58.7	65.3	68.0	70.7	68.0
Precision	97.0	95.5	97.0	94.0	82.1
F1-measure	73.1	77.6	80.0	80.7	74.4
#Detected Persons	25	20	18	13	9

Table 64: The experimental results for the talk show sequence when the maximum number of best faces representing a face sequence is varied.

Clustering Termination Threshold	0.024	0.027	0.03	0.033	0.036
Recall	62.7	66.7	70.7	68.0	40.0
Precision	97.0	97.0	94.0	82.1	51.6
F1-measure	76.1	79.0	80.7	74.4	45.1
#Detected Persons	20	18	13	8	4

Table 65: The experimental results for the talk show sequence when the clustering termination threshold is varied.

Overall, although a number of parameters must be set in our system, the first series of experiments indicates that the system performance is not critically sensitive to certain parameter settings.

In the second series of experiments, the impact of in-plane rotation and self-supervised learning was investigated. The following parameter settings were used based on the best results of the experiments described above. The minimum number of detection hits was set to 5. A face sequence was accepted if there were 11 faces correctly detected and tracked with a skin color probability of at least 0.1. The number of best faces that were extracted per face sequence was set to 9. For the talk show sequence, 67 out of 75 frontal face sequences were extracted correctly without any false alarms using the system's face detection and face sequence generation. The experimental results are displayed in Table 66 for two different thresholds for clustering termination. Recall and precision are shown for the whole clustering result as well as for each person separately. The termination criterion t is displayed in the first row. Overall recall is about 66-70% with a high precision of about 95%, hence, clusters that represent a person mainly hold faces of the same person.

Threshold t	0.027		0.03	
	Rec.	Prec.	Rec.	Prec.
Person 1	87.0	100.0	100.0	100.0
Person 2	73.7	100.0	84.2	94.1
Person 3	72.7	100.0	72.7	80.0
Person 4	50.0	100.0	50.0	100.0
Person 5	33.0	100.0	-	-
Person 6	30.0	100.0	30.0	100.0
Overall	66.7	97.0	70.7	94.0
Recognition	74.7	97.0	79.1	94.0
F1-measure	79.0		80.7	
#Detected Persons	18		13	

Table 66: The experimental results for the talk show sequence using the baseline system. Recognition is computed based on the detected face sequences only.

Threshold t	*0.025*		*0.027*	
	Rec.	Prec.	Rec.	Prec.
Person 1	100.0	100.0	100.0	100.0
Person 2	78.9	94.7	84.2	94.1
Person 3	72.7	80.0	72.7	80.0
Person 4	83.3	100.0	83.3	100.0
Person 5	-	-	-	-
Person 6	60.0	100.0	60.0	100.0
Overall	76.0	94.0	77.3	94.0
Recognition	85.1	94.0	86.5	94.0
F1-measure	84.0		84.8	
#Persons	10		9	

Table 67: The experimental results for the talk show sequence when in-plane rotation of faces was performed. Recognition is computed based on the detected face sequences only.

Classifier	Adaboost (30 features), 1*30	
	Rec.	Prec.
Person 1	100.0	100.0
Person 2	94.7	94.7
Person 3	90.9	83.3
Person 4	83.3	100.0
Person 5	-	-
Person 6	70.0	87.5
Overall	84.0	94.0
Recognition	94.0	94.0
F1-measure	88.7	
#Persons	5	

Table 68: The experimental results for the talk show sequence when inplane-rotation was performed and Adaboost was used to learn facial features of face clusters. Recognition is computed based on the detected face sequences only.

Classifier	SVM (30 features), 1*30		3-SVM Ensemble (30 features), 3*10	
	Rec.	Prec.	Rec.	Prec.
Person 1	100.0	100.0	100.0	95.8
Person 2	94.7	94.7	94.7	94.7
Person 3	90.9	83.3	90.9	83.3
Person 4	83.3	100.0	83.3	83.3
Person 5	-	-	-	-
Person 6	80.0	80.0	80.0	80.0
Overall	85.3	92.7	85.3	90.1
Recognition	95.5	92.7	95.5	92.7
F1-measure	88.8		87.6	
#Persons	5		5	

Table 69: The experimental results for the talk show sequence when inplane-rotation was performed and SVM or a 3-SVM ensemble, respectively, was used to learn facial features of face clusters.

Number of Selected Features	5	15	30	45	90
Recall	82.7	84.0	84.0	84.0	84.0
Precision	92.5	94.0	94.0	94.0	94.0
F1-measure	87.3	88.7	88.7	88.7	88.7
#Detected Persons	5	5	5	5	5

Table 70: The experimental results for the talk show sequence when the number of selected features is varied using the Adaboost classifier.

The experimental results for the system that was enabled to deal with in-plane rotation are presented in Table 67 (again for two different thresholds for clustering termination); some examples for correcting in-plane rotation are presented in Figure 4, left. Using this technique, an overall recall of 76-77% could be achieved while precision could be preserved clearly above 90%. Along with that, the number of clusters was noticeably reduced and approximated nearly the correct number of persons in the best clustering. In the next experiment, it has been investigated whether it is possible to learn characteristic facial features and face models directly from the given video sequence, based on the initial clustering result (default settings + in-plane rotation handling, *t*=0.027). Only clusters with at least three different face sequences were considered for training. The following scenarios were tested: Training of an Adaboost classifier with 45 features, and training of a SVM or of 3 SVMs, respectively, in order to form an ensemble of SVMs using 30 features selected by Adaboost. The experimental results for the Adaboost classifier is presented in Table 7. In Table 8, results for a SVM and an ensemble of SVMs are shown.

The results in Table 68and Table 69 indicate that the system was indeed able to learn characteristic facial features of the different persons from the initial clustering result by itself. The best results were achieved with the single Adaboost classifier (30 features) yielding an overall recall of 84% while preserving a precision of 94%, and with the SVM classifier (30 features) resulting in an overall recall of 85.3% and a precision of 92.7%. The f1-measure was increased with self-supervised learning from 84.8 (baseline system + removal of in-plane rotation) up to 88.8. Finally, Adaboost-based learning with different numbers of selected features has been tested for the talk show sequence. The results are presented in Table 70. Interestingly, a small number of five features is sufficient to achieve a very good result.

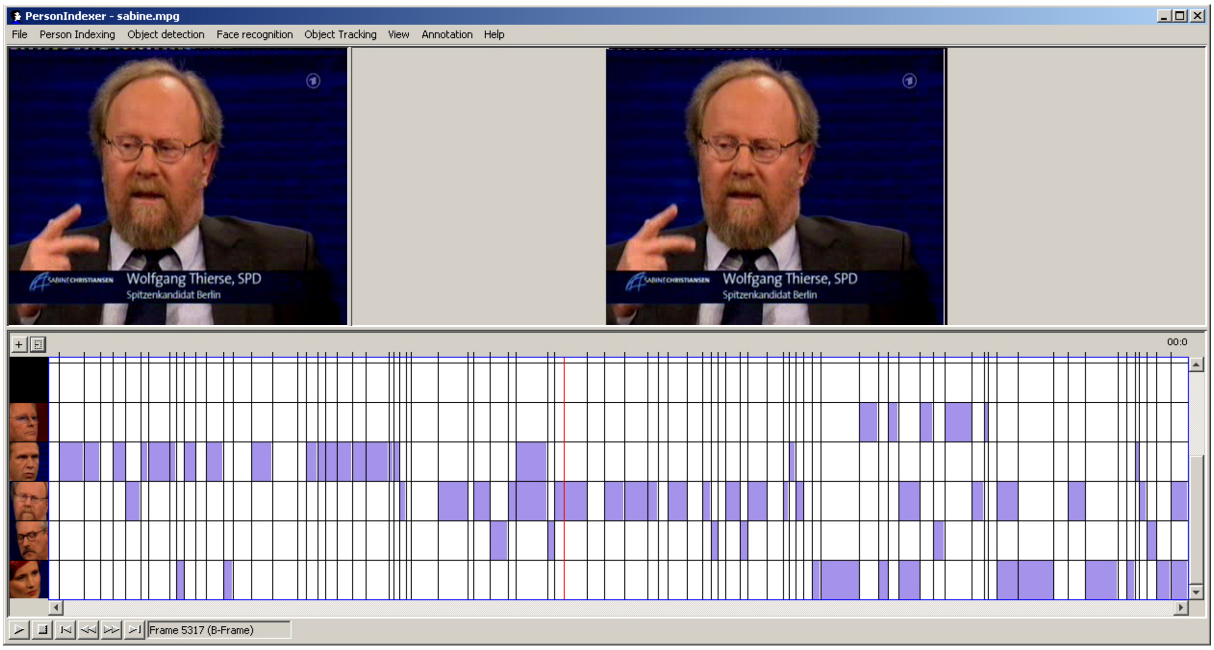


Figure 41: Example for a shot-based person indexing result for the talk-show sequence.

Furthermore, the potential of self-supervised learning has been investigated for the video “camiloefilho” (duration: 38 minutes, 4 characters) from the MPEG-7 [14] video test set. Using a given scene segmentation, aggregation of face sequences in dialogs was applied. The parameters are slightly adjusted according to the longer duration of the video. The clustering termination threshold was set to 0.04. Only clusters with a minimum number of 10 face sequences were used to train a face model. This way, the recognition procedure could be boosted for three of four actors. In Figure 4, right, the selected best 5 features are shown for two characters from this test video. Unfortunately, character 4, who does not appear very often in this video, was lost since the number of face sequences related to him was quite small. Nevertheless, using the Adaboost classifier with 45 features for each person, the overall recognition rate increased from 71.1% up to 81.7% while a high precision could be preserved (89.1% after learning, 90.7% before). The same features were used to train a SVM for each person: here, a similar overall recognition rate of 81.2% and a precision of 88.5% was achieved. An example for an indexing result for the talk show sequence is presented in Figure 41, in Figure 42 these results are compared to the related ground-truth data.

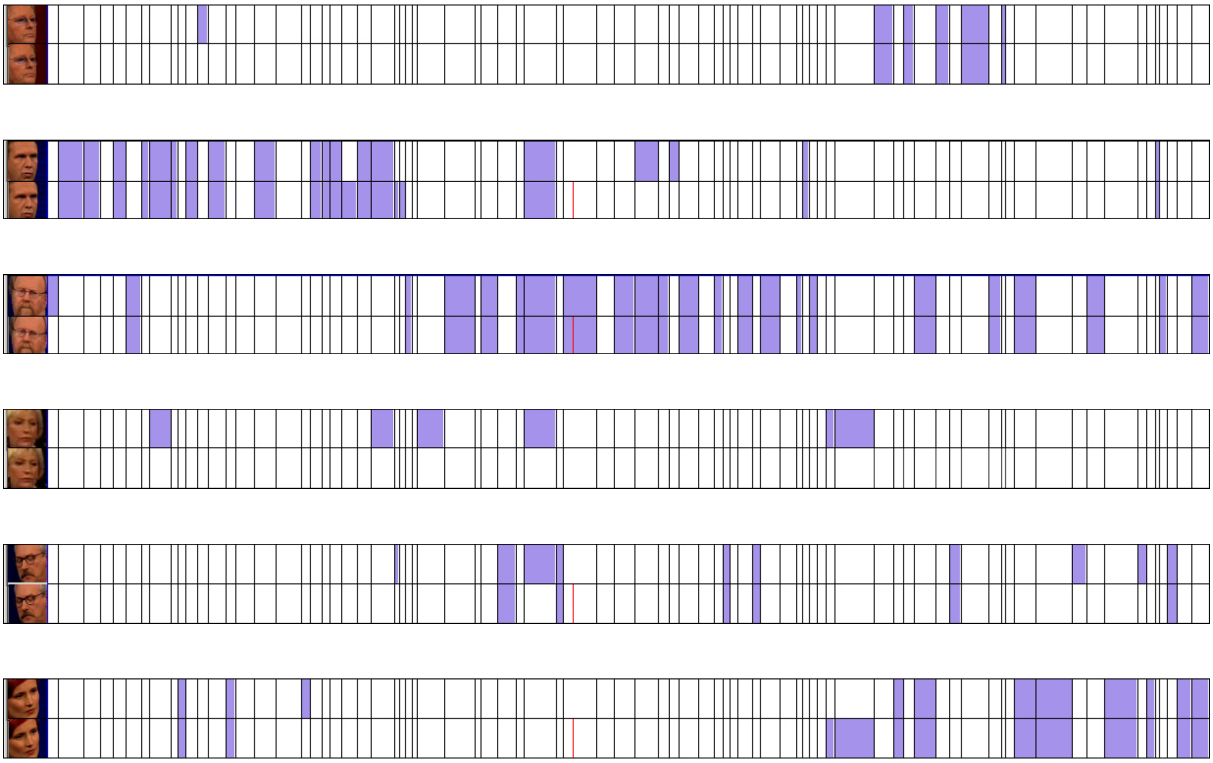


Figure 42: Example shot-based indexing results for the six persons of the talk-show sequence. Vertical lines represent the shot boundaries. For each person, the upper timeline represents the ground-truth data, and the bottom timeline represents the automatic indexing result.

A comparison to earlier systems is difficult. In contrast to our approach, the systems proposed by Acosta et al. [1], by Everingham and Zisserman [50, 51], and by Satoh [134], respectively, require a training stage using labeled face data. Thus, the systems can only recognize persons which are known in advance. Comparable approaches are presented by Eickeler et al. [34], Fitzgibbon and Zisserman [53], and Raytchev and Murase [129, 131]. Eickeler et al. [34] used only a short news video in the experiments - the proposed baseline system achieved a perfect result for a similar video as well, even without self-supervised learning. Fitzgibbon and Zisserman [53] present only visually the cast face result list but no results are given in terms of recall/precision. Raytchev and Murase [129, 131] address a constrained environment in which people walk towards a fixed camera. Overall, the proposed system achieves in more demanding settings at least comparable results to [34, 53, 129, 131] and, in addition (in contrast to [1, 50, 134]), no labeled training data are required in our approach.

6.5 Summary

In this chapter, a novel automatic video annotation system with respect to a person's occurrence was presented. It was demonstrated how face detection results can be exploited to estimate the eye positions precisely in order to automatically cope with in-plane rotated faces. Experimental results showed the effectiveness of the proposed in-plane rotation removal. Furthermore, it was investigated in which way an initial face clustering result can be further improved by self-supervised learning, in particular with feature selection and appropriate re-classification. For this purpose, the task was considered as a setting of transductive learning and only the face samples present in the given video were used. Several possibilities to train Adaboost classifiers, SVM classifiers, and to train an ensemble of these (transductive) classifiers on a video were presented and compared. Finally, experimental results demonstrated that it is sufficient to train a single classifier: the best performance measures were achieved by the Adaboost and a SVM classifier, respectively, each increasing the F1-measure from 84.8 up to 88.8.

Chapter Seven

7 Semi-Supervised Learning for Semantic Video Retrieval

7.1 Introduction

The ambitious goal of research efforts in the domain of multimedia retrieval is the automatic understanding of audiovisual content. If it were feasible to automatically understand what is shown in a video shot or what is said in a news cast, then it would be much easier to answer user queries for multimedia databases. However, automatically recognizing objects and events in an image or in a video is a difficult problem, although there has been some success for particular object recognition tasks, such as face detection [187]. Naphade and Smith [117] state that first TRECVID [161] efforts showed that approaches such as query by content using low-level features are inadequate to successfully search a large video collection. This is the reason why current research in the field of video retrieval focuses on the detection of so-called high-level features (concepts or topics) in order to index video shots to finally support different kinds of queries. Recent approaches to high-level concept detection are typically based on automatically extracted low-level features which are used by supervised machine learning methods to infer about semantic scene content. Such low-level features are, for example, color histograms, texture and shape descriptors [106], motion information [81] for video content, and the zero crossing rate ratio, the short-time energy ratio or the spectrum flux [102] for audio content. The incompatibility of low-level features that can be extracted automatically for an audiovisual scene and the high-level meaning associated by humans is considered as the "semantic gap" (e.g., Dorai and Venkatesh [33], Gevers and Smeulders [59]).

In an arbitrary video, the number of object classes, objects (or class instances), events and topics is very large, and it is not reasonable to use a specialized detector for each possible concept. Furthermore, even for partially solved object detection or object recognition problems like face recognition, the system performance heavily depends on pose and illumination of a face [127]. In addition, in an arbitrary video, the context of events and person occurrences determines the complexity of the recognition task as well, for example a person's appearance depends on clothes and age. This does not only hold for person recognition but also for the overwhelming majority of semantically relevant objects and events. Often, the meaning or the appearance of a certain event or concept is strongly determined by contextual information. For example, the appearance of a high-level concept, such as maps or news anchors in a certain news program, is strongly determined by the editing layout which is specific for a certain broadcasting station.

Since the appearance of certain concepts is related with a particular video source or program, the task of detecting these concepts can be considered as a transductive learning setting. In this chapter, two methods to adaptively learn the appearance of certain objects or events for a particular video with the aim of enhancing the retrieval quality are proposed to exploit this setting: 1.) a semi-supervised learning ensemble approach based on our framework described in Chapter 3 and 2.) an approach using transducitve SVM. In the preceding Chapters 4 and 6, it has been shown that an initial object model obtained for a particular video via unsupervised learning can be improved adaptively for this video via our proposed framework. Therefore, the given video content analysis task has been considered as a transductive learning setting. The transductive learning ensemble (as proposed in Chapter 3) has been realized in a self-supervised learning manner because only unlabeled data have been used in these tasks. In Chapter 7.3, it is shown that a transductive learning approach can also improve the retrieval performance for detecting high-level concepts, although the performance of the available baseline classifiers (based on supervised learning) is relatively low. For this task, the transductive learning ensemble is realized in semi-supervised way since it relies on supervised baseline classifiers. To improve results for particular videos, the fact is exploited that there are concepts whose appearance is strongly related to a certain video source (e.g., maps in a news cast). The novel idea is to estimate the relevant features for a given concept in a particular test video *v*. It is assumed that an initial concept model is available which has been obtained via supervised learning using a set of appropriate training videos. Based on this initial model, shots are ranked separately for each test video *v*. Then, features are selected with respect to the concept's appearance in this video *v* and new classifiers are trained using only the best and worst ranked shots as positive and negative training samples. The feature set is split in order to train additional classifiers with different views to assemble a robust ensemble of classifiers which is finally used to re-classify or to re-score the shots of this particular test video *v*. The second proposal (Chapter 7.4) considers the scenario as a transductive setting as well and transductive SVMs are applied to improve concept detection in particular videos. Experimental results for the MediaMill Challenge [18] part of the TRECVID 2005 video set demonstrate the feasibility of the proposed approaches for certain concepts. The work presented in this chapter has been partially published in [48, 49].

7.2 Related Work

First, state-of-the-art approaches to concept detection are presented, such as the IBM Video Retrieval System [5] and the MediaMill system [147]. A review about concept detection systems is given by Hauptmann and Christel [70]. These approaches have in common that a mapping is learned between low-level and high-level features using machine learning techniques. Furthermore, several classifiers are trained on different feature subsets and on the whole feature set, respectively.

Second, since unlabeled data are augmented in the proposed semi-supervised learning framework, related approaches that incorporate unlabeled data in the training process in order to reduce the labeling effort are discussed.

Amir et al. [5] present a set of different machine learning techniques to map low-level features to high-level semantic concepts. The machine learning techniques are: Support Vector Machine, Gaussian Mixture Models, Maximum Entropy Methods, a modified Nearest-Neighbor classifier, and Multiple Instance Learning. Furthermore, different fusion methods are investigated. The authors have conducted extensive experiments on TRECVID video data (the 2003 and 2005 development set) to select the best audiovisual and textual features for this task. A semantic model vector (according to Smith et al. [142]) is built consisting of confidence scores for 39 LSCOM-Lite concepts (LSCOM: Large Scale Concept Ontology for Multimedia, www.lscom.org). The idea of semantic model vectors is to train a binary classifier for each concept and combine the classifiers' outputs or their confidence scores, respectively, in the so-called semantic model vector. Optionally, Principal Component Analysis can be applied to a model vector if dimensionality reduction is needed. Semantic model vectors represent the video shots and are finally used to compare different videos for retrieval purposes. Amir et al [5] conduct several experiments where several strategies to fuse features or classifiers are investigated.

Snoek et al. [147] suggest the semantic pathfinder to extract semantic concepts from video shots. The idea is based on an authoring metaphor where videos are considered as edited entities which are produced with a certain intention and purpose. Videos are processed in the semantic pathfinder in several analysis steps: the "content analysis" step, the "style analysis" step and the "context analysis" step. In the content analysis step, the following modalities are considered: visual features and text which is obtained from transcribed speech. A multi-class SVM is trained for a number of pre-defined proto-concepts using only a few training samples. The feature vector representing the visual content consists of the percentage of pixels per proto-concept. For textual analysis, for each concept *x* a separate lexicon is created that contains the words (after stop word removal) that co-occur with *x* in the training set shots. The textual feature vector per shot consists of a histogram for the words related to concept *x*. The visual feature vector *v* and textual feature vector *t* are fused and serve as input to train a visual model for each concept using SVM which represents the first step: "content analysis". The second step, "style analysis", considers: 1.) layout (shot length, overlaid text, silence, voice-over), 2.) content (faces, face position, cars, object motion, frequent speaker; length of overlaid text, video text named entity; voice named entity), 3.) capture (camera distance, camera motion, camera motion type) and 4.) context, which serves to enhance or reduce the correlation between semantic concepts. The last step, "context analysis", uses the probabilities of style analysis that a concept is present in a shot: these probabilities are fused in the context vector *c*. Finally, a held-out validation set is used to find for each concept individually the best path through those three analysis steps for content, style, and context. This best concept path is then used to retrieve shots.

In a way, these approaches are representative for the concept detection part of the current generation of video retrieval systems. In recent years, first works have considered the incorporation of unlabeled data into the supervised learning process. Blum and Mitchell [13] suggest co-training in order to augment unlabeled data into the classification process. It is assumed that there are two different independent views of the classes or two different independent views of the data, respectively. The feature set is divided according to these views for each sample and each classifier. Each classifier tries to classifiy unlabeled data and passes the data with the highest confidence score to the other classifier as training data. This process is repeated several times. Yan and Naphade [184] extend the co-training approach for semantic concept detection in video shots. They divide the feature set for a video shot into the textual and the visual representation and thus meet the prerequisite for co-training. Since classifiers for semantic video concepts are not sufficiently accurate and the use of incorrectly labeled data might degrade performance, they integrate a human annotator in the processing loop who reviews and eventually corrects the automatically labeled samples. They report experiments with several co-training iterations for four semantic concepts from the TRECVID 2004 high-level feature detection task: airplane, basketball, Bill Clinton and people. Experiments show that the performance degrades with the number of iterations when the original co-training approach has been used, presumably due to the low accuracy of the baseline classifiers. In case when the automatically labeled data were corrected by a human reviewer, the classification accuracy increased with the number of iterations, at least for most of the iterations. Anyway, compared to the baseline performance, an improvement was achieved for any concept and any iteration number, which was not always the case for the standard co-training procedure.

Yan and Naphade [185] present another extension called "semi-supervised cross feature learning" for co-training. First, initial classifiers are trained on the labeled part of the training set. Then, in each iteration, samples that are classified with highest and lowest confidence by one classifier form a training set for the other classifier which is then trained only on the automatically labeled training set (without the samples from the original, manually labeled training set). Third, the performance of the newly trained classifiers is tested on a validation set and a corresponding weight is computed for each classifier. This weight is possibly 0 in case the classifier would degrade the performance of the initial classifier. The authors show theoretically that the minimal risk (to make an error) is never higher for the ensemble created in this way. Finally, they extend the learning approach for multiple views. Their experimental results for 11 concepts from the TRECVID 2003 data set show that the proposed semi-supervised approach slightly improves average precision of the baseline system from 21.6% up to 23.3% whereas co-training leads to a lower average precision of 17.7%. In case the whole labeled training set was used, an average precision of 24% was achieved.

Wu et al. [180] state that the appearance of concepts changes over time and address the problem of concept drifting in videos. They use Gaussian Mixture Models (GMM) to model a concept and propose an incremental online learning framework to cope with concept drifting. For this purpose, videos are processed in a batch mode. The first batch of pre-labeled data is used to learn a global GMM for each concept. The next batch of data (five videos are considered as one batch in their experiments) is employed to learn a set of locally optimized GMMs for each concept from the first unlabeled portion of the new data, aiming at an optimal classification performance on the current test batch. At first, the local models are used to classify the test data. Then, they are used to update the global GMMs. This online process is repeated for each newly upcoming test batch. Their experiments show that locally optimized GMMs outperform the updated global GMMs.

Yan and Hauptmann [183] argue that textual information is the most useful information source for video retrieval but claim that additional audiovisual features can improve a retrieval result. Based on this argumentation, they use textual information to achieve a first ranking and employ other features (audio, visual, motion) to re-rank and refine a retrieval result. However, learning based on a retrieval result must deal with noisy labels because the retrieval result is not perfect. The authors argue that the first k returned documents represent sufficiently positive training samples and consider the remaining set of documents as negative training samples. This training set is used to perform a boosting process that is augmented with audiovisual features, called co-retrieval. The authors report that the boosted re-ranking approach improves the precision compared with the baseline system which used only textual features.

7.3 Semi-Supervised Learning for Semantic Video Retrieval

As discussed above, a common approach to detect semantic concepts in videos is based on a mapping between low-level features and the high-level features. It can be observed for some concepts that their visual appearance is strongly related to a particular video source. For example, weather news are typically related to the following concepts. A map usually presents the area of interest, and some symbols indicate rain, clouds, wind and sun. Furthermore, displayed text gives information about locations, temperatures, all together explained and moderated by a human expert. For a particular instance of weather news, those general elements take a concrete shape, for example the map color, the font type and size of text and the style of the symbols are identical for a certain TV program. In addition, the spatial composition of the shot will be specific for this TV cast, for example the moderator's position and the camera distance will be specific as well.

The proposed approach is aimed at learning the specific appearance of a certain concept in a particular unlabeled test video *v* based on an initial model. To improve the retrieval performance for such concepts, the following concept detection and retrieval framework is suggested to exploit the specific appearance of a concept. Its main processing steps are as follows. First, a baseline system using supervised learning (the SVM data of the MediaMill Challenge system [148] are used in our prototype) is used to map low-level features to high-level concepts. Second, the learned model is used to achieve a first separate ranking for the shots of each test video *v* according to the SVM confidence scores. Third, the best features are selected for this test video *v* and optionally split into two disjoint feature sets. These feature sets are used to train new SVM classifiers using only the automatically labeled data from the current test video *v* under consideration. Then, the additional classifier(s) and the original classifier form an ensemble which is used to re-compute a total confidence score for each shot in the test video *v*.

Finally, the confidence scores for all shots of all test videos *v* are used to rank them, and shots are returned to the user according to this ranking. The algorithmic steps are now explained in more detail (see also the pseudo code in Figure 43). To give a complete picture of the system, the employed low-level features which have been donated by Snoek et al. [148] are discussed also in section 7.3.1.

7.3.1 Learning the Initial Concept Model

In general, any classifier that produces a confidence score regarding a concept's presence in a video shot can be used as the baseline classifier in our system. For ease of implementation, the visual features, SVM models and confidence scores of the MediaMill Challenge system [148] are used in our system, which have been kindly donated by Snoek et al. [148]. These features are aimed at describing a complex scene as a composition of 15 so-called proto-concepts, such as building, car, desert, maps, mountain, road, sky, snow, water etc. Based on a small training set per proto-concept (between 20 and 320 samples), the texture characteristics are captured for each proto-concept. Therefore, the color channels are decorrelated and then an invariant edge detector (equivalent to a Gaussian derivative filter) is applied to each of the newly obtained color channels. The distribution of edges in an image region is then represented by a Weibull distribution which is described by three parameters: μ describes the origin of distribution, β describes the width of the distribution, and γ stands for the peakness of the distribution of edge responses [165]. So far, only models of the proto-concepts are obtained. An image is now divided into a number of regions and each region is described by a set of features that reflect the similarity to each of the pre-defined proto-concepts. A similarity measure C^2 has been suggested by van Gemert et al. [165] to compare two Weibull distributions *F* and *G* reflecting the squared error between them:

$$C^2(F,G) = \sqrt{\frac{\min(\gamma_F,\gamma_G)}{\max(\gamma_F,\gamma_G)}\frac{\min(\beta_F,\beta_G)}{\max(\beta_F,\beta_G)}} \tag{88}$$

```
Input:  Initial SVM model $svm_C$ for concept C; Set of test videos V in the
        database;
Output: List of ranked shots of all test videos;

Algorithm:

Semi-Supervised-Learning-for-Retrieval($svm_C$, V)
  for each test video v ∈ V
      Rank shots(v) according to $svm_C$ confidence score in descending order;
      // generate training samples for this video
      nrPotentialPositiveSamples := |{shot s ∈ shots(v)}| with
      $svm\_confidence_C$(s) >= min_Confidence}|;

      // if not enough training data exist
      if   nrPotentialPositiveSamples < minNrPosShots then
             Normalize $svm_C$ scores for all shots ∈ v to ensemble score and
             store them;


      P:= Select best max_percentage_Pos shots s with $svm\_confidence_C$(s) >=
      min_Confidence as positive training samples;
      N:= Select worst max_percentage_Neg shots s with $svm\_confidence_C$(s) <
      min_Confidence as negative training samples;

      // Get list of ranked features for concept C for this
      // particular video v using a modified Adaboost;
      F=AdaboostFeatureSelection(P,  N);

      Split F in two sets $F_{odd}$ and $F_{even}$;

      Train $newClassifier_1$ with features $F_{odd}$ and training data P and N;
      Train $newClassifier_2$ with features $F_{even}$ and training data P and N;
      for each shot s ∈ shots(v)
             Obtain two confidence scores $confidence1_C(s)$ and $confidence2_C(s)$
             using $newClassifier_1$ and $newClassifier_2$;
             total_score(s)= $a_1$*$svm\_confidence_C$(s)+ $a_2$*$confidence1_C(s)$+
             $a_3$*$confidence2_C(s)$; // $a_i$ are weights
      end for;
  end for;

  return ranking of all shots of all test videos according to total_score(s) in
  descending order;
```

Figure 43: Pseudo code for the semi-supervised retrieval scheme.

Finally, two different histograms are created for each proto-concept to describe an image. The first one, H_{accu} adds the similarity values for all pairs of image regions and annotated training samples, whereas H_{best} is the maximum of this set of similarity values. Hence, H_{accu} captures global information about the image and H_{best} captures local image information. Since both features are extracted for two different Gaussian smoothing filters and two different regions sizes, eight features are obtained for each of the proto-concepts, yielding a total number of 120 features. For further details, the reader is referred to [165].

As mentioned above, these features, SVM models and related confidence scores for 101 concepts were provided by Snoek et al. [148] for the TRECVID 2005 training video set. SVM models are learned based on 70% of this training set, while the remaining 30% are used as the *test* set. Confidence scores are computed for the test set partition for each concept using the SVM models.

7.3.2 Adapting a Concept Model to a Video

In this section, a semi-supervised learning scheme to learn a specific concept model for its appearance in a certain test video *v* and utilize it for the retrieval process is proposed.

7.3.2.1 GENERATING TRAINING DATA FOR A VIDEO

To adapt a model to its appearance in a particular test video *v*, the baseline SVM model is used to rank the shots in this video according to their probability containing the concept. Then, the *p* shots with the highest probability and the *n* shots with the lowest probability serve as positive and negative samples for the subsequent adaptive learning process. The positive samples must exceed a minimum SVM confidence score *min_Confidence* to be considered for the subsequent training process. In case that the number of positive samples is below another threshold *minNrPosShots* defining a minimum number of positive samples, the semi-supervised learning process is not applied for this test video *v*. In this case, the initial scores are normalized with respect to the final number of ensemble classifiers. In addition, two further thresholds define the maximum percentage of shots in a video which are used as positive and negative training samples, respectively. The reader is referred to the pseudo code for these algorithmic steps in Figure 43.

7.3.2.2 SELECTING FEATURES

Once the positive and negative training samples have been obtained automatically for a test video v, feature selection is conducted using a slightly modified version of the Adaboost procedure as described in Chapter 3.3.1. Adaboost is a meta-classifier that provides an ensemble decision rule that is a weighted sum of the n different classifiers. It was shown by Freund and Schapire [55] that Adaboost minimizes the error on the training data as the number of training rounds (and hence the number of classification models) increases. A very nice property of Adaboost is that this is guaranteed as long as each selected classification model achieves an error rate below 0.5, thus, Adaboost is able to build a classification model (a "strong classifier") with an arbitrarily low error rate on the training data based on the estimated combination of possibly "weak classifiers".

As stated above, Adaboost combines a number of *n* (possibly weak) classifiers to build a strong classifier within *n* rounds of training. Therefore, a classification error is estimated for each (weak) classifier by estimating the best threshold that separates positive and negative samples using the one-dimensional data of a single feature. The classification error is calculated based on the current sample weights of misclassified samples. The training samples are weighted equally in the beginning. Training samples which are misclassified by the current classification model are re-weighted such that they have more impact in the next training round for the next "weak classifier". Thus, a newly selected feature has a higher probability to correctly classify those training samples that have been misclassified in preceding rounds.

7.3.2.3 BUILDING AN ENSEMBLE OF CLASSIFIERS

After the selected features have been obtained, they can be used to train any other classifier for the test video v under consideration: in our system, a SVM is used [23]. Another possibility is to build an ensemble of classifiers using majority voting. In this case, the classifiers forming an ensemble should have a certain level of independence. There is evidence [89] that a reasonable degree of independence of ensemble classifiers improves accuracy and guarantees at least the accuracy of the weakest classifier in the ensemble if the classifiers' accuracies exceed a certain value. There is no generally accepted measure for the diversity of classifier ensembles, and it has not been shown theoretically how to successfully build a classifier ensemble, as remarked by Brown et al. [20]. Nevertheless, as stated before, independence between the ensemble members is beneficial. In the proposed approach for self-supervised cut detection (Chapter 4.7), the boosting procedure of Adaboost has been successfully exploited to assign features alternating to two different classifiers according to their odd or even rank in the feature selection process. This approach is motivated by the fact that during the Adaboost process, the weights of the samples which have been misclassified by the preceding weak classifier are increased. Thus, the next selected classifier should be partially independent of its direct predecessor. This property is our motivation to split the feature set depending on the rank of features for subsequent training in order to increase the independence of the obtained feature sets. In case of *n* classifiers, the feature with rank *k* is assigned to classifier *k* modulo *n*. In our proposed system, the feature set is split into two disjoint feature sets where each of them is used in a separate subsequent SVM training process. In this SVM training, concept models are learned based only on the automatically labeled training examples taken from the current test video *v*, yielding two new SVM models. Finally, three SVM models are available for each video, the initial (global) model and two (local) models that are learned with training data that have been labeled automatically for this particular test video *v* using the initial model.

7.3.2.4 RE-RANKING ALL VIDEO SHOTS

At this stage, the question is how to utilize the SVM models, in particular with respect to those models which were trained adaptively for a test video *v*, to achieve a retrieval result for all (test) videos in the database. After the video-specific learning of concept models, three different classifiers are available for each shot and the concept under consideration: the initial model learned on the whole training set and two SVM classifiers which have been learned specifically for a test video *v*. The two newly trained SVM models are now used to obtain additional confidence scores which indicate the probability that a shot exhibits a certain concept. Then, a final confidence score is computed for each shot using the three models (weighted sum of these three scores) which is finally used to rank the shots of all test videos.

7.3.3 EXPERIMENTAL RESULTS

In our experiments, the usefulness of the proposed learning framework has been investigated for several video concepts which were expected to have a specific appearance in a certain video or TV program, respectively. The TRECVID 2005 training set consisting of 137 videos was used for this purpose. As mentioned in the previous section, the MediaMill challenge [148] features, SVM models and related confidence scores are used as our baseline system. In the challenge scenario, the features are extracted from the TRECVID 2005 training video set and SVM models have been learned using 70% of this training set, whereas the remainder is used as the test set. Hence, the proposed system is tested on the remaining 30% as well. The minimum probability that must be assigned to a shot to serve as a positive training sample was set to 0.01, and at least one shot in a test video *v* must have fulfilled this condition. Otherwise, semi-supervised learning is not conducted for this test video *v*. The libSVM [23] is used in our implementation to learn SVM models and to obtain confidence scores. Average precision is used to measure the retrieval performance.

The following high-level concepts were expected to have an appearance related to a specific video: anchor, charts, maps, and overlaid text. Several experiments were conducted for these high-level features. First, the performance of the proposed semi-supervised framework is compared with the MediaMill baseline system. In this scenario, 90 features are selected and this feature set is split to train two additional SVMs for each video.

For this experiment, the results are presented for each concept separately in terms of average precision in Table 71 - Table 74 for several sizes of retrieved shots (documents). The semi-supervised learning scheme improves the average precision for three of those four concepts, except for the concept “overlaid text”. The average precision with respect to 100 retrieved documents is more than 10% higher for the concepts ”maps” and “charts” compared with the baseline system.

In case of the concept "anchor", the top-100 and top-1000 average precision of the baseline is already rather high, so there is only a little improvement. However, average precision increased by about 3% with respect to the top-2000 result. Let us consider the results for the other concepts at the level of top-2000 average precision which is used in the TRECVID evaluation series: for the concept "maps", average precision increased by about 4%, for the concept "anchor" average precision increased by about 3%, and by 4% and 6%, respectively, in case when all shots are considered as retrieved. For the concepts "maps" and "charts" the performance improvement is rather high for the top-100 and top-1000 average precision. This is a useful system property for a user who first wants to browse through these returned shots to access relevant shots more quickly.

Further experiments have been conducted for those concepts for which the retrieval performance could be improved. Five additional learning strategies were investigated: 1.) Only one newly trained model and the related score is used (without the initial model) to finally rank all test video shots in the database. 2.) The initial baseline SVM model and one additional newly learned SVM model is employed, all features are used; 3.) The initial baseline SVM model and one additional newly learned SVM model is employed, 30 features are selected for each video; 4.) As 3.), but 90 features are used; 5.) The proposed semi-supervised learning with two additional SVM models and 90 features. The experimental results are presented for each concept separately in terms of average precision in Table 75 - Table 77. First, it is observable that using only the newly trained model deteriorates average precision significantly. However, the results also show that the proposed semi-supervised learning scheme using two additional SVM models is superior in nearly all cases in terms of the top-100 and top-1000 average precision, while the use of only one additional model sometimes achieves a slightly better result. Since the improvement is clearer for the former results, the semi-supervised framework that uses three classifiers in total seems to be superior for most scenarios. Overall, it has been shown that the proposed semi-supervised learning scheme can learn the specific appearance of certain concepts in a video and improve the retrieval performance. Some example retrieval results for the concepts "maps" and "anchor" are presented in Figure 44, Figure 45, and Figure 46, respectively.

"Maps" Average Precision [%]	Baseline system	Semi-Supervised Learning	Improvement
Top-100	76.0	89.6	+13.6
Top-1000	58.2	69.0	+10.8
Top-2000	52.7	56.6	+3.9
All shots	47.6	51.8	+4.2

Table 71: Experimental results for the high-level feature „maps": Average precision values for the MediaMill challenge baseline system and the proposed semi-supervised approach are presented.

"Anchor" Average Precision [%]	Baseline system	Semi-Supervised Learning	Improvement
Top-100	97.1	98.0	+0.9
Top-1000	83.1	84.8	+1.7
Top-2000	75.3	78.2	+2.9
All shots	63.1	69.4	+6.3

Table 72: Experimental results for the high-level feature „anchor": Average precision values for the MediaMill challenge baseline system and the proposed semi-supervised approach are presented.

"Charts" Average Precision [%]	Baseline system	Semi-Supervised Learning	Improvement
Top-100	60.6	73.4	+12.8
Top-1000	44.4	51.3	+6.9
Top-2000	40.5	42.6	+2.1
All shots	32.7	35.7	+3.0

Table 73: Experimental results for the high-level feature „charts": Average precision values for the MediaMill challenge baseline system and the proposed semi-supervised approach are presented.

"Overlaid Text" Average Precision [%]	Baseline system	Semi-Supervised Learning	Improvement
Top-100	97.5	96.9	-0.6
Top-1000	87.9	85.1	-2.8
Top-2000	83.9	80.4	-3.5
All shots	66.9	62.1	-4.8

Table 74: Experimental results for the high-level feature „overlaid text": Average precision values for the MediaMill challenge baseline system and the proposed semi-supervised approach are presented.

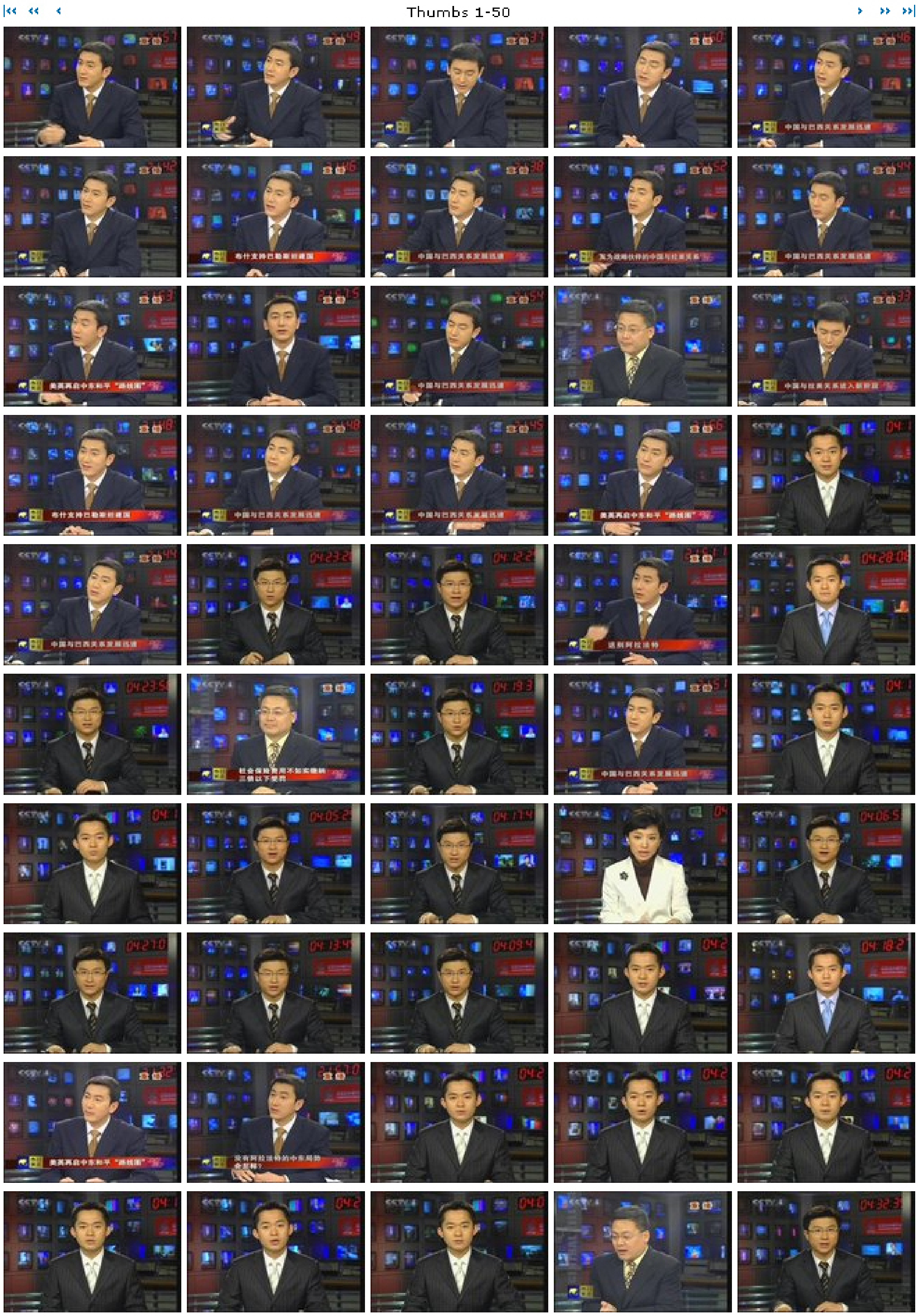


Figure 44: Example retrieval result (top-50) for the concept "anchor" when using the semi-supervised learning ensemble approach.

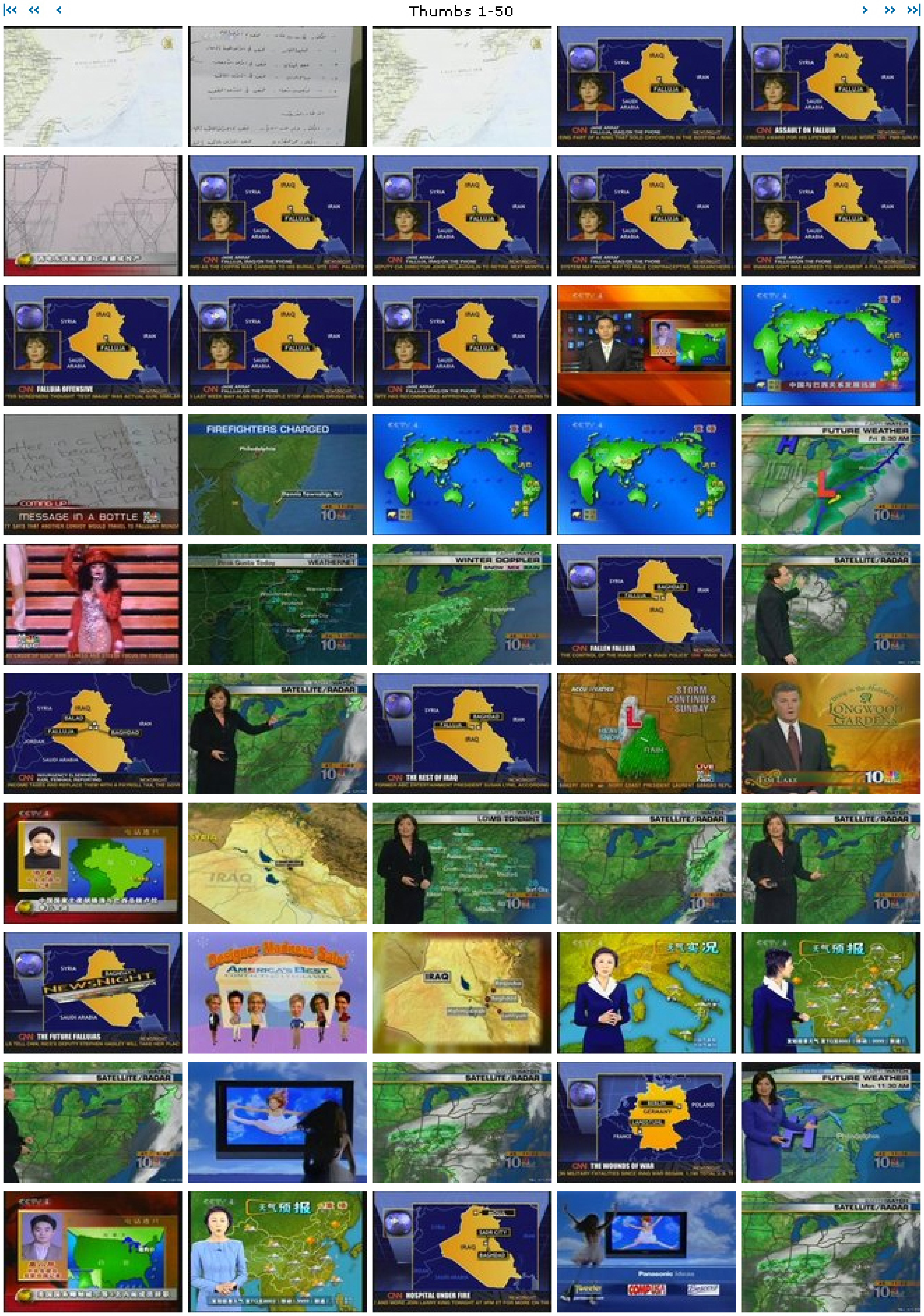


Figure 45: Example retrieval result (top-50) for the concept "maps" when using the Mediamill baseline system.

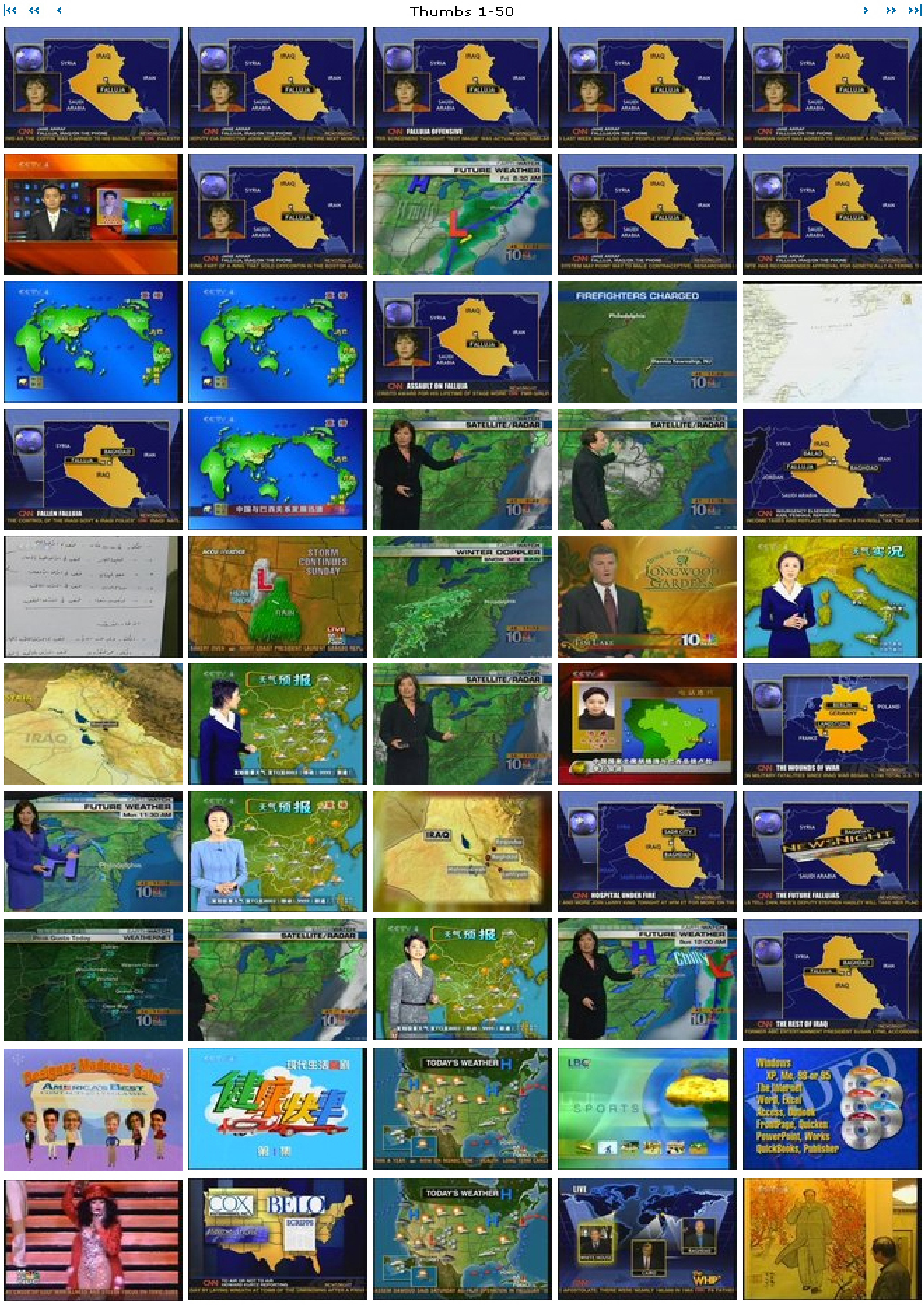


Figure 46: Example retrieval result (top-50) for the concept "maps" when using the semi-supoervised learning ensemble approach. The "non-maps" shots are shifted downwards when using the semi-supervised approach compared to the baseline system.

“Maps” Average Precision [%]	Only new SVM model (30)	Initial model & 1 new SVM model (all)	Initial model & 1 new SVM model (30)	Initial model & 1 new SVM model (90)	Initial model & 2 new SVM models (90)
Top-100	35.4	85.3	86.6	87.8	**89.6**
Top-1000	18.7	64.0	67.3	68.5	**69.0**
Top-2000	15.0	58.9	58.4	**59.7**	56.6
All shots	13.4	52.5	52.0	**52.5**	51.8

Table 75: Experimental results for the high-level feature „maps” for different learning strategies. The number in brackets indicates the number of selected features by Adaboost.

“Anchor” Average Precision [%]	Only new SVM model (30)	Initial model & 1 new SVM model (all)	Initial model & 1 new SVM model (30)	Initial model & 1 new SVM model (90)	Initial model & 2 new SVM models (90)
Top-100	46.9	**98.9**	97.8	97.1	98.0
Top-1000	50.5	83.8	83.7	84.5	**84.8**
Top-2000	47.4	76.6	76.8	75.5	**78.2**
All shots	42.8	68.3	68.1	66.8	**69.4**

Table 76: Experimental results for the high-level feature „anchor” for different learning strategies. The number in brackets indicates the number of selected features by Adaboost.

“Charts” Average Precision [%]	Only new SVM model (30)	Initial model & 1 new SVM model (all)	Initial model & 1 new SVM model (30)	Initial model & 1 new SVM model (90)	Initial model & 2 new SVM models (90)
Top-100	11.5	63.1	70.1	68.8	**73.4**
Top-1000	8.3	48.1	50.5	49.7	**51.3**
Top-2000	7.6	41.7	**45.1**	43.9	42.6
All shots	6.4	34.9	36.5	**36.7**	35.7

Table 77: Experimental results for the high-level feature „charts” for different learning strategies. The number in brackets indicates the number of selected features by Adaboost.

7.4 TRANSDUCTIVE SVM FOR SEMANTIC VIDEO RETRIEVAL

Semi-supervised learning comprises the class of learning methods that do not only use labeled training data but in addition incorporate unlabeled data in the learning process in order to improve class models [136]. Transductive learning can be considered as a special case of semi-supervised learning: In this setting, the labels of all training samples are known, and the set of unlabeled data consists only of the test samples. So far, this is the usual setting of supervised learning. But in contrast to traditional supervised learning methods, a transductive learner incorporates the unlabeled test samples into the learning process to obtain a model that separates the test data appropriately into the corresponding classes. This yields another difference compared to inductive learners: An inductive learner produces a generalized model that aims to classify all possible class instances in an optimal way, whereas a transductive learner produces a model that is aimed at being optimal only for the test data. In a way, this is according to Vapnik's principle [166] targeted at high-dimensional estimation problems: "When trying to solve some problem, one should not solve a more difficult problem as an intermediate step".

Why should transductive learning be advantageous for the task of high-level concept detection in videos? Often, the meaning or the appearance of a certain event or concept is strongly determined by contextual information. For example, the appearance of certain high-level concepts, such as maps or news anchors in a news program, is strongly related to the editing layout which might be specific for a broadcasting station. In the previous chapter, it has been shown that a semi-supervised learning scheme can improve the retrieval performance for high-level concepts, although the performance of the available baseline classifiers (based on supervised learning) is relatively low.

Up to now, transductive learning has not been applied for the task of high-level concept detection and video retrieval. In this chapter, a novel transductive concept detection system for the purpose of semantic video retrieval is proposed. The following scenario is assumed: A training set of videos is given whose shots are labeled with respect to the appearance of concept *c*, and a test set of videos is given whose shots are not labeled with respect to the appearance of concept *c*. First, a baseline model is learned using the complete training set via traditional supervised learning (inductive support vector machine). Then, for each video *v* in the test set, this model of concept c is adapted with respect to its appearance in video *v* via transductive learning. In our system, transductive support vector machines [82] are employed to achieve this model adaptation. In this way, prediction values are obtained based on the corresponding transductive model for all shots of a particular video in the test set. These values indicate whether the concept *c* is present in them or not. Finally, the predictions for all test video shots in the database are used to rank them according to the probability that the concept *c* is present in them. Experimental results on TRECVID 2005 video data demonstrate the feasibility of the proposed transductive learning system for several high-level concepts.

As discussed above, a common approach to detect semantic concepts in videos is based on a mapping between low-level features and the high-level features. It can be observed for some concepts that their visual appearance is strongly related to a particular video source, as explained in Chapter 7.1.

Considering the related work of semi-supervised learning for semantic video retrieval, it is evidaent that the video-specific appearance of certain concepts has not been exploited before to improve the concept detection and retrieval performance (except our previous proposal). Furthermore, transductive learning has not been applied to concept detection and video retrieval. In this section, a novel approach to learn the specific appearance of a certain concept in a particular video using transductive support vector machines (TSVM) is described.

The following scenario is assumed (see also flow diagram in Figure 47). A set of labeled training videos is available and a set of unlabeled test videos must be classified and indexed with respect to the occurrence of a semantic concept c. Each shot in a video is desribed by a number of visual features. The feature representation used in our system is described in Chapter 7.3.1. Furthermore, for each shot of a training video, the class label for concept *c* is known, whereas all shots of the test videos are not labeled. Now, a transductive setting can be defined which allows us to exploit the specific appearance of a concept in a particular test video *v* by adapting a class model of the semantic concept *c* to test video *v*. For this purpose, each test video *v* is considered separately, and the unlabeled feature vectors of *v* are incorporated in the learning process. This process is aimed at computing a model that separates the classes (a shot belongs to concept c or not) in an optimal way for this test video. As a result, a class model is computed for each test video *v* separately and, based on this model, prediction values are computed for each shot of this video which indicate whether the concept is present in a shot or not. Finally, the prediction values for all shots of all test videos *v* are used to rank the shots, and shots are returned to the user according to this ranking.

7.4.1 Applying Transductive SVMs for Video Indexing and Retrieval

The main idea in this step is to employ transductive learning to adaptively refine the appearance model of a high-level concept with respect to a particular video. The following scenario is assumed: A set of m training video shots is given where all shots are labeled according to the appearance of concept *c* (training set). This complete training set is utilized to learn a base model via an inductive

SVM. Then, for all *k* unlabeled shots of a new test video v in the database (test data), a transductive SVM is used to adapt and refine the SVM base model for concept c with respect to its appearance in a particular test video *v* via transductive learning. Formally, the shots of the training videos are represented by the feature vectors x_i with labels y_i ($i \in \{1, ..., m\}$) according to concept *c*, while the *k* shots of a particular test video *v* yield the unlabeled test data for each transductive setting, represented by feature vectors x_j^* and unknown labels y_j^* ($j \in \{1, ..., k\}$). Given *t* test videos in the database, the transductive SVM is applied to each test video *v* separately in order to refine the concept model for this video. In this way, based on the particular transductive model TSVM(*c*, *v*), a prediction value is obtained for each of the *k* shots in a test video *v* indicating whether the concept *c* is present in it or not. Finally, assuming a user query in a retrieval scenario, the predictions for all test shots in the database are used to rank them according to the probability that the concept c is present in them.

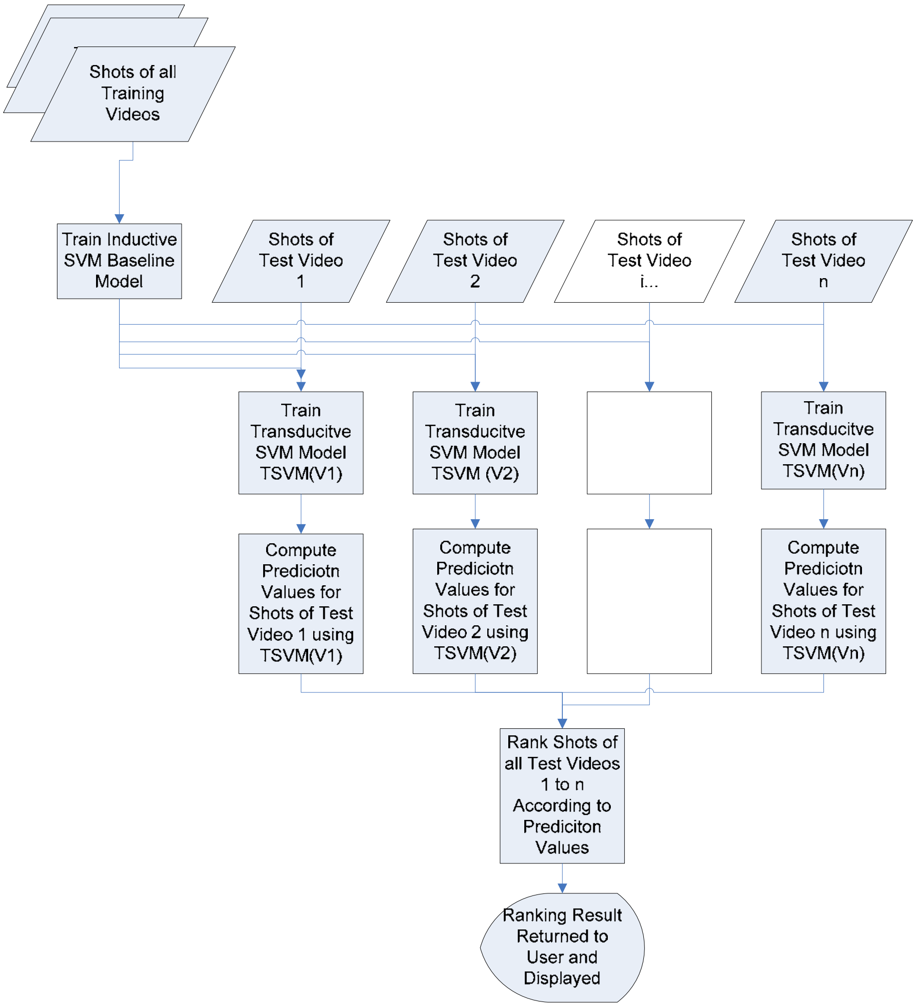

Shots of all Training Videos
Train Inductive SVM Baseline Model
Shots of Test Video 1
Shots of Test Video 2
Shots of Test Video i...
Shots of Test Video n
Train Transducitve SVM Model TSVM(V1)
Train Transducitve SVM Model TSVM (V2)
Train Transducitve SVM Model TSVM(Vn)
Compute Prediciotn Values for Shots of Test Video 1 using TSVM(V1)
Compute Prediciotn Values for Shots of Test Video 2 using TSVM(V2)
Compute Prediciotn Values for Shots of Test Video n using TSVM(Vn)
Rank Shots of all Test Videos 1 to n According to Prediciton Values
Ranking Result Returned to User and Displayed

Figure 47: Scheme for semantic video indexing and retrieval based on transductive SVM learning.

7.4.2 EXPERIMENTAL RESULTS

In our experiments, the usefulness of the proposed transductive learning approach has been investigated for several video concepts which were expected to have a specific appearance in a certain video or TV program, respectively. The performance of the transductive SVM is compared with the previously proposed system in Chapter 7.3, which employs a semi-supervised ensemble of classifiers. The TRECVID 2005 video training set consisting of 137 videos was used for this purpose. As mentioned in the previous section, the MediaMill challenge [148] features are used in our baseline system. In the challenge scenario, the features are extracted from the TRECVID 2005 training video set and SVM models have been learned using 70% of this training set, whereas the remainder is used as the test set. The proposed system is tested on the remaining 30% as well.

The SVM implementation svmLight <http://svmlight.joachims.org> is used to compute transductive SVM models. Not only the SVM models provided by Snoek et al. are used as a baseline because it should be avoided that different performance measurements are caused only by different implementations or parameter settings of the SVMs. To achieve this, the inductive SVM of "svmLight" is also used to compute the prediction values of our baseline system. Thus, it is guaranteed that different performance measures are caused only by the transductive extension of the SVM but by a particular SVM implementation.

The performance of our baseline SVM achieves at least the performance of the MediaMill challenge SVM models for the investigated concepts. It could be observed that performance of transductive SVM degrades noticeably when the percentage of positive samples is below 2%. The performance degradation could be avoided when the number of samples, which are labeled by the inductive baseline SVM as positive, is set to 0. In this way, no falsely assigned positive labels deteriorated the computation of the hyperplane, while the unlabeled negative samples still influenced the maximum margin hyperplane. In this way, best results could be achieved for such concepts. This had to be done for the concepts "charts" and "maps". In all SVM experiments, a RBF kernel was used. Average precision is used to measure the retrieval performance.

"Anchor" Average Precision [%]	Baseline system	Transductive SVM	Improvement
Top-100	98.9	93.8	-5.1
Top-1000	78.0	81.6	+3.6
Top-2000	71.8	76.9	+5.1
All shots	61.8	66.6	+4.8

Table 78: Experimental results for the high-level feature „anchor": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented.

"Charts" Average Precision [%]	Baseline system	Transductive SVM	Improvement
Top-100	70.2	84.2	+14.0
Top-1000	55.7	64.5	+8.8
Top-2000	52.3	60.6	+8.3
All shots	36.8	42.5	+5.7

Table 79: Experimental results for the high-level feature „charts": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented.

"Entertainment" Average Precision [%]	Baseline system	Transductive SVM	Improvement
Top-100	45.5	73.3	+27.8
Top-1000	46.8	63.4	+16.6
Top-2000	42.2	54.9	+12.7
All shots	29.0	32.7	+3.7

Table 80: Experimental results for the high-level feature „entertainment": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented.

"Maps" Average Precision [%]	Baseline system	Transductive SVM	Improvement
Top-100	83.1	85.9	+2.8
Top-1000	66.0	68.0	+2.0
Top-2000	61.6	61.6	0.0
All shots	51.9	51.5	-0.4

Table 81: Experimental results for the high-level feature „maps": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented.

The following high-level concepts were expected to have an appearance related to a specific video and hence, they are investigated in our experiments: anchor, charts, entertainment, maps, overlaid text, and studio. Several experiments were conducted for these high-level features. First, the performance of the proposed transductive SVM approach framework is compared with the baseline SVM system for the six selected high-level concepts. For this experiment, the results are presented for each concept separately in terms of average precision in Table 78 - Table 83 for several sizes of retrieved shots (documents).

"Overlaid Text" Average Precision [%]	Baseline system	Transductive SVM	Improvement
Top-100	89.5	88.6	-0.9
Top-1000	82.5	82.5	0.0
Top-2000	79.8	80.0	+0.2
All shots	65.7	65.8	+0.1

Table 82: Experimental results for the high-level feature „overlaid text": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented.

"Studio" Average Precision [%]	Baseline system	Transductive SVM	Improvement
Top-100	95.7	97.3	+1.6
Top-1000	87.3	90.2	+2.9
Top-2000	79.9	83.8	+3.9
All shots	61.8	65.8	+4.0

Table 83: Experimental results for the high-level feature „studio": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented.

"All Concepts" Average Precision [%]	Baseline system	Transductive SVM	Improvement
Top-100	80.5	87.2	+6.7
Top-1000	69.4	75.0	+5.6
Top-2000	64.6	69.6	+5.0
All shots	51.2	54.2	+3.0

Table 84: Experimental results for all six high-level features which were expected to benefit from transductive learning. The average of all average precision values is presented for the baseline SVM system and the proposed transductive SVM approach.

The transductive SVM improves the average precision for four of six concepts ("anchor", "charts", "entertainment" and "studio"). The average precision with respect to 100 retrieved documents is more than 10% higher for the concepts "charts" and "entertainment" compared with the baseline system. Let us consider the results for the other concepts at the level of top-2000 average precision which is used in the TRECVID evaluation series: The average precision for the concepts "anchor", "charts", "entertainment" and "studio" is improved noticeably, between 3.9% and 12.7 %, in terms of average precision regarding top-2000 shots (and by 7.5% on the average). Considering all six concepts, the results are improved on the average by 5% for top-2000 average precision (see Table 84). In case of the concept "anchor", the top-100 average precision is below that of the baseline approach but the results for the other shot sizes are improved noticeably, up to 5%. Some example retrieval results using the baseline SVM and the transductive SVM are presented in Figure 48 and Figure 50 for the concept "anchor", whereas example retrieval results for the concept "charts" are presented in Figure 50 and Figure 51.

"All concepts" Top-2000 Average Precision [%]	Baseline Challenge [148]	Semi-Sup. Approach	Baseline svmLight	TSVM
Anchor	75.3	78.2	71.8	76.9
Charts	40.5	42.6	52.3	60.6
Entertainment	19.1	15.4	42.2	54.9
Map	52.7	58.6	61.6	61.6
Studio	85.3	81.0	79.9	83.8
Average	54.6	55.2	61.6	67.6

Table 85: Comparison of the MediaMill challenge baseline system [148], our baseline SVM (svmLight), the semi-supervised learning approach presented in Chapter 7.3, and the proposed transductive SVM approach. Only those concepts are considered for which either the semi-supervised approach or the transductive approach yielded an improvement.

Furthermore, the performance of five high-level concepts of the transductive learning approach has been compared with our previously proposed semi-supervised learning approach. Only those concepts are considered for which at least one of both methods yielded better results. The results are displayed in Table 85. Transductive learning clearly outperforms our previous approach for four concepts and achieves a similar performance for the remaining one. Thus, it is concluded that the transductive learning is very well suited for the given task.

7.5 SUMMARY

In this chapter, two transductive learning methods for video content analysis with respect to high-level feature (concept) detection have been proposed. The first proposal is realized by a semi-supervised learning ensemble, whereas the second method employs transductive SVM. Both methods exploit the fact that there are concepts whose appearance or layout is strongly related to a certain video source or TV program (e.g., maps in a news cast). Experimental results on the TRECVID 2005 training set have demonstrated the feasibility of the proposed approaches for certain concepts which are strongly related to a video source or TV program. The approach using transductive SVM achieved the best results and worked for more concepts as the approach based on the semi-supervised learning ensemble.

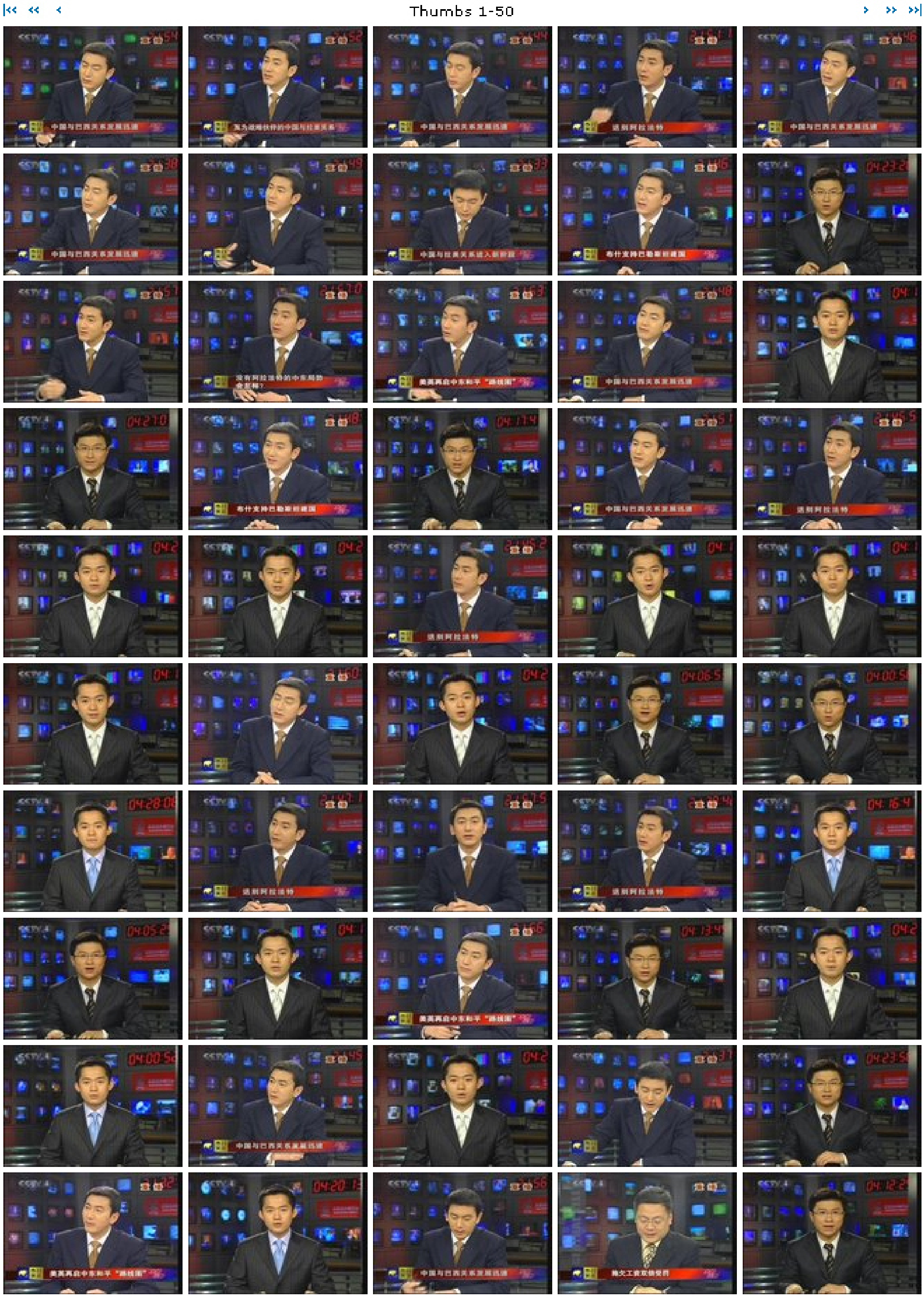


Figure 48: Example retrieval result (top-50) for the concept “anchor” when using the baseline SVM.

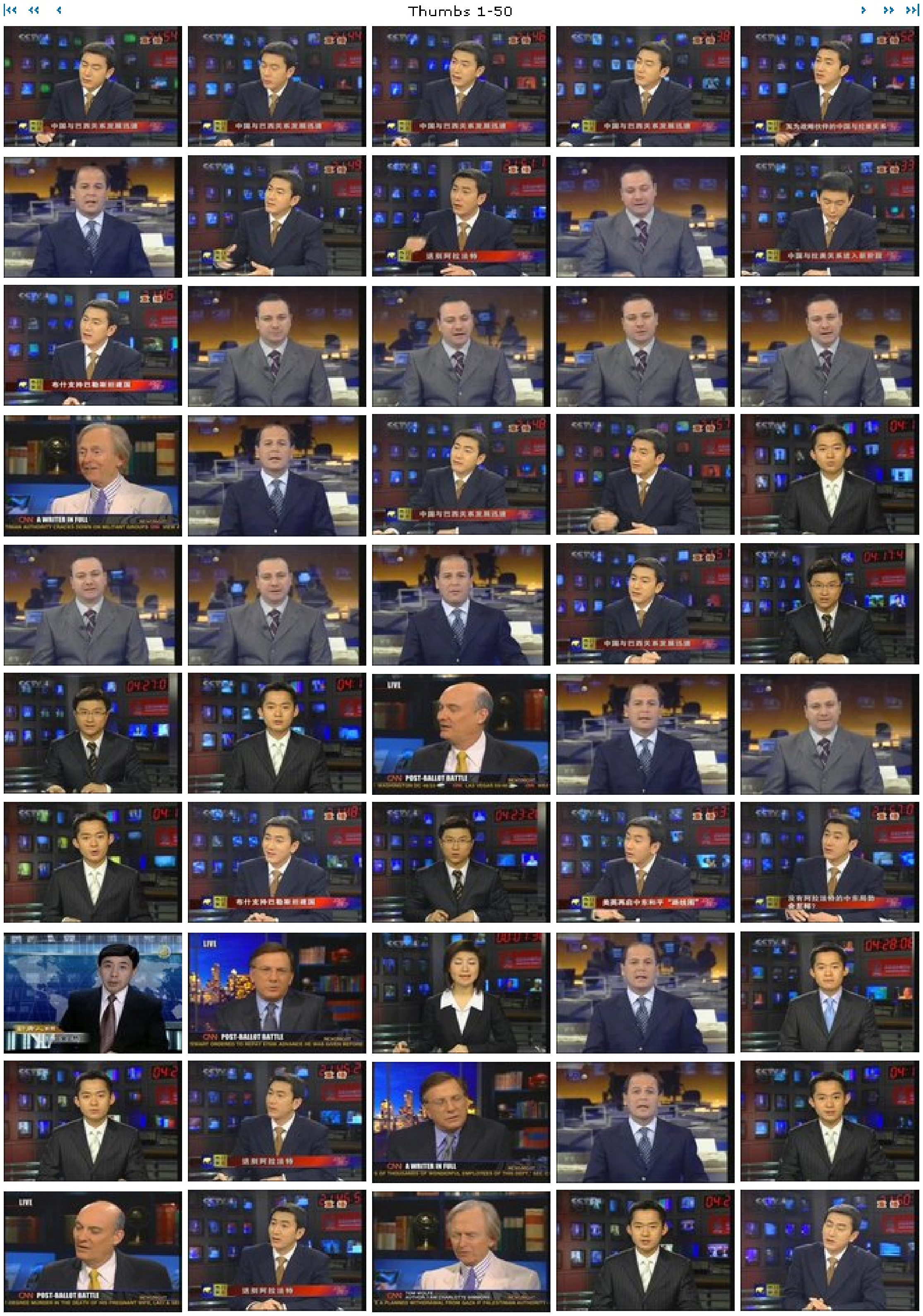


Figure 49: Example retrieval result (top-50) for the concept "anchor" when using the transductive SVM. Interestingly, the retrieved top-50 shots using the transductive SVM are from several different programs. When using the baseline SVM, all retrieved shots are from the same program.

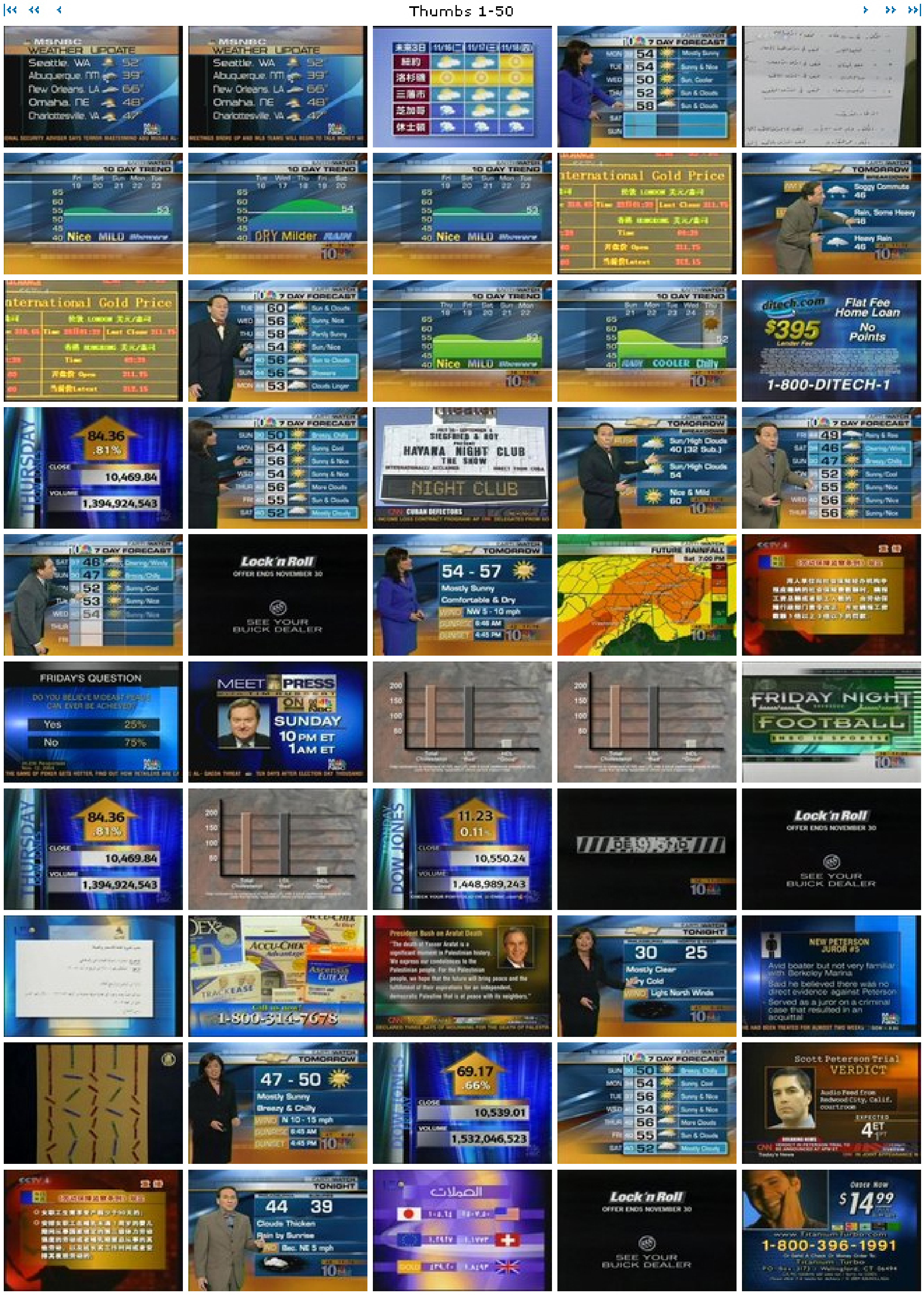


Figure 50: Example retrieval result (top-50) for the concept "charts" when using the SVM baseline system.

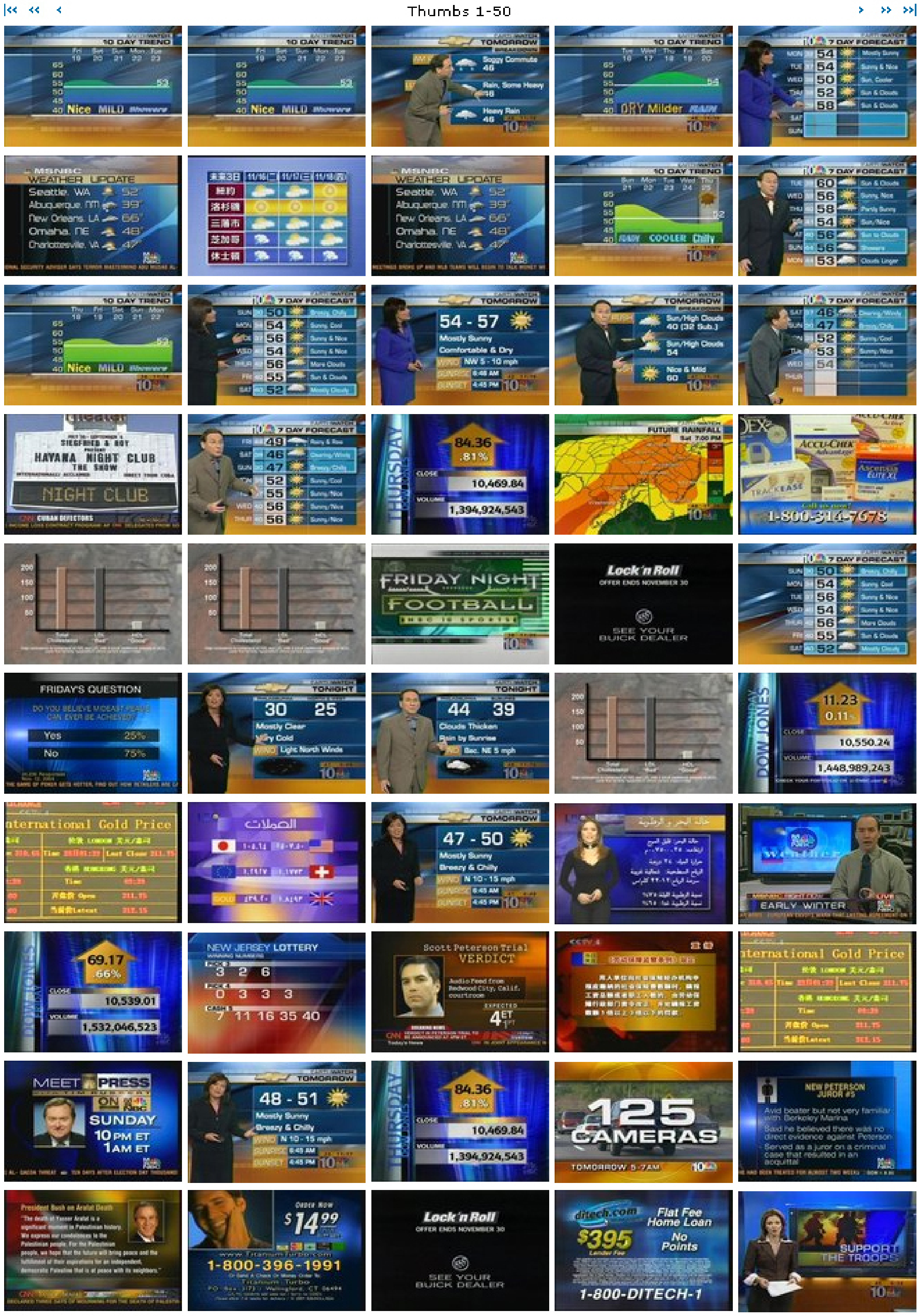


Figure 51: Example retrieval result (top-50) for the concept "charts" when using the transductive SVM.

Chapter Eight

8 Semantic Video Analysis for Psychological Research by Semi-Supervised Learning

8.1 Introduction

Computer games play a very important role in today's entertainment media and belong to the most popular entertainment products. Unfortunately, the number of computer games containing serious violence increases. There is an extensive ongoing debate about the question whether playing violent games causes aggressive cognitions, aggressive affects or aggressive behavior, in particular with respect to teens and young adults.

The neurophysiologic perspective of mass communication research concentrates on emotional responses to video game playing. Mathiak and Weber [110] developed neurophysiologically grounded measures for the "human experience of media enjoyment". The study continues their prior work (Weber et al. [172]) on video game playing in which functional magnetic resonance imaging (fMRI) scans were taken during video game playing. Through this neurophysiologic perspective, they demonstrated that a specific neurological mechanism is activated when playing a first-person-shooter game. One central finding is that cognitive areas seem to suppress affective areas during the (virtually) violent interactions. This mechanism helps to better understand a potential link between playing certain types of violent video games and aggressive cognitions and affects.

The experimental design presented by Weber et al. [172] is based on the definition of certain game states and captures a player's brain activity via fMRI while he (only male players were investigated) is playing a violent computer game. Several semantic game events are distinguished: 1.) inactive; 2.) preparation; 3.) search and explore; 4.) danger; 5.) under attack, and 6.) fighting and killing. Once the game recordings are annotated with these semantic categories, the interrelationship of violent game events and the underlying neurophysiologic basis (brain activity) of the player can be investigated. Normally, human annotators are required to index such game content according to the current game state, but this is a very time-consuming task. In this context, computer-based automatic video content analysis of computer game recordings promises several advantages: Human annotation efforts can be reduced significantly, and the annotation process is speeded up and is based on reproducible and objective criteria only. At the same time, researchers are enabled to investigate a larger number of computer game videos to gather more experimental data.

In this chapter, an automatic semantic video analysis system is presented that supports the experimental design described above by automatically identifying the game states (i.e., categories). The system is aimed at minimizing the human annotation effort and thus requires only manual annotations for a single video. The task is also considered as a transductive learning setting and, according to the transductive learning ensemble framework presented in Chapter 3, a solution based on semi-supervised learning is investigated as well. Content analysis relies on audiovisual low-level features as well as on mid-level features. The considered mid-level features are the results of shot boundary detection (as proposed in Chapter 4.5), camera motion estimation (as proposed in Chapter 5), audio segmentation, text detection [62] and face detection [77]. For each game category, a support vector machine (SVM) is trained using the low- and mid-level features. In our approach, only a single video sequence with a duration of 12 minutes is required to provide training data and hence, human annotation effort is kept at a minimum. Afterwards, new videos are automatically analyzed using these SVM models. To achieve a more robust result, an automatic semi-supervised correction step is employed separately for each video: Based on the initial classification result, the system automatically labels the frames in a new video and adapts its concept models to this video by employing feature selection and adaptively building a specialized classifier for a particular game video. Finally, the graphical user interface of our software system Videana allows a human expert to refine or to correct the annotation results, if needed. Experimental results demonstrate the very good performance of the proposed approach, which is thus indeed applicable to this interdisciplinary research field. The work presented in this chapter has been partially published in [116].

8.2 RELATED WORK

To the best of our knowledge, neither video content analysis methods have been applied to computer game recordings nor automatic video content analysis has been suggested for the field of behavioral sciences. Nevertheless, there exist many semantic video analysis systems which are specialized for a certain genre, for example sports videos or news videos.

There are many approaches addressing the analysis of news videos. This emphasis might have been enforced by the TRECVID evaluation series [http://www-nlpir.nist.gov/projects/t01v/] in which comprehensive news video test collections have been provided and used for evaluation purposes. A summary of semantic concept detection approaches regarding news videos is presented by Naphade and Smith [117]. The authors state that in most approaches, concept detection is considered as a supervised pattern recognition problem.

In a way, sports videos can be considered as somewhat related to the genre of computer games investigated in this paper: Since both genres are rule-driven, the amount of possibly appearing content is limited in both sports and computer games ("e-sports"). The automatic indexing of sports videos has been extensively studied in recent years. As noted by Sadlier and O'Connor [133], many specific approaches exist for several sports domains such as Formula-1, cricket, tennis, American football, and Gaelic football.

Apart from specific approaches, frameworks have been proposed that cover more than only a single type of sports. For example, Xu and Chua [182] propose a framework for event detection in team sports videos that is based on audiovisual features, domain knowledge, and external information sources.

Tong et al. [159] suggest a framework for semantic shot representation of sports videos. This framework is applicable to field sports, and shots are classified based on camera distance, displayed subject and edited video layout.

Sadlier and O'Connor [133] present an event detection system for field sports as well. They argue that it is not feasible to build a generic supervised event detection system for any kind of sports and find the limitation to field sports reasonable. The following features are employed in a supervised learning process: image crowd detection, speech-band audio activity, on-screen graphics tracking, motion activity measure, field line orientation and some other features.

8.3 Semantic Analysis of Computer Games by Semi-Supervised Learning

In this section, a system to support interdisciplinary research in media and behavioral sciences via automatic multimodal video content analysis is presented. First, in section 8.3.1 the semantic classes are described which must be recognized for the experiment conducted by Weber et al. [172]. Then, the proposed system is presented in sections 8.3.2 to 8.3.4. It utilizes automatically extracted audiovisual low-level and mid-level features to infer about the semantic game classes via supervised learning or semi-supervised learning, respectively. Two main targets have been pursued. First, the system is supposed to remain a generic video content indexing system and thus does not contain any specific content detectors (restricting its applicability to a certain computer game would offer a lot of tuning possibilities). Second, the annotation effort that is needed to apply a machine learning approach should be kept at minimum, that is only a single labeled video is used for training. The following parts of our system are discussed in more detail in sections 8.3.2 – 8.3.4: audiovisual feature extraction, feature selection, classification, and the semi-supervised classification approach.

8.3.1 Semantic Classes for the Computer Game Experiment

Participants of the experiment conducted by Weber et al. [172] played the "mature" rated first-person-shooter game "Tactical Ops: Assault on Terror" [http://www.tactical-ops.de/]. As mentioned above, the experiment was aimed at gaining insight into the interrelationship of playing violent computer games and changes in the consumer's brain activities. Therefore, several game states were defined, and the dependence of the players' brain activity is set in relation to these game states. Brain activity was measured via fMRI scans. In this section, a system that is able to classify the following semantic classes with an acceptably high accuracy is presented:

1.) "inactive": The player's avatar (PA) is dead or the game has not started yet.

2.) "preparation": The PA is buying equipment in the beginning of a new round.

3.) "search/explore/danger": The PA explores the virtual world and searches for hostages, enemies and weapons.

4.) "violence": The PA is fighting and/or injured.

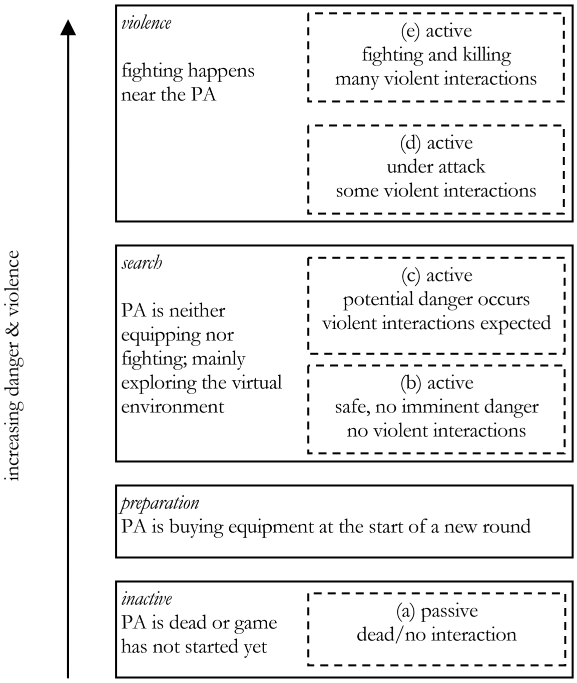

increasing danger & violence
violence
fighting happens near the PA
(e) active
fighting and killing
many violent interactions
(d) active
under attack
some violent interactions
search
PA is neither equipping nor fighting; mainly exploring the virtual environment
(c) active
potential danger occurs
violent interactions expected
(b) active
safe, no imminent danger
no violent interactions
preparation
PA is buying equipment at the start of a new round
inactive
PA is dead or game has not started yet
(a) passive
dead/no interaction

Figure 52: The four boxes explain the different semantic game classes used in this study and how they relate to the categories used by Mathiak and Weber [110], which are displayed in the dashed nested boxes. The classes are ordered from bottom to top in terms of increasing violent content, where PA stands for "player's avatar".

In the original study, the semantic game categories were distinguished and annotated more sophistically (see Figure 52). Category 3 was further divided into "search" and "potential danger", and for category "violence" it is distinguished whether the PA is injured/attacked or fighting actively. However, automatic distinction of these semantic classes would not be feasible without neglecting the target to have a generic video content analysis system. For example, consider the highly abstract semantics regarding the distinction of "search" and "danger". When the PA currently is in the state "search" (no imminent danger) and spots another character, its state switches to (potential) "danger". Now, according to whether this character is identified as an enemy or not, the state switches to "violence", because the PA shoots at the enemy, or back to "search" when the appearing character is harmless. Normally, state "danger" endures only for a few seconds before the states evolve further in the mentioned manner. Furthermore, the appearance of new characters in the PA's field of view often takes place near the horizon, where avatars are only a few pixels in size, and it is extremely difficult to perform the necessary friend-or-foe identification with a reasonable precision. Furthermore, our system does not distinguish between "active" and "passive" violence. In practice, "passive" violence is a very short segment before either "active" violence or "inactive" (player's avatar is dead) take place. This is the reason for the definition of the four classes described above: In this way, an automatic and generic annotation system is feasible and the remaining manual revisions are minimized. Figure 53 shows example frames for each of the four semantic game categories.

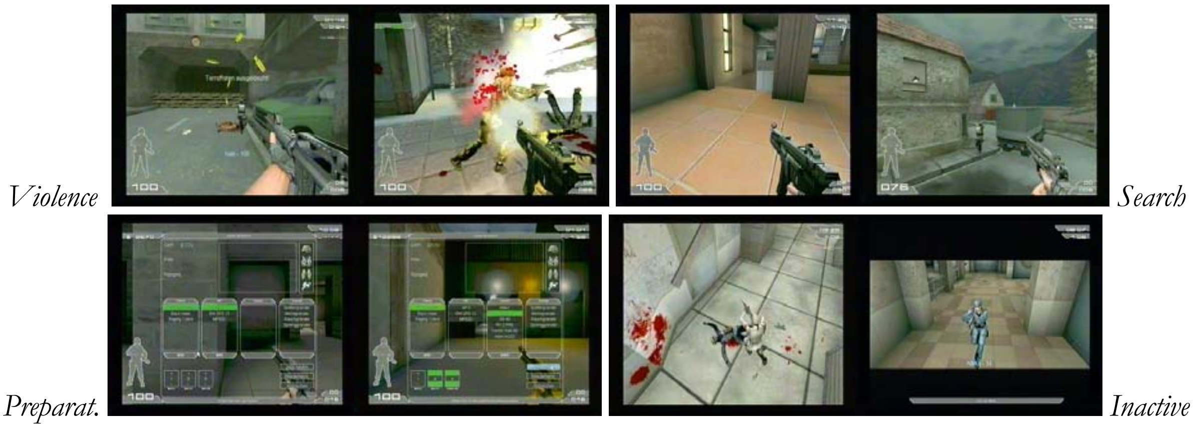


Figure 53: Example frames of the four different game classes.

8.3.2 Extraction of Audio Features

The semantic content of computer games is present in all modalities of their recordings: fighting and killing, for example, is visible in the video domain by the presence of enemies, muzzle flash and blood; it is also audible in the accompanying soundtrack by means of shots or explosive sounds as well as moans. The automatic content analysis system extracts a number of general audio low-level features which support the recognition of the semantic classes. The following features are extracted from non-overlapping 25ms frames [102] and are fed directly into the annotation system:

1. Eighth-order Mel Frequency Cepstrum (MFC) Coefficients: Capturing the broad envelope of the spectrum;

2. Zero Crossing Rate: A measure of oscillation and intra-frame variation;

3. Short Time Energy: Corresponding with loudness;

4. Sub-band Energy Distribution: Loudness ratio for four successive frequency bands;

5. Brightness and Bandwidth: The spectrum's frequency centroid and spread;

6. Spectrum Flux: Inter-frame spectral variation;

7. Band Periodicity: Periodicity of the four subbands;

8. Noise Frame: Noisiness corresponding to lack of periodicity.

Additionally, these features are fed into a content-based audio classification and segmentation system based on the approach of Lu et al. [102]. The system produces mid-level features on a per-second (sub-clip) basis in the form of acoustic class labels and related probabilities for silence (SIL), pure-/non-pure speech (PSp/NpSp), music, background (BG) and action sounds (ACTN). The low-level features are therefore aggregated per second, normalized and then concatenated to form one feature vector per sub-clip, which is processed by a hierarchical tree of SVMs, if it was not previously classified as silence by a threshold based classifier. Figure 54 shows this classification tree. It is trained on more than 32 hours of audio – TIMIT [100] data for clean speech, NOIZEUS [75] and broadcast speech data for non-pure speech, pop and instrumental music, various movie sound samples from broadcast material, and free web resources for the different types of noise. Five-fold cross-validation on a subset of 15000 feature vectors has been used to find the best parameter settings for a one-class SVM with RBF (radial basis function) kernel via libSVM [23]. The final acoustic class labels and their corresponding probabilities are fed into the game state learning algorithm as mid-level features to further guide the discovery of semantic patterns.

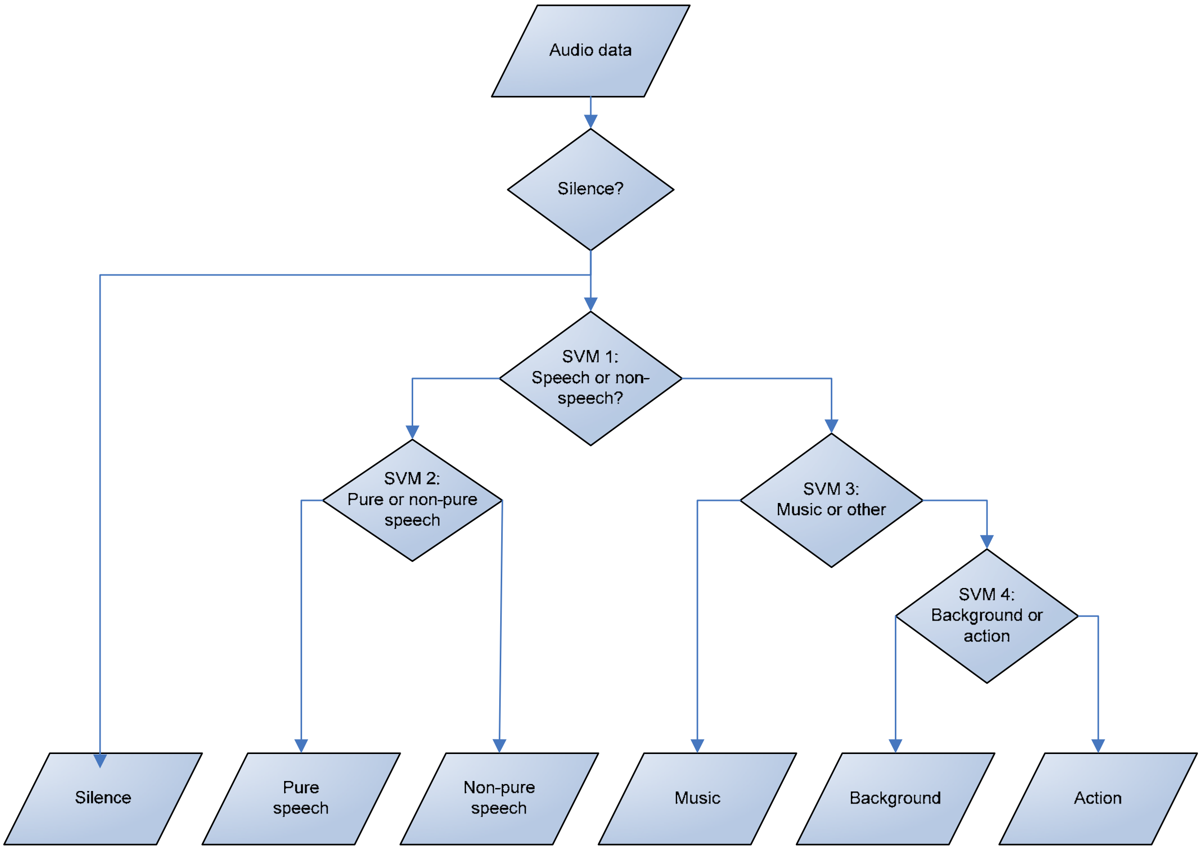


Figure 54: Scheme of the hierarchical audio type classifier: A single feature vector per sub-clip serves as input; output is a single acoustic class label and its corresponding probability.

8.3.3 Extraction of Visual Features

Several visual features are extracted for each video frame. In addition to low-level features as color moments and texture features, several mid-level features are extracted automatically by utilizing camera motion estimation (as proposed in Chapter 5 [39, 42]), face detection [77] and text detection [62]. In the following, the extracted features are briefly described:

- Color moments: Color moments are extracted at two different granularities. The first three global color moments are computed for the whole image. Corresponding values are extracted for each region of a 3 x 3 grid in HSV (Hue, Saturation, Value) color space. The *i*-th pixel of the *j*-th color channel of an image region is represented by c_{ij}. Then, the first three color moments are defined as:

$$mean_j = \frac{1}{N} \cdot \sum_{i=0}^{N-1} c_{ij} \qquad (89)$$

$$stdev_j = \sqrt{\frac{1}{N} \cdot \sum_{i=0}^{N-1} (c_{ij} - mean_j)^2} \tag{90}$$

$$skew_j = \sqrt[3]{\frac{1}{N} \cdot \sum_{i=0}^{N-1} (c_{ij} - mean_j)^3} \tag{91}$$

- Texture features: The gray-scale image co-occurrence matrices mk are constructed at 8 orientations. The following matrices are used to extract the following values representing the global texture:

$$energy_k = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (m_{kij})^2 \tag{92}$$

$$contrast_k = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} (i-j)^2 \cdot m_{kij} \tag{93}$$

$$entropy_k = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} m_{kij} \cdot log(m_{kij}) \tag{94}$$

$$homogeneity_k = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \frac{m_{kij}}{1+|i-j|}, \tag{95}$$

where *N* is the number of gray values and m_{kij} is the value of the co-occurrence matrix m_k at position (*i*,*j*).

- Camera motion features: Videos are segmented into shots using the cut detection approach described in Chapter 4.5.2. Motion vectors embedded in MPEG videos are employed to compute camera motion at the granularity of P-frames, according to the approach presented in Chapter 5. The following camera motion types are distinguished: translation along the x-axis and y-axis, respectively, rotation around the x-axis, respectively y-axis and z-axis, and zoom.

- Text features: A robust text detection approach [62] is applied which can automatically detect horizontally aligned text with different sizes, fonts, colors and languages. First, a wavelet transformation is applied to the image and the distribution of high-frequency wavelet coefficients is considered to statistically characterize text and non-text areas. Then, the k-means algorithm is used to classify text areas in the image. The detected text areas undergo a projection analysis in order to refine their localization. The detected text areas are used to derive the following features: number of text elements, distribution of text elements, and text frame coverage.

- Face features: Frontal faces are detected in each video frame using the face detector provided by Intel's OpenCV library [www.intel.com/technology/computing/opencv]. The number of detected faces and the face frame coverage are considered as feature values.

The camera motion features are useful to recognize the game state of searching and exploring, whereas text detection and texture features help recognizing the preparation state. A player steps into the preparation state with the intention to maintain his equipment. This screen contains several menus and is characterized by a high proportion of overlaid text. Thus, text features are assumed to be very good criteria to detect preparation states. However, text detection in the used game videos is a challenging task, because the text is printed on complex background and the frames include many MPEG artifacts. For the first-person-shooter game "Tactical Ops: Assault on Terror" color moment features seem to be useful to detect the state inactive because of the mostly appearing black areas at the top and bottom of the screen.

8.3.4 Semantic Classification

The goal of the proposed system is to learn models for the high-level semantic states of video games described in section 8.3.1 based on the extracted audiovisual low-level and mid-level features. As stated above, the system does not focus on special properties of the computer game under consideration. Instead of using a specific and narrow approach that only works for a single video game, a generic video content analysis system is utilized that can be easily adapted to other games or video genres. The SVM as suggested by Platt [128] with improvements of Keerthi et al. [85] has been used to learn the mapping between the extracted audiovisual features and the semantic game classes. An early fusion scheme is used to employ multimodal analysis. The datasets consequently consist of concatenated audio and visual features. The training of the SVM is realized by Sequential Minimal Optimization [128]. This is a fast training method which scales somewhere between linear and quadratic in the training set size. Several strategies to classify the computer game videos have been investigated which are described below.

8.3.4.1 CLASSIFICATION USING THE BASELINE SYSTEM

Several SVMs (one for each game class) must be combined to solve our problem, since SVMs are binary classifiers. To make a decision about the game state of a certain frame, the SVM models are employed to provide probability scores for a test instance (frame). These scores are compared and the class with the highest score is chosen.

8.3.4.2 CLASSIFICATION USING TEMPORAL NEIGHBORHOOD

It is observable that the appearance of a certain class is reflected also by the probability scores which are assigned to neighbored frames by an initial SVM classifier. This is the motivation for our second strategy to classify the computer game content. In addition to the audiovisual features, some time series information is utilized. The basic idea of this strategy is to obtain information about the temporal neighborhood of a frame using the probability scores of the initial SVM classifier. Based on the classification results, the relative frequency of each class in the temporal environment is computed for the current frame. The relative frequency of class *c* in the neighborhood of frame *k* is calculated according to the following formula:

$$freq_c(instance_k) = \frac{1}{2w+1} \cdot \sum_{i=k-w}^{k+w} t_c(instance_i) \qquad (96)$$

with $t_c(instance_i)$=1, if frame *i* is classified as class *c* and 0 otherwise, and *w* defines the window size. For example, if the relative frequency of violence is 0.5 for a frame, it follows that 50% of the neighboring frames are classified as violence. Furthermore, a smoothing filter is applied to the class probabilities obtained by the initial classifier. In both cases, a sliding window size of 25 frames is applied. The probability scores of the initial classifier, the frequencies and the smoothed values (4 features each) are used as new features and then re-train another classifier that makes the final decision. The processing steps are displayed in Figure 55.

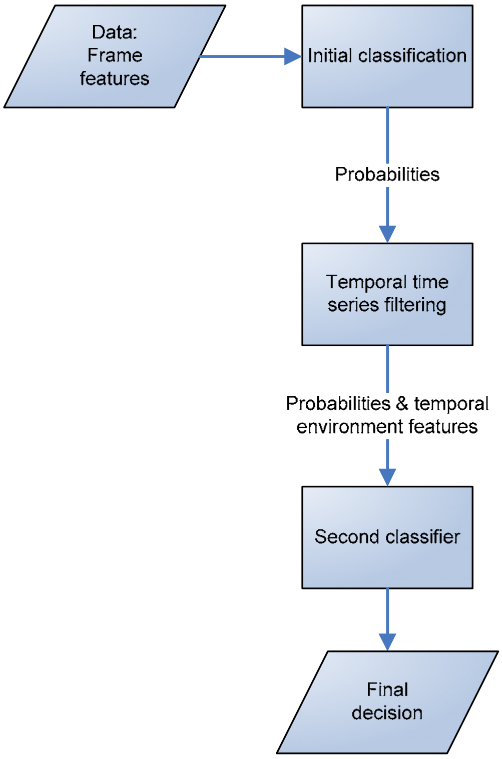


Figure 55: Concatenation of classifiers in order to employ temporal information.

8.3.4.3 REFINEMENT USING SEMI-SUPERVISED LEARNING

In the setting of the addressed psychological experiment, the consumers always play the same game but at different levels and hence, they explore different virtual environments. Thus, it is possible that the SVM models learned from the training video are not suited well to distinguish between the different game classes in the test video. In the previous chapters (4, 6 and 7), it has been shown that an initial model obtained via unsupervised learning can be improved adaptively for a particular video. In order to achieve a more robust classification for a particular game video in our scenario, a similar idea is employed and a semi-supervised learning approach is proposed.

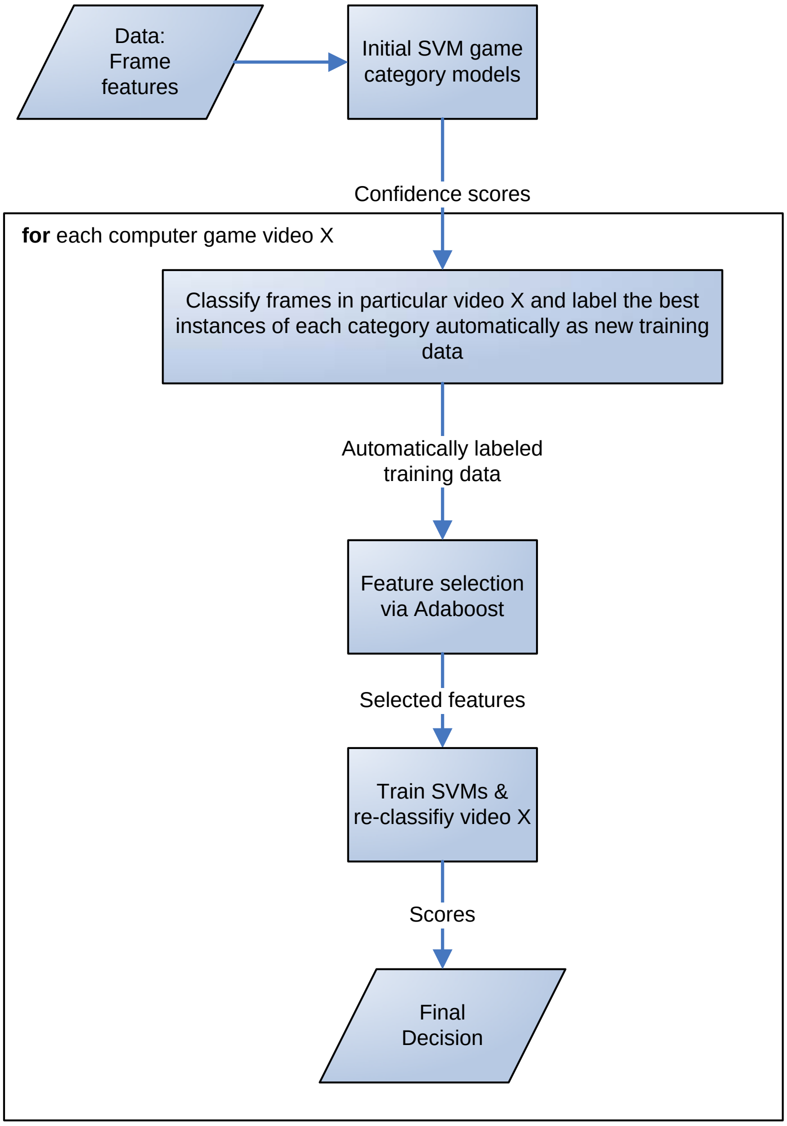

Data:
Frame
features
Initial SVM game category models
Confidence scores
for each computer game video X
Classify frames in particular video X and label the best instances of each category automatically as new training data
Automatically labeled training data
Feature selection via Adaboost
Selected features
Train SVMs & re-classifiy video X
Scores
Final Decision

Figure 56: Main processing steps of the semi-supervised learning approach.

Several strategies are investigated to improve the initial result.First, two variations of the proposed semi-supervised learning framework are employed. The main processing steps of the first variation are depicted in Figure 56 and are as follows. First, the training video is used to build a classifier consisting of the initial game category models. The initial classifier is used to classify the instances (frames) of a test game video as described in section 8.3.4.1. Then, the instances are ranked separately for each game category based on the probabilities of the detected classes. These rankings are then used to choose the instances with the highest confidence of each class. The top fifty percent of each class of the automatically labeled instances are chosen as positive training samples. Based on these automatically labeled most relevant instances of the test video, relevant features are selected using Adaboost [167]. The most relevant 77 features are chosen for subsequent use. An additional classifier is built using the previously chosen instances and selected features. Finally, this semi-supervised classifier, consisting of four newly trained SVMs, is used to classify the test video.

In the second realization of the proposed framework, the set of selected features is split into two parts, according to the odd and even ranks in the feature selection process, and two new SVM classifiers are trained directly on the test video. Finally, the initial video and the two new training videos form an ensemble. As a consequence, there are three classifiers for each of the four game classes. A frame is labeled as class X in case when the number of votes of the "X-classifiers" is higher than those of the other classes. In case there is a equal number of votes for two classes X and Y, the sum of the SVM scores is exploited to obtain a decision for a class label.

Finally, a transductive learning approach has been applied for adaptive game categegory classification via transductive SVM [82].

8.4 Experimental Results

In this section, several experiments to test the system's applicability for the psychological study are presented. The main goal is to significantly reduce the human annotation effort while achieving an accuracy that is comparable to a manual annotation. In the original experimental setting, the human annotators needed 120 hours to label the entire video collection [172]. In addition, the goal was to keep the video content analysis approach generic.

Four computer game videos were used to evaluate the system performance. All computer game videos show a resolution of 352 x 288 pixels and a video frame rate of 25 frames per second. Table 86 presents the distribution of the semantic game categories for each of the videos. The ground truth data were created by Weber et al. [172].

	Preparation	Search	Violence	Inactive	Total
Game-vmj3_7	2390	11657	488	2155	16690
Game-vmj6_3	1665	8574	525	5601	16365
Game-vmj6_4	2364	6445	2630	5251	16690
Game-vmj6_5	2157	10023	1211	2581	15972

Table 86: Number of frames referring to semantic game categories for each of the used computer game videos

A "leave k-1 videos out" cross validation scheme is used: Since the main goal is the reduction of human annotation effort, only one video is used as training data in each test while the remaining three videos are used as test videos. The SVM has been implemented using the WEKA library

[175]. A radial basis function kernel was used for the SVM. Adaboost has been implemented according to the description given by Viola and Jones [167].

The following system variations were tested: 1.) The first one is the baseline system as described in 8.3.4.1. All features mentioned in section 8.3.2 and 8.3.3 are used to learn a SVM model for each semantic game class; 2.) After an initial SVM training, further features are generated that capture temporal characteristics of classes as described in section 8.3.4.2; 3.) The semi-supervised learning scheme as described in section 8.3.4.3: after an initial classification of a test video, the frames that are classified with highest confidence are used as training data. These training data are used to learn new SVM models, and finally the same video is classified using these models. The results for these experiments are presented in Table 87 - Table 89. The measure "total recall" is defined as the percentage of frames which are classified correctly with respect to all semantic game classes.

[%]	Preparation	Search	Violence	Inactive	Total recall
Recall	84.3	92.3	53.9	88.5	87.5
Precision	86.0	87.5	68.7	93.4	
F1	85.1	89.8	60.4	90.9	

Table 87: "Baseline" system: Recall, precision and f1-measure for each of the four semantic classes as well as the total recall.

[%]	Preparation	Search	Violence	Inactive	Total recall
Recall	83.1	92.6	56.7	91.6	88.5
Precision	87.7	88.5	68.4	94.1	
F1	85.4	90.5	**62.0**	92.8	

Table 88: "Baseline + temporal features": Recall, precision and f1-measure for each of the four semantic classes as well as the total recall.

[%]	Preparation	Search	Violence	Inactive	Total recall
Recall	92.6	93.7	55.2	92.1	**90.4**
Precision	96.6	90.1	57.5	98.0	
F1	**94.6**	**91.9**	56.3	**95.0**	

Table 89: "Baseline + Semi-Supervised Learning": Recall, precision and f1-measure for each of the four semantic classes as well as the total recall.

Several observations can be made. At first, our automatic baseline system achieves a frame-based total recall of 87.5% on the average. This is a very good result if one considers that the inter-coder reliability in the original psychological experimental setting between the human annotators was 0.85

[172]. In nearly any experiment, preparation, search and inactive states were recognized well, whereas the recognition of violent states is rather difficult. In terms of total recall, the semi-supervised approach outperforms the alternative approaches (see Table 92). The approach using temporal neighborhood information achieves the best performance for the most difficult concept "violence" and recognizes more than half of the violent actions correctly while keeping the precision at nearly 70% at the same time. The confusion matrix in Table 93 allows one to gain insights in the system failures of the best system (semi-supervised learning). The diagonal represents the number of correctly classified frames. For example, the most frequent error is that a violence frame is misclassified as search, and vice versa, whereas for example a violence frame was never classified as preparation. Overall, it can be concluded that the proposed system achieves a very satisfying performance. It demonstrates the ability to reduce human annotation efforts to a minimum because the system automatically determines relevant game events with high reliability.

In addition, the ensemble variation of the semi-supervised framework was tested as described in the previous section as well as a transductive SVM based on Joachim's implementation. For the transductive SVM, a polynomial kernel was used with exponent *d*=2. The results for the semi-supervised ensemble are presented in Table 90 and for the TSVM approach in Table 91. The semi-supervised ensemble approach is slightly better than the semi-supervised approach which used only the newly trained SVM for classification. The transductive SVM approach yields a significant degradation of annotation quality. Overall, it is observable that using the semi-supervised learning framework improves the results for the more frequent game categories (inactive, search, preparation) but not for the game category violence.

[%]	Preparation	Search	Violence	Inactive	Total recall
Recall	92.0	94.8	52.0	92.1	90.6
Precision	93.4	89.9	63.0	98.4	
F1	92.7	92.3	57.0	95.2	

Table 90: Results for the semi-supervised ensemble approach. Recall, precision and f1-measure for each of the four semantic classes as well as the total recall.

[%]	Preparation	Search	Violence	Inactive	Total recall
Recall	88.6	85.8	40.1	78.5	81.1
Precision	92.6	84.9	41.5	77.7	
F1	90.6	85.4	40.8	78.1	

Table 91: Results for the transductive SVM approach. Recall, precision and f1-measure for each of the four semantic classes as well as the total recall.

[%]	Baseline	Temporal	Semi-Sup.-Learning	Semi-Sup.-Learning Ensemble	Transductive SVM
Total recall	87.5	88.5	90.4	90.6	81.1

Table 92: Total recall for each of the tested systems.

	Prep. (GT)	Search (GT)	Violence (GT)	Inactive (GT)
Det. Prep.	23670	1202	328	133
Det. Search	2012	104362	6659	3005
Det. Violence	42	3851	7567	551
Det. Inactive	4	682	8	43075

Table 93: Confusion matrix for the experiment which used the semi-supervised ensemble. For example, the most frequent error is that a violence frame is misclassified as search and vice versa.

8.5 Summary

In this chapter, an automatic semi-supervised semantic video analysis system that supports psychological experiments on violence in computer games has been presented. In the addressed interdisciplinary study, annotations are required to find interrelationships between the consumer's brain activity and game events during the recorded game sessions, in particular with respect to violent actions. Our proposed system automatically labels such videos and achieves a total recall of up to 91% in the best case using a semi-supervised learning approach based on the transductive learning framework presented in Chapter 3. This approach adaptively refines a model on a particular video: Based on the initial classification result, the approach automatically labels the frames in a new video and adapts its concept models to this video by employing feature selection to adaptively learn a classifier for a particular game video.

Considering the fact that Weber et al. [172] observed an inter-coder reliability of 0.85 for human annotators, our automatic system demonstrates an excellent performance. In addition, since our semi-supervised approach needs labeled training data for a single video only, the required human supervision could be kept at a minimum in this interdisciplinary study. The graphical user interface of our software Videana enables a human expert to refine or to correct the annotation results: As a basic requirement, the annotations must be as accurate as possible to investigate the interrelationship with a player's brain activity. However, such a correction step must also be applied when only human annotators label the videos. Overall, it is concluded that the experimental results demonstrate the applicability of the proposed system for the interdisciplinary studies in the field of media and behavioral sciences.

Chapter Nine

9 Conclusions

9.1 Summary

It is obvious that videos vary in many ways with respect to encoding, layout, genre, and of course with respect to content. As a consequence, it is difficult to build video indexing and retrieval approaches which work reliably for different video sources and content. In this thesis, the question has been investigated how the robustness of video analysis and indexing can be improved via transductive learning methods. A transductive learning ensemble framework has been proposed that exploits an initial classification model or clustering model which is adapted to a particular video in order to improve indexing or retrieval quality. Apart from this approach, it has been also analyzed how compressed data can be successfully exploited without reducing the indexing and retrieval quality.

Up to now, the development of video specific classification models has not been considered in the field of video indexing and retrieval, except for the proposals made in this thesis. Some recent works in other fields such as handwriting recognition [123] and computer vision [54, 138] address this issue. In this respect, there is one of the main contributions of this thesis: To the best of our knowledge, the usefulness of transductive learning for video content analysis and indexing has been motivated and addressed explicitly in this thesis for the first time. The need for robust and adaptive video content analysis methods is identified and clearly motivated, in particular by an experimental evaluation of high-quality cut detection approaches on several test sets. It has been demonstrated even for the well known problem of cut detection, that the quality of the best approach of our comparison study varies in dependence of the analyzed video – it even completely fails on a particular video. Thus, it is obvious that there is also a need for robust and adaptive methods for other video indexing problems which are more difficult than cut detection. The goal of such robust video indexing methods is to automatically obtain reliable results for any particular video. It has been investigated how clustering or classification models can be adapted to and optimized for a particular video via transductive learning methods, which are realized either in a self-supervised or in a semi-supervised manner. To achieve the envisaged goal, the (unlabeled) data of the analyzed video itself have to be incorporated in the learning process. Several fundamental approaches have been pursued, including some approaches which are not based on the proposed framework, namely:

1. Employing and extending of an unsupervised learning method (applied to cut detection);

2. Transductive learning (transductive SVM);

3. A transductive learning ensemble framework based on feature selection and ensemble classification.

Several proposals have been presented for different video indexing problems which automatically cope with different video sources, compression artifacts and different appearances of objects and events in order to achieve robust indexing results. The particular contributions are briefly summarized below.

9.1.1 Transductive Learning Methods for Robust Video Content Analysis

The proposed transductive learning ensembles, realized by a self-supervised or a semi-supervised learning scheme (Chapter 3), respectively, aim to recognize the specific appearance of objects or events in a video for robust analysis. The schemes are based on the assumption that an appropriate baseline system with sufficient classification accuracy exists. The scheme is called self-supervised in case when the baseline system relies on unsupervised learning. In case when the initial system consists of a supervised learning result, the learning scheme is called semi-supervised. The idea is to employ an initial clustering or classification result to obtain automatically labeled training data for this video – and to train additional transductive classifiers on and for this video using these training data. The best features to classify the objects of interest are selected using Adaboost based only on the automatically labeled training data. To achieve a robust final classification result, the newly created transductive classifiers and the initial classifier form an ensemble using majority voting. The newly trained classifiers are trained on different feature sets, the set of best features is split according to the odd and even ranks in the selection process. This transductive learning ensemble scheme has been successfully applied to the following tasks:

- video cut detection,
- face recognition in video,
- video retrieval based on high-level concepts, and
- semantic annotation of computer game sequences.

The achieved results for these tasks are briefly summarized in Table 94 with respect to the applied initial baseline system, the number of features used by the baseline system, the percentage of automatically labeled training samples per video which were selected for subsequent re-training, the number of selected features that was sufficient to achieve the best result for a task, ratio of test and training data per test video, as well as information about the fact which system performed best (single transductive classifer, transductive learning ensemble, or transductive SVM), and how it compared to the corresponding baseline system. The question is: What can we learn or conclude from Table 94 and all the experiments reported in this thesis about the transducitve learning framework?

	Cut detection (Chapter 4)	Face recognition (Chapter 6)	Semantic video retrieval (Chapter 7)	Semantic indexing of computer games (Chapter 8)
Initial baseline classification/clustering	K-means clustering	Agglom. clustering	SVM classification	SVM classification
Baseline performance	F1: 87.8	F1: 84.8	63.1.% average precision (libSVM)	87.5% (accuracy)
Percentage of selected (automatically labeled) positive training samples per video	90% (better than 100%)	100% (no selection)	Max. 10%	50% (better than 10%, 20%, 33%)
Initial number of features	2	3600	120	178
Sufficient number of features	45	15	90	77
Ratio test data:training data per test video	-	**-**	Ca. 1:100	Ca. 1:1
Transductive learning framework with only 1 new classifier is better than baseline	Yes (baseline+2.9)	**Yes** (baseline+4.0)	*No* (baseline-35.8%)	**Yes** (baseline+2.9%)
Transductive learning ensemble is better than using only a single transductive classifier	**Yes** (baseline+4.2)	*No* (baseline+2.8)	Yes (baseline+1.4%)	**Yes** (baseline+3.1%)
Transductive learning ensemble is better than baseline system	**Yes** (baseline+4.2)	Yes (baseline+2.8)	Yes (baseline+1.4%)	**Yes** (baseline+3.1%)
Transductive SVM is better than transductive learning ensemble framework (baseline is SVM-lite)	-	-	**Yes** (baseline+6.0%)	*No* (baseline-7.4%)

Table 94: Summarization of the results of the proposed transductive learning framework for several tasks.

First, it is obvious that the investigated tasks are very different. The feature sets vary with respect to size and used feature types. The baseline performance ranges from 63.1% average precision for semantic video retrieval up to an f1-measure of 87.8 for cut detection. In case of semantic video retrieval, there are videos which even do not contain the concept of interest (e.g., maps are not present in each news video). Of course, the most interesting question is which approach performs best. While the realization of the proposed framework using only a single re-trained classifier improves performance moderately for three of four tasks, its performance degrades dramatically for the semantic video retrieval task (high-level feature retrieval). It is probably caused by the low number of positive training samples in some videos in this scenario: in some cases there are only few or even no positive training samples which leads to bad SVM models. With respect to a low number of positive samples in the test data, the transductive learning ensemble approach is advantageous since the initial classification model, possibly trained with a large number of samples, is not discarded but included in the final ensemble decision. The transductive SVM approach worked very well for semantic video retrieval task, however, its performance for the task of semantic computer game indexing was bad and the analysis of the reasons for that remains future work. A possible reason is that the ratio of training data and test data were of equal size, in contrast to the semantic video retrieval task. Overall, it has been demonstrated that adaptive and robust video analysis can be achieved via the proposed framework. The ensemble approach is always a good choice, it achieved the best performance for three of four tasks and its performance did never degrade compared to the baseline system.

9.1.2 Un- /Self-supervised Learning for Robust Shot Boundary Detection

In this thesis, the field of shot boundary detection in videos has been studied extensively. Recent approaches have been surveyed including the approaches that achieved best results at the TRECVID 2005 evaluation. A comprehensive comparison study has been conducted that summarizes the current state-of-the-art of shot boundary detection techniques. The literature review has been supplemented by an experimental comparison study of recent approaches which had not been evaluated on publicly available test sets before. Interestingly, it has been shown that the best approach (based on metric selection and SVM) does not work reliably on arbitrary videos: it even failed completely for a video of the TRECVID 2007 test.

This has illustrated the need for developing robust robust and adaptive approaches. In a first step, a robust unsupervised shot boundary detector has been presented. The unsupervised approach for cut detection works reliably without any pre-defined parameters and thresholds. For this purpose, the fact that the classification problem of shot boundary detection can be transformed into a clustering problem has been exploited. The important parameter of sliding window size could be estimated automatically using a clustering validity measure (silhouette coefficient). In addition, no training is required to learn a classification function or to estimate empirically thresholds, respectively. As a surplus, it has been observed that the silhouette coefficient is quite strongly related to the precision of a cut detection result, and formulas to estimate precision and recall have been derived and validated on the 2005 TRECVID test set. Furthermore, in case when training data are available it has been demonstrated that this unsupervised approach can be extended to an ensemble approach by adding supervised classifiers which yielded an improved detection performance. Finally, the task of cut detection has been considered as a transductive learning setting and the unsupervised video cut detector has been extended with the functionality of self-supervised learning, which improved detection performance.

9.1.3 Estimation of Arbitrary Camera Motion in MPEG Videos

An approach for camera motion estimation has been proposed for the MPEG domain. In contrast to other approaches for this domain, its main advantage is the fact that it potentially distinguishes between rotation around the y-axis (x-axis) and translation along the x-axis (y-axis). Furthermore, the incorporation of an outlier removal algorithm reduces the noise in motion vector fields. For the purpose of experimental validation, a comprehensive video test set that allows to entirely control the camera motion parameters in these videos has been created. It has been demonstrated that outlier removal increases the detection performance significantly. Most detection errors are due to the difficult distinction between the corresponding translational and rotational camera movements. In addition, we participated in TRECVID's 2005 evaluation task for camera motion detection (low-level feature task 2005: detection of pan, tilt, and zoom) and achieved very good results on a large real-world test set. In terms of the f1-measure the best result for zoom detection was achieved among 12 participants at TRECVID 2005, and the second best result for tilt detection.

9.1.4 Self-Supervised Learning for Person Recognition in Video

A novel automatic video annotation system with respect to a person's occurrence has been presented. It has been demonstrated how face detection results can be exploited to estimate the eye positions precisely in order to automatically cope with in-plane rotated faces. Furthermore, it has been investigated in which way an initial face clustering result can be further improved by self-supervised learning, in particular with feature selection and appropriate re-classification. For this purpose, only the face samples present in the given video were used. Several possibilities to train Adaboost and SVM classifiers and to train an ensemble of these classifiers on a video have been presented and compared. The experimental results have shown that it is sufficient to train a single classifier. The experimental results have demonstrated the effectiveness of the proposed in-plane rotation removal. Finally, the best performance has been achieved by the Adaboost and a SVM classifier, respectively, each increasing the f1-measure from 84.8 up to 88.8.

9.1.5 Semi-Supervised Learning for Semantic Video Retrieval

The semi-supervised learning framework has also been applied to the task of high-level concept detection in order to improve the precision of retrieval results. For this purpose, the observation that there are concepts whose appearance is strongly related to a certain video source has been exploited (e.g., anchor person in a news cast). The problem has been considered as a learning task in a transductive setting. To this extent, it was investigated whether transductive learning approaches are also applicable to high-level concept detection. The average precision could be improved for some concepts using the proposed semi-supervised learning framework. A transductive SVM has been applied for this task as well and the experimental results demonstrated a further improvement in terms of average precision. Experimental results on the TRECVID 2005 training set identified high-level concepts, for example "anchor", "entertainment" and "maps", which are strongly related to a video source.

9.1.6 Semi-Supervised Learning for Semantic Annotation of Computer Games

The transductive learning ensemble approach has also been used for the semantic annotation of computer game sequences. An automatic semi-supervised semantic video analysis system that supports psychological experiments with respect to violence in computer games has been presented. These annotations are required to find interrelationships between the consumer's brain activity and game events during the recorded game sessions, in particular with respect to violent actions. The proposed system automatically labels such videos and achieves a total recall of up to 91% in the best case using the semi-supervised learning approach.

9.1.7 Robust Video Indexing in the MPEG Compressed Domain

Information embedded in a compressed MPEG video stream is potentially useful for video content analysis. Furthermore, computational costs to fully decode video frames can be saved through its use. However, compression artifacts might lead to noisy representations of visual information and might degrade indexing quality. In this respect, a systematic bias in frame dissimilarity measurements in MPEG videos was identified which potentially hinders video cut detection performance. Two solutions were presented to deal with these artefacts: frame difference normalization (FDN) and GoP-oriented frame difference normalization (GFDN).

Regarding the estimation of arbitrary camera motion, a solution that is based on motion vectors embedded in compressed MPEG videos has been presented. To deal with noisy motion vectors that do not represent "true" motion, an effective outlier removal algorithm has been applied. Experimental results have shown that the incorporated outlier removal algorithm improved the performance significantly.

9.1.8 Summary of Contributions

Overall, the contributions of this thesis can be summarized as follows:

1. Identification and motivation of the need for robust and adaptive video content analysis approaches; in particular; it was motivated that video content analysis tasks should be considered in a transductive learning setting, which has not been done before in the field of video content analysis and indexing.

2. Development of a novel transductive learning framework for robust video content analysis based on feature selection and ensemble classification;

3. Identification of a systematic bias in frame dissimilarity measurements for MPEG videos, and presentation of two proposals to remove compression artefacts from frame dissimilarity measurements (FDN and GFDN) in order to improve video cut detection;

4. Development of an unsupervised approach for robust shot boundary detection, including

    a. automatic estimation of the important parameter of the sliding window size;

    b. investigation of possible improvements by extending the unsupervised cut detection approach with additional supervised classifiers to form an ensemble;

    c. extension of the gradual transition detector with an effective approach to remove false alarms based on a high-quality camera motion estimation approach;

5. Development of an approach for automatic performance prediction for video cut detection. To the best of our knowledge, this is the first proposal of this kind for the field of video indexing;

6. Application of the transductive learning framework (self-supervised) to video cut detection;

7. A comprehensive comparison study regarding recently suggested shot boundary detection approaches, including experiments for recent approaches which have not been evaluated before on common test sets (such as TRECVID test sets);

8. Development of a camera motion estimation approach for MPEG videos that deals adequately with noisy motion vector fields and potentially distinguishes between rotational and translational movements;

9. Application of the transductive learning framework (self-supervised) to the problem of face recognition in video for the purpose of an automatic video indexing system with respect to person appearances;

10. Application of the transductive learning framework (semi-supervised) to video shot retrieval based on high-level concepts;

11. Application of transductive SVM to the task of high-level concept detection;

12. Supporting interdisciplinary research by providing semantic annotation of computer game sessions, based also on the semi-supervised realization of transductive learning framework.

9.2 Future Work

In this thesis, a transductive learning ensemble framework and a number of particular solutions for several video content analysis tasks were proposed. As a matter of fact, a lot of work remains to be done to build robust and adaptive methods in practice. The fields of shot boundary detection, camera motion estimation, face recognition in video, and high-level concept detection leave more or less room for future improvements. The possibilities for future work are now discussed in detail for several research fields.

9.2.1 Transductive Learning Ensemble for Robust Video Content Analysis

It was shown experimentally that the proposed self-supervised and semi-supervised learning framework can be successfully applied for several video indexing and retrieval tasks but, of course, several questions remain. First, under which circumstances are the training data offered by a video sufficient to successfully learn classifiers for this video? Second, what must be the minimum accuracy for an initial baseline classifier? Third, how many positive and negative examples must be available for a certain concept, object or event to be learned? Finally, in which way do these aspects interact with one another? In practice, the concrete realization varied for the different video indexing and retrieval tasks, and there are many possibilities to modify components in the framework which are discussed now separately:

9.2.1.1 INITIAL CLASSIFICATION STAGE

Here, the first question regards the selection of features: In which cases is it recommendable to use the complete available feature set or only a subset of the features to obtain a first stable classification result that provides a good basis for subsequent learning. Furthermore, is it important to achieve the best possible classification result in the initial classification stage?

9.2.1.2 SUBSEQUENT LEARNING STAGE

The main question is how to train and to form an ensemble of classifiers which finally achieves a robust classification result that at least reaches the performance of the initial classifier. In this thesis, an ensemble approach was utilized in most cases but was not better than a single re-trained classifier in all scenarios. When several classifiers are trained for an ensemble, the following unsolved research question arises: how much diversity (and how should diversity be measured?) is needed between the different ensemble experts to achieve an optimal classification accuracy?

9.2.1.3 SELECTION OF TRAINING SAMPLES FOR THE LEARNING STAGE

Currently, all training samples that are classified with highest confidence are incorporated to train a classifier. Alternatively, subsets of these samples could be used in order to obtain different classifiers.

9.2.1.4 ITERATIVE LEARNING VERSUS NON-INTERATIVE LEARNING

In the current framework, learning is conducted in a non-iterative way in order to save computation time. However, the potential to further improve accuracy by using an iterative learning scheme should be investigated.

9.2.1.5 MISCELLANEOUS

Furthermore, in the future, the impact of both classifier independence and baseline system accuracy should be investigated. The use of other feature selection methods and classifiers within the framework should be analyzed as well as the application to other video indexing or pattern recognition tasks. Finally, recent advances in the field of machine learning should be considered with respect to their applicability for robust and adaptive video indexing and retrieval. Although semi-supervised learning or transductive learning methods (as transductive SVM) were used for the identified issue, some recently published methods might be well suited for this task and thus should be investigated. For example, some recent work of Bickel et al. [12] addresses the issue of building classifiers for differing training and test distributions.

9.2.2 SHOT BOUNDARY DETECTION IN VIDEOS

The main issue that remains in the field of shot boundary detection is the reliable detection of gradual transitions. The top approaches for gradual transition detection achieve a performance of about 80% in terms of recall and precision. From our point of view, better results will come along only with better object recognition methods and a better understanding of scene content. If a system would better recognize objects and events in terms of content, gradual transitions between different shots could be detected much easier.

9.2.3 ESTIMATION OF ARBITRARY CAMERA MOTION IN VIDEOS

Most detection errors are still due to the difficult distinction between translational and rotational camera movements. This problem could be investigated in future work by considering context information about objects in a scene. The distinction between camera motion and object motion should be considered in more detail in the future.

9.2.4 PERSON RECOGNITION IN VIDEO

There are several areas for future work. The robustness of the face detector must be further improved. In particular, the Adaboost based profile face detector is currently not well suited to detect profile faces in a video reliably. This is the reason that currently only frontal faces are processed, although the system is already capable of clustering and recognizing profile faces. Furthermore, a reasonable automatic clustering termination criterion is needed. The self-supervised learning procedure can be improved by the use of a larger feature set, possibly enriched with features from holistic or 3D face recognition approaches.

9.2.5 Detection of High-Level concepts

The reliable generic detection of arbitrary high-level concepts is still a challenging research problem. For the application of semi-supervised and transductive learning, it should be investigated further whether the retrieval performance for other concepts can be improved using this learning scheme. In addition, the adaptation of learned models will become more and more important in the future. Recent research indicates [71] that a number of 1000 – 3000 concepts is necessary to reach the retrieval quality of today's text search engines. State-of-the-art concept detectors are based on supervised learning, and a reasonable amount of positive training samples is required. The annotation of a large video set with respect to such a large number of concepts is a very time-consuming task. Consequently, there is a problem if the models are, for example, learned on news videos, but the genre of the target video archive is completely different (movies, sports videos), or the type of video sources is completely different (e.g., youTube.com, or any other unedited videos, such as home user videos and rushes material). Hence, the issue of adapting existing models to new video collections will become a very important issue in practice.

9.2.6 Semantic Indexing of Computer Games

It became apparent that the automatic annotation of semantic concepts like “danger” is very difficult. This concept depends on the detection of person occurrences and particularly on a “friend-or-foe” distinction. However, these persons appear in very small sizes in the game, and it is even hard for a human annotator to make the decision whether the situation is actually “dangerous” or not. Finally, temporal state transitions promise to entail additional useful information, for example, the state “inactive” is always preceded by the state “violence”. Such temporal relationships should also be incorporated into the automatic annotation system.

ANNEX A. LIST OF FIGURES

Figure 1: Typical model for statistical pattern recognition, (according to [78], slightly modified). 20

Figure 2: Framework for multimodal video indexing (according to Snoek and Worring [143]). 23

Figure 3: Schema for content-based video retrieval. 26

Figure 4: Scheme of the encoding process for I-frames according to the MPEG-1 standard. This coding scheme is similar for JPEG image encoding. 29

Figure 5: A typical frame ordering for a group of pictures in an MPEG-1 video. Below the encoding and the display order are displayed. 31

Figure 6: Example for a movement of a macroblock. Right: The macroblock that must be encoded. Left: Its best block match in a previous reference frame. 32

Figure 7: Example for a fade-in from a black monochrome frame. Three representative frames, taken from the MPEG-7 video riscos-sl.mpg. 33

Figure 8: Example for a dissolve. Three representative frames, taken from the MPEG-7 video jornaldanoite.mpg. 33

Figure 9: Examples for different shot sizes: from left to right and top to bottom: close-up shot, close shot, medium shot, full shot and long shot. 34

Figure 10: Example for a cut frequency diagram for a short music video clip, generated with *Videana*. 36

Figure 11: Pseudo code for k-means clustering. 40

Figure 12: Pseudo code for EM clustering. 41

Figure 13: Pseudo code for agglomerative clustering. 43

Figure 14: An example for a well class-separating hyperplane (left) and a non-optimal separating hyperplane (right). 46

Figure 15: Pseudo code for co-training according to [13]. 49

Figure 16: Example how a transductive SVM might refine and improve a maximum margin hyperplane achieved by an inductive SVM. Class 1 is represented by black rectangles, class 2 is represented by white triangles. Unlabeled data are displayed as grey circles. The SVM hyperplane is shown as a dashed line, whereas the TSVM hyperplane is displayed by a solid line. 50

Figure 17: Pseudo code for bagging model generation. 53

Figure 18: Pseudo code for classification using bagging. 53

Figure 19: Pseudo code for Adaboost model generation. 54

Figure 20: Pseudo code for Adaboost classification. 54

Figure 21: Pseudo code for the simplex algorithm of Nelder and Mead. 58

Figure 22: Framework for transductive learning ensembles for video content analysis. The process is applied to each video separately. Dashed lines indicate optional processing steps. 61

Figure 23: Pseudo code for feaure selection based on Adaboost. 62

Figure 24, left: Average difference values for different GOP positions normalized in relation to the minimum difference for "riscos-sl.mpg", using decoded full frames. Right: Average difference values for the same video, normalized in relation to the minimum difference as well, but approximated DC-frames were used. 92

Figure 25, left: Difference for approximated DC-frame and DC-frames that are reconstructed based on the fully decoded frame. Right: Example for a possible false alarm detection at frame 2882: there are nearly no changes between frame 2881 and 2882 but frame 2882 is an I-frame. 93

Figure 26: Pseudo code for GoP-oriented frame difference normalization. 97

Figure 27: Pseudo code for the algorithm to estimate the sliding window size. 105

Figure 28: Distribution of recall/precision for shot detection at TRECVID 2005 depending of the video sources [137]. 126

Figure 29: Reduction of error rate for each video of the TRECVID 2005 shot boundary test set. Error rate reduction approach when using the self-supervised approach is given in relation to the original unsupervised approach. Error rate is given separately for each video from the TRECVID 2005 shot boundary test set. 131

Figure 30, top: The silhouette coefficient (SC, scaled by 100), precision and estimated precision based on SC (both in %) for 33 videos from the MPEG-7 test set. Bottom: The data for recall. 136

Figure 31, top: The silhouette coefficient (SC, scaled by 100), precision and estimated precision based on SC (both in %) for 12 videos of the TRECVID 2005 test set. Bottom: The data for recall. 137

Figure 32: Various types of camera motion. 139

Figure 33: Two examples for outlier criteria showing a motion vector and its neighbors. Left: "smoothness"; Right: "neighborhood". 142

Figure 34: The 3D camera model that is used in the proposed approach. 143

Figure 35: The main processing steps of the self-supervised system are displayed here. Optionally, aggregation of face sequences appearing in dialogs can be applied. 152

Figure 36, left: Example for face detections at slightly different scales and position. positions. Right: Only features in the inner face area are used for tracking. 153

Figure 37: Settings to capture a dialog of two persons with two and three cameras (taken from [6]). 155

Figure 38: The principle of aggregating two face sequences A and B of two persons in a dialog scene (taken from [6]). 155

Figure 39: The face graph structure as used in the proposed system. Left: Frontal face graph. Right: Profile face graph. 156

Figure 40: Examples of automatically corrected in-plane rotated faces. The original faces as extracted from the video are displayed in the top row. Below are the results after correcting in-plane rotation. 157

Figure 41: Example for a shot-based person indexing result for the talk-show sequence. 168

Figure 42: Example shot-based indexing results for the six persons of the talk-show sequence. Vertical lines represent the shot boundaries. For each person, the upper timeline represents the ground-truth data, and the bottom timeline represents the automatic indexing result. 169

Figure 43: Pseudo code for the semi-supervised retrieval scheme. 177

Figure 44: Example retrieval result (top-50) for the concept “anchor” when using the semi-supervised learning ensemble approach. 183

Figure 45: Example retrieval result (top-50) for the concept “maps” when using the Mediamill baseline system. 184

Figure 46: Example retrieval result (top-50) for the concept “maps” when using the semi-supoervised learning ensemble approach. The “non-maps” shots are shifted downwards when using the semi-supervised approach compared to the baseline system. 185

Figure 47: Scheme for semantic video indexing and retrieval based on transductive SVM learning. 189

Figure 48: Example retrieval result (top-50) for the concept “anchor” when using the baseline SVM. 194

Figure 49: Example retrieval result (top-50) for the concept “anchor” when using the transductive SVM. Interestingly, the retrieved top-50 shots using the transductive SVM are from several different programs. When using the baseline SVM, all retrieved shots are from the same program....................................195

Figure 50: Example retrieval result (top-50) for the concept “charts” when using the SVM baseline system. 196

Figure 51: Example retrieval result (top-50) for the concept “charts” when using the transductive SVM. 197

Figure 52: The four boxes explain the different semantic game classes used in this study and how they relate to the categories used by Mathiak and Weber [108], which are displayed in the dashed nested boxes. The classes are ordered from bottom to top in terms of increasing violent content, where PA stands for “player’s avatar”. 202

Figure 53: Example frames of the four different game classes. 203

Figure 54: Scheme of the hierarchical audio type classifier: A single feature vector per sub-clip serves as input; output is a single acoustic class label and its corresponding probability. 205

Figure 55: Concatenation of classifiers in order to employ temporal information. 209

Figure 56: Main processing steps of the semi-supervised learning approach. 210

ANNEX B. LIST OF TABLES

Table 1: Composition of the test sets for the shot boundary detection task at TRECVID in the years from 2003-2007. 65

Table 2: Performance results in terms of F1-measure for selected approaches which have been evaluated on a TRECVID benchmark test set. Chua et al. report an increasement with respect to their baseline system of 10.0 in terms of f1-measure [26]. 77

Table 3: Performance results in terms of F1-measure for selected approaches which have been evaluated on parts of MPEG-7 video test set. 78

Table 4: Experimental results for cut detection on the TRECVID 2005 test data for the different approaches using the parameter settings as described in the related papers. 83

Table 5: Experimental results for cut detection on the TRECVID 2005 test data for the different approaches using optimized parameter settings. 85

Table 6: Experimental results for cut detection on the TRECVID 2006 test data for the different approaches using optimized parameter settings. 85

Table 7: Experimental results for cut detection on the TRECVID 2007 test data for the different approaches using optimized parameter settings. 85

Table 8: Experimental results for cut detection on the MPEG-7 video test set for the different approaches using optimized parameter settings. 85

Table 9: Experimental results for cut detection on the MPEG-7 video test set for the different approaches using optimized parameter settings. 86

Table 10: Experimental results for cut detection on the news video part of the TRECVID 2005 test data for the different approaches using optimized parameter settings. 86

Table 11: Experimental results for cut detection on the NASA video part of the TRECVID 2005 test data for the different approaches using optimized parameter settings. 86

Table 12: Experimental results on the TRECVID 2007 test data for the local thresholding approach depending on the used training set for learning. 87

Table 13: Experimental results on the TRECVID 2007 test data for the statistical approach depending on the used training set for learning. 88

Table 14: Experimental results on the TRECVID 2007 test data for the approach of Bescos [11] depending on the used training set for learning. 88

Table 15: Experimental results for gradual transition detection on the TRECVID 2005 data, according to TRECVID evaluation. 88

Table 16: Experimental results for gradual transition detection on the TRECVID 2007 data, according to TRECVID evaluation. 88

Table 17: Experimental results for gradual transition detection on the TRECVID 2005 data. In addition, short gradual transitions (with at least one transitional frame involved) are considered, in contrast to the results in Table 15 and Table 16. 89

Table 18: Experimental results for gradual transition detection on the TRECVID 2005 data. Also short gradual transitions (with at least one transitional frame involved) are considered, in contrast to the results in Table 15 and Table 16. 89

Table 19: Results are presented for the MPEG-7 test set for both methods: FDN and GFDN. GFDN outperforms the baseline implementation and the FDN method. 99

Table 20: Cut detection results for the SVM extension of Bescos' approach [11] for different sliding window sizes on the TRECVID SBD test set 2005. 110

Table 21: Cut detection results for Hanjalic's approach [68] for different sliding window sizes on the TRECVID SBD test set 2005. 110

Table 22: Cut detection results for the proposed approach for different sliding window sizes on the TRECVID SBD test set 2005. 110

Table 23: Cut detection results for the proposed approach for different sliding window sizes on the TRECVID SBD test set 2006. 110

Table 24: Results for different ranges for automatic sliding window size estimation on TRECVID 2005 test set. 111

Table 25: Results for different ranges for automatic sliding window size estimation on TRECVID 2006 test set. 111

Table 26: Cut detection results on TRECVID data 2005. Comparison of the proposed unsupervised approach with the approach of [11] extended with SVM. 111

Table 27: Cut detection results on TRECVID data 2006. Comparison of the proposed unsupervised approach with the approach of [11] extended with SVM. 112

Table 28: Cut detection results on TRECVID data 2007. Comparison of the proposed unsupervised approach with the approach of [11] extended with SVM. 112

Table 29: Cut detection results on MPEG-7 video data set. Comparison of the proposed unsupervised approach with the approach of [11] extended with SVM. 112

Table 30: Cut detection results with respect to all data sets. Comparison of the proposed unsupervised approach with the approach of [11] extended with SVM. 112

Table 31: Results for a frame distance of 6 for different sliding window sizes. 113

Table 32: Results for a frame distance of 10 for different sliding window sizes. 113

Table 33: Results for a frame distance of 20 for different sliding window sizes. 113

Table 34: Results for a frame distance of 25 for different sliding window sizes. 114

Table 35: Results for a frame distance of 30 for different sliding window sizes. 114

Table 36: Results for a frame distance of 40 for different sliding window sizes. 114

Table 37: Results for a frame distance of 50 for different sliding window sizes. 114

Table 38: Results for thr combinations of different frame distances. Two clustering types are distinguished: using the feature vectors for different frame distances in one clustering process (“all), or in separate clustering processes (“separate”). 115

Table 39: Results for different strategies of false alarm removal. 115

Table 40: Performance for gradual transition detection when only one feature is used. Tested in conjunction with different false alarm removal strategies. 116

Table 41: Performance for gradual transition detection when only one feature is used but two frame distances. Tested in conjunction with different false alarm removal strategies. 116

Table 42: Performance for gradual transition detection when thrre frame distances are used. Tested in conjunction with different false alarm removal strategies and features. 116

Table 43: Results for six or seven different frame distances, combined with different parameter settings. 117

Table 44: Results for six or seven different frame distances used in conjunction with the feature *max* only, combined with different false alarm removal strategies. 117

Table 45: Results for six or seven different frame distances used in conjunction with sliding window size parameter 24, combined with different false alarm removal strategies 117

Table 46: Results for gradual transition detection on the TRECVID 2005 test set for the proposed approach and the local thresholding approach of Yeo and Liu [184]. 118

Table 47: Recall and precision for different maximum sliding window sizes, using optionally an ensemble of classifiers or false alarm removal. 122

Table 48: Experimental results for the classifiers used in the ensemble approach. 122

Table 49: Experimental results on the TRECVID 2005 test set for the unsupervised baseline system, for a SVM classifier and for some Adaboost classifiers using a different number of features. Each of them was trained separately on each video based all automatically positive labeled training samples. The number of negative training samples depends on the number of positive samples. 129

Table 50: Experimental results on the TRECVID 2005 test set for the unsupervised baseline system, for a SVM classifier and for some Adaboost classifiers using a different number of features. Each of them was trained separately on each video based only on the "best" 90% positive training samples. The factor of negative training samples remained unchanged. 129

Table 51: Experimental results on the TRECVID 2005 test set for several classifier ensemble schemes. Supervised experts in these ensembles are trained only on the automatically labeled data for the given video. 130

Table 52: Experimental results on the TRECVID 2005 test set for three different self-supervised variations. 130

Table 53: Experimental results on the TRECVID 2006 test set for the ensemble scheme with two SVMs using alternating features. Supervised experts in these ensembles are trained only on the automatically labeled data for the given video. 130

Table 54: Experimental results on the TRECVID 2007 test set for the ensemble scheme with two SVMs using alternating features. Supervised experts in these ensembles are trained only on the automatically labeled data for the given video. 131

Table 55: Results for the TRECVID 2005 shot boundary test set: The silhouette coefficient for each cuts cluster which was obtained after k-means clustering, the estimated precision and the real precision, and the estimation error..................135

Table 56: The detection results listed separately for each motion type for the baseline implementation and its extension with outlier removal. 144

Table 57: The results for each motion type if the number of motion vectors used in the optimization algorithm is reduced after outlier removal. 145

Table 58: The results if translation and the similar rotation type are considered as a single motion type. 145

Table 59: Experimental results in terms of recall and precision for our submission for the TRECVID 2005 low-level feature task. 145

Table 60: Experimental results in terms of F1- measure for our submission for the TRECVID 2005 low-level feature task. 145

Table 61: The Gabor wavelet specification as used in our system. 158

Table 62: The experimental results for the talk show sequence when the number of detection hits is varied in the process of face sequence generation. 164

Table 63: The experimental results for the talk show sequence when the minimum number of faces forming a face sequence is varied in the face sequence generation process. 164

Table 64: The experimental results for the talk show sequence when the maximum number of best faces representing a face sequence is varied. 164

Table 65: The experimental results for the talk show sequence when the clustering termination threshold is varied....................164

Table 66: The experimental results for the talk show sequence using the baseline system. Recognition is computed based on the detected face sequences only. 165

Table 67: The experimental results for the talk show sequence when in-plane rotation of faces was performed. Recognition is computed based on the detected face sequences only....................166

Table 68: The experimental results for the talk show sequence when inplane-rotation was performed and Adaboost was used to learn facial features of face clusters. Recognition is computed based on the detected face sequences only.......166

Table 69: The experimental results for the talk show sequence when inplane-rotation was performed and SVM or a 3-SVM ensemble, respectively, was used to learn facial features of face clusters. 167

Table 70: The experimental results for the talk show sequence when the number of selected features is varied using the Adaboost classifier. 167

Table 71: Experimental results for the high-level feature „maps": Average precision values for the MediaMill challenge baseline system and the proposed semi-supervised approach are presented. 182

Table 72: Experimental results for the high-level feature „anchor": Average precision values for the MediaMill challenge baseline system and the proposed semi-supervised approach are presented. 182

Table 73: Experimental results for the high-level feature „charts": Average precision values for the MediaMill challenge baseline system and the proposed semi-supervised approach are presented. 182

Table 74: Experimental results for the high-level feature „overlaid text": Average precision values for the MediaMill challenge baseline system and the proposed semi-supervised approach are presented. 182

Table 75: Experimental results for the high-level feature „maps" for different learning strategies. The number in brackets indicates the number of selected features by Adaboost. 186

Table 76: Experimental results for the high-level feature „anchor" for different learning strategies. The number in brackets indicates the number of selected features by Adaboost. 186

Table 77: Experimental results for the high-level feature „charts" for different learning strategies. The number in brackets indicates the number of selected features by Adaboost. 186

Table 78: Experimental results for the high-level feature „anchor": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented. 190

Table 79: Experimental results for the high-level feature „charts": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented. 191

Table 80: Experimental results for the high-level feature „entertainment": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented. 191

Table 81: Experimental results for the high-level feature „maps": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented. 191

Table 82: Experimental results for the high-level feature „overlaid text": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented. 192

Table 83: Experimental results for the high-level feature „studio": Average precision values for the baseline SVM system and the proposed transductive SVM approach are presented. 192

Table 84: Experimental results for all six high-level features which were expected to benefit from transductive learning. The average of all average precision values is presented for the baseline SVM system and the proposed transductive SVM approach. 192

Table 85: Comparison of the MediaMill challenge baseline system [143], our baseline SVM (svmLight), the semi-supervised learning approach presented in Chapter 7.3, and the proposed transductive SVM approach. Only those concepts are considered for which either the semi-supervised approach or the transductive approach yielded an improvement. 193

Table 86: Number of frames referring to semantic game categories for each of the used computer game videos 211

Table 87: "Baseline" system: Recall, precision and f1-measure for each of the four semantic classes as well as the total recall. 212

Table 88: "Baseline + temporal features": Recall, precision and f1-measure for each of the four semantic classes as well as the total recall. 212

Table 89: "Baseline + Semi-Supervised Learning": Recall, precision and f1-measure for each of the four semantic classes as well as the total recall. 212

Table 90: Results for the semi-supervised ensemble approach. Recall, precision and f1-measure for each of the four semantic classes as well as the total recall. 213

Table 91: Results for the transductive SVM approach. Recall, precision and f1-measure for each of the four semantic classes as well as the total recall. 213

Table 92: Total recall for each of the tested systems. 214

Table 93: Confusion matrix for the experiment which used the semi-supervised ensemble. For example, the most frequent error is that a violence frame is misclassified as search and vice versa. 214

Table 94: Summarization of the results of the applications of the proposed learning framework for several tasks. 217

ANNEX C. REFERENCES

1. Acosta, E., Torres, L., Albiol, A., and Delp, E. *An Automatic Face Detection and Recognition System for Video Indexing Applications*. In Proceedings of the 27th IEEE Conference on Acoustics, Speech and Signal Processing, Vol. 4, Orlando, Florida, USA, 2002, 3644-4647.

2. Adams, B. Amir, A., Dorai, C. Ghosal, S., Iyengar, G., Jaimes, A., Lang, C., Lin, C.-Y., Natsev, A., Naphade, M., Neti, C., Nock, H., J., Permuter, H. H, Singh, R., Smith, J. R., Srinivasan, S., Tseng, B. L., Ashwin, T. V., and Zhang, D. *IBM Research TREC-2002 Video Retrieval System*. In NIST Special Publication 500-250: The Tenth Text REtrieval Conference (TREC 2002), http://trec.nist.gov//pubs/trec10/index_track.html#video, visited; August, 24, 2006.

3. Amir, A. Berg, M., Chang, S.-F., Hsu, W., Iyengar, G., Lin, C.-Y., Naphade, M., Natsev, A. P., Neti, C., Nock, H., Smith, J. R., Tseng, B., Wu, Y., and Zhang, D. *IBM Research TRECVID-2003 Video Retrieval System*. TREC Video Retrieval Online Proceedings, (2003), http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, visited: 24th of August, 2006

4. Amir, A., Argillander, J., Berg, M., Chang, S.-F., Franz, M., Hsu, W., Iyengar, G., Kender, J. R., Kennedy, L., Lin, C.-Y., Naphade, M. R., Natsev, A., Smith, J.R., Tesic, J., Wu, G., Yan, R., and Zhang, D. *IBM Research TRECVID-2004 Video Retrieval System*. In: TREC Video Retrieval Online Proceedings, (2004), http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, visited: 14th of June, 2006

5. Amir, A., Argillander, J., Campbell, M., Haubold, A., Iyengar, G., Ebadollahi, S., Kang, F., Naphade, M. R., Natsev, A., Smith, J.R., Tesic, J., and Volkmer, T. *IBM Research TRECVID-2005 Video Retrieval System*. In TREC Video Retrieval Online Proceedings, (2005), http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, visited: 14th of June, 2006.

6. Arijon, D. *Grammar of the Film Language*. Silman-James Press, 1991.

7. Aslandogan, Y. A. and Yu, C. T. *Techniques and Systems for Image and Video Retrieval*. In IEEE Transactions on Knowledge and Data Engineering, Vol. 11, No. 1, 1999, 56-63.

8. Bailer, W., Schallauer, P., and Thallinger, G. *Joanneum Research at TRECVID 2005 – Camera Motion Detection*. In TREC Video Retrieval Online Proceedings, (2005), http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, visited: 14th of June, 2006.

9. Bashir, F. I., Khanvilkar, S., Schonfeld, D., and Khokhar, A. *Multimedia Systems: Content-Based Indexing and Retrieval*. Chapter 6 in "The Electrical Engineering Handbook", ed. Wai Chen, Academic Press, 2004.

10. Bescos, J., Cisneros, G., and Menendez, J.M. *Multidimensional Comparison of Shot Detection Algorithms*. In Proceedings of IEEE International Conference on Image Processing 2002, Volume II, IEEE Press, 2002, 401-404.

11. Bescos, J. *Real-Time Shot Change Detection Over Online MPEG-2 Video*. In IEEE Transactions on Circuits and Systems for Systems and Video Technology, Volume 14, No. 4, 2004, IEEE Press, 2004, 475-484.

12. Bickel, S., Brückner, M, and Scheffer, T. *Discriminative Learning for Differing Training and Test Distributions*. In Proc. of 24th International Conference on Machine Learning, Corvallis, OR, ACM Press, 2007, 81-88.

13. Blum, A. and Mitchell, T. *Combining Labeled and Unlabeled Data with Co-Training*. In Proceedings of the 11th Conference on Computational Learning Theory, Madison, Wisconsin, USA, 1998, 92-100.

14. Bober, M. *MPEG-7 Visual Shape Descriptors*. In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 6, IEEE Press, 2001, 716-719.

15. Boccignone, G., Chianese, A., Moscato, V., and Picariello, A. *Foveated Shot Detection for Video Segmentation*. In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 15, No. 3, IEEE Press, 2005, 365-377.

16. Böhm, C, Berchtold, S., and Keim D. A. *Searching in High-Dimensional Spaces – Index Structures for Improving the Performance of Multimedia Databases*. In ACM Computing Surveys, Vol. 33, No. 3, ACM Press, 2001, 322-373.

17. Bolme, D. S. *Elastic Bunch Graph Matching*. Master Thesis, Colorado State University, 2003.

18. Boreczky, J. S. and Rowe, L. A. *Comparison of Video Shot Boundary Detection Techniques*. In Proceedings of IS&T/SPIE Conf. Storage and Retrieval for Image and Video Databases IV, Vol. SPIE 2670 (1996), 170–179.

19. Bouguet, J.-Y. *Pyramidal Implementation of the Lucas Kanade Feature Tracker*. In OpenCV Documentation, Intel Corporation, Microprocessor Research Labs, 1999.

20. Brown, G. Wyatt, J., Harris, R., and Yao, X. *Diversity Creation Methods: A Survey and Categorisation*. In Information Fusion 6 (2005), Elsevier, 2005, 5-20.

21. Burges, C. *A Tutorial on Support Vector Machines for Pattern Recognition*. In Data Mining and Knowledge Discovery 2, 2, Kluwer Academic Publishers, 1998, 121-167.

22. Chandrashekhara, A., Feng, H. M. and Chua, T.-S. *Temporal Multi-Resolution Framework for Shot Boundary Detection and Keyframe Extraction*. In Proceedings of The Eleventh Text Retrieval Conference (TREC 2002), 492-496, 2002. Onlince proceedings also available at: http://trec.nist.gov//pubs/trec11/index.track.html#video.

23. Chang, C.-C. and Lin, C.-J. *LIBSVM - A Library for Support Vector Machines*. 2001. Software available at: www.csie.ntu.edu.tw/~cjlin/libsvm/, visited on 30th August, 2006.

24. Chang, S.-F., Sikora, T., and Puri, A. *Overview of the MPEG-7 Standard*. In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 6, 2001, 688-695.

25. Chapelle, O., Schölkopf, B., and Zien, A. (eds.) *Semi-Supervised Learning*. MIT Press, Cambridge, Massachusetts, 2006.

26. Chua, T.-S., Kankanhalli, M., and Lin, Y. *A General Framework for Video Segmentation Based on Temporal Multi-Resolution Analysis*. In Proc. of International Workshop on Advanced Image Technology, Fujisawa, Japan 2000, 119-124.

27. Chua, T.-S., Feng, H.M., and Anantharamu, C. *An Unified Framework for Shot Boundary Detection via Active Learning*. In Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing 2003, Hong Kong, Band II, IEEE Press, 2003, 845-848.

28. Cortes, C. and Vapnik, V. *Support Vector Networks*. In Machine Learning, Vol. 20, No. 3, 1995, 273-297.

29. Cronen-Townsend, S., Zhou, Y., and Croft, W. B. *Predicting Query Performance.* In Proceedings of the 25th Annual International ACM SIGIR conference on Research and Development in Information Retrieval (SIGIR 2002), ACM Press, Tampere, Finland, 2002, 299-306.

30. Cronen-Townsend, S., Zhou, Y., and Croft, W. B. *Precision Prediction based on Ranked List Coherence.* In Information Retrieval, Volume 9, Issue 6, Kluwer Academic Publishers, Hingham, MA, USA, 2006, 723-755.

31. Dante, A. and Brookes, M. *Precise Real-Time Outlier Removal from Motion Vector Fields For 3D Reconstruction.* In Proc. of IEEE International Conference on Image Processing, Vol. 1, Barcelona, 2003, 393-396.

32. Dempster, A. P., Laird, N. M., and Rubin, D. B. *Maximum Likelyhood From Incomplete Data via the EM Algorithm.* In Journal of the Royal Statistical Society, Series B, 39(1), 1977, 1-31.

33. Dorai, C. and Venkatesh, S. *MEDIA COMPUTING – Computational Media Aesthetics.* Kluwer Academic Publishers, Boston, 2002.

34. Eickeler, S., Wallhoff, F., Iurgel, U., and Rigoll, G. *Content-Based Indexing of Images and Video Using Face Detection and Recognition Methods.* In Proc. of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vol. III, Salt Lake City, Utah, USA, 2001, 1505-1508.

35. Ester, M. and Sander, J. *Knowledge Discovery in Databases.* Springer Verlag, Berlin, 2000.

36. Ewerth, R. and Freisleben, B. *Frame difference normalization: An approach to reduce error rates of cut detection algorithms for MPEG videos.* In Proc. of the IEEE International Conference on Image Processing, Vol. 2, Barcelona, IEEE Press, 2003, 1009-1012.

37. Ewerth, R. and Freisleben, B. *Improving Cut Detection in MPEG Videos by GoP-Oriented Frame Difference Normalization.* In Proceedings of the 17th International Conference on Pattern Recognition, Cambridge (UK), Volume 2, IEEE Press, 2004, 807-810.

38. Ewerth, R., Gllavata, J., Gollnick, M., Mansouri, F., Papalilo, E., Sennert, R., Wagner, J., Freisleben, B., and Grauer, M. *Methoden und Werkzeuge zur rechnergestützten medienwissenschaftlichen Analyse.* In: Siegener Periodicum zur Internationalen Empirischen Literaturwissenschaft, 20, H. 2, 2003, 306-320.

39. Ewerth, R., Schwalb, M., Tessmann, P., and Freisleben, B. *Estimation of Arbitrary Camera Motion in MPEG Videos*. In Proceedings of the 17th International Conference on Pattern Recognition, Vol. I, Cambridge, UK, IEEE Press, 2004, 512-515.

40. Ewerth, R. and Freisleben, B. *Video Cut Detection without Thresholds*. In Proceedings of the 11th International Workshop on Signals, Systems and Image Processing, Poznan, Poland, 2004, 227-230.

41. Ewerth, R., Friese, T., Grube, M., and Freisleben, B. *Grid Services for Distributed Video Cut Detection*. In Proceedings of the International Symposium on Multimedia Software Engineering, Miami (USA), 2004, 164-168.

42. Ewerth, R., Beringer, C., Kopp, T., Niebergall, M., Stadelmann, T., and Freisleben, B. *University of Marburg at TRECVID 2005: Shot Boundary Detection and Camera Motion Estimation Results*. In Online Proceedings of TRECVID Conference Series 2005: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html

43. Ewerth, R. and Freisleben, B. *Self-Supervised Learning for Robust Video Indexing*. In Proceedings of the IEEE Conference on Multimedia & Expo 2006, Toronto, 2006, 1749-1752.

44. Ewerth, R., Mühling, M., and Freisleben, B. *Self-Supervised Learning of Face Appearances in TV Casts and Movies*. In Proceedings of the IEEE Symposium on Multimedia, San Diego, CA, USA, 2006, 78-85.

45. Ewerth, R., Mühling, M., Stadelmann, T., Agel, B., Seiler, D., and Freisleben, B. *University of Marburg at TRECVID 2006: Shot Boundary Detection and Rushes Task Results*. In Online Proceedings of TRECVID Conference Series 2006: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, 2006

46. Ewerth, R. and Freisleben, B. *Computerunterstützte Filmanalyse mit Videana*. In: Augen-Blick: Hefte zur Medienwissenschaft, Heft 39, Schüren-Verlag, Marburg, 2007, 54-66.

47. Ewerth, R., Mühling, M., and Freisleben, B. *Self-Supervised Learning of Face Appearances in TV Casts and Movies*. In International Journal on Semantic Computing, World Scientific, 2007, June, 185-204.

48. Ewerth, R., and Freisleben, B. *Semi-Supervised Learning for Semantic Video Retrieval.* In Proceedings of ACM International Conference on Image and Video Retrieval, Amsterdam, ACM Press, 2007, 154-161.

49. Ewerth, R. and Freisleben, B. *Adapting Appearance Models of Semantic Concepts to a Particular Video via Transductive Learning.* In Proceedings of 9th ACM SIGMM Int'l Workshop on Multimedia Information Retrieval, in conjunction with ACM Multimedia 2007, Augsburg, Germany, 2007, 187-196.

50. Everingham, M. R. and Zisserman, A. *Automated Visual Identification of Characters in Situation Comedies.* In Proceedings of the 17th International Conference on Pattern Recognition (ICPR), Cambridge, UK, 2004, 983-986.

51. Everingham, M. R. and Zisserman, A. *Automated Person Identification in Video.* In Proceedings of the 3rd International Conference on Image and Video Retrieval, (CIVR), Dublin, Ireland, 2004, 289-298.

52. Fitzgibbon, A. and Zisserman, A. *On Affine Invariant Clustering and Automatic Cast Listing in Movies.* In Proceedings of the 7th European Conference on Computer Vision, Vol. 3, Copenhagen, Denmark, Springer-Verlag, 2002, 304-320.

53. Fitzgibbon, A. and Zisserman, A. *Joint Manifold Distance: A New Approach to Appearance Based Clustering.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Vol. 1, Madison, WI, USA, 2003, 16-26.

54. Fritz, M., Kruijff, G.-J. M., and Schiele, B. *Cross-Modal Learning of Visual Categories using Different Levels of Supervision.* In Proceedings of International Conference on Computer Vision Systems, Bielefeld, Germany, 2007.

55. Freund, Y. and Schapire, R. E. *A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting.* In Journal of Computer and System Sciences, 55(1), 1997, 119-139.

56. Gao, X., and Tang, X. *Unsupervised Video-Shot Segmentation and Model-Free Anchorperson Detection for News Video Story Parsing.* In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 12, No. 9, 2002, 765-776.

57. Gargi, U., Kasturi, R., and Antani, S. *Performance Characterization and Comparison of Video Indexing Algorithms*. In Proc. of IEEE Conference on Computer Vision and Pattern Recognition, 1998, 559-565.

58. Gargi, U., Kasturi, R., and Strayer, S. H. *Performance Characterization of Video-Shot-Change Detection Methods*. In IEEE Transaction on Circuits and Systems for Video Technology, Vol. 10, No. 1, 2000, 1-13.

59. Gevers, T. and Smeulders, A.W. M. *Image Search Engines: An Overview*. In G. Medioni and S. B. Kang (eds.), *Emerging Topics in Computer Vision*. Prentice Hall, 2004.

60. Gllavata, J., Ewerth, R., and Freisleben, B. *Finding Text in Images via Local Thresholding*. In Proceedings of the IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), Darmstadt, IEEE Press, 2003, 539-542.

61. Gllavata, J., Ewerth, R., and Freisleben, B. *A Robust Algorithm for Text Detection in Images*. In: Proceedings of the 3rd International Symposium on Image and Signal Processing and Analysis, Rome, IEEE Press, 2003, 611-616.

62. Gllavata, J., Ewerth, R., and Freisleben, B. *Text Detection in Images Based on Unsupervised Classification of High-Frequency Wavelet Coefficients*. In Proceedings of 17th International Conference on Pattern Recognition, Vol. 1, Cambridge (UK), IEEE Press, 2004, 425-428.

63. Gllavata, J., Ewerth, R., and Freisleben, B. *Tracking Text in MPEG Videos*. In Proceedings of ACM Multimedia 2004, New York, ACM Press, 2004, 240-243.

64. Gllavata, J., Ewerth, R., and Freisleben, B. *A Text Detection, Localization and Segmentation System for OCR in Images*. In Proceedings of the 6th IEEE International Symposium on Multimedia Software Engineering, Miami (USA), IEEE Press, 2004, 310-317.

65. Gllavata, J. and Freisleben, B. *Adaptive Fuzzy Text Segmentation in Images with Complex Backgrounds using Color and Texture*. In Proceedings of the International Conference on Computer Analysis of Images and Patterns (CAIP '05), Springer Verlag, Paris, France, 2005, 756-765.

66. Gllavata, J., Qeli, E., and Freisleben, B. *Detecting Text in Videos Using Fuzzy Clustering Ensembles*. In Proceedings of the IEEE International Symposium on Multimedia (ISM '06), IEEE Press, San Diego, USA, 2006, 283-290.

67. Gllavata, J., Ewerth, R., Stefi, T., and Freisleben, B. *Unsupervised Text Segmentation Using Color and Wavelet Features*. In Proceedings of the 3rd International Conference on Image and Video Retrieval 2004, Lecture Notes on Computer Science LNCS 3115, Dublin, Springer-Verlag, 2004, 216-224.

68. Günsel, B., Ferman, A. M., Tekalp, A. M. *Temporal Video Segmentation Using an Unsupervised Clustering and Semantic Object Tracking*. In SPIE Journal of Electronic Imaging 7 (3), 1998, 592-604.

69. Hanjalic, A. *Shot Boundary Detection: Unraveled and Resolved?*. In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 12, 2002, 533-544.

70. Hauptmann, A. G. and Christel, M. G. *Successful Approaches in the TREC Video Retrieval Evaluations*. In Proceedings of the ACM Conference on Multimedia 2004, New York, USA, ACM Press, 2004, 668-675.

71. Hauptmann, A., Lin, W-H., and Yan, R. *How Many High-level Concepts Will Fill the Semantic Gap in News Video Retrieval?*. In Proceedings of ACM International Conference on Image and Video Retrieval, Amsterdam, ACM Press, 2007, 627-634.

72. He, B. and Ounis, I. *Inferring Query Performance Using Pre-Retrieval Predictors*. In 11th Proceedings of International Conference String Processing and Information Retrieval (SPIRE 2004), Lecture Notes in Computer Science 3246, Padova, Italy, Springer-Verlag, 2004, 43-54.

73. Hickethier, K. *Film- und Fernsehanalyse*. 3. Auflage, Verlag J. B. Metzler, Stuttgart/Weimar, 2001.

74. Ho, T. K. *The random subspace method for constructing decision forests*. In IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 20, Issue 8 (Aug. 1998), 832-844.

75. Hu, Y. and Loizou, P. *Subjective Comparison of Speech Enhancement Algorithms*. In Proceedings of the International Conference on Acoustics, Speech, and Signal Processing, Vol. 1, Toulouse, France, 2006, 153-156.

76. Huang, C. Ai, H., Li, Y., and Lao, S. *Vector Boosting for Rotation Invariant Multi-View Face Detection*. In Proceedings of the Tenth International Conference on Computer Vision, Vol. 1, Bejing, China, 2005, 446-453.

77. Intel Open Source Computer Vision Library, www.intel.com/technology/computing/opencv/index.htm

78. Jacobs, A., Miene, A., Ioannidis, G. T., and Herzog, O. *Automatic shot boundary detection combining color, edge, and motion features of adjacent frames.* In Online Proceedings of TRECVID Conference Series 2004, Gaithersburg, Maryland, USA, http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, 2004.

79. Jain, A. K., Duin, R., and Mao, J. *Statistical Pattern Recognition: A Review.* In IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 1, 2000, 4-37.

80. Jarre, F. and Stoer, J. *Optimierung.* Springer-Verlag Berlin. 2004.

81. Jeannin, S. and Mory, B. *Video Motion Representation for Improved Content Access.* In IEEE Transactions on Consumer Electronics, Vol. 46, No. 3, 2000, 645-655.

82. Joachims T. *Transductive Inference for Text Classification using Support Vector Machines.* In Proceedings of 16th International Conference on Machine Learning (ICML), Bled, Slovenia, 1999, 200-209.

83. Joly, P. and Kim, H.-K. *Efficient Automatic Analysis of Camera Work and Microsegmentation of Video Using Spatiotemporal Images.* In Signal Processing: Image Communication 8, Elsevier Science Ltd., 1996, 295-307.

84. Kaufmann, L. and Rousseuw P. J. *Finding Groups in Data: An Introduction to Cluster Analysis.* John Wiley & Sons, 1990.

85. Keerthi S., Shevade S., Bhattacharyya C., and Murthy K. *Improvements to Platt's SMO Algorithm for SVM Classifier Design.* In Neural Computation, Vol. 13, 2001, 637-649.

86. Kim, J.-G., Chang, H. S., Kim, J., and Kim, H.-M. *Efficient Camera Motion Characterization for MPEG Video Indexing.* In Proceedings. of IEEE International Conference on Multimedia and Expo, New York, USA, Vol. 2, 2000, 1171 -1174.

87. Koprinska, I. and Carrato, S. *Temporal Video Segmentation: A Survey.* In Signal Processing: Image Communication 16, Elsevier Sc. Ltd., 2001, 477-500.

88. Korte, H. *Einführung in die Systematische Filmanalyse.* 2 Auflage, Erich Schmidt Verlag, 2004.

89. Kreyß, J., Röper, M., Alshuth, P., Hermes, T., and Herzog, O. *Video Retrieval by Still Image Analysis with ImageMiner.* In Proceedings of IS&T/SPIE Symposium on Electronical Imaging Sciene & Technology (Storage and Retrieval for Images and Video Databases), San Jose, CA, USA, 1997, 236–247.

90. Kuncheva, L. I., Whitaker, C. J., Shipp, C. A., and Duin, R. P. W. *Limits on the Majority Vote Accuracy in Classifier Fusion.* In Pattern Analysis and Applications, 6, 2003, Springer-Verlag, 22-31.

91. Kuo, T. C. T., and Chen, A. L.P. *A Mask Matching Approach for Video Segmentation on Compressed Data.* In Information Sciences 141, Elsevier Ltd., 2002, 169-191.

92. Lefèvre, S., Hollera, J., and Vincent, N. *A Review of Real-time Segmentation of Uncompressed Video Sequences for Content-Based Search and Retrieval.* In Real-Time Imaging, Elsevier Sc. Ltd., Volume 9, Issue 1, 2003, 73-98.

93. Lelescu, D., and Schonfeld, D. *Statistical Sequential Analysis for Real-Time Video Scene Change Detection on Compressed Multimedia Bitstream.* In IEEE Transactions on Multimedia, Vol. 5, No. 1, IEEE Press, 2003, 106-117.

94. Li, D. and Sethi, I. *MPEG Developing Classes.* http://www.cs.wayne.edu/~dil/research/mdc/docs.

95. Lieb, D. Lookingbill, A. and Thrun, S. *Adaptive Road Following using Self-Supervised Learning and Reverse Optical Flow.* In Online Proceedings of Robotics: Sciencs and Systems, Cambridge, USA, http://www.roboticsproceedings.org/index.html, 2005.

96. Lienhart, R. *Comparison of Automatic Shot Boundary Detection Algorithms.* In Image and Video Processing VII 1999, SPIE Proc. Vol. 3656-29, 1999, 290-301.

97. Lienhart, R. *Reliable Transition Detection in Videos: A Survey and Practitioner's Guide.* In International Journal of Image and Graphics, Vol. 3., 2001, 469-486.

98. Lienhart, R. *Reliable Dissolve Detection.* In Storage and Retrieval for Media Databases 2001, Proceedings of SPIE 4315, 2001, 219-230.

99. Lienhart, R., Liang, L., and Kuranov, A. *A Detector Tree of Boosted Classifiers for Real-time Object Detection and Tracking*. In Proceedings of IEEE International Conference on Multimedia and Expo (ICME), 2003, Vol. 2, Baltimore, Maryland, USA, 2003, 277-280.

100. Linguistic Data Consortium. *The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus*. 1990, available online (9th of February, 2007): http://www.ldc.upenn.edu/Catalog/readme_files/timit.readme.html

101. Liu, Z., Gibbon, D., Zavesky, E., Shahraray, B., and Haffner, P. *AT&T Research at TRECVID 2006*. In TREC Video Retrieval Online Proceedings, available online (14th of August, 2007): http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, 2006.

102. Lu, L., Zhang, H.-J., and Li, S. Z. *Content-Based Audio Classification and Segmentation by Using Support Vector Machines*. In Multimedia Systems 8, Springer-Verlag, 2003, 482–492.

103. Lucas, B. and Kanade, T. *An Iterative Image Registration Technique with an Application to Stereo Vision*. In Proceedings of the 7th International Joint Conference on Artificial Intelligence, Vancouver, Canada, 1981, 674-679.

104. Luttermann, H., Freisleben, B., Grauer, M., Kamphusmann, T., Kelter, U., Merten, U., Rößling, G., Unger, T., and Waldhans, J. *Mediana: Eine Workbench zur rechnergestützten Analyse von Mediendaten*. In Wirtschaftsinformatik, 44, Nr. 1, 2002, 41-51.

105. MacQueen, J. B. *Some Methods for classification and Analysis of Multivariate Observations*. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, 1967, 281-297.

106. Manjunath, B. S., Ohm, J.-R., Vasudevan, V., and Yamada, A. *Color and Texture Descriptors*. In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 11, No. 6, June 2001, 703-715.

107. Margineatu, D. D. and Dietterich, T. G. *Pruning Adaptive Boosting*. In Proceedings of the International Conference on Machine Learning 1997, Morgan Kaufmann, 1997, 211-218.

108. Markkula, M. and Sormunen, E. *Video Needs at the Different Stages of Television Program Making Process*. In Proceedings of the 1st International Conference on Interaction of Context, Copenhagen, Denmark, ACM Press, 2006, 111-118.

109. Martinez, J. M. *MPEG–7 Overview.* Technical Report N4980, ISO/IEC JTC1/SC29/WG11, Klagenfurt, AT, 2002.

110. Mathiak, K. and Weber, R. *Towards Brain Correlates of Natural Behavior: fMRI During Violent Video Games.* Human Brain Mapping, Vol. 27, 2006, 957-962.

111. MPEG-1: ISO/IEC Draft International Standard (DIS) 11172: *Information technology - Coding of moving pictures and associated audio for digital storage media up to about 1,5 Mbit/s (MPEG).* International Organization for Standardization, Geneva, 1992.

112. MPEG-2: ISO/IEC 13818: *Information technology - Generic coding of moving pictures and associated audio information.* International Organization for Standardization, 1993.

113. MPEG-4: ISO/IEC 14496: *Information technology - Coding of audio-visual objects.* International Organization for Standardization, 1999.

114. MPEG-7: ISO/IEC 15938: *Information Technology - Multimedia Content Description Interface Part 2: Description Definition Language.* International Organization for Standardization, 2002.

115. Mühling, M., Ewerth, R., M., Stadelmann, T., Sih, B., Zöfel, C., and Freisleben, B. *University of Marburg at TRECVID 2007: Shot Boundary Detection and High-Level Feature Extraction.* In Online Proceedings of TRECVID Conference Series 2007: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, 2007.

116. Mühling, M., Ewerth, R., Stadelmann, T., Freisleben, B., Weber, R., Mathiak, K. *Semantic Video Analysis for Psychological Research on Violence in Computer Games.* In Proceedings of ACM International Conference on Image and Video Retrieval, Amsterdam, ACM Press, 2007, 611-618.

117. Naphade, M. R. and Smith, J. R. *On the Detection of Semantic Concepts at TRECVID.* In Proceedings of the ACM Conference on Multimedia, 2004, New York, ACM Press, 2004, 660-667.

118. Nelder, J.A. and Mead, R. *A Simplex Method for Funtion Minimization.* In Computer Journal, 7, 1965, 308-313.

119. Niemann, H. *Klassifikation von Mustern*. Lehrstuhl für Mustererkennung, Universität Erlangen-Nürnberg, 2nd edition, http://www5.informatik.uni-erlangen.de/Personen/niemann/klassifikation-von-mustern/m00links.html, 2003.

120. Nigam, K., Mccallum, A. K., Thrun, S. and Mitchell, T. *Text Classification from Labeled and Unlabeled Documents Using EM*. In Machine Learning, Springer Netherlands, Volume 39, No. 2-3, 2000, 103-134.

121. Nigam, K. and Ghani, R. *Analyzing the Effectiveness and Applicability of Co-training*. In Proceedings of the 9th International Conference on Information and Knowledge Management, McLean, Virgina, USA, 2000, 86-93.

122. Ngo, C.-W., Pong, T.-C., and Zhang, H.-J. *Motion Analysis and Segmentation Through Spatio-Temporal Slices Processing*. In IEEE Transactions on Image Processing, Vol. 12, No. 3, 2003, 341-355.

123. Oudot, L., Prevost, L. Moises, A., and Milgram, M. *Self-Supervised Writer Adaption using Perceptive Concepts: Application to On-Line Text Recognition*. In Proc. of the 17th International Conf. on Pattern Recognition, Vol. II, Cambridge, UK, 2004, 598-601.

124. Over, P., Ianeva, T., Kraaijz, W., and Smeaton, Alan F. *TRECVID 2005 - An Overview*. In TREC Video Retrieval Online Proceedings (14th of June, 2006): http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, 2005.

125. Park, J.-I., Inoue, S., and Iwadate, Y. *Estimating Camera Parameters from Motion Vectors of Digital Video*. In Proc. of IEEE Second Workshop on Multimedia Signal Processing, Redondo Beach, USA, 1998, 105-110.

126. Petersohn, C. *Wipe Shot Boundary Determination*. In Proceedings of IS&T/SPIE Electronic Imaging 2005, Storage and Retrieval Methods and Applications for Multimedia, San Jose, CA, USA, 2005, 337-346.

127. Philips, P. J., Moon, H., Rizvi, S. A., and Rauss, P. J. *The FERET Evaluation Methodology for Face-Recognition Algorithms*. In IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 10, 2000, 1090-1104.

128. Platt, J. *Fast Training of Support Vector Machines using Sequential Minimal Optimization*. Advances in Kernel Methods - Support Vector Learning, MIT Press, 1999, 185-208.

129. Raytchev, B. and Murase, H. *Unsupervised Face Recognition from Image Sequences.* In Proc. of IEEE International Conference on Image Processing, Vol. 1, Thessaloniki, Greece, 2001, 1042-1045.

130. Raytchev, B. and Murase, H. *Unsupervised Face Recognition from Image Sequences Based on Clustering with Attraction and Repulsion.* In Proceedings of International Conference on Computer Vision and Pattern Recognition (CVPR) 2001, Volume II, Hawaii, USA, 2001, 25-30.

131. Raytchev, B. and Murase, H. *VQ-Faces-Unsupervised Face Recognition from Image Sequences.* In Proc. of IEEE International Conference on Image Processing, Vol. 1, Rochester, New York, USA, Greece, 2002, 809-812.

132. Rosenberg, C., Hebert, M., and Schneiderman, H. *Semi-Supervised Self-Training of Object Detection Methods.* In Proceedings of the 7th IEEE Workshop on Applications for Computer Vision, Vol. 1, Breckenridge, Co, USA, 29-36.

133. Sadlier, D. and O'Connor, N. *Event Detection in Field Sports Video using Audio-visual Features and a Support Vector Machine.* In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 15 (10), 2005, 1225-1233.

134. Satoh, S. *Towards Actor/Actress Identification in Drama Videos.* In Proceedings of the Seventh ACM International Conference on Multimedia (ACM MM 1999), Orlando, Florida, USA, 1999, 75 - 78.

135. Schneiderman, H., and Kanade, T. *Object Detection Using the Statistics of Parts.* In International Journal of Computer Vision, Springer-Verlag, Volume 56 (3), February 2004, 151-177.

136. Schober, J.-P., Hermes, T., and Herzog, O. *Picturefinder: Description Logics for Semantic Image Retrieval.* In Proceedings of IEEE International Conference on Multimedia and Expo, 2005. ICME 2005, Amsterdam, 2005, 1571–1574.

137. Seeger, M. *A Taxonomy for Semi-Supervised Learning Methods.* In Chapelle, Olivier, Schölkopf, Benhard, and Zien, Alexander (eds.). *Semi-Supervised Learning.* MIT Press, 2006, 17-32.

138. Seeman, E., Fritz, M., and Schiele, B. *Towards Robust Pedestrian Detection in Crowded Image Sequences.* In Proc. of International Conference on Computer Vision and Pattern Recognition, Minneapolis, 2007, 1-8.

139. Shen, K., and Delp, E. J. *A Fast Algorithm for Video Parsing Using MPEG Compressed Sequences.* In Proc. of IEEE International Conf. on Image Processing 1995, Washington, DC (1995) 252-255.

140. Smeaton, A. and Over, P. *TRECVID 2005: Shot Boundary Detection Task Overview.* In Online Proceedings of TRECVID Conference Series 2005 (14th of February, 2006): http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, 2005.

141. Smeaton, A. and Over, P. *TRECVID 2006: Shot Boundary Detection Task Overview.* In Online Proceedings of TRECVID Conference Series 2006 (19th of February, 2008): http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html, 2006.

142. Smith, J. R., Naphade, M. R., and Natsev, A. *Multimedia Semantic Indexing Using Model Vectors.* In Proceedings of the IEEE International Conference on Multimedia & Expo 2003, Volume 2, Baltimore, Maryland, USA, 2003, 445-448.

143. Smith, J. R., Naphade, M. R., Natsev, A, and Tesic, J. *Multimedia Research Challenges for Industry.* In Proceedings of the International Conference on Image and Video Retrieval, Lecture Notes on Computer Science Vol. 3568, Springer-Verlag, Singapore, 2005, 28-37.

144. Smith, J. R., Srinivasan, S., Amir, A., Basu, S., Iyengary, G., Lin, C.-Y., Naphade, M., Ponceleon, D., and Tseng, B. *Integrating Features, Models, and Semantics for TREC Video Retrieval.* In NIST Special Publication 500-250: The Tenth Text REtrieval Conference (TREC 2001), (24th of August, 2006): http://trec.nist.gov//pubs/trec10/index_track.html#video, 2006.

145. Snoek, C. *Camera Distance Classification: Indexing Video Shots based on Visual Features.* Master of Science Thesis, University of Amsterdam, October 2000.

146. Snoek, C. G. M. and Worring, M. *Multimodal Video Indexing: A Review of the State-of-the-art.* In Multimedia Tools and Applications, 25(1). Springer-Verlag, 2005, 5-35.

147. Snoek, C. G. M., Worring, M., Geusebroek, J.-M., Koelma, D. C., Seinstra, F. J., and Smeulders, A. W. M. *The Semantic Pathfinder: Using an Authoring Metaphor for Generic Multimedia Indexing.* In IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 28 (10), 2006, 1678-1689.

148. Snoek, C. G. M., Worring, M., van Gemert, J. C., Geusebroek, J.-M., and Smeulders, A. W.M. *The Challenge Problem for Automated Detection of 101 Semantic Concepts in Multimedia.* In Proceedings of ACM Multimedia, Santa Barbara, USA, ACM Press, 2006, 421-430.

149. Snoek, C. G. M., Worring, M., Koelma, D. C. and Smeulders, A. W. M. *A Learned Lexicon-Driven Paradigm for Interactive Video Retrieval.* In IEEE Transactions on Multimedia, Vol. 9, Issue 2, Feb. 2007, 280-292.

150. Sochman, J. and Malas, J. *AdaBoost with Totally Corrective Updates for Fast Face Detection.* In Proceedings of the Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004, Seoul, Korea, 445-450.

151. Srinivasan, M. V., Venkatesh, S., and Hosie, R. *Qualitative Estimation of Camera Motion Parameters from Video Sequences.* In Pattern Recognition, Vol. 30, No. 4, Elsevier Science Ltd., 1997, 593-606.

152. Stadelmann, T. and Freisleben, B. *Fast and Robust Speaker Clustering Using the Earth Mover's Distance and MIXMAX Models.* In Proceedings of the 31st International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Toulouse, France , Volume I, IEEE Press, 2006, 989-992.

153. Tahaghoghi, S. M. M., Thom, J. A., Williams, H. E., and Volkmer, T. *Video Cut Detection Using Frame Windows.* In Proc. of the Twenty-Eighth Australasian Computer Science Conf., Vol. 38, Newcastle, Australia, 2005, 193-199.

154. Tan, Y.-P. Saur, D. D. Kulkarni, S. R., and Ramadge, P. J. *Rapid Estimation of Camera Motion from Compressed Video ith Application to Video annotation.* In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 10, No. 1, 2000, 133-146.

155. Taskiran, C., Chen, J.-Y., Albiol, A., Bouman, C. A, and Delp, E. J. *A Compressed Video Database Structured for Active Browsing and Search.* In Proc. of IEEE International Conference on Image Processing, Vol. 3, Chicago, 1998, 133-137.

156. Taskiran, C., Chen, J.-Y., Albiol, A., Torres, L., Bouman, C. A, and Delp, E. J. *ViBE: A Compressed Video Database Structured for Active Browsing and Search.* In IEEE Transactions on Multimedia, Vol. 6, Issue 1, Feb. 2004, 103-118.

157. Tekalp, A. M. *What can Video Analysis Do for MPEG Standards?* In Proc. of 8th Int'l Workshop of Visual Content Processing and Representation, LNCS 2849, Madrid, Spain, 2003, 3-5.

158. Terrillon, J.-C., Shirazi, M. N., Fukamachi, H., and Akamatsu, S. *Comparative Performance of Different Skin Chrominance Models and Chrominance Spaces for the Automatic Detection of Human Faces in Color Images*. In Proceedings of the International Conference on Face and Gesture Recognition, Grenoble, France, 2000, 54-61.

159. Tong, X., Liu Q., Duan, L., Lu, H., Xu C., and Tian, Q. *A Unified Framework for Semantic Shot Representation of Sports Video*. In Proceedings of the 7th ACM SIGMM International Workshop on Multimedia Information Retrieval, Hilton, Singapore, 2005, 127-134.

160. Torres, L., Lorente, L., and Vilà, J. *Face Recognition Using Self-Eigenfaces*. In Proceedings of the International Symposium on Image/Video Communications Over Fixed and Mobile Networks, Rabat, Marokko, 2000, 44-47.

161. TRECVID: *TREC Video Retrieval Evaluation*. At (6th of September, 2007): http://www-nlpir.nist.gov/projects/t01v.

162. TRECVID: *Evaluation software for shot boundary detection task*. At (August 29th, 2006): http://www-nlpir.nist.gov/projects/trecvid/trecvid.tools/shot.boundary.evaluation/.

163. Truong, B. T., Venkatesh S., and Chitra D. *Scene Extraction in Motion Pictures*. In IEEE Transactions on Circuits and Systems for Video Technology: Special issue on Multimedia Content Description. Vol. 15, No. 1, 2003, 5-15.

164. Truong, B. T., Dorai, C., and Venkatesh, S. *New Enhancements to Cut, Fade, and Dissolve Detection Processes in Video Segmentation*. In Proceedings of ACM Multimedia 2000, 219-227.

165. van Gemert, J., Geusebroek, J., Veenman, C., Snoek, C., and Smeulders, A. *Robust Scene Categorization by Learning Image Statistics in Context*. In Proceedings of Int'l Workshop on Semantic Learning Applications in Multimedia, in conjunction with CVPR'06, New York, USA, 2006, 105-112.

166. Vapnik, V. *Transductive Inference and Semi-Supervised Learning*. In (Chapelle, O., Schölkopf, B., and Zien, A. (eds.)): Semi-Supervised Learning. MIT Press, 2006, 453-472.

167. Viola, P. and Jones, M. *Robust Real-Time Face Detection.* In International Journal of Computer Vision, Volume 57 (2), (2004), Kluwer Academic Publishers, 2004, 137–154.

168. Vogel, J. and Schiele, B. *On Performance Characterization and Optimization for Image Retrieval.* In Proceedings of European Conference on Computer Vision 2002, Vol. 4, Lecture Notes on Computer Science (LNCS) 2353, Springer-Verlag, Copenhagen, Denmark, 2002, 49-63.

169. Vogel, J. and Schiele, B. *Performance Evaluation and Optimization for Content-Based Image Retrieval.* In Pattern Recognition 39 (2006), Elsevier Ltd., 2006, 897-909.

170. Volkmer, T. and Tahaghoghi, S. M. M. *RMIT University Video Shot Boundary Detection at TRECVID 2005.* In TREC Video Retrieval Evaluation Online Proceedings, 2004, at (August 29th, 2006): http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html

171. Watanabe, S. *Pattern Recognition: Human and Mechanical.* New York, Wiley, 1985.

172. Weber, R., Ritterfeld, U., and Mathiak, K. *Does Playing Violent Video Games Induce Aggression? Empirical Evidence of a Functional Magnetic Resonance Imaging Study.* Media Psychology, Vol. 8, 2006, 39-60.

173. Westerman, U. and Klas, W. *An Analysis of XML Database Solutions for the Management of MPEG-7 Media Descriptors.* In ACM Computing Surveys, Vol. 35, No. 4, ACM Press, 2003, 331-373.

174. Wiskott, L., Fellous, J.-M., Krüger, N., and von der Malsburg, C. *Face Recognition by Elastic Graph Bunch Matching.* In IEEE Transactions on Pattern Analysis and Machine, Volume 19 , Issue 7 (1997), IEEE Press, 1997, 775-779.

175. Witten, Ian H. and Frank, Eibe. *Data Mining - Practical Machine Learning Tools and Techniques.* Morgan Kaufmann Publishers, Elsevier Inc., San Francisco, 2005.

176. Wolpert, D.H. *Stacked Generalization.* Neural Networks, Pergamon Press Vol. 5, 1992, 241-259.

177. Wu, X. *Incorporating Large Unlabeled Data to Enhance EM Classification.* In Journal of Intelligent Information Systems, 26, Springer-Verlag, 2006, 211-226.

178. Wu, W. Chen, D. and Yang, J. *Integrating Co-Training and Recognition for Text Detection.* In Proc. of IEEE International Conf. on Multimedia and Expo 2005, Amsterdam, Netherlands, 2005, 1166-1169.

179. Wu, Y. and Huang, T. S. *Self-Supervised Learning for Visual Tracking and Recognition of Human Hand.* In Proc. of the 17th National Conference on Artificial Intelligence, Austin, USA, 2000, 243-248.

180. Wu, J., Ding, D., Hua, X.-S., and Zhang, B. *Tracking Concept Drifting with an Online-Optimized Incremental Learning Framework.* In Proc. of the 7th ACM Int'l Workshop on Multimedia Information Retrieval, Singapore, 2005, 33-40.

181. Xiao, R., Li, M.-J., Zhang, H.-J. *Robust Multipose Face Detection in Images.* In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 14, No. 1, (2004), IEEE Press, 2004, 31-41.

182. Xu, H. and Chua, T. *Fusion of AV Features and External Information Sources for Event Detection in Team Sports Video.* In ACM Transactions on Multimedia Computing, Communications, and Applications, Vol. 2 (1), 2006, 44-67.

183. Yan, R. and Hauptmann, A. G. *Co-Retrieval: A Boosted Reranking Approach for Video Retrieval.* In Proc. of the Int'l Conf. on Image and Video Retrieval, Dublin, Ireland, 2004, 60-69.

184. Yan, R. and Naphade, M. *Co-Training Non-Robust Classifiers for Video Semantic Concept Detection.* In Proceedings of the IEEE International Conference on Image Processing 2005, Vol. 1, Singapore, 1205-1208.

185. Yan, R. and Naphade, M. *Semi-Supervised Cross Feature Learning for Semantic Concept Detection in Videos.* In Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 2005, Vol. 1, San Diego, CA, USA, 657-663.

186. Yang, M-H., Roth, D., and Ahuja. *A SNoW-based Face Detector.* In Advances in Neural Information Processing Systems 12, 1999, 53, 855-861.

187. Yang, M.-H., Kriegman, D. J., and Ahuja, N. *Detecting Faces in Images: A Survey.* In IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 1 (2002), ÍEEE Press, 2002, 34-58.

188. Yeo, B. and Liu, B. *On the extraction of DC sequence from MPEG compressed video.* In Proceedings of the IEEE International Conference on Image Processing, Volume 2, Washington, DC, 1995, 2260-2263.

189. Yeo, B. L. and Liu, B. *Rapid Scene Analysis on Compressed Video.* In IEEE Transactions on Circuits and Systems for Video Technology, Vol. 5, No. 6, 1995, 533-544.

190. Yoshitaka, A. and Ichikawa, T. *A survey on content-based retrieval for multimedia databases.* In IEEE Transactions on Knowledge and Data Engineering, Vol. 11, No. 1, 1999, 81-93.

191. Yuan, J., Li, J., Lin, F., and Zhang, B. A. *Unified Shot Detection Framework Based on Graph Partition Model.* In Proceedings of the ACM Conference on Multimedia, Singapore, 2005, 539-542.

192. Yuan, J., Wang, H., Xiao, L., Ding, D., Zuo, Y., Tong, Z., Liu, X., Xu, S., Zheng, W., Li, X., Si, Z., Li, X., Lin, F., and Zhang, B. *Tsinghua University at TRECVID 2005.* In Online Proceedings of TRECVID Conference Series 2005: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html

193. Yuan, J., Zhang, B., and Lin, F. *Graph Partition Model for Robust Temporal Data Segmentation.* In Lecture Notes on Artificial Intelligence, Vol. 3518: Proc. of the Nineth Pacific-Asia Conference on Knowledge Discovery and Data Mining, Hanoi, Vietnam, Springer-Verlag, 2005, 758-763.

194. Yuan, J., Wang, H., Xiao, L., Zheng, W., Li, J., Lin, F., and Zhang, B. *A Formal Study of Shot Boundary Detection.* In IEEE Transaction on Circuits and Systems for Video Technology, Vol. 17, No. 2, 2007, 168-186.

195. Zhang, H., Kankanhalli, A., and Smoliar, S. W. *Automatic partitioning of full motion video.* In Multimedia Systems, Springer, 1993, 10-28.

196. Zhao, W., Chellappa, R., Phillips, P. J., and Rosenfeld, A. *Face Recognition: A literature survey.* In ACM Computing Surveys, Volume 35, Issue 4 (2003), ACM Press, 2003, 399-458.

197. Zheng, W., Yuan, J., Wang, H., Lin, F., and Zhang, B. *A Novel Shot Boundary Detection Framework.* In Proceedings of the SPIE Volume 5960 "Visual Communications and Image Processing", 2005, 410-420.

Annex D. Lebenslauf (CV)

1 PERSÖNLICHE DATEN

Name:	Ralph Ewerth
Geburtsdatum:	29.07.1972
Geburtsort:	Hanau
Familienstand:	ledig
Staatsangehörigkeit:	Deutsch

2 SCHULBILDUNG

08/1979 - 07/1983	Grundschule Bruchköbel-Roßdorf
08/1983 - 07/1989	Heinrich-Böll-Schule, Gesamtschule, Bruchköbel
08/1989 - 07/1992	Georg-Christoph-Lichtenberg-Oberstufen-Gymnasium, Bruchköbel
	Abschluss: Abitur

3 STUDIUM

10/1993 - 09/1998:	Johann Wolfgang Goethe-Universität Frankfurt/Main
	Diplom-Informatik
	Nebenfach Psychologie, Schwerpunkt: Pädagogische Psychologie
	Vordiplom
10/1998 - 03/2002:	Philipps-Universität Marburg
	Diplom-Informatik, Schwerpunkt: Software Engineering
	Nebenfach Psychologie
	Abschluss: Diplom-Informatiker („sehr gut“)

4 BERUFSERFAHRUNGEN

08/1992 - 12/1992	Pedro-Jung-Schule, Hanau Sonderschule für Lernbehinderte, Abteilung für Körperbehinderte Zivildienst
12/1992 - 02/1993	Deutsche Post Aushilfstätigkeit als Briefzusteller
03/1993 - 09/2002	Firma Astech GmbH und Co. KG, Wöllstadt Softwareentwickler
10/2002 – 09/2005	Sonderforschungsbereich „Medienumbrüche“ Universität Siegen Wissenschaftlicher Mitarbeiter
10/2005 -	SFB „Medienumbrüche“ Philipps-Universität Marburg Wissenschaftlicher Mitarbeiter