Adaptive Methods for User-Centered Organization of Music Collections

Sebastian Stober

DKE
Data & Knowledge Engineering Group

ADAPTIVE METHODS FOR USER-CENTERED ORGANIZATION OF MUSIC COLLECTIONS

DISSERTATION
zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

angenommen durch die Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg

von Dipl.-Inform. Sebastian Stober

geboren am 10. Dezember 1980 in Halberstadt

Gutachterinnen/Gutachter:
Prof. Dr. Andreas Nürnberger
Dr. habil. Marcin Detyniecki
Prof. Dr. Emilia Gómez

Ort und Datum des Promotionskolloquiums:
Magdeburg, den 15. November 2011

Sebastian Stober: *Adaptive Methods for User-Centered Organization of Music Collections,* Dissertation, © November 2011

Dedicated to the loving memory of Karl Sagebaum.
1924 – 2008

ABSTRACT

Music Information Retrieval (MIR) systems have to deal with multi-faceted music information and very heterogeneous users. Especially when the task is to organize a music collection, the diverse perspectives of users caused by their different level of expertise, musical background or taste pose a great challenge. This challenge is addressed here by proposing adaptive methods for several elements of MIR systems: Data-adaptive feature extraction techniques are described that aim to increase the quality and robustness of the information extracted from audio recordings. The classical genre classification problem is approached from a novel user-centric perspective – promoting the idea of idiosyncratic genres that better reflect a user's personal listening habits. An adaptive visualization technique for exploration and organization of music collections is elaborated that especially addresses the common and inevitable problem of projection errors introduced by dimensionality reduction approaches. Furthermore, it is outlined how this technique can be applied to facilitate serendipitous music discoveries in a recommendation scenario and to enable novel gaze-supported interaction techniques. Finally, a general approach for adaptive music similarity is presented which serves as the core of many adaptive MIR applications. Application prototypes demonstrate the usability of the described approaches.

ZUSAMMENFASSUNG

Music Information Retrieval (MIR) Systeme müssen fazettenreiche Informationen verarbeiten und gleichzeitig mit heterogenen Nutzern umgehen können. Insbesondere wenn es darum geht, eine Musiksammlung zu organisieren, stellen die verschiedenen Sichtweisen der Nutzer, verursacht durch deren unterschiedliche Kompetenz, musikalischen Hintergrund und Geschmack, eine große Herausforderung dar. Diese Herausforderung wird hier adressiert, indem adaptive Verfahren für verschiedene Elemente von MIR Systemen vorgeschlagen werden: Datenadaptive Techniken zur Merkmalsextraktion werden beschrieben, welche zum Ziel haben, die Qualität und Robustheit der aus Audioaufnahmen extrahierten Informationen zu verbessern. Das klassische Problem der Genreklassifikation wird von einer neuen nutzerzentrierten Sichtweise behandelt – anknüpfend an die Idee idiosynkratischer Genres, welche die persönlichen Hörgewohnheiten eines Nutzer besser widerspiegeln. Eine adaptive Visualisierungstechnik zur Exploration und Organisation von Musiksammlungen wird entwickelt, die insbesondere Projektionsfehler adressiert, welche ein weit verbreitetes und unumgängliche Problem von Techniken zur Dimensionsreduktion darstellen. Darüber hinaus wird umrissen, wie diese Technik eingesetzt werden kann, um die Interessantheit von Musikempfehlungen zu verbessern, und neue blickbasierte Interaktionstechniken ermöglicht. Schließlich wird ein allgemeiner Ansatz für adaptive Musikähnlichkeit vorgestellt, welcher als Kern für eine Vielzahl adaptiver MIR-Anwendungen dient. Die Einsatzmöglichkeiten der beschriebenen Verfahren werden an verschiedenen Anwendungsprototypen gezeigt.

Happiness is only real when shared.
Christopher McCandless

ACKNOWLEDGMENTS

I would like to thank all people that supported me during the past years and in particular during the development of this thesis. At the first place, I owe my sincere gratitude to my mentor and supervisor, Andreas Nürnberger, for giving me the opportunity to be part of his young research group from the very beginning. He granted me all the freedom to pursue a topic of my own choice (with all the resulting challenges) and supported me throughout my work – especially during my first steps in the unfamiliar music information retrieval research community. I also want to thank Marcin Detyniecki and Emila Gómez for kindly agreeing to review my thesis, taking their time to read it and give feedback. I am especially grateful to Perfecto Herrera who took precious time out from his busy schedule to serve as my external reader even though he could not be a reviewer himself. I also owe my thanks to Christian Hentschel, Janick Martinez Esturo, Christian Moewes, Georg Ruß, and Sophie Stellmach whose comments, corrections and criticism made this thesis more readable and understandable.

I thank my colleagues and the whole staff of the *Faculty of Computer Science* at the *Otto-von-Guericke-University*, especially those from the *Department of Technical and Operational Information Systems* and the *Department of Knowledge and Language Engineering*, for the pleasant working atmosphere. I would further like to give special thanks to all the colleagues and students that – in one way or another – were involved in some bits of this work. In particular, Christian Moewes and Matthias Steinbrecher showed their data mining expertise in analyzing the listening logs and the survey data. Tobias Germer kindly provided his original implementation of the *SpringLens* distortion technique and ideas on how to elaborate it. Christian Hentschel took care of the image feature extractors and helped to conduct the user study for the evaluation of the focus-adaptive *SpringLens* interface and interpret its results – but above all, he has been a great person to discuss ideas with over breakfast. The folk song analysis experiments were done in collaboration with Korinna Bade and Jörg Garbers. Sophie Stellmach introduced me to the field of gaze-supported interaction and developed and evaluated the respective extension of *MusicGalaxy* almost entirely on her own. She has also been a valuable consultant on eye tracking technology in general. I am very grateful to all the "EU Bisons" (members of the EU project *BISON – Bisociation Networks for Creative Information Discovery*) for introducing me to the concept of bisociations and many fruitful and sometimes not so fruitful discus-

sions. In particular, Stefan Haun helped to transfer and apply their approaches in the music domain. Micheal Kauert and his colleagues from the *Technology-Transfer-Center* kindly supported me during the many *CeBIT* fares I attended to present my work and gather feedback. Furthermore, the following students of the many, which I had the pleasure – and sometimes challenge – to supervise, contributed with their internships, master- or diploma theses, or as research assistants: Gregor Aisch, Christian Beyer, Matthias Busch, Alexander Duda, Sebastian Dorok, Georgi Dzhambazov, Marcel Hermkes, Valentin Laube, Sebastian Loose, Konstanze Lorenz, Stefan Räbiger, Johannes Reinhard, Fabian Schmidt, Mike Stephan, and Martin Tobies. I would also like to thank the many participants of the user studies and surveys for their valuable time and effort.

This thesis has been typeset using LaTeX with the beautiful *classicthesis* style by André Miede, whom I also thank for his help concerning my style customizations. For the implementation of the algorithms and prototypes described in this thesis, I relied on a number of freely available software libraries and web services, which has facilitated my work considerably. I would like to thank everybody involved in the development of these tools. Furthermore, I also thank Christopher Harte and the people behind *Magnatagatune* and *MusicBrainz* who kindly shared their music information datasets.

Special thanks go to the *German Research Foundation (DFG)* for funding my research as the project *AUCOMA – Adaptive User-Centered Organization of Music Archives*. Furthermore, I am gratefully indebted to the *German National Merit Foundation (Studienstiftung des deutschen Volkes)*, whose Ph.D. scholarship went far beyond plain financial support and has also been a great honor and motivation.

Thankfully, my path of becoming a doctor has been littered with many enjoyable distractions. First of all, I want to thank all my friends and especially the *IFx*00 and *VT* 75-24 fellows, without whom this time would not have been half as much fun. I am very grateful for having been part of the *OvGU Bigband*, whose members I thank for many exciting rehearsals and memorable concerts as well as for their patience while I struggled with the swing rhythms. The many university sport courses that I took or led during the last years helped a great deal to balance my life. I am especially grateful to my qigong and taijiquan teachers and their masters for sharing and passing on their knowledge and experience. Most of all, I thank René Taurer who introduced me to the Chinese martial arts, Susanne Gärtner who motivated me to start practicing qigong, and Andrea Skroblien for her continuing encouragement and guidance.

Finally, I owe my deepest thanks to my family who always supported and encouraged me without a second thought in everything I have ever done. Most of all, I want to thank Kathrin for her love and sharing so many happy moments with me along the way.

PUBLICATIONS

Some ideas and figures have appeared previously in the following publications (in chronological order):

[pub:1] Duda, A., Nürnberger, A., and Stober, S.: "Towards Query by Singing/Humming on Audio Databases." In: *Proceedings of the 8th International Conference on Music Information Retrieval (ISMIR'07)*. Ed. by Dixon, S., Bainbridge, D., and Typke, R. Vienna, Austria: ÖCG, 2007, pp. 331–334 (cit. on pp. 61–64, 184).

[pub:2] Stober, S. and Nürnberger, A.: "User Modelling for Interactive User-Adaptive Collection Structuring." In: *Adaptive Multimedial Retrieval: Retrieval, User, and Semantics. 5th International Workshop, AMR 2007, Paris, France, July 5-6, 2007, Revised Selected Papers*. Ed. by Detyniecki, M. and Nürnberger, A. Vol. 4918. LNCS. Heidelberg / Berlin: Springer Verlag, 2008, pp. 95–108 (cit. on pp. 127, 159, 184).

[pub:3] Stober, S. and Nürnberger, A.: "AUCOMA - Adaptive Nutzerzentrierte Organisation von Musikarchiven." In: *Fortschritte der Akustik: Plenarvorträge und Fachbeiträge der 34. Deutschen Jahrestagung für Akustik DAGA 2008, Dresden*. Ed. by Jekosch, U. and Hoffmann, R. German Acoustical Society (DEGA). Berlin, Germany: German Acoustical Society (DEGA), 2008, pp. 547–548 (cit. on p. 184).

[pub:4] Reinhard, J., Stober, S., and Nürnberger, A.: "Enhancing Chord Classification through Neighbourhood Histograms." In: *Proceedings of the 6th International Workshop on Content-Based Multimedia Indexing (CBMI'08)*. London, UK, 2008 (cit. on pp. 61, 65–66, 68, 70, 73, 184).

[pub:5] Laube, V., Moewes, C., and Stober, S.: "Browsing Music by Usage Context." In: *Proceedings of the 2nd Workshop on Learning the Semantics of Audio Signals (LSAS'08)*. Ed. by Burred, J. J., Nürnberger, A., Peeters, G., and Stober, S. Paris, France: IRCAM, 2008, pp. 19–29 (cit. on pp. 77, 83, 184).

[pub:6] Stober, S. and Nürnberger, A.: "User-Adaptive Music Information Retrieval." In: *KI* 23.2 (2009), pp. 54–57 (cit. on p. 184).

[pub:7] Bade, K., Nürnberger, A., and Stober, S.: "Everything in its right place? Learning a user's view of a music collection." In: *Proceedings of NAG/DAGA 2009, International Conference on Acoustics, Rotterdam*. German Acoustical Society (DEGA). Berlin, Germany: German Acoustical Society (DEGA), 2009, pp. 344–347 (cit. on p. 184).

[pub:8] Bade, K., Garbers, J., Stober, S., Wiering, F., and Nürnberger, A.: "Supporting Folk-Song Research by Automatic Metric Learning and Ranking." In: *Proceedings of the 10th International Conference on Music Information Retrieval (ISMIR'09)*. International Society for Music Information Retrieval. Kobe, Japan, 2009, pp. 741–746 (cit. on pp. 126, 137, 184).

[pub:9] Stober, S., Steinbrecher, M., and Nürnberger, A.: "A Survey on the Acceptance of Listening Context Logging for MIR Applications." In: *Proceedings of the 3rd Workshop on Learning the Semantics of Audio Signals (LSAS'09)*. Ed. by Baumann, S., Burred, J. J., Nürnberger, A., and Stober, S. Graz, Austria, 2009, pp. 45–57 (cit. on pp. 86, 91, 184).

[pub:10] Stober, S. and Nürnberger, A.: "Towards User-Adaptive Structuring and Organization of Music Collections." In: *Adaptive Multimedia Retrieval. Identifying, Summarizing, and Recommending Image and Music. 6th International Workshop, AMR 2008, Berlin, Germany, June 26-27, 2008. Revised Selected Papers*. Ed. by Detyniecki, M., Leiner, U., and Nürnberger, A. Vol. 5811. LNCS. Heidelberg / Berlin: Springer Verlag, 2010, pp. 53–65 (cit. on pp. xxiii, 126, 141, 143–144, 184, 194–195).

[pub:11] Stober, S., Hentschel, C., and Nürnberger, A.: "Multi-Facet Exploration of Image Collections with an Adaptive Multi-Focus Zoomable Interface." In: *Proceedings of 2010 IEEE World Congress on Computational Intelligence (WCCI'10)*. Barcelona, Spain, 2010, pp. 2780–2787 (cit. on pp. 115, 184).

[pub:12] Stober, S. and Nürnberger, A.: "A Multi-Focus Zoomable Interface for Multi-Facet Exploration of Music Collections." In: *Proceedings of 7th International Symposium on Computer Music Modeling and Retrieval (CMMR'10)*. Málaga, Spain, 2010, pp. 339–354 (cit. on pp. 104, 114, 184).

[pub:13] Stober, S. and Nürnberger, A.: "Visualisierung von großen Musiksammlungen unter Berücksichtigung projektionsbedingter Verzerrungen." In: *36. Jahrestagung für Akustik DAGA 2010, Berlin*. German Acoustical Society (DEGA). Berlin, Germany: German Acoustical Society (DEGA), 2010, pp. 571–572 (cit. on pp. 114, 184).

[pub:14] Stober, S. and Nürnberger, A.: "MusicGalaxy - An Adaptive User-Interface for Exploratory Music Retrieval." In: *Proceedings of 7th Sound and Music Computing Conference (SMC'10)*. Barcelona, Spain, 2010, pp. 382–389 (cit. on pp. 110, 184).

[pub:15] Stober, S. and Nürnberger, A.: "Automatic Evaluation of User Adaptive Interfaces for Information Organization and Exploration." In: *SIGIR Workshop on the Simulation of Interaction (SimInt'10)*. Geneva, Switzerland, 2010, pp. 33–34.

[pub:16] Stober, S. and Nürnberger, A.: "MusicGalaxy - An Adaptive User-Interface for Exploratory Music Retrieval." In: *11th International Conference on Music Information Retrieval (ISMIR'10) - Late Breaking Demo Papers*. Utrecht, Netherlands, 2010 (cit. on pp. xxiii, 114, 184).

[pub:17] Stober, S. and Nürnberger, A.: "Similarity Adaptation in an Exploratory Retrieval Scenario." In: *Proceedings of 8th International Workshop on Adaptive Multimedia Retrieval (AMR'10)*. Linz, Austria, 2010 (cit. on pp. 126, 146, 159, 184).

[pub:18] Stober, S., Hentschel, C., and Nürnberger, A.: "Evaluation of Adaptive SpringLens - A Multi-focus Interface for Exploring Multimedia Collections." In: *Proceedings of 6th Nordic Conference on Human-Computer Interaction (NordiCHI'10)*. Reykjavik, Iceland, 2010, pp. 785–788 (cit. on pp. 115, 184).

[pub:19] Stober, S. and Nürnberger, A.: "A Multi-Focus Zoomable Interface for Multi-Facet Exploration of Music Collections." In: *Extended Proceedings of 7th International Symposium on Computer Music Modeling and Retrieval (CMMR'10)*. Ed. by al., K. J. et. Vol. 6684. LNCS. Berlin / Heidelberg: Springer Verlag, 2010, pp. 273–302 (cit. on pp. 97, 184).

[pub:20] Stellmach, S., Stober, S., Dachselt, R., and Nürnberger, A.: "Designing Gaze-supported Multimodal Interactions for the Exploration of Large Image Collections." In: *Proceedings of 1st International Conference on Novel Gaze-Controlled Applications (NGCA'11)*. Karlskrona, Sweden, 2011, pp. 1–8 (cit. on pp. 173–174, 177–179, 184).

[pub:21] Stober, S. and Nürnberger, A.: "Analyzing the Impact of Data Vectorization on Distance Relations." In: *Proceedings of 3rd International Workshop on Advances in Music Information Research (AdMIRe'11)*. Barcelona, Spain, 2011 (cit. on pp. 184, 193).

[pub:22] Stober, S.: "Adaptive Distance Measures for Exploration and Structuring of Music Collections." In: *Proceedings of AES 42nd Conference on Semantic Audio*. Ilmenau, Germany, 2011 (cit. on pp. 126, 184).

[pub:23] Stober, S. and Nürnberger, A.: "An Experimental Comparison of Similarity Adaptation Approaches." In: *Proceedings of 9th International Workshop on Adaptive Multimedia Retrieval (AMR'11)*. Barcelona, Spain, 2011 (cit. on pp. 147, 184).

[pub:24] Stober, S., Haun, S., and Nürnberger, A.: "Creating an Environment for Bisociative Music Discovery and Recommendation." In: *Proceedings of Audio Mostly 2011 – 6th Conference on Interaction with Sound*. Coimbra, Portugal, 2011 (cit. on pp. 163, 184).

Furthermore, the following diploma theses of students that I had the pleasure to supervise substantially contributed to this work:

[stud:1] Duda, A.: "Query-by-Singing/Humming with Low-Level Feature Extraction." Diploma Thesis. Otto-von-Guericke-University Magdeburg, 2007 (cit. on pp. 61–62, 64, 184).

[stud:2] Reinhard, J.: "Akkorderkennung in Audiodateien." Diploma Thesis. Otto-von-Guericke-University Magdeburg, 2008 (cit. on pp. 61, 65, 184).

[stud:3] Laube, V.: "Analysis and Visualization of Music Listening Behaviour." Diploma Thesis. Otto-von-Guericke-University Magdeburg, 2008 (cit. on pp. 77, 95, 184).

[stud:4] Loose, S.: "MusicGalaxy: Eine interaktive Visualisierung zur multifokalen Exploration von Musikbibliotheken." Diploma Thesis. Otto-von-Guericke-University Magdeburg, 2010 (cit. on pp. 109, 184).

Credit is given in margin notes in the respective sections.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF MEDIA CLIPS

The following media clips serve as additional illustration and can be downloaded from http://www.dke-research.de/aucoma/thesis/

Audio Clip 1 VoiceExtraction-1-Original.wav
Original 22 seconds clip from *"Have a little faith in me"* by Joe Cocker. 64

Audio Clip 2 VoiceExtraction-2-Karaoke.wav
Approximated karaoke version of Audio Clip 1. . 64

Audio Clip 3 VoiceExtraction-3a-GlobalNoiseProfile.wav
Filtered melody extracted from audio clip 1 using a global noise profile from Audio Clip 2. Figure 16 (top) shows a visualization of the frequency spectrum and the waveform. 64

Audio Clip 4 VoiceExtraction-3b-LocalNoiseProfile.wav
Filtered melody extracted from audio clip 1 using a local noise profile from Audio Clip 2. Figure 16 (bottom) shows a visualization of the frequency spectrum and the waveform. 64

Video Clip 1 MusicGalaxyH264.mov
Demonstration video clip for the *MusicGalaxy* user interface prototype created for the ISMIR'10 late breaking demo [pub:16]. 110, 167

Video Clip 2 PhotoSelectionTaskH264.mov
Excerpt of a recorded session as part of the evaluation of the focus-adaptive *SpringLens* visualization technique (cf. Section 7.5). The gaze trajectory is visualized by red lines and filled red circles (size increases with time). Mouse clicks are highlighted by red (left button) and green (right button) circles. 117

Video Clip 3 BeatlesExplorerH264.mov
Demonstration video clip for the *BeatlesExplorer* user interface prototype created for [pub:10]. 141

LIST OF ACRONYMS

API	Application Programming Interface . . . 99
ARL	Advanced Relationship Link . . . 169
BPM	Beats Per Minute . . . 22
DAG	Directed Acyclic Graph . . . 133
EM	Expectation Maximization . . . 34
FFT	Fast Fourier Transform . . . 35
GHSOM	Growing Hierarchical Self-Organizing Map . . . 41
GMM	Gaussian Mixture Model . . . 25
GSOM	Growing Self-Organizing Map (cf. Section 4.3.1) . . . 55
GUI	Graphical User Interface . . . 157
HMM	Hidden Markov Model . . . 25
ISMIR	International Symposium for Music Information Retrieval (since 2009: International Society for Music Information Retrieval) . . . 7
LDA	Linear Discriminant Analysis . . . 43
LMDS	Landmark Multidimensional Scaling (cf. Section 4.4.1) 57
LMNN	Large-Margin Nearest Neighbor . . . 43
LVQ	Learning Vector Quantization . . . 25
MAP	Maximum A Posteriori . . . 36
MDS	Multidimensional Scaling (cf. Section 4.4) . . . 49
MFCC	Mel-Frequency Cepstral Coefficient . . . 25
MIDI	Musical Instrument Digital Interface . . . 16
MIR	Music Information Retrieval . . . 1
MIREX	Music Information Retrieval Evaluation eXchange . . . 13
NCA	Neighborhood Component Analysis . . . 43
OMR	Optical Music Recognition . . . 17
OPE	Object Position Error . . . 144
PA-L-EX	Passive-Aggressive Linear Expanded
PCA	Principal Component Analysis . . . 99
POE	Partial Order Embedding . . . 43
PUID	Portable Unique IDentifier . . . 169
QBSH	Query-by-Singing/Humming . . . 22
QP	Quadratic Programming . . . 52

RBFN	Radial Basis Function Network	42
RCA	Relevant Component Analysis	43
SOM	Self-Organizing Map (cf. Section 4.3)	41
SVM	Support Vector Machine (cf. Section 4.2)	37
TreeQ	Tree Vector Quantizer	37
VRPN	Virtual Reality Peripheral Network	178
XML	Extensible Markup Language	17

Music is the language of us all

"How to explain"
The Cat Empire

If we knew what it was we were doing,
it would not be called "research", would it?
Albert Einstein

1 INTRODUCTION

One of the big challenges of computer science in the 21st century is the digital media explosion. Steadily growing hard drives are filled with personal media collections comprising, e. g., music, photos and videos. With increasing collection size, maintenance becomes a more and more tedious task, but without manual organization effort it gets harder to access specific pieces of media or even to keep an overview. Typically, a large portion of the digital content is just collecting dust because the user has simply forgotten about it. Here, computer science and especially information retrieval techniques can help to improve awareness and accessibility of such data.

One means to ease access to media collections is automatic structuring, be it for organization or for exploration. Moreover, users would greatly benefit if a system would not just structure the collection for easier access but would organize it in a way that is intuitively understandable for the individual user by adapting to personal preferences and needs. Unfortunately, such aspects of individualization have been only a minor issue of research in the field of multimedia retrieval. At best, interfaces for media collection access allow for manual adaption by the user. However, they are largely lacking the ability to learn from user actions and to adapt on their own without explicit intervention of the user. The overall goal of this thesis project is therefore *to develop intuitive, non-obtrusive, user-adaptive methods for media collection access with special focus on Music Information Retrieval (MIR)*.

Dealing with music information, the following considerations serve as motivation for the project: Firstly, music can be described by a large variety of ways comprising, e. g., simple tags (artist, title etc.), content-based features ranging from simple loudness to complex timbre descriptions, harmonics, meter and tempo, instrumentation, and lyrics but also information about the production and publishing process as well as the general reception in the public expressed in reviews or chart positions. This diversity of features makes music especially interesting from the information retrieval point of view and allows to transfer results to different domains. Secondly, perception of music is highly subjective and may depend on a person's background. A musician, for instance, might especially look after structures, harmonics or instrumentation (possibly paying – consciously or unconsciously – special attention to his own instrument). Non-musicians will perhaps focus more on overall timbre or general mood. Others, in turn, may have a high interest in the lyrics as long as they are able to understand

the particular language. Finally and most importantly, music can be considered as an integral part of daily life even though it may often only play a background role. There may be common contexts in which music is consumed as well as contexts that are particular to an individual listener. Either way, the choice of music listened to in each context is supposed to be highly individual. The large variety of usage contexts makes this especially interesting for research in the area of user modeling and personalization. Given these considerations, this thesis addresses the following specific research tasks:

1. An analysis of existing adaptive MIR approaches
   - establishing a suitable generalized model of adaptive systems for system analysis and design, and
   - systematically categorizing the existing approaches.
2. An investigation of the potential of listening context information for collection organization
   - gathering evidence for the existence and usefulness of idiosyncratic genres relating to personal listening habits, and
   - identifying possibilities for automatic logging of listening context information.
3. The design of a generic model for adaptive music similarity
   - combining multiple facets of music information,
   - allowing manual adaptation (by users), and
   - supporting automatic adaptations by algorithms that can learn from user actions in interactive scenarios.
4. The development of an adaptive visualization of music collections for exploration and organization based on a user's similarity model, in particular:
   - providing means for browsing/navigation,
   - enabling interactive manipulation by reorganization,
   - supporting large collections (by means of a zoom function or hierarchical organization), and
   - allowing and possibly combining multiple (adapted) views on the collection.

1.1 THESIS OUTLINE

This thesis is structured into three main parts and an appendix. Part i provides the reader with a context and the fundamental knowledge for the understanding of this thesis: Chapter 2 introduces MIR as a field of research with its innate challenges and common practices. In particular, the model of the general retrieval process described

in Section 2.2.1 serves as contextual frame for the specific problems addressed by the approaches in Part ii. Chapter 3 takes a closer look at adaptive approaches within the field of MIR (cf. Task 1). To this end, a working definition for adaptivity and a generic model for adaptive systems are elaborated in Section 3.2.1 that allow a systematic analysis of the different approaches. The fundamental techniques that are applied in the context of this thesis are explained in Chapter 4 as a prerequisite for a deeper understanding.

Part ii proposes various adaptive approaches that cover different elements of MIR systems: Chapter 5 addresses the subject of adaptive feature extraction – the first step in the general retrieval process. It summarizes the relevant adaptive aspects of two diploma theses that have been supervised in the context of this work. One deals with the problem of extracting the melody from an audio recording and the other with the correction of misclassifications in chord recognition. Chapter 6 turns to the problem of genre classification which happens to be one of the best studied subjects in MIR. However, a different perspective is taken here, addressing Task 2 and arguing that user-specific genres emerging from personal usage patterns would be of more use than the common categories because users could directly relate to them. A pilot study investigates possibilities for automatic listening context logging and a subsequent survey analyzes how potential users would accept the different proposed options. Chapter 7 focuses on the visualization of music collections for exploration and organization (cf. Task 4). A focus-adaptive visualization technique is elaborated that addresses the common and inevitable problem of projection errors introduced by dimensionality reduction approaches. Based on this technique, the *MusicGalaxy* prototype – a user interface for music collection exploration – is developed in a user-centered design process. Finally, Chapter 8 takes a look at music similarity as a key element of MIR systems (cf. Task 3). A generalized approach is presented, which allows to model and learn individual distance measures for comparing music pieces based on weighted facets. Three application scenarios with different objectives exemplify how the proposed method can be employed in various contexts, guided either by domain-specific expert information or by user actions in an interactive setting.

Part iii gives an outlook for future research that is based on the work presented in Part ii but takes different directions: Chapter 9 proposes a way to increase the chance of serendipitous music recommendations using the *MusicGalaxy* user interface by exploiting the concept of bisociations. Furthermore, Chapter 10 investigates gaze-supported interaction techniques with *MusicGalaxy*. Concluding, Chapter 11 summarizes this thesis and its contributions.

Part I

FUNDAMENTALS

Writing about music is like dancing about architecture – it's a really stupid thing to want to do.

Elvis Costello
(as quoted in [243])

2 MUSIC INFORMATION RETRIEVAL

According to Wikipedia [url:53], Music Information Retrieval (MIR) is

> "[...] the interdisciplinary science of retrieving information from music."

This common consensus, which is consistent with the descriptions given in the scientific publications cited below, can be seen as the attempt of a least restrictive definition of a still evolving field of research. The term MIR has already been used in the 1960s in the context of computer-supported musicology [106]. However, as a discipline, MIR has been maturing only since the late 1990s – driven by the increasing availability of music in digital form and decreasing costs for storage and processing power at the same time. It is thus much younger than and "decades behind" [30] classic information retrieval which has been dealing with text documents for more than 50 years. Further, though belonging to the wider field of multimedia retrieval, MIR still remains largely unnoticed compared to the long-established sister disciplines of image and video retrieval.

Futrelle and Downie [70] describe MIR as an interdisciplinary research area encompassing computer science (in particular information retrieval, machine learning, and user interface design), musicology and music theory, audio engineering and digital signal processing, cognitive science and psychology, philosophy, library science, publishing, and law. – Arguably, this list could even be further extended, e. g., including computer music [56] as an early discipline that paved the way for MIR. In 2000, the first International Symposium for Music Information Retrieval (ISMIR) [29] was held as an explicit attempt to gather together representatives of all the related disciplines and research areas. Since then, it has developed into a yearly event playing a key role for MIR research. In 2009, the International Society for Music Information Retrieval [url:20] emerged from the community formed by this conference and adopted its acronym.

This chapter aims to give a brief overview of MIR with a special focus on topics and concepts that are particularly relevant in the context of this work. More thorough overviews are provided by Downie [60], Typke, Wiering, and Veltkamp [237], Orio [176] and most recently Casey et al. [34] and Gouyon et al. [79]. Further, the cumulative ISMIR proceedings are publicly available at no cost [url:7]. An analysis of the proceedings of the past ISMIR conferences between 2000 and 2008 is given by Grachten et al. [82] and Lee, Jones, and Downie [120]

visualizing and interpreting the change of hot ISMIR topics in the course of time.

2.1 CHALLENGES OF MUSIC INFORMATION RETRIEVAL

DOWNIE [60] identifies several aspects of music and music information that pose challenges for the development of MIR systems:

1. Music is multi-cultural (Section 2.1.1).
2. Music information has many facets (Section 2.1.2) and
3. can be represented in multiple ways (Section 2.1.3).
4. Users of MIR systems are very heterogeneous and have varying information needs (Section 2.1.4).
5. Music can be experienced in many ways leading to different perceptions (Section 2.1.5).

The concept of facets plays a key role in this thesis. Therefore, it is covered more thoroughly in Section 2.1.2 which also serves as an introduction to very basic musical terms used throughout this thesis. Further, Section 2.1.4 and Section 2.1.5 are of special interest here because they provide a main motivation for the development of adaptive MIR systems.

2.1.1 *Multi-Cultural Challenge*

Almost every known culture in the current and probably past world has created music. Nevertheless, music has developed differently in distant parts of the world leading not only to a manifold of musical styles but also to different systems. So far, work in MIR has almost exclusively focused on so-called *Western tonal music* – especially classical and popular music – though methods are to some extent also applicable for non-western music [127]. DOWNIE [60] names three causes for this bias: the availability of both, symbolic and audio encodings, the familiarity of the researchers with this music, and the size of the potential user base.

This bias is also maintained in this thesis as covering non-western music is far beyond its scope. Therefore, in the following, the context of Western tonal music is assumed if not explicitly stated otherwise. Some of the particularities of Western tonal music are covered by the next section – most importantly w.r.t. tonality (Section 2.1.2.1) and harmony (Section 2.1.2.2).

2.1.2 *Multi-Faceted Challenge*

Considering music as a perceptual phenomenon based on physics, several basic parameters can be defined in relation to physical properties that have an impact on perceptual sensations. For instance, Byrd and Crawford [30] mention four general parameters of (definite-pitched) musical notes:

1. *pitch* (the perceptual analog of frequency),
2. *duration* (the temporal length) – alternatively described by the note *onset* and *offset* (the start end ending of the note respectively),
3. *loudness* (the perceptual analog of amplitude), and
4. *timbre* or tone quality.

They further point out that these *parameters* are not cleanly separable as, e. g., short notes are perceived less loud that longer ones. Consequently, it is also hard to separate the *facets* of music discussed in the following. In fact, while it appears obvious that music information is multi-faceted, various views exist on what the actual facets are: Downie [60] differentiates seven "facets" of music information playing a variety of roles in defining the MIR domain. Alternatively, Lesaffre et al. [126] consider six basic "categories" for a taxonomy for MIR feature extraction. Orio [176] similarly names seven main "dimensions" effectively usable for MIR. Finally, in a recent survey on content-based MIR, Casey et al. [34] mention several high-level music "features". All these can be considered as facets of music information. Table 1 summarizes their coverage in the different publications.

author(s)	facets
Downie [60]	pitch, harmony, temporal, timbre, editorial, textual, bibliographic
Lesaffre et al. [126]	melody, harmony, rhythm, timbre, dynamics, expression
Orio [176]	melody, harmony, rhythm, timbre, orchestration, acoustics, structure
Casey et al. [34]	pitch, melody, key, harmony, rhythm, timbre, lyrics, structure, non-western music

Table 1: Overview of music information facets considered in the literature [34, 60, 126, 176].

At their core (melody, harmony, rhythm, timbre) the views largely correspond and deviations are most likely due to the slightly differing scope. Specifically, the pitch, temporal, and timbre facet cover the respective aforementioned parameters of musical notes. The loudness parameter is covered by the dynamics facet (but also plays an important role for rhythm). Downie [60] groups pitch, melody and key

together as they are closely related. However, the label "tonal facet" appears to be more suitable here. The temporal facet described by Downie [60] is a generalization of rhythm and tempo. The textual and lyrics facet are largely identical. Orchestration – in the sense used by Orio [176] – belongs to the editorial facet of Downie [60]. However, the choice of the instruments also has a significant impact on the overall timbre of a recording. Acoustics – comprising, e. g., room acoustics, background noise and audio post-processing – can be considered a sub-facet of timbre. Finally, the "non-western music" feature mentioned by Casey et al. [34] mainly refers to differing tonal systems and thus relates to the tonal facet as well.

The following sections address the different facets in detail. Further, two additional facets are proposed here that take recent developments in the field of MIR into account: the visual (Section 2.1.2.10) and the contextual facet (Section 2.1.2.11).

2.1.2.1 *Tonal Facet*

Western tonal music builds upon an "alphabet" of notes with definite and discretized *pitch* – the perceived fundamental frequency of a tonal sound (*tone*). The pitch difference between two notes is called *interval*. An *octave* – the interval between one musical pitch and another with half or double its frequency – is divided into 12 semitones which in the commonly used *equal temperament* tuning are equally spaced out. Pitches with octave distance are perceived as very similar and mapped to the same *pitch class*. This results in a total of 12 pitch classes – one for each semitone of an octave.

Further, a *melody* can be considered as a (temporal) sequence of tones – usually adding up to a recognizable whole. The respective sequence of intervals of a melody forms the *melody contour*. Melodies shifted in pitch – called *transposed* – still share the same contour and are perceived by listeners as equivalent although their pitches are different. Downie [60] explicitly mentions the musical *key* as a sub-facet of pitch as it can be regarded as the tonal center of gravity of a piece. The key may also change within a piece which is called *modulation*.

2.1.2.2 *Harmonic Facet*

Harmony results from multiple pitches sounding simultaneously which is also called *polyphony* – in contrast to *monophony* with only one pitch at a time. In Western tonal music, harmonically related tones are commonly grouped into *chords* which adds another layer of abstraction. The same set of tones may however be interpreted in several ways resulting in different chords. Thus, to determine an actual chord label (unless it is given explicitly), usually the harmonic context of the surrounding phrase or the whole piece has to be taken into account such

as the key. To further complicate things, listeners can often perceive and recognize chords even if the tones are not played simultaneously but sequentially or delayed[1], or if other non-chord tones are present.

2.1.2.3 *Temporal / Rhythm Facet*

According to Downie [60], the rhythmic component of a musical piece results from the complex interplay of five elements: tempo indicators, meter, pitch duration, harmonic duration, and accents. He differentiates absolute (e. g., in beats per minute), general (e. g., "adagio") and relative (e. g., "faster") tempo information. Further, tempo distortions such as rubato, accelerando, rallentando, and ritardando are possible and for some playing styles such as Jazz, deviations from the actual score are implicitly expected.

2.1.2.4 *Dynamics Facet*

While the dynamic instructions are part of the editorial facet (Section 2.1.2.7), the actual dynamics in a recording form a separate one. There is dynamic on a micro-time scale referring to the aforementioned loudness parameter of single notes. Slightly varying loudness of single notes can be essential for a rhythmic pattern and largely contribute to its liveliness. Further, on a larger time scale, dynamics can also vary between parts of a musical piece. e. g., the repetition of some part could be louder. The *dynamic range* as the difference between the quietest and loudest volume can also be regarded as an important property of a musical piece [url:51]. However, over-compression of recordings (i. e., reduction of the dynamic range to increase the loudness) has become common practice in music production processes – especially for popular music [url:44].

2.1.2.5 *Timbral Facet*

This facet comprises all aspects of tone color that make it, e. g., possible to distinguish between different instruments playing the same note or between different playing techniques of a single instrument such as mutings, pedalings and bowings. Downie [60] remarks that information about such performance methods as well as the orchestration can also be attributed to the editorial facet – in so far as it refers to the respective instructions in contrast to the aural effect. Apart from instrument-related aspects of timbre, there are also qualities of the timbre sensation that do not need to be associated with a single instrument but characterize the general way a recording sounds – such as "harsh", "airy", "noisy", or "transparent."

1 Because of this, harmony may also be considered inside the tonal facet.

2.1.2.6 *Structural Facet*

Musical structure builds upon various basic elements already covered by the aforementioned facets, e. g., melody, harmony, key, rhythm, meter, tempo, timbre (orchestration), and dynamics. It is induced by the change of these elements throughout a musical piece on a higher-level time scale and can be described by segments having a certain time range and label(s). Segments with identical labels are considered as occurrences of a certain structural part. A segmentation can also be hierarchical, i. e., parts may consist of sub-parts.

There are certain structural patterns that pervade many musical genres. For instance, the *12-bar blues scheme* (typically a set of three related chords played repeatedly over 12 bars) is very popular in blues music but also in other genres such as rock, pop and jazz music. Furthermore, a *intro-verse-chorus-verse-chorus-outro* structure is used in many rock and pop songs. This can also be described as *A-B-A'-B'* pattern, which is related to the more general *theme plus variations* form used in many classical pieces.

Orio [176] argues that *"the ability to perceive and recognize a musical structure depends on the musical education, knowledge of the styles of the music genre, exposure to previous musical works, and active listening"* and that among musicologists, various approaches to music structure exist. Most of them however focus on Western classical music and thus are only partly applicable for popular music. Finding a robust description of music structure that can be applied to any kind of music is not trivial. A multi-dimensional description that accomplishes this by super-imposing different view-points (musical role, acoustical similarity, instrument role) is proposed by Peeters and Deruty [195]. Paulus, Müller, and Klapuri [188] provide an overview of audio-based music structure analysis techniques.

2.1.2.7 *Editorial Facet*

This facet is primarily related to score-based music information. Here, Downie [60] considers primarily performance instructions such as dynamic instructions or ornamentation, but also parts of the music itself as, e. g., solos written out by the editor which originally had been intended as improvisations by the composer. He further points out that often the difference between editions make it hard to choose a "definitive" version to be included in a MIR system.

2.1.2.8 *Textual / Lyrics Facet*

The lyrics of songs – or in general any sung text belonging to a musical piece – form the textual facet. Here, well-established techniques from classic text retrieval can be applied, but there are also some particularities for song texts. Sometimes, different lyrics exist for a single melody, such as translations into other languages or adaptations.

At the same time, a text may relate to multiple musical pieces. Further, there exist many musical pieces without any text. Recently, a toolbox for lyrics analysis has been made publicly available [154].

2.1.2.9 *Bibliographic Facet*

This is the last facet of music information that DOWNIE [60] describes and the only one (mentioned by him) that is not derived from the content of a work.[2] Typically, it comprises information about the title and artist(s) (composer, arranger, lyrics author, producer, performer), as well as information about the publishing process, e. g., the publisher, edition, catalog number, publication date, and especially for popular music the album title and track number. Here lies the strongest connection to traditional library science and all of its inherent difficulties also apply here. DOWNIE [60] does not mention copyright information, though the above listed information could be used to identify potential copyright holders. However, explicit information about the copyright such as the type of license (e. g., public domain or creative commons) needs to be taken into account as well.

2.1.2.10 *Visual Facet*

This facet does *not* comprise visual *representations* of music information (such as sheet music) but visual information itself that is closely related to music. Especially in popular music, there is often a concept behind a record release which also includes cover and album artwork. This could be considered as an additional visual facet linking MIR to image retrieval. While it does not describe the music directly, it is closely related to it and gives a context.[3] In fact, this visual context may have a great impact on the perception of a musical piece. For some people, it may even influence the decision of whether or not to buy a specific record as, e. g., documented in a study by LAPLANTE [115]. Moreover, as popular music genres usually have specific imagery associated to them, images can be used as sources for music similarity, artist relatedness or genre classification.

Furthermore, a link to video retrieval can be drawn when also taking into account music videos that are specifically produced for music pieces. Moreover, a lot of music is used in movie soundtracks and not seldom especially composed as such. Including such information, new query scenarios emerge. e. g., DUNKER et al. [63] describe a MIR

2 Some of the bibliographic information may in fact be predicted from the content though currently not with a high precision as this is still a very difficult problem. For instance, there are tasks for the identification of the (performing) artist or classical composer from an audio recordings as part of the Music Information Retrieval Evaluation eXchange (MIREX) [url:34].

3 Hence, the visual facet may also be seen as a special sub-facet of the contextual facet.

system that retrieves suitable background music for a photo slide show.[4] Similarly, this could be done for video snippets as query.

2.1.2.11 *Contextual Facet*

During the last years, the scope of MIR has been widely extended towards further contextual information. BROCHU, DE FREITAS, and BAO [23] argue that *"an essential part of human psychology is the ability to identify music, text, images or other information based on associations provided by contextual information of different media"*. A broad overview is given by SCHEDL and KNEES [214]. Apart from the song lyrics and visual information already covered in Section 2.1.2.8 and Section 2.1.2.10 respectively, they differentiate the following types of contextual information usable to enrich the notion of similarity in MIR systems:

- playlists (or usage context in general),
- term profiles extracted from related textual web resources such as artist websites or CD reviews,
- collaborative tags obtained from social tagging platforms such as *Last.fm* [url:22],
- page counts and web co-occurrences of music entities (usually artists) – either on arbitrary web pages or on specific platforms or services,
- peer-to-peer network co-occurrences of music entities (songs, albums or artists) in shared collections of users,

The contextual facet gains increasing importance. A survey of music information needs, uses, and seeking behaviors [119] indicated a high interest in contextual metadata. At the ISMIR 2009 conference even a whole tutorial was dedicated to "Mining the Social Web for Music-Related Data" [5].

2.1.3 *Multi-Representational Challenge*

Traditionally, MIR approaches can be divided into two "worlds" according to the representation of the music content they work with – either an *acoustic* or a *symbolic* representation as illustrated by Figure 1. The two forms carry significantly different and to some extent complementary information and are of interest for different types of users and their information needs. However, looking at the variety of facets covered by Section 2.1.2, this turns out to be rather an oversimplification because there are facets that belong to neither of the two worlds (e. g.,

4 The reference serves here only as an example for the query scenario. In fact, a different approach is taken than the one motivated here: Both, photos and music pieces, are mapped into a mood space where similarity can be computed.

the visual facet addressed in Section 2.1.2.10). Further, more and more hybrid MIR systems emerge that combine acoustic, symbolic and other representations such as the *Score-Lyrics-Audio-Video-Explorer (SLAVE)* [235]. The following sections address the acoustic and symbolic representation in detail and further take a glimpse at other possibilities for representing music information.

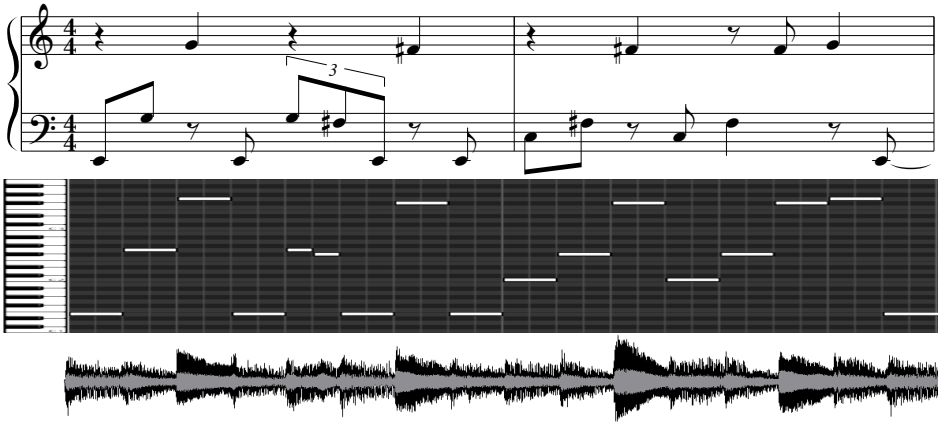


Figure 1: Different representations of music content: symbolic score (top) and time-stamped event-based piano roll (middle) in contrast to an audio recording of a guitar shown as waveform (bottom).

2.1.3.1 *Acoustic Representation*

Recordings of *performances* of a work – either in a studio or live – are commonly stored as digital audio data. This can be either compressed (e. g., MP3) or uncompressed (e. g., WAV). A recording comprises one or more *channels* – up to multiple channels per instrument or voice but this is very uncommon. Most audio recordings publicly available are mixed for stereo playback and thus contain only two channels (left and right). The data in each channel describes an audio *waveform* as shown in Figure 1 (bottom) – a time series of the (sound) air pressure measured by a microphone. The *sample rate* specifies how many amplitude values – called *samples* – are stored in the time series per second, e. g., 44100 for CD quality. A higher sample rate results in a finer frequency resolution. To represent a sine wave with frequency f, a sample rate of at least $2f$ is required according to the Nyquist-Shannon sampling theorem [143, Chapter 5]. The amplitude resolution is determined by the *bit-depth* – the number of bits per sample, e. g., 16bits for CD quality.

Music information in this representation is widely available and it is relatively easy to build large collections if intellectual property issues are resolved. However, it takes a lot of storage space. Further, in this representation the content can hardly be called "information"

– rather "data" – because it is so low-level as it merely describes a signal. Nevertheless, the signal contains valuable information associated with various facets – most importantly about the dynamics (of the recording) and the timbre that can hardly be obtained from other representations. In fact, if it is the recorded performance of a work from score, it should contain all the information carried by the original symbolical representation – to some extent augmented by the interpretation of the performance artists. However, the extraction of this information is computationally expensive and in many cases not yet sufficiently well solved. Recordings of performances may be the only available form of a work. This is often the case for traditional music or jazz.

2.1.3.2 *Symbolic Representation*

Here, ORIO [176] refers to musical *scores*, *"a structured organization of symbols, which correspond to acoustic events and describe the gestures needed for their production"*. He differentiates two groups of musical parameters that can directly be represented by scores: general parameters (main tonality, modulations, time signatures, tempo, musical form, number of voices and instruments, and repetitions) and local parameters (individual tones with relative position, duration and possibly intensity). On the other hand, scores can hardly cover aspects of the performance, recording and production such as timbre, articulation, room acoustics, and *spatialization*[5] of sound sources. It is possible to consider the score as an approximate representation of a musical work as it is impossible to represent all the possible nuances of musical gestures with a compact symbolic representation. However, the opposite view is possible as well where the symbolic representation is seen as the ideal version of a musical work, to which performances are only approximations.

BYRD and CRAWFORD [30] further differentiate between *music notation* (actual scores) comprising sheet music of any form (Figure 1; top) and representations based on *time-stamped events*. The most prominent example for the latter is the Musical Instrument Digital Interface (MIDI) format [url:32] that is oriented towards the representation of musical gestures performed on digital instruments such as keyboards or electronic drum sets. As MIDI can be used directly to control sound generators (e. g., audio synthesizers or samplers) and this way produce some kind of performance, it can also be considered as a compromise intermediate format bridging the symbolic and acoustic world [176]. A large variety of other formats based on time-stamped events exists [218] – most of them with a special application or purpose.[6] A typical visualization for event-based music information is the *piano roll* (using

5 Spatialization is the spatial positioning of sound sources in the audio mix (usually stereo) as a compositional element in recording and production.

6 An exhaustive list is maintained at [url:5].

the metaphor of punched rolls used in automatic pianos) shown in Figure 1 (middle). The horizontal axis represents time and the vertical axis pitch such that notes are horizontal lines with the onset as the beginning and the length according to the duration.

Whilst the event-based representation has only little structure, the structure information in music notation is very rich: e. g., voices are clearly separated and repetitions and parts are annotated. Such information can be represented with some of the aforementioned formats as, e. g., the popular *MusicXML* format [76] that exploits the powerful features of Extensible Markup Language (XML) as a portable format for information exchange.

Byrd and Crawford [30] point out parallels to text considering the event-based representation as the analog for plain text and music notation for structured text with markup respectively. In this analogy, the equivalent for audio data is (recorded) speech. The comparison can be extended by additionally taking into account sheet music that is only available in printed form or as hand-written manuscript. This is similar to printed or hand-written text. Both can – with some limitations – be scanned and converted into a (machine readable) symbolic representation.[7] For music, this processing step is called Optical Music Recognition (OMR) [204].

2.1.3.3 *Other Representations*

The variety of facets described in Section 2.1.2 underlines that music information is truly multi-modal and goes beyond the musical content covered by acoustic or symbolic representations. Lyrics (Section 2.1.2.8) are usually represented as plain text but also more structured formats are possible – e. g., many karaoke programs use text formats that include timetamps and possibly (approximate) pitch information. Metadata – e. g., bibliographic ((Section 2.1.2.9) – and other contextual information (Section 2.1.2.11) are often represented as text or tags. Further, more sophisticated structures such as graphs can code relations between musical entities, e. g., connecting collaborating artists. For associated images and videos (Section 2.1.2.10) respective formats exist – but these shall not further be discussed here. The next section turns away from the facets of music information and their representations and instead addresses the users of MIR systems and their information needs.

2.1.4 *Multi-Disciplinarity Challenge*

As the introduction of this chapter pointed out, MIR is a truly multi-disciplinary field of research. In his description of the multi-

7 Before the conversion, printed or hand-written sheet music is already in a symbolic representation. However, while this representation of the music information would be well suited for an artist, it is not directly accessible for a machine.

disciplinarity challenge, DOWNIE [60] only focused on problems caused by the variance in evaluation paradigms between the disciplines. These have largely been solved in the meantime – mostly thanks to the Music Information Retrieval Evaluation eXchange (MIREX), which has been a part of the annual ISMIR conferences ever since 2005. Whilst MIR researchers are multi-disciplinary, likewise, the audience that may benefit from MIR is very diverse. Here, TYPKE, WIERING, and VELTKAMP [237] distinguish three main user groups or roles related to music:

1. *industry* – e. g., recording, broadcasting, performance
2. *consumers*
3. *professionals* – e. g., performers, teachers, musicologists

They further identify the following four most important levels that such users of an MIR system may address:

1. *(work) instance* – the individual score or audio object
2. *work* – a set of instances considered as essentially the same[8]
3. *artist* – creator or performer of the work
4. *genre* – music that is similar at a very generic level

These audiences and specificity levels define the two dimensions of the "task space" shown in Figure 2. TYPKE, WIERING, and VELTKAMP [237] map the following typical MIR tasks onto that space (together with references to existing MIR systems which are, however, omitted here):

- *copyright and royalties* – receive payments for broad-cast or publication of music
- detection of *plagiarism* – the use of musical ideas or stylistic traits of another artist under one's own name
- *recommendation* – find music that suits a personal profile
- *sounds as* – find music that sounds like a given recording
- *mood* – find music that suits a certain atmosphere[9]
- *emotion* – find music that reflects or contradicts an emotional state

8 Here, the term "work" is used to refer to an abstract piece of music (e. g., a song) in contrast to an artist's lifework.

9 Whilst the terms "mood" and "emotion" are used interchangeably in a recent survey on music emotion recognition by KIM et al. [107], they are not synonyms as, e. g., pointed out by THAYER [234]. Most importantly, moods differ from emotions in that they are longer lasting, less specific, and less intense.

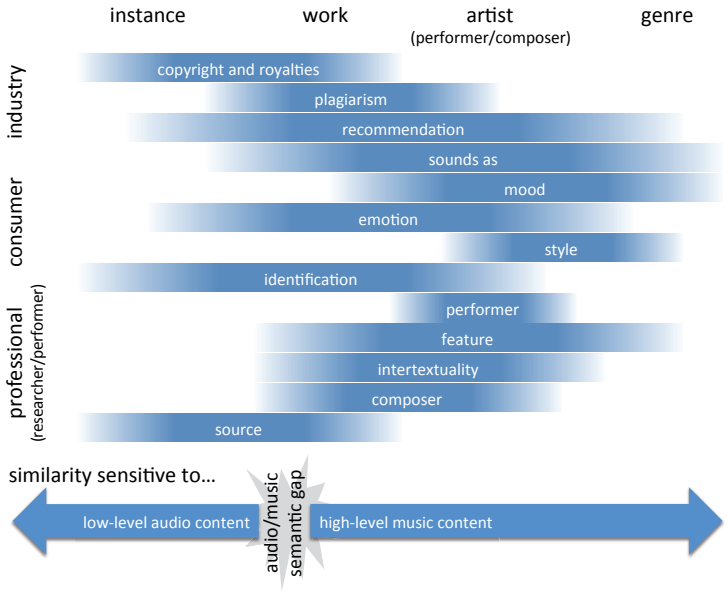


Figure 2: Typical MIR tasks mapped according to target user group and specificity level. (Adapted from [237] and [34].)

- *style* – find music that belongs to a generic category, however defined
- *identification* – ascribing a work or instance to an artist or finding works containing a given theme (query by humming)
- *performer* – find music by (type of) performer
- *feature* – employ technical features to retrieve works in a genre or by an artist
- *intertextuality* – finding works that employ the same material or refer to each other by allusion
- *composer* – find works by one composer
- *source* – identifying the work to which an instance belongs, for example because metadata are missing

The list is most likely not at all exhaustive and the specificity level is somewhat fuzzy. However, it demonstrates the large variety of tasks and information needs that along with the differing background and expectations between the possible user groups make it hard to build MIR systems for the general audience. As, e. g., pointed out by Orio [176], it is most likely that the relevance of the facets described in Section 2.1.2 varies with the retrieval task and the level of the user's expertise. The next section further elaborates on this problem by focusing on the personal experience of music.

2.1.5 *Multi-Experiential Challenge*

DOWNIE [60] points out that music ultimately comes to life in the mind of its perceiver and that there are many ways to experience music. A vast – though most likely not even exhaustive – variety of *listening modes* has been identified by HURON [97], e. g.:

- *allusive listening* – relating moments or features of the music to similar ones in other musical works,
- *reminiscent listening* – reminding the listener of past experiences or circumstances (including emotions) in which the music was previously heard or encountered,
- *identity listening* – trying to answer "what is" questions, e. g., identifying instruments, chords, language, composer, or style,
- *retentive listening* – aiming to memorize what is being heard,
- *directed listening* – focusing the attention, e. g., on a single instrument,
- *ecstatic listening* – causing "shivers", and
- *kinesthetic listening* – motivating the listener to move along to the music

The currently engaged listening mode as well as the general mood may be very different from user to user and to some extent influenced by the musical background including past listening experiences and musical education. Consequently, MIR systems that aim at a wide spectrum of intended audience and intended use have to face questions about the very nature of music similarity and relevance [60]:

1. How can the similarity of a user's experience of one piece with others in a collection be assessed?
2. How can a desired mood or physiological effect be considered as "similar to" or "associated with" a musical piece?
3. How should an "experiential" similarity measure be modified over time to account for the change of the individual users' mood and perceptions?
4. How can relevance judgments be adjusted in a scenario of ever-shifting moods and perceptions?

These questions still remain far from being solved. They demand for the application of adaptive techniques and are to some extent addressed in this thesis.

2.2 COMMON APPROACHES

After the rather theoretical discussion of the general challenges of MIR, this section now takes a more practical perspective by outlining common approaches used in MIR. First, Section 2.2.1 describes the general retrieval process. Then, Section 2.2.2 and Section 2.2.3 cover common practices of dealing with symbolic and acoustic representations of music, respectively. Finally, Section 2.2.4 addresses music similarity as central issue in MIR and the key to many applications.

2.2.1 *The General Retrieval Process*

The general information retrieval process within a MIR system can roughly be divided into three parts as illustrated by Figure 3: At the center, there is the actual core retrieval system consisting of a retrieval model, some index and database structures and the respective algorithms for structuring, ranking or classification. The user-interface part facilitates interaction with the user. It allows to pose queries and presents the retrieval result. On the other end, the data-interface part provides the connection to the music (and music-related) data. In some cases, an additional component for gathering the data may be involved such as a crawler which is, however, not further covered here.

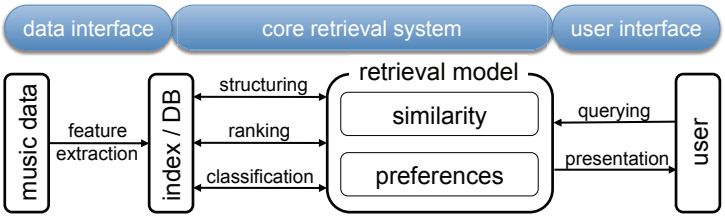


Figure 3: Model of the general information retrieval process based on NÜRNBERGER and DETYNIECKI [171].

Arguably, this model, which is derived from traditional information retrieval, has several shortcomings. Most importantly, it does not elaborate on the user interaction, which very much depends on the specific retrieval scenario. For instance, the query and result presentation may be incrementally refined in an iterative process which incorporates some form of *relevance feedback*. In the exploratory search scenario described below, there is no explicit query in the traditional sense. Instead, it is expressed implicitly in the navigation path. Nonetheless, the model is well suited to illustrate the general relations of the essential system components and will hereinafter be used as orientation guide by putting them into context.

As already pointed out in Section 2.1.4, there is a large variety of tasks and information needs which a MIR system may address. In general, the following retrieval scenarios can be differentiated:

- *Search by Query*:
  In this classic retrieval scenario, the user poses a specific query and the system usually returns a list of results ranked by relevance. Here, appropriate index structures facilitate a fast lookup of relevant objects and the retrieval model is used to compute the relevance scores required for ranking. Queries can be descriptive or examples. The former requires a higher-level feature representation of the objects in the database – properties that the user can refer to in the query to describe his information need, e. g., "genre=pop type=ballad year=1990..1999 tempo=70..80" for pop ballads from the 1990s with a tempo between 70 and 80 Beats Per Minute (BPM). The latter also works with low-level features which can ideally be extracted from the example in the same manner as from the objects in the database. A special form of this more popular query-by-example scenario is Query-by-Singing/Humming (QBSH). Several QBSH systems are, for instance, covered in the survey by TYPKE, WIERING, and VELTKAMP [237].

- *Exploratory Search*:
  In the exploratory scenario, the user searches for relevant information by navigating or browsing the information space. This is typically used in situations when the user is unable to formulate a specific query or just wants to get familiar with a dataset. Here, appropriate means for navigation and orientation are required. Often, this is supported by sophisticated visualization techniques like maps or networks. Many exploratory MIR systems furthermore rely on structuring or classification techniques.

- *Structuring*:
  In this scenario, the user is looking for internal structures of the dataset which can be considered as meta-information. Structures may be based on similarity or – more abstractly – on links/connections. Similarity-based structuring – usually referred to as *clustering* – aims to group similar objects together such that the inner-group similarities are high compared to the intra-group similarities. For instance, a folk song researcher might want to identify clusters of similar tunes that may share a common origin as described in Section 8.4. Link-based structuring approaches, on the other hand, connect related objects and this way construct complex network structures. Both types of structures can also be used internally for the other scenarios – most importantly for exploration by providing an aggregated overview (clustering) or a structure for navigation (network).

- *Classification/Prediction*:
  Here, the MIR system has to predict class labels – or more gener-

ally feature values. As a typical example, classic recommender systems aim to identify (previously unknown) songs, albums or artists that are likely to be of interest to the user. Furthermore, classification/prediction may also be used in an internal process of the core retrieval system with the goal to derive higher-level features which then can be used in other scenarios. For instance, *auto-tagging* aims to automatically assign descriptive tags based on existing manual annotations.

2.2.2 *Working with Symbolic Content*

It is possible to code monophonic melodies as simple one-dimensional strings of characters – each describing pitch and duration of one note or one pair of consecutive notes. Alternatively, characters can represent intervals or simply the direction of the melody (up, down or same tone). This way, the melody contour is captured. Using such string codings, common string matching algorithms can be applied such as *Knuth-Morris-Pratt* and *Boyer-Moore* for exact matches or dynamic programming techniques like the *Levenshtein distance* (commonly referred to as "edit distance") for approximate matching. Also local sequence alignment algorithms popular in bioinformatics can be used.

For polyphonic music, more complex codings and distance measures are required. For instance, Typke et al. [238] propose to interpret scores (and queries) as sets of weighted points in two-dimensional space where coordinates reflect the pitch and onset time of notes and weights depend on the notes' duration and importance. These point sets can be compared using transportation distances such as the *earth mover's distance*.

Further research focuses on rhythmic aspects of melody similarity which appear to be especially important for folk-song research as a rhythmic stability within the oral transmission of melodies can be observed [242]. Another direction of research is to model harmonic progression (i. e., sequences of chords) as parse trees obtained through automatic analysis with a special grammar [49]. As the main focus of this thesis lies, however, in the acoustic domain, MIR approaches working with symbolic content shall not be discussed any deeper.

2.2.3 *Working with Acoustic Content*

Working with audio signals directly is not feasible because this is a too fine temporal scale and also a low level of abstraction to represent perceptually salient information. Instead, meaningful features have to be extracted that describe a recording in a more abstract way and cover perceptively relevant aspects. Based on such features and appropriate distance or similarity measures, recordings can be compared. Depending on how close a feature is to the original audio signal or – from the

Table 2: Common low-level features used to describe audio recordings that are referred to in this thesis.

Feature	Description
Spectral Centroid	Center of gravity of the (frequency) spectrum – related to the perceived brightness of the sound.
Spectral Flux	Change of the spectrum determined by the (usually Euclidean) distance between two consecutive spectra.
Pitch-Class Profile (Chromagram)	Vector containing the energy per pitch class (aggregated over all octaves) with 12 dimensions (or multiples for higher resolution).
Mel-Frequency Cepstral Coefficients (MFCCs)	Coefficients describing the nonlinear spectrum of the frequency spectrum with frequency bands equally spaced on the *mel scale* approximating the response of the human auditory system. (The term "cepstrum" results from reversing the first syllable of "spectrum".)

other perspective – how strong its abstraction, it is called a *low-level* or *high-level* feature respectively. Some features may also be classified as *mid-level*. Gouyon et al. [79] give a comprehensive overview of state-of-the-art algorithms for the automatic description of music audio signals. They also attempt a systematic distinction between low-, mid- and high-level features alternatively referring to them as "signal-centered", "object-centered" and "user-centered"/"semantic" descriptors respectively.

2.2.3.1 *Low-Level Features*

Low-level (or signal-centered) features are commonly computed directly from the music audio signal. Whilst it is also possible to derive global low-level features that describe a recording as a whole, low-level features often refer only to short, possibly overlapping segments of the audio signal called *frames*. The length of the frames differs from feature to feature and is usually chosen in such a way that there are not multiple distinguishable events covered by one frame. It can be constant – resulting in frames of equal size – or variable such as *beat-synchronous* segmentations that align frames to musical beat boundaries.

Many low-level audio features are based on the short-time frequency spectrum of the audio signal. It can be computed by "sliding" a window function – usually a non-negative smooth bell-shaped curve such as a Gaussian – over the signal and apply a (short-time) Fourier transform to obtain the sinusoidal frequency and phase content of each windowed section. Some common low-level features that are referred to in this thesis are listed in Table 2. For further reading, Herrera, Serra, and Peeters [90] describe the low-level audio descriptors defined as part of the *MPEG-7* standard. Furthermore, Peeters [193] covers a large set of low-level audio features for sound description developed within the *CUIDADO* project.

2.2.3.2 *Feature Aggregation*

Low-level features computed per frame/segment can be aggregated in several ways which may lead to some higher abstraction. CASEY et al. [34] differentiate the following common aggregation methods:

- *Bag-of-frames (or bag-of-features) models* capture global statistics of feature values (per recording). The value distributions are commonly described by single Gaussian distribution or Gaussian Mixture Models (GMMs). A popular application is to use GMMs of Mel-Frequency Cepstral Coefficients (MFCCs) to model the general timbre of a recording as, e. g., described by MANDEL and ELLIS [141].
- *State models* capture fine spectral-temporal structure by mapping (usually multi-dimensional) features to simple states. For instance, Hidden Markov Models (HMMs) are especially popular to predict chord labels for pitch class profiles [186].
- *Audio Shingles* are (possibly overlapping) sequences of feature values – each concatenated into a single high-dimensional vector – that represent coarse temporal context. Such an aggregation technique can, e. g., be used to identify near duplicates [33].

The first approach completely neglects sequential information in contrast to the others. It is motivated by an analogy to the popular *bag-of-words* model in classical (text) information retrieval that neglects word order and grammar. Instead, it considers a text document as collection (multi-set) of terms – only taking their (statistical) distribution into account. With such a model, only the parameters of the distribution need to be stored for each document. For texts, this is a vector of term weights that can be interpreted as a histogram of the term frequencies in a document.[10] For audio recordings, usually a Gaussian distribution or a GMM of the feature values is assumed whose parameters (means and covariances) need to be stored.

Looking at the analogy to bag-of-words, there is no direct correspondence for a word or term – basically, only the general idea of working with the value distribution has been adopted. However, it is possible to further elaborate the analogy by adding a *quantization* step originally proposed in the image retrieval domain [43]: A *dictionary* or *codebook* is built by clustering all (usually multi-dimensional) feature values for all frames/segments of all recordings in the collection. Using a prototype-based clustering technique such as k-Means or Learning Vector Quantization (LVQ) to group similar feature values into clusters,

10 This is a simplification for the sake of illustration: Actually, (term) weight vectors in (text) information retrieval usually contain so called *TFxIDF* weights that depend not only on the frequency of the term within the document (term frequency) but also on the frequency of the term throughout the whole collection (inverse document frequency). Therefore, the weight vectors are not histograms in a strict sense.

a set of "typical" features values (one for each cluster) is derived - the analog form for terms, finally. This set, which has a predetermined size, is the dictionary. Based on this dictionary, a histogram-like vector can be computed for each recording – much like for texts – by mapping all feature values that occur within to the dictionary and aggregating the frequencies. An advantage of this compared to the above described conventional bag-of-frames approach is the much simpler vector representation of the documents and consequentially the applicability of basic distance measures such as the Euclidean distance. Further refinements of this approach are possible. For instance, dimensionality reduction techniques can be applied additionally as, e. g., proposed by McFee, Barrington, and Lanckriet [151] – similar to latent semantic indexing [50] in text information retrieval.

2.2.3.3 *High-Level Features*

Low-level audio features and aggregated representations are often only the first stage in a bottom-up processing strategy with the goal to cross the infamous *semantic gap*[11] between these close-to-signal descriptors and the concepts that user actually rely upon when relating to music. This problem is, for instance, described in detail by Celma, Herrera, and Serra [37]. High-level descriptions are (mostly) more intuitive for humans to manipulate and search – and may enable high-level retrieval scenarios otherwise hardly possible. For instance, to effectively retrieve cover versions of a song, musical content needs to be compared on a very abstract level incorporating high-level features such as melody, harmony, key, rhythm, and structure.

Extracting high-level music content descriptions is subject to intensive research in MIR. Popular MIR tasks covering many of the facets discussed in Section 2.1.2 are:

- beat tracking (including tempo estimation and detection of note onsets, meter and measure beginnings)
- melody and bass estimation (usually based on source separation techniques)
- chord and key recognition
- music structure analysis and segmentation (on a larger scale)
- instrument/performer/composer recognition
- mood/emotion recognition

Most of these disciplines exist also for symbolic content but the main focus of research is on audio. Further, some work has also been done on extracting high-level features from other modalities than symbolic

11 The McGraw-Hill Dictionary of Scientific & Technical Terms defines the semantic gap as *"the difference between a data or language structure and the objects that it models."* [153]

or acoustic content: For instance, MAYER, NEUMAYER, and RAUBER [149] introduce style and rhyme features for lyrics processing and HIRJEE and BROWN [92] propose high-level features for the comparison of rhyming styles.

2.2.4 *Specificity of Music Similarity*

Musical similarity is a central issue in MIR and the key to many applications as illustrated by Figure 3 on page 21 but not a simple concept as, e. g., pointed out by WIERING [245]:

1. There are many interrelated features and facets (Section 2.1.2) that have to be considered.
2. There is the aspect of perception and experience of music – raising questions about the very nature of music similarity and relevance (Section 2.1.5).
3. There are different "types" of music similarity that depend on the context given by the user group and the task at hand (Section 2.1.4).

CASEY et al. [34] call the latter "similarity specificity".[12] They arrange common MIR tasks according to their similarity specificity in a spectrum that corresponds largely with the specificity levels described by TYPKE, WIERING, and VELTKAMP [237] (Section 2.1.4). It is therefore incorporated into the illustration of the MIR task space in Figure 2 (bottom) on page 19. On the left end of the spectrum are fingerprinting systems that allow queries for the same *audio* content (i. e., an instance of a work) with high specificity. High-to-mid specificity systems retrieve nearest-neighbor sequences instead – such as remixes or recordings that contain samples (i. e., parts) of the query. On the other side of the audio-music semantic gap are applications that require a similarity that is invariant to specific *audio* content and instead more sensitive to high-level *music* concepts. CASEY et al. [34] note that all of the tasks with mid-specificity are addressed by approaches that consider sequences of features whereas aggregated bag-of-frame representations that ignore sequence information are applied for low-specificity tasks such as those related to genres.

2.3 SUMMARY

This chapter introduced the research field of Music Information Retrieval (MIR) as the general context for the underlying work of this thesis. As shown in Section 2.1, dealing with music information poses

12 CASEY et al. [34] consider only acoustic similarity but their findings can be generalized to music similarity at large.

several challenges. Most importantly, music information has many facets and can be represented in multiple (multi-modal) ways. At the same time, users of MIR systems are very diverse – w.r.t. their musical background and roles related to music – resulting in very different information needs and perception of music. Consequently, MIR systems need to be able to adapt in order to address a broader audience or support different tasks.

Section 2.2 outlined the general MIR process and introduced the different retrieval scenarios. Furthermore, the common practices for dealing with symbolic and acoustic content representations and addressing different levels of similarity specificity have been described. Still, the question remains, how MIR systems can deal with a large variety of content and users.

The next chapter will take a close look at the state of the art in MIR under the aspect of adaptivity and point out how the various challenges are currently addressed. The system model illustrated in Figure 3 will, in the remaining chapters, be used as orientation guide by putting them into context.

Intelligence is the ability to adapt to change.
Stephen Hawking

3

ADAPTIVE MUSIC RETRIEVAL – A STATE OF THE ART

Originally intended as a State-of-the-Art-Report (STAR) for the ISMIR 2010 conference, this survey has been submitted as an article for the Multimedia Tools & Applications Journal.

Following up on the general introduction of MIR, this chapter takes a closer look at adaptive approaches in this field of research. With the development of more and more sophisticated MIR approaches, aspects of adaptivity are becoming an increasingly important research topic. Even though adaptive techniques have already found their way into MIR systems and contribute to robustness or user satisfaction, they are not always identified as such. This survey attempts a structured view on the last decade of MIR research from the perspective of adaptivity in order to increase awareness and promote the application and further development of adaptive techniques. To this end, different approaches from a wide range of application areas that share the common aspect of adaptivity are identified and systematically categorized.

3.1 INTRODUCTION

Why is adaptivity interesting in the context of MIR? Even though the term "adaptive" does not appear directly in recent surveys by Orio [176] and Casey et al. [34] nor in the analysis of the "ISMIR cloud" by Grachten et al. [82], adaptivity is a core element of many MIR applications. Whilst not (yet) being a primary research topic in MIR, studying adaptivity has a long tradition in the field of control theory [4, 158], artificial neural networks and nature-inspired systems in general [71, 145]. In classic (text) information retrieval the term "adaptive information retrieval" was coined in the late eighties by Belew [12] and with "adaptive hypermedia" a whole new research field has emerged in the nineties – tightly coupled with user-modeling [26, 27]. Adaptive systems can help in application areas with high diversity,

- where some decisions cannot be made during system design but have to be taken during runtime,
- where certain initial model assumptions are expected not to be static and dynamic parameters are required, or
- where goals cannot be stated directly a priori but only indirectly by means of feedback during runtime.

To come back to and answer the initial question: Especially in MIR, which has to cope not only with a large diversity in music signals but

also with users with different backgrounds, tastes and perceptions of music and different goals and roles, adaptive techniques have the potential to improve system performance significantly.

This survey looks back at the last decade of MIR research from the perspective of adaptivity and highlights existing MIR approaches that reveal at different levels some kind of adaptivity. The goal is to increase the awareness of adaptive techniques and promote their application and further development in the future. Though the author tried to give a broad and systematic overview of the subject to the best of their knowledge, the coverage is most likely far from being complete. Nevertheless, it gives an impression of the diversity of possible applications and provides a lot of pointers for the reader to dig deeper into this broad field of research.

This chapter is structured as follows: The following section introduces a working definition of adaptivity and proposes a generic model for adaptive systems. Both will be used throughout this thesis to categorize the various approaches. Section 3.3 is the main part where a broad range of adaptive systems grouped by application area is described. Afterwards, Section 3.4 gives a systematic overview and categorization of the works covered in Section 3.3. Finally, Section 3.5 concludes the chapter.

3.2 ADAPTIVE SYSTEMS

Before addressing adaptive music retrieval, the concept of adaptivity needs to be clarified. This is all but trivial as the term is used in the context of several research areas and mostly in an intuitive manner. However, in order to tell whether an approach is to be considered adaptive or not, specific characteristics are required. Therefore, this section attempts to give a definition of the crucial terms used throughout this thesis and in particular this chapter – in the hope that they prove to be useful beyond its scope. The ideas throughout this section are based on recent work by Broy et al. [25] who proposed a universally practicable definition of adaptive systems by means of system engineering.[1]

3.2.1 *A Definition of Adaptable and Adaptive Systems*

Let S be a black-box system whose internal structure and processes are not relevant for the following considerations. Further, S shall be interactive in the sense that it continuously interchanges information with

1 Most importantly, the definition by Broy et al. [25] is generalized here w.r.t. the system context (instead treating user and environment separately) and simplified regarding inputs and outputs. Further, the aspect of goal orientation is added and a generic internal structure of the system is proposed that links adaptable and adaptive systems.

its environment. For the sake of simplicity – but without losing generality – let S have only one input and one output channel.[2] Through the input, S may receive data as well as commands (e. g., issued by a user of the system) which result in some observable or measurable output. The term "output" is used here in the most generic way. It could, for instance, be a (classification) label, a distance or similarity value, a ranked list of objects, a complex visualization, or even a command that controls another system. The *behavior* of S is specified by a set-valued input/output (I/O) function that defines a relation between the input and the output. Specifying the function does not require knowledge about the internals of S as it can be represented by the history of all input/output value-pairs. S is *adaptable* iff it provides means to change its behavior externally by some parameters and this system property is referred to as *adaptability*. The respective change w.r.t. the (invisible) internal structure and (visible) behavior of S is called *adaptation*. Again this is to be understood as generic as possible. An adaptation may range from changing a single parameter value to a complete internal restructuring of S.

Based on this concept, the notion of adaptivity can be defined as follows: For this, let $C(S)$ be the *context* of S in the broadest possible sense. It comprises the (operational) environment, the user context (cf. DEY [52], DEY and ABOWD [53]), and the data (i. e., input/output values and their characteristics). S is called *(context) adaptive* and the respective system property *(context) adaptivity* iff

1. S behaves differently in different contexts given the same input, and
2. the respective adaptation (i. e., the difference in behavior) of S is goal-driven in that it aims to optimize the system's behavior in the given context according to some pre-defined measure.

Although it seems tempting to solely demand (1), this would imply that a system producing random output is considered adaptive – which is obviously not desirable. Additionally, (2) enables evaluation of the system w.r.t. its adaptivity.

3.2.2 *A Generic Model for Adaptive Systems*

Based on the above definition, this section proposes a generic architecture for adaptive systems that can be utilized for system analysis, comparison or classification as shown later in this chapter. Figure 4 shows the schematic of such a generic adaptive system. As the definition implies, an adaptive system requires some means of (partially) sensing its context in a pro-active manner, some kind of knowledge

2 In case of multiple inputs or outputs, these can be combined into a single meta-input or -output respectively.

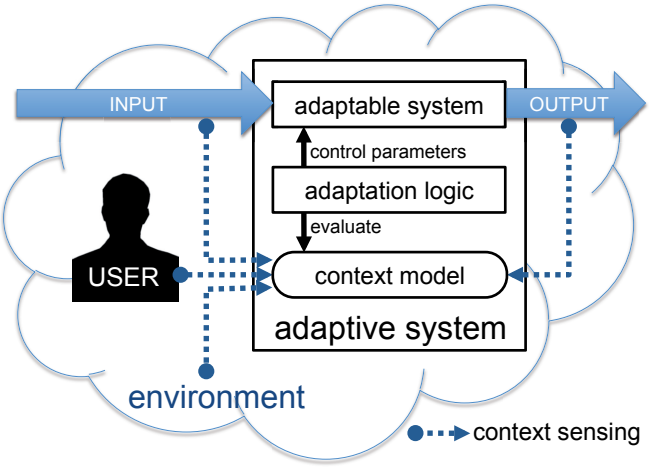


Figure 4:
A generic model of adaptive systems.

representation of the context (*context model*) and some means to change the behavior of the system based on this knowledge. For analytic reasons, the latter part can be – rather artificially – subdivided into an encapsulated *adaptable system* which is responsible for the actual processing of the input and an *adaptation logic* that evaluates the context model and chooses appropriate parameters (according to the goal) for the adaptable system. Optionally, the means for (partially) adapting the internal adaptable system may be made accessible from the outside making the system both, adaptable and adaptive. Additionally, a dynamic adaptation logic can be achieved by substituting it with another adaptable or adaptive system. Through recursive substitution, arbitrarily complex adaptive systems can be modeled. Similarly, it is possible to concatenate or cascade adaptive systems in case this is helpful for system analysis and modeling.

3.3 APPLICATIONS IN MIR

There are various possibilities to integrate adaptive techniques into the general retrieval process introduced in Section 2.2.1. Figure 5 shows how this section subdivides the different approaches according to their field of application.

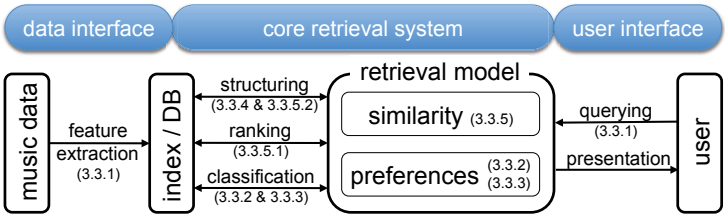


Figure 5: Model of the general information retrieval process (described in Section 2.2.1) with references to sections that address the respective application areas.

Table 3: Typical adaptation goals of the different retrieval components shown in Figure 5.

adaptive component	adaptation goal
user-adaptive querying	better "understanding" of the user's information need
user-adaptive presentation	increase understandability
user-adaptive structuring	structures that reflect the user's individual way of structuring
data-adaptive structuring	respond to changes in the dataset
user-adaptive ranking	rank according to the user's understanding of relevance
data-adaptive ranking	increase diversity within the result list
user-adaptive classification	reflect the user's classification criteria
data-adaptive feature extraction	increase robustness and quality of the extracted features

A brief overview of typical adaptation goals for the different components is given in Table 3. In general, components belonging to the user interaction part can be adapted to the user: For query components, the goal might be to better "understand" what the user means by the query - i. e., to reduce its "fuzziness". This may, for instance, involve the compensation of errors or enriching the query with implicit information derived from the context. The adaptation of presentation parameters usually aims to increase the understandability of the retrieval results and hence facilitate a better user experience. In contrast to this, the feature extraction part can benefit from data-adaptivity which may increase the robustness and overall quality of the extracted information. The core retrieval system may comprise both, data- and user-adaptive components: Data-adaptive structuring techniques can respond to changes in the dataset whereas user-adaptive methods aim to construct structures that are easier (and at best intuitively) understandable for a specific user by reflecting his way of structuring. Rankings can be adapted w.r.t. the data in order to increase the diversity or according to some learned user preferences such as a feature weighting. User-adaptive classification aims to reflect the user's classification criteria.

3.3.1 *Adaptive Feature Extraction*

A lot of research in MIR is focused on improving the quality and robustness of content-based features. As shown in the following, adaptive techniques can already be applied at a very early stage of the extraction process. At this level, adaptivity itself is mostly only a minor aspect of the specific contributions. Nevertheless, applying adaptive techniques can still make a significant difference. Often, the general rationale is here to guide the extraction of a critical feature by means of the exploitation of other co-occurring feature(s) or of larger spectro-temporal contexts.

3.3.1.1 *Peak Picking with Dynamic Thresholding*

There is a wide range of feature extraction approaches that apply adaptive thresholding to identify peaks in the input signal. Publicly available MIR tools such as *aubio* [24] or the *MIRToolbox* [116] have long been providing methods for peak picking with dynamic thresholds. The dynamic threshold is derived from a short (sliding) window of the input values. For onset detection, peaks above this threshold with sufficient loudness can be selected [39, 203, 252]. Apart from this common application, peak picking with dynamic thresholding has also been used in the context of audio identification [16]. Further areas where dynamic thresholding is applied comprise audio to score matching [162] and segmentation of audio recordings [144, 232]

3.3.1.2 *Context-Sensitive Chord and Key Recognition*

When it comes to infer chords or key information from audio data, context information has proven to be very valuable as most approaches make assumptions that do not necessarily always hold: For instance, not all music recordings adhere to the standard tuning frequency of 440 Hz as pointed out by Lerch [125] and several approaches have been proposed to adapt the tuning accordingly [87, 194, 255]. Using beat-synchronous chroma features (which requires context information about onsets) can increase performance over frame-by-frame approaches [14]. Furthermore, certain (vague) knowledge about the chord probabilities can be exploited to correct possible misclassifications in a post-processing step. In the case of the popular HMM-based approaches [219] the post-processing is implicitly done in the Viterbi decoding using the chord history as context information. The Viterbi decoding is an algorithm that finds the optimal path in a sequence of states based firstly on the similarity of the observed input to the state output probability distribution and secondly on the transition probabilities contained in the transition matrix. The transition matrix may be learned [219] or manually defined encoding musical knowledge [14, 186]. For additional adaptation, the HMM can also be trained on the currently processed piece using the standard Expectation Maximization (EM) algorithm [14]. HMMs have also been proven suitable for the closely related task of key detection [121, 170]. The context information about the key of the piece can be exploited as a third option for smoothing chord predictions [137, 220]. An approach that integrates bass pitch estimation in a probabilistic framework for hypothesis-search-based chord recognition is described by Sumi et al. [233]. Mauch and Dixon [148] use a dynamic Bayesian network with four hidden source layers jointly modeling metric position, key, chord, and bass (pitch class) and two observed layers corresponding to low-level audio features for bass and treble tonal content as shown in Figure 6. This sophisticated modeling of the musical context allows to differentiate between 109

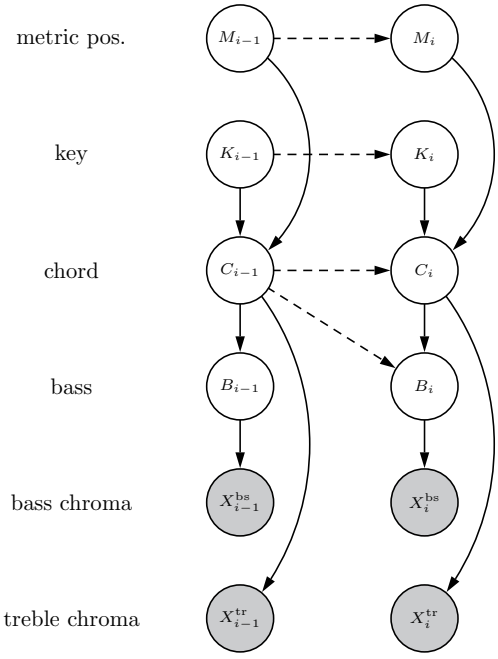


Figure 6: Illustration from [147] of the sophisticated modeling of direct dependencies between observable low-level audio features (gray) and high-level concepts for 2 consecutive beats.

different chord classes – compared to the 24 major and minor chords commonly addressed – with a classification accuracy that exceeds the previous state of the art significantly.

3.3.1.3 *Tempo Estimation with Adaptive Window Size*

Müller et al. [164] extract tempo parameters from expressive music performances using an onset-dependent adaptive window size. The approach is based on the assumption that note onsets are the main source for inducing tempo information. The motivation is, that in order to obtain a meaningful tempo estimation, a larger window is needed in passages where only few notes are played whereas otherwise a smaller window size suffices. Thus, instead of specifying the window size in terms of a fixed number of input values, the authors propose a definition in terms of a fixed number of inter-onset-intervals.

3.3.1.4 *Key Recognition with Adaptive Frequency Weighting*

Chuan and Chew [40] describe a fuzzy analysis technique for pitch-class determination for polyphonic audio key finding which incorporates an adaptive level weighting step. This step aims to increase the robustness of the pitch values obtained from a Fast Fourier Transform (FFT) of the signal. (From the resulting values, pitch class profiles are derived and later used to determine the key.) The respective

weight for each predefined pitch range is inferred from information about the density of the signal in that range. The authors state that this weighting is particularly effective in clarifying low pitches in late romantic music.

3.3.1.5 *Pitch Detection with an Adaptive Strategy*

An approach for automatic violin transcription with an adaptive strategy for pitch estimation is presented by Loscos, Wang, and Boo [133]. The pitch detection process is guided by a harmonic descriptor that classifies the input signal as monophonic or polyphonic. Based on this information, either simple or more demanding pitch estimation techniques are applied accordingly.

3.3.1.6 *Adaptive Optical Music Recognition*

OMR approaches have to deal with a large content variability – especially for older sheet music. During the normal usage of any OMR tool, errors are manually corrected by the user. This is valuable feedback information that can be used by an adaptive OMR system to adapt on-the-fly to the currently processed material. Such an approach is described by Pugin, Burgoyne, and Fujinaga [200]. The system uses an HMM that is pre-trained for the general OMR task but initially does not have any training information w.r.t. the book of scores to be processed. As soon as first corrections are made by the user, this feedback is incorporated into a book-dependent HMM using Maximum A Posteriori (MAP) adaptation. The authors report that the adaptation after only a couple of pages causes a reduction of recognition failures by a factor of nearly two and an increase of the precision at the same time.

3.3.1.7 *Fo-Estimation with Adaptive Tone Models*

Goto [77] describes an extension of an approach for predominant fundamental frequency estimation (PreFEst) that makes use of adaptive tone models. Model parameters are adapted for each song on the basis of the MAP probability estimation by using the EM algorithm. Experiments to estimate the fundamental frequencies of the melody and bass lines in compact-disc recordings with the extended system showed improvements in performance and robustness. The approach is extended by Fujihara et al. [69] to identify vocal sections where the vocal melody line is actually represented by using a two-state HMM with vocal and non-vocal states.

3.3.1.8 *Singer-Adaptation in QBSH*

Query-by-Singing/Humming (QBSH) systems transcribe a sung or hummed query and search for related musical themes in a database,

returning the most similar themes. A QBSH system that is able to adapt to different singers by learning from user-provided feedback on the search results is described by LITTLE, RAFFENSPERGER, and PARDO [129]. To this end, a trainable note segmentation system and an easily parametrized singer error model are proposed. Suitable model parameters are found by a straightforward genetic algorithm. The system was tested on a database containing 1001 melodies with 10 singers and 15 queries each that were drawn from the QBSH dataset of the 2006 MIREX. Results show significant improvement in performance given only ten example queries from a particular user.

3.3.2 *User-Adaptive Recommendation*

The development of music recommender systems has become a well established field of research in MIR – with obvious commercial potential. Adapting recommendations to the user can significantly increase user satisfaction. User preference modeling aims to learn about the user's taste, i. e., which tracks, artists or genres a user likes or dislikes. As taste may change over time, recommender systems should be able to adapt accordingly.

HOASHI, MATSUMOTO, and INOUE [94] train a Tree Vector Quantizer (TreeQ) to discriminate songs that a user likes and dislikes on a small initial training set. The TreeQ is used to generate reference histograms from the bin assignments of all songs belonging to one class. It further quantizes unknown songs into individual histograms that can be compared with the references for classification (both interpreted as vectors). The user can give relevance feedback on the classifications which is incorporated using a method based on the original Rocchio approach for text retrieval [207]. This results in a continuous update of the user model.

MANDEL, POLINER, and ELLIS [140] use a Support Vector Machine (SVM) to classify songs as positive or negative based on user feedback. To refine the classifier, the system asks the user to label only the most informative songs, i. e., those close to the decision boundary. This iterative active learning approach requires half as many labels to achieve the results of the passive learning scheme or, alternately, can increase the precision-at-20 by ten percent with the same number of labels.

Good Vibrations [211] lets the user tag songs with whatever semantic association the user may want to create. Using predictive models, built from content-based audio descriptors and the annotations already provided by the user, the system automatically proposes tags for other songs. Feedback from tag corrections is collected and incorporated into the user model. Based on the concepts, playlists can be generated.

The system described by MOH and BUHMANN [161] gathers information about whether a user likes or dislikes an artist. A user model

is derived using an extension of the online passive-aggressive algorithm – a discriminative kernel algorithm that learns sequentially – with kernel expansion (PA-L-EX). This approach avoids the tendency of kernelized online large margin methods to use more and more support vectors which inevitably leads to constraint violations and high computational costs and space. However, it is still capable of recalling historically learned events and can adapt to changing data distributions as shown in experiments.

3.3.3 *User- & Context-Adaptive Playlist Generation*

Automatic playlist creation is a popular MIR application, which can also be used in a recommendation scenario. One way to adapt playlists to an individual user is to learn from feedback such as skips (Section 3.3.3.1). Further, the playlist generation process can be guided by information about the listening context (Section 3.3.3.2). In this case, the term "context" refers to context-of-use, i. e., the environment w.r.t. to the generic model shown in Figure 4 on page 32.

3.3.3.1 *Learning from Feedback*

PATS (Personalized Automatic Track Selection) by Pauws and Eggen [190] is an adaptive system for playlist generation that learns from user feedback. The system generates a playlist for a specific user context through dynamic clustering. The user can then select songs in the playlist that in his opinion do not fit the current context-of-use. From this preference feedback, new feature weights in the underlying similarity measure are derived by an inductive learning algorithm based on the construction of a decision tree that uncovers the feature values classifying songs into the categories "preferred" and "rejected".

The system described by [248] uses a set of classifiers – each one representing a different high-level characteristic – to generate ranked lists for a seed song.[3] For each song contained in at least one ranked list, a profile vector is created storing the inverse ranks w.r.t. to the classifiers and the seed song. The ranked lists are aggregated and presented to the user who can rate songs as positive or negative. From the profiles of the rated songs, the system can derive which characteristics are most discriminative and adapt the result list accordingly.

The *SatisFly* system uses constraint-satisfaction [192] and a local search procedure based on simulated annealing [191] to construct playlists that meet user-defined constraints. As a fully manual specification of the constraints is required, this system is only adaptable. However, it could easily be turned into an adaptive (and more inter-

How this could be done is described in Chapter 8.

3 The seed song is initially provided by the user. Later, the system uses positively rated songs as seed.

active) system by letting it pro-actively infer constraints from user actions.

Several song selection heuristics for dynamic playlist generation based on content similarity and the user's skipping behavior are proposed by PAMPALK, POHLE, and WIDMER [184]. The goal is to minimize the number of skips which at the same time is supposed to maximize satisfaction. The heuristics are evaluated by several simulations including a simulated transition of user preference from one genre to another. The best performing heuristic was basically to recommend songs close to any of the user's favorite songs, and far away from banned songs. Compared to a purely content-based approach this heuristic drastically reduces the number of skips as an indicator for user satisfaction. For future work, the authors propose to introduce an artist filter that avoids having a large number of pieces from the same artist right after each other and thus increases diversity. PAMPALK and GASSER [181] further elaborate the approach: A playlist generator is presented that allows users to define personalized radio stations (such as "wake-up music") by selecting a seed song or artist. Users can further control the frequency with which songs and artists are repeated. Instead of collecting implicit feedback for skips, this system requires the user to rate recommended songs explicitly. While this makes recommendations more transparent, it also requires a lot more manual adaptation by the user. Heuristics for automatically updating the music on a mobile player based on personal listening behavior are investigated by POHLE, SEYERLEHNER, and WIDMER [199]. The aim is to automatically discard those pieces of music from the player which the listener is fed up with, while new music is automatically selected from a large repository of available music. The same song selection heuristics as proposed by PAMPALK, POHLE, and WIDMER [184] are applied to select new songs. Further, heuristics for removal based on the user's skipping behavior are proposed. The heuristics are tested in a simulation experiment and show significant improvement over the random baseline when weighted according to the number of times a song has actually been listened to.

3.3.3.2 *Adapting to the Listening Context*

Apart from skipping behavior or preferences on what kind of music a user prefers, there has been recent development to exploit information about the listening or usage context. Several motivating papers can be found pointing out possible benefits from such information [44, 206]. In the field of ubiquitous computing, recommender systems for music have been proposed that use easily measurable environmental data to differentiate between listening contexts:

The M^3 music recommender system described by LEE and LEE [118] uses a two-step case-based reasoning approach for context-aware recommendation. First, information about season, month, day of the

week, weather and temperature is used to infer whether the user wants to listen to some music. This decision is made through case-based reasoning on the user's listening history. If music is likely to be desired, a second case-based reasoning step infers whether the music should be slow, fast or may have any tempo. Here, the tempo is estimated from the genre tag, assuming that songs belonging to the genres "Ballad" and "R&B" are slow whereas songs belonging to "Rock/Metal" and "Dance" are fast.

Park, Yoo, and Cho [187] describe a context-aware music recommendation system that apart from weather (temperature, humidity, weather and forecast) and time (season and time of day) data also incorporates the ambient noise level recorded by a microphone and the illumination measured by a sensor. The continuous data is discretized to fuzzy membership vectors with respect to predefined fuzzy sets. The resulting data is processed by a Bayesian network that infers the current context. Explicitly created user-profiles are then used to recommend songs with regard to the context.

Finally, a "music recommender system for the smart office" is proposed by Guan et al. [83]. The system uses basic content-based classifiers to assign the available songs to distinct genre and mood classes. Songs are recommended that comply with the genres specified in the user profile and match the user's current mood. The user's mood is predicted by a naive Bayesian classifier that takes into account the user's location, the time of day, what other people are in a room with him, the weather outside and his stock portfolio.

PAPA (Psychology and Purpose-Aware Automatic Playlist Generation) [174] uses sensors that measure certain bio-signals (such as the pulse) as immediate feedback for the music currently played. This information is then used to learn which characteristics of music have a certain effect on the user. Based on this continuously adapting model, playlists for certain purposes can be created. The *MPTrain* player [175], for instance, selects suitable music for an optimal workout.

Similarly, *PersonalSoundtrack* [64] works with sensor data from an accelerometer and user feedback. According to the authors, the system follows an affective model of arousal, valence and stance where tempo (arousal) directly affects enjoyment and receptivity (valence and stance). Given this model, the goal is to synchronize user arousal (in steps per minute) with music tempo to promote positive valence and open stance. In case a song is explicitly skipped, the playback probability of that song is modified using a very basic update formula.

Another system that automatically selects music appropriate for running exercises by making a correlation between musical tempo and runner's step frequency measured by an accelerometer is described by Niitsuma et al. [168]. First commercial products that include the required sensor hardware are already available such as the *Yamaha BODiBEAT* music player [url:56] or the *Nike+iPod Sport Kit* [url:40].

3.3.4 *Data-Adaptive Collection Structuring*

In the field of MIR, Self-Organizing Maps (SOMs) have become very popular for collection structuring and visualization. They are inherently data-adaptive with the general goal to reflect the (possibly changing) data distribution in the collection: During training, each presented data point leads to a modification of the internal parameters (weights and thresholds) such that the response is amplified the next time. SOM-based systems comprise the *SOM-enhanced Jukebox (SOMeJB)* [202], the *Islands of Music* [180, 185] and *nepTune* [108], the *MusicMiner* [163], the *PlaySOM-* and *PocketSOM-Player* [165] (the latter being a special interface for mobile devices), the *SoniXplorer* [134, 135], the *Globe of Music* [123] and the tabletop applications *MUSICtable* [230], *MarGrid* [93], *SongExplorer* [105] and [54]. Apart from this basic data-adaptivity, it is furthermore possible to allow the SOM to change its structure which adds another level of adaptivity: Growing versions of SOMs as used by DOPLER et al. [58], PAMPALK, FLEXER, and WIDMER [178], RAUBER, PAMPALK, and MERKL [202] can adapt incrementally to changes in the data collection whereas other approaches may always need to generate a new structuring from scratch. Moreover, the approaches described by DOPLER et al. [58], PAMPALK, FLEXER, and WIDMER [178], RAUBER, PAMPALK, and MERKL [202] learn a Growing Hierarchical Self-Organizing Map (GHSOM). Finally, there exist several approaches to similarity-based structuring that are able to adapt their underlying metric according to user preference. Section 3.3.5.2 covers these approaches in detail.

Section 4.3 explains the fundamental concepts of SOMs.

3.3.5 *Adaptive Music Similarity*

A suitable similarity measure is crucial for the performance of many commonly used approaches for clustering, classification or ranking. Hence, metric learning has been a topic of interest in general information retrieval for some time. The general objective is either to get a query closer to the relevant objects (in a classic retrieval scenario) or to refine the decision boundary between relevant and irrelevant objects (in a classification or clustering scenario which does not necessarily require a query).

In the field of MIR, early studies of ELLIS et al. [65] and BERENZWEIG et al. [15] compared existing measures of similarity and addressed issues of consistency in human perception of music and artist similarity. The highly subjective nature of music perception and the large variety of ways to represent and compare music in many "plausible" ways make it hard to manually define and tweak a metric according to the characteristics of the input data and the specific retrieval task. Consequently, there exist only few adaptable systems that employ direct manipulation of the metric as described, e. g., by BAUMANN

and Halloran [10] and Vignoli and Pauws [240]. Instead of asking the user to explicitly state how he compares music, adaptive systems aim to learn a suitable metric from user feedback in an interactive way. Existing approaches can be divided into two groups: Those that generate rankings based on the user-adaptive metric and those that use the metric as means to structure a music collection according to user preferences.

3.3.5.1 *Adaptive Ranking*

The so called "learning to rank" problem has been a topic of much attention in general information retrieval in recent years. Here, machine learning techniques are applied to improve the ranked lists produced by a retrieval system. Approaches include the work on relevance feedback, automatically parameter tuning for existing information retrieval models, or learning optimal feature combinations. Liu [130] gives a comprehensive overview on this subject.

The content-based MIR system for symbolic music described by Rolland [208] adjusts its similarity model based on user feedback received during successive interactions with the user (search sessions). To model the similarity between a transcribed query and a melody, the concept of *pairings* is introduced: A pairing is a part of an alignment (between query and melody) that may comprise several notes and rests. Pairings can be classified into types and for each type, a weight is defined that specifies the importance of the pairing type in the similarity computation. In a ranked list of search results, the user can point out the correct match and optionally some reasonable secondary matches. Given this feedback, the weight for each pairing type is reinforced by a constant update factor if it contributes more to the similarity in the correct match than in the higher ranked false matches or otherwise decreased respectively. This way the system can adapt to the user's way of comparing melodies.

Mandl and Womser-Hacker [142] propose to use the *MIMOR (Multiple Indexing and Method-Object Relations)* fusion approach to cope with the diversity of user needs and data properties. The idea is to linearly combine several information retrieval systems – each one presumably optimized for different tasks – using weights that can be learned from relevance feedback of the users. Though the general idea appears promising, no publications could be found that report about an actual MIR application of the system which has been originally developed for classic text-retrieval.

The goal of the *MUSIPER* system [229] is to construct music similarity perception models of its users through relevance feedback. The system applies Radial Basis Function Networks (RBFNs) – a special form of neural network – to incrementally learn which subset of a set of (content-based) objective features approximates more accurately the subjective music similarity perception of a specific user. An evaluation

that involved 100 users verified the relation between subsets of objective features and individualized music similarity perception, while significant improvement was exhibited in individualized perceived similarity in subsequent music retrievals.

McFee and Lanckriet [152] compute a multi-modal distance metric from social, semantic, and acoustic features that matches the human perception of artist similarity using Partial Order Embedding (POE) with multiple kernels. This is done by mapping the artists into m different non-linear spaces (using m different kernel matrices), learning a separate transformation for each kernel, and concatenating the resulting vectors. The Euclidean distance in the resulting embedding space corresponds to the perceived similarity. The constraints required to guide the learning algorithm are obtained by analyzing subjective artist similarity judgments collected earlier through the web-based *MusicSeer* game for the study of Ellis et al. [65]. This results in about 98.000 partial order constraints of the form *"artists A and B are more similar than artists A and C"* that can be represented as directed edges in a constraint graph with artist pairs as vertices. In such a directed graph, inconsistencies in the subjective judgments can easily be detected and removed by checking for cycles. This methodology for dealing with inconsistencies is generally applicable in scenarios where similar constraints are used. Moreover, although primarily intended to improve artist recommendations, the learning approach can also be applied for visualizing a collection by directly embedding into the (usually two-dimensional) display space. Further work of McFee, Barrington, and Lanckriet [151] focuses on adapting content-based song similarity by learning from a sample of collaborative filtering data. Here, an extension of the structural SVM algorithm called "metric learning to rank" is applied for learning.

3.3.5.2 *User-Adaptive Collection Structuring*

Slaney, Weinberger, and White [227] apply several algorithms based on second-order statistics (whitening, Linear Discriminant Analysis (LDA), and Relevant Component Analysis (RCA)) and optimization techniques (Neighborhood Component Analysis (NCA) and Large-Margin Nearest Neighbor (LMNN)) to learn Mahalanobis distance metrics for clustering songs by artist, album or blog they appear on. For the optimization, an objective function that mimics the k-nearest neighbor leave-one-out classification error is chosen. Songs are represented as vectors containing various acoustic features. From their experiments, the authors conclude that all algorithms lead to a significant improvement over the baseline. In particular, NCA and RCA showed higher robustness with (artificially generated) noisy features.

The *SoniXplorer* [135] shown in Figure 7 is a SOM-based system that adapts a weighted linear combination of basic similarities. Here, the SOM is displayed as video-game-like virtual 3-D landscape accompa-

nied by spatialized playback of songs. Apart from moving songs on the map, the user can raise or lower the terrain to increase or decrease barriers between regions. For the adaptation, a target distance matrix is derived from the arrangement. Then a linear regression learner adapts the weighting accordingly.

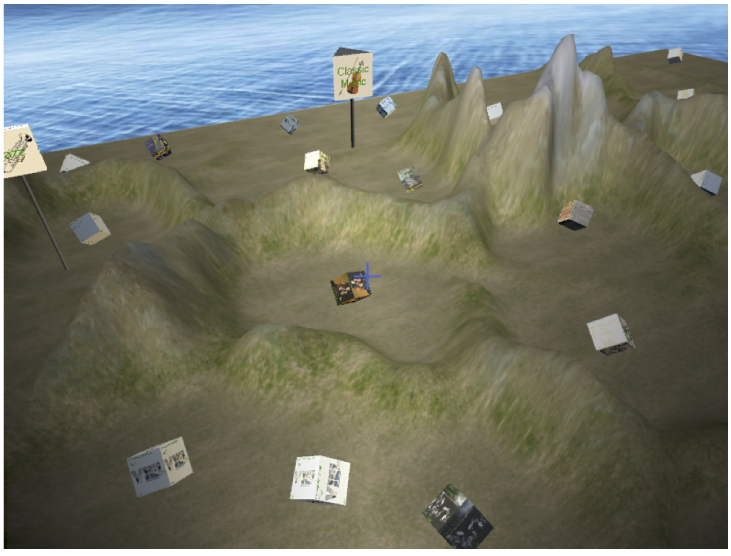


Figure 7: Illustration from [135] of the *SoniXplorer* – a prototype interface for user-adaptive collection structuring.

3.4 CLASSIFICATION OF THE COVERED APPROACHES

In order to provide a systematic overview of the adaptive music retrieval approaches covered in the preceding section, a classification scheme is employed that has originally been used by BRUSILOVSKY [26] for the characterization of adaptive hypermedia systems. Apart from the application area, the following dimensions are covered by the overview given in Table 4 with additional references to the related publications and the respective sections:

- *Adaptation technique:* What general technique is applied as adaptation logic?
  This is the key property by which the entries of Table 4 are sorted. Further, the techniques are grouped conceptually.
- *Source for adaptation:* What context information is used by the adaptation logic?
  According to the generic system model introduced in Section 3.2.2 this refers to the information gathered pro-actively by the system from its context to built the internal context model.
- *Target of adaptation:* What is adapted by the adaptation logic?
  This refers to the means of adaptation provided by the internal

adaptable system that is encapsulated in the generic model for adaptive system discussed in Section 3.2.2.

- *Adaptation guidance:* Where does the system know from how well it performs?
  This directly refers to property (2) of the definition of adaptive systems proposed in Section 3.2.1. In order to adapt properly, i. e., to increase its performance, an adaptive system needs to know how well it is doing or at least whether it is doing the right thing or not. The system may have some internal objective function that it can directly use to measure or at least estimate or predict its performance. Sometimes, such a function cannot be defined a priori. This is typically the case when a user is involved with some unknown user needs that possibly even cannot be stated explicitly. But it might as well be that the system is just a part in a bigger system whose goals it does not know. In this case the system requires some means to infer information about its performance from the context.

3.5 CONCLUSIONS

Since the first mentioning of adaptive music retrieval by ROLLAND [208] almost a decade ago, the MIR community has grown rapidly and approaches have matured. As this survey shows, adaptive techniques have found their way into MIR systems – enabling them to cope with the inherent diversity of both, music and users.

As the overview shows, there has been a lot of recent development covering user-adaptive recommendation and similarity measures. These two challenging areas with a lot of potential for user-adaptation are also addressed in this thesis: Chapter 6 investigates a promising option for future development in user-adaptive genre classification whereas Chapter 9 proposes a way for improving recommendations. Chapter 8 focuses on adaptive music similarity as the key to many MIR applications. An approach for adaptive presentation – the only application area of the general retrieval process where no adaptive approaches could be identified (cf. Figure 5) – is presented in Chapter 7 and further elaborated in Chapter 10. Before, however, Chapter 4 explains the fundamental techniques that are applied in the context of this thesis as a prerequisite for a deeper understanding.

Table 4: Overview of approaches sorted by adaptation technique.

technique (adaptation logic)	application area (section)	references	source for adaptation (context information)	target of adaptation	guidance of adaptation
BASIC DATA-DRIVEN TECHNIQUES USING DYNAMIC PARAMETERS					
dyn. thresholding	onset detection (3.3.1.1)	[39, 203, 252]	data distribution	threshold	internal objective function
	audio identification (3.3.1.1)	[16]			
	audio-score matching (3.3.1.1)	[162]			
	segmentation (3.3.1.1)	[144, 232]			
adaptive window size	tempo estimation (3.3.1.3)	[164]	inter-onset-intervals	window size	internal objective function
fuzzy freq. weighting	key recognition (3.3.1.4)	[40]	signal density distribution	pitch range weights	internal objective function
PROBABILISTIC (CLASSIFICATION) TECHNIQUES					
HMM / Viterbi	chord recognition (3.3.1.2)	[14, 186, 219]	data history	path - state	internal objective function
	key recognition (3.3.1.2)	[121, 170]			
	Fo-estimation (3.3.1.7)	[69]			
MAP (EM)	chord recognition (3.3.1.2)	[14]	data distribution	model parameters	internal objective function
	optical music recognition (3.3.1.6)	[200]			explicit user feedback (corrections)
	Fo-estimation (3.3.1.7)	[77]			internal objective function
naïve Bayes	rec. by listening-context (3.3.3.2)	[83]	location, time, people in the room, weather, stock portfolio	user mood	explicit user feedback
(fuzzy) Bayesian net	rec. by listening-context (3.3.3.2)	[187]	weather, time, ambient noise, illuminance	context (state)	explicit user preference
hypothesis search	chord recognition (3.3.1.2)	[233]	estimated bass pitch	chord probabilities	internal objective function
ARTIFICIAL NEURAL NETWORK TECHNIQUES					
RBFN	ranking (3.3.5.1)	[229]	track ratings	feature selection	explicit user feedback
SOM	collection structuring (3.3.4)	[54, 93, 105, 108, 123, 135, 163, 165, 180, 185, 230]	data	weights & thresholds	internal (amplify response w.r.t. input)
GHSOM	collection structuring (3.3.4)	[58, 178, 202]	data	structure (hierarchy)	internal objective function

technique (adaptation logic)	application area (section)	references	source for adaptation (context information)	target of adaptation	guidance of adaptation
OPTIMIZATION TECHNIQUES					
genetic algorithms	QBSH (3.3.1.8)	[129]	singer errors	singer model	explicit user feedback
simulated annealing	playlist generation (3.3.3)	[191, 192]	n/a	n/a	explicit constraints (adaptable!)
linear regression	collection structuring (3.3.5.2)	[135]	user interaction	similarity metric	implicit (derived distance matrix)
weight reinforcement	ranking (3.3.5.1)	[208]	alignment (pairings)	pairing type weights	explicit user feedback
whitening, LDA, RCA, NCA, LMNN	ranking (3.3.5.1)	[227]	artist/album/blog labels	similarity metric	explicit constraints
multi-kernel POE	ranking (3.3.5.1)	[152]	subjective judgments	similarity metric	explicit constraints
structural SVM	ranking (3.3.5.1)	[151]	collaborative filtering information	similarity metric	derived constraints
STRATEGY SELECTION TECHNIQUES					
rule-based	pitch detection (3.3.1.5)	[133]	harmonic descriptor (simple/polyphonic)	pitch estimator selection	internal objective function
(OTHER) CLASSIFICATION TECHNIQUES					
Rocchio	preference modeling (3.3.2)	[94]	track ratings	reference vectors	explicit user feedback
SVM	preference modeling (3.3.2)	[140]	track ratings	decision boundary	explicit user feedback
PA-L-EX	preference modeling (3.3.2)	[161]	track ratings	kernel	explicit user feedback
decision tree induction	playlist generation (3.3.3)	[190]	track ratings	decision tree	explicit user feedback
classifier fusion	playlist generation (3.3.3)	[248]	track ratings	fusion parameters	explicit user feedback
case-based reasoning	rec. by listening-context (3.3.3.2)	[118]	season, month, day of the week, weather and temperature	case selection	implicit (listening behavior)
heuristics	playlist generation (3.3.3)	[184, 199]	track skips	song selection/removal	implicit (skipping behavior)
	playlist generation (3.3.3)	[181]	track ratings	radio station profile	explicit user feedback

If I have seen further it is only by standing on the shoulders of giants.
Isaac Newton

4 FUNDAMENTAL TECHNIQUES

The work presented in this thesis builds upon various fundamental techniques. This chapter aims to provide basic knowledge of these techniques that is required for a deeper understanding of the developed approaches covered in Part ii: Gradient descent (Section 4.1) and (linear) Support Vector Machines (SVMs) (Section 4.2) are two of the techniques used in Chapter 8 to learn facet distance weights. Self-Organizing Maps (SOMs) (Section 4.3) are used by the *BeatlesExplorer* prototype described in Section 8.5 for the initial structuring of the music collection. Multidimensional Scaling (MDS) (Section 4.4) is applied in Chapter 7 as dimensionality reduction technique to obtain a two-dimensional visualization of a music collection. Furthermore, the vectorization technique described in Appendix A relies on MDS.

4.1 OPTIMIZATION BY GRADIENT DESCENT

Gradient descent is a generic optimization technique that finds a local minimum of a function $f(\mathbf{x}) : \mathbb{R}^n \to \mathbb{R}$ which usually refers to some sort of local error of a system. It follows a very simple fundamental idea: If $f(\mathbf{p})$ is defined and differentiable for some point $\mathbf{p}$, the fastest way to find a minimum is to follow the direction of the steepest gradient "downhill".[1] Usually starting from a random point or initial guess $\mathbf{p}_0$, this is done iteratively until some stopping criterion is met. In particular, for some point $\mathbf{p}_t$ as the result of the t-th step, the next point $\mathbf{p}_{t+1}$ is computed by the following update formula:

$$\mathbf{p}_{t+1} = \mathbf{p}_t - \eta_t \nabla f(\mathbf{p}_t) \tag{4.1}$$

Here, ∇ is the *nabla operator* which returns the *gradient* – a vector with the direction and magnitude of the greatest local ascent of f at the point $\mathbf{p}_t$. The negative sign results in the opposite direction (i. e., "downhill") and the *learning rate* η controls the step size. The learning rate is a critical parameter, because a too small value could require too many iterations and thus too much time to reach a minimum as illustrated by Figure 8 (left). With a too high learning rate, the algorithm may overshoot a minimum (i. e., land on the other side) and in the worst case never reach it because the steps are too wide as shown in Figure 8 (middle). Often, a dynamic learning rate is used

1 Going "uphill" instead would correspond to *gradient ascent* which can be used to find local maxima. Furthermore, there also exist local search strategies such as *hill climbing* that can be applied if no gradient can be computed.

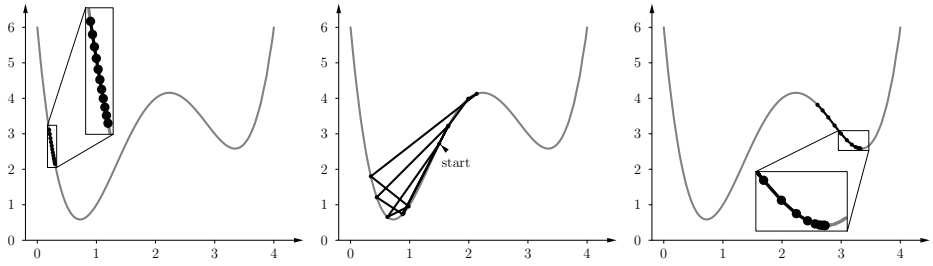


Figure 8: Illustration from [22] of the effect of different learning rates and initializations on the result obtained by gradient descent. Left: $\mathbf{p}_0 = 0.2$ and $\eta = 0.001$. Middle: $\mathbf{p}_0 = 1.5$ and $\eta = 0.25$. Right: $\mathbf{p}_0 = 2.6$ and $\eta = 0.05$.

that depends on the number of steps already taken. This allows to quickly approach a minimum with larger steps and find the optimal value with smaller ones.

Other important parameters are the stopping criterion and the initialization. A multitude of options exists for the stopping criterion which can also be combined, e. g.:

- The number of iterations exceeds some threshold.
  Variant: The processing time exceeds some threshold.
- The improvement (difference of the function values) over the last k steps is below some threshold.
  Variant: There has been no improvement over the last k steps.

With an appropriate learning rate and stopping criterion, the algorithm will find a minimum. However, unless f is a convex function, it might only be a local minimum. Depending on the initial point $\mathbf{p}_0$, different (local) minima may be found as illustrated by Figure 8. Therefore, it is a common strategy to run the algorithm multiple times with different initializations and afterwards choose the best solution.

Furthermore, various advanced extensions of this basic gradient descent algorithm exist like adding a *momentum term* or working with ensembles which, however, shall not be discussed here. For further information, the interested reader may refer to standard literature such as [18, 138].

4.2 LINEAR CLASSIFICATION WITH MAXIMUM MARGIN USING SUPPORT VECTOR MACHINES

This section describes the fundamental idea of Support Vector Machines (SVMs) in the context of linear classification. The following elaboration is confined solely to the aspects that are of interest in the context of this thesis. Specifically, kernelization techniques that allow non-linear classification are not covered elaborately. For further details

on SVMs and working with kernels in general, the interested reader may, e. g., refer to SCHÖLKOPF and SMOLA [216].

Classification is a common supervised learning problem: Given some training instances with associated class labels, a classifier learns to assign labels to previously unseen instances. In *binary classification*, there are only two class labels. These are in the following denoted as $+1$ and -1. Furthermore, the instances to be classified shall be vectors $\mathbf{x} \in \mathbb{R}^n$. (Obviously, more complex representations of instances are also possible but not of interest in the scope of this thesis.) A *linear classifier* aims to divide this space with a $(n-1)$-dimensional hyperplane into two parts – each containing only instances belonging to one of the two classes. If the data is *linearly separable* – i. e., a hyperplane exists that perfectly separates both classes – there is usually not a unique solution. Figure 9 illustrates this for points in two-dimensional space. The *maximum-margin hyperplane* (h_2 in Figure 9) is the one with the greatest separation margin between the training instances of the two classes. Intuitively, the larger margin results in a higher robustness to noise in the data.

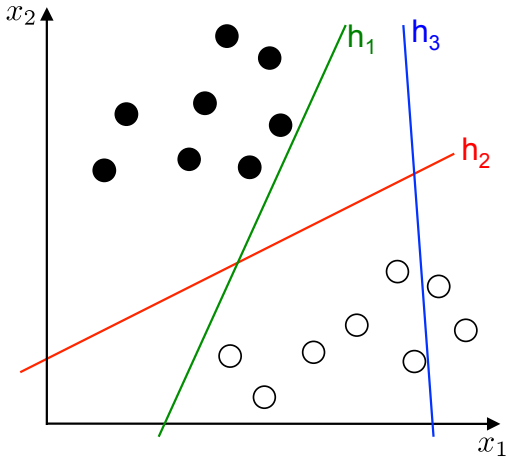


Figure 9: Hyperplanes in $\mathbb{R}^2$: h_1 and h_2 separate the two classes ($\circ$ and $\bullet$) of points whereas h_3 does not. h_2 is the maximum margin separating hyperplane.

The maximum margin criterion can be formalized mathematically as follows: According to the Hessian normal form, a hyperplane comprises all points $\mathbf{x}$ satisfying:

$$\mathbf{w}^T\mathbf{x} - b = 0 \tag{4.2}$$

with the *normal vector* $\mathbf{w}$ which is perpendicular to the plane and the *bias* b which corresponds to the euclidean distance from the origin. Hence, the output of the linear classifier can be written as:

$$f(\mathbf{x}) = \operatorname{sign}(\mathbf{w}^T\mathbf{x} - b) \tag{4.3}$$

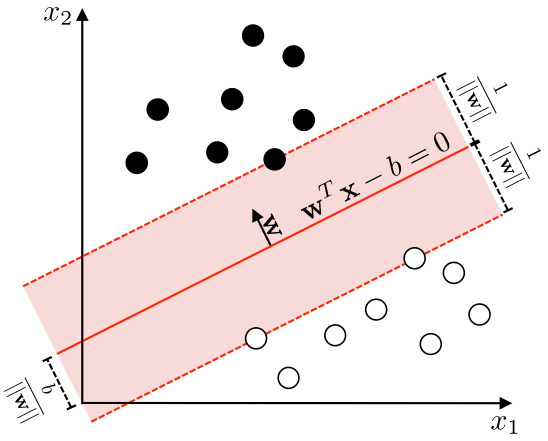


Figure 10: Maximum margin separating hyperplane for the example shown in Figure 9 with parallel hyperplanes corresponding to the margin.

and its *functional margin* for a set T of training instances $\mathbf{x}_i$ with class labels y_i can be computed as:

$$\gamma = \min_{(\mathbf{x}_i, y_i) \in T} y_i(\boldsymbol{w}^T \mathbf{x}_i - b) \tag{4.4}$$

The functional margin is however not a suitable measure for the robustness of a separating hyperplane because it also depends on the scaling of $\boldsymbol{w}$ and b. Normalizing by the length of the normal vector results in the scale-independent *geometric margin*:

$$r = \min_{(\mathbf{x}_i, y_i) \in T} y_i \frac{\boldsymbol{w}^T \mathbf{x}_i - b}{\|\boldsymbol{w}\|} \tag{4.5}$$

For any separating hyperplane, there exist two parallel hyperplanes – one includes the closest instance(s) from the negative class and the other the closest instance(s) from the positive class respectively. The respective closest instances are called *support vectors*. If the separating hyperplane is in the middle of the resulting "corridor" as illustrated in Figure 10, the distance to both borders is exactly r and the resulting margin $2r$ consequently. Choosing a scaling that results in a functional margin of $\gamma = 1$, i. e.

$$y_i(\boldsymbol{w}^T \mathbf{x}_i - b) \geqslant 1 \tag{4.6}$$

with the value 1 obtained for the support vectors, yields $r = \frac{1}{\|\boldsymbol{w}\|}$ from Equation 4.5. For the maximal margin hyperplane, r needs to be maximized which consequently means to minimize $\|\boldsymbol{w}\|$. As this would involve the square root, $\frac{1}{2}\|\boldsymbol{w}\|^2$ is minimized instead, where the factor $\frac{1}{2}$ is only introduced for convenience. This can be formulated as the following Quadratic Programming (QP) problem:

$$\min_{\boldsymbol{w} \in \mathbb{R}^n} \frac{1}{2}\|\boldsymbol{w}\|^2 \tag{4.7}$$

subject to

$$y_i(\boldsymbol{w}^T\boldsymbol{x}_i - b) \geqslant 1 \qquad \forall(\boldsymbol{x}_i, y_i) \in T \tag{4.8}$$

which can be solved with standard QP solvers or in the more efficient SVM-way by constructing a dual problem. However, this shall not be discussed further here.

Sometimes, it is impossible to linearly separate the training instances. In order to cope with this case, non-negative *slack variables* ξ_i can be introduced which soften the constraints such that:

$$y_i(\boldsymbol{w}^T\boldsymbol{x}_i - b) \geqslant 1 - \xi_i \quad \xi_i \geqslant 0 \qquad \forall(\boldsymbol{x}_i, y_i) \in T \tag{4.9}$$

Additionally, a penalty term is added to the objective function:

$$\min_{\boldsymbol{w}\in\mathbb{R}^n} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_i \xi_i \tag{4.10}$$

where the constant C allows to control the impact of the penalty.

Another option to cope with training instances that are not linearly separable is to apply the *kernel trick*. The idea is to map the data to a higher-dimensional feature space using a non-linear function

$$\phi\colon \mathbb{R}^n \to \mathbb{R}^{n'}, \boldsymbol{x} \mapsto \phi(\boldsymbol{x}) \tag{4.11}$$

Choosing an appropriate mapping function, ϕ, there is a separating hyperplane in the target space. For SVMs, this trick can be implemented very elegantly in a way that requires only to specify a distance function on the transformed feature vectors, $\phi(\boldsymbol{x})$, instead of explicitly describing the target space. As this technique is not applied in the context of this thesis, further details are omitted here. A detailed description is, e. g., given by Schölkopf and Smola [216].

4.3 STRUCTURING DATA COLLECTIONS WITH SELF-ORGANIZING MAPS

Self-Organizing Maps (SOMs) belong to the *unsupervised learning* techniques – i. e., in contrast to the supervised learning approach of SVMs covered by the preceding section, no class labels are required here. Kohonen [109] first described SOMs in 1982. Hence, sometimes the name "Kohonen (feature) maps" is used synonymously. They can be considered as a special type of artificial neural network, partly motivated by how the human brain processes visual, auditory and other sensory information with neighboring parts of the network responding similarly. Despite this background, a rather pragmatic perspective is chosen in the following which focuses on the (generic) structuring aspect.

SOMs are commonly applied for structuring data collections through *prototype-based clustering* and furthermore as nonlinear dimensionality

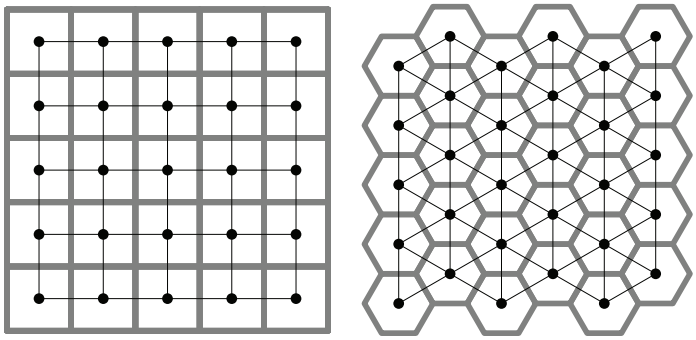


Figure 11: Illustration from [22] showing a grid of quadratic cells (left) and a grid of hexagonal cells (right). Thin black lines indicate nearest neighbors of a cell. Thick gray lines indicate regions assigned to a cell for visualization.

reduction technique which is especially suitable for analysis and visualization of high-dimensional datasets. Clustering means to group similar objects together into so-called *clusters* with the general objective to obtain a high intra-cluster similarity (between objects of the same cluster) and at the same time a low inter-cluster similarity (between objects from different clusters). Prototype-based clustering approaches like SOMs represent each cluster by a *prototype* which has the same form as the objects to be clustered. Here, objects and cluster-prototypes are generally represented by vectors $\mathbf{x} \in \mathbb{R}^n$. (Like for SVMs, more complex representations are possible but require rather severe modifications which are not covered here.) In particular, the prototypes of a SOM are arranged in a grid structure. Most commonly, this grid is two-dimensional and consists of hexagonal or square *cells* with each cell corresponding to a prototype. Furthermore, the grid may be mapped onto the surface of a sphere or torus and consequently have no outer boundary in contrast to a planar grid. Figure 11 shows planar grids with quadratic and hexagonal cells.

Training a SOM means to adapt the prototypes of its grid to the distribution of the data to be clustered. This is accomplished in the following iterative process: At first, the prototypes are randomly initialized – e. g., by choosing fully random vectors or randomly selecting objects from the dataset to copy values from. In each iteration, the prototypes compete for the objects of the dataset as each object is assigned to the most similar prototype. The assignment of an object represented by a vector $\mathbf{x}$ results in an update of all prototypes $\mathbf{p}$ according to the following update rule:

$$\mathbf{p}' = \mathbf{p} + N\left(\mathbf{p}, \mathbf{p}^{(win)}\right) \eta_t \left(\mathbf{x} - \mathbf{p}\right) \tag{4.12}$$

where $\mathbf{p}^{(win)}$ is the "winner" – i. e., the most similar prototype, η_t is the *learning rate* in the t-th iteration and N a *neighborhood function*. This update rule moves each prototype vector to some extent into the direction of the input vector $\mathbf{x}$. The magnitude of this change depends on two factors: the learning rate which is usually dynamic

to decrease the impact over time (similar to the learning rate for the gradient descent technique described in Section 4.1) and the neighborhood function. The neighborhood function is the crucial part of the update rule as it incorporates the neighborhood relations between the prototypes into the computation. The idea is that the impact of the update should be highest for the winner prototype and decay with increasing distance on the grid. Typical neighborhood functions are the Gaussian and the Mexican Hat function – whereby the latter even results in negative update factors for surrounding prototypes. Furthermore, a time-dependency may be introduced here as well for better convergence of the prototype vectors. Again, several termination criteria for the learning process are possible such as fixing the number of iterations, using a threshold on the magnitude of the change, or checking how many objects have been assigned to a different cluster compared to the previous iteration.

As a result of this neighborhood-sensitive training mechanism, neighboring prototypes "see" similar input data and thus adapt similarly. This leads to neighborhoods of clusters in the grid with similar prototypes and similar content. Consequently, the mapping of the objects to the cluster cells is approximately topology-preserving, i. e., neighboring objects in the original (possibly high-dimensional) vector space are most likely to be neighbors on the grid as well.[2] This makes SOMs very attractive to generate an overview or to structure collections.

Several extensions of the basic SOM approach exist such as hierarchical SOMs [114] or Growing Self-Organizing Maps (GSOMs). The latter are applied in the context of this thesis (Section 8.5) and thus are further explained in the following section.

4.3.1 *Growing Self-Organizing Maps*

The basic SOM approach has the disadvantage that the structure of the SOM has to be specified prior to training. Usually, several maps with different sizes need to be learned until a satisfying structuring is obtained. Further problems arise if the collection changes significantly after the map has been trained and generating a completely new map may not be desirable – e. g., if a user has become familiar with the initial map. A possible solution to these problems is to allow the map to grow. This means to also adapt the structure of the SOM with some specified boundaries. The GSOM approach applied in the context of this thesis has been described by Nürnberger and Detyniecki [172]. They work with hexagonal cell structures and restrict the growth of the grid to its boundaries (i. e., new cells can only be added at the outside). The algorithm can be summarized as follows: For the initialization,

2 The visualization technique presented in Chapter 7 specifically addresses cases where this fails and neighborhoods are torn apart. In this context, general problems of dimensionality reduction techniques are discussed in depth.

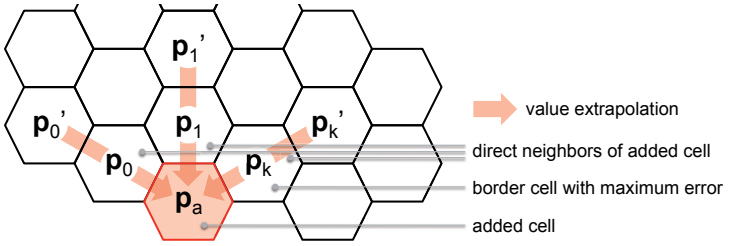


Figure 12: Illustration of the extrapolation rule used to initialize new cells in a GSOM as described by Nürnberger and Detyniecki [172].

a small grid of usually 2×2 cells is chosen. After a regular training phase with a fixed number of iterations, an internal error is computed for each cell as a measure of quality for each cluster. To this end, e. g., the pairwise distance of the objects contained in a cluster can be used. Unless the maximum error is below a threshold which is specified as stopping criterion, a new cell is added next to the border cell $\mathbf{p}_k$ with the highest error.[3] If several possibilities exist where to place the new cell, the position next to the most dissimilar neighbor of $\mathbf{p}_k$ is chosen. For the initialization of the prototype vector $\mathbf{p}_a$ of the added cell, the following extrapolation rule is used:

$$\mathbf{p}_a = \frac{1}{|N_a|+1}\left(\mathbf{p}_k + (1-\lambda)\left(\mathbf{p}_k - \mathbf{p}_k'\right) + \sum_{\substack{\mathbf{p}_i \in N_a \\ \mathbf{p}_i \neq \mathbf{p}_k}} \left(\mathbf{p}_i + \lambda\left(\mathbf{p}_i - \mathbf{p}_i'\right)\right)\right) \tag{4.13}$$

where N_a is the set of direct neighbors of the added cell, $\mathbf{p}_i'$ refers to the direct neighbor of $\mathbf{p}_i$ which is opposite to the new cell, and λ is a distance weight with default value 0.7. Figure 12 illustrates this rule for an example grid.

Only a single cell is added in each expansion step. Afterwards, the algorithm starts over with the regular training phase. The above described way of initializing a new cell causes some of the objects previously contained in neighboring border cells to be reassigned to the added cell. This way, the overall error in the neighborhood is reduced. In their experiments, Nürnberger and Detyniecki [172] also observe that objects from inner cells with high error are "pushed" to the border in the training phase. Therefore, they conclude that considering only growth at the border is not limiting the capabilities of their GSOM approach.

4.4 MULTIDIMENSIONAL SCALING

In general, Multidimensional Scaling (MDS) comprises several techniques for reducing the dimensionality of a dataset which are well-

3 As a cell is directly represented by its prototype vector, the notation is simplified in the following by always referring to the prototype vector instead of having a designated notation for the cells additionally.

suited to compute low-dimensional projections – e. g., for visualization as described by RÖSSL and THEISEL [209]. Apart from *classical* MDS, which has been applied in this thesis, a variety of other approaches exist that are not covered here such as metric least squares scaling or non-metric MDS which involve numerical minimization of a cost function using gradient descent. Generally, MDS takes only a symmetric matrix of pairwise distances (or dissimilarities) as input without any requirements concerning the corresponding objects and their representations. This is in contrast to SOMs which can also be used for dimensionality reduction as described above but generally require a vector representation of the objects.

Given the distances (or dissimilarities) for a set of objects S and a parameter $0 < m \leqslant |S|$, classical (metric) MDS as, e. g., described in [86, Chapter 15.2] finds an embedding into m-dimensional Euclidean space that maintains the objects' distances as good as possible in the least squares sense. I. e., each object $o_i \in S$ is mapped to an m-dimensional vector $\mathbf{x}_i \in \mathbb{R}^m$ such that the Euclidean distance between the vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ is (as close as possible to) the original distance d_{ij} between the corresponding objects o_i and o_j.

In the context of this thesis, the following algorithm based on the description by HÄRDLE and SIMAR [86, Chapter 15.2] is used to compute the MDS: First, a mean-centered matrix $\mathbf{B}$ is constructed from the squared distances by the following two steps:

1. Define matrix $\mathbf{A} = (a_{ij})$ with $a_{ij} = -\frac{1}{2} d_{ij}^2$
2. Define matrix $\mathbf{B} = (b_{ij})$ with $b_{ij} = a_{ij} - \overline{a}_i - \overline{a}_j + \overline{a}$
   for the row mean $\overline{a}_i = \frac{1}{n} \sum_{j=1}^{n} a_{ij}$,
   the column mean $\overline{a}_j = \frac{1}{n} \sum_{i=1}^{n} a_{ij}$,
   and the overall mean $\overline{a} = \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij}$ of $\mathbf{A}$.

The m-dimensional coordinate vectors $\mathbf{x}_i$ can then be computed as $\mathbf{x}_i = \gamma_i \sqrt{\lambda_i}$ from the m largest positive eigenvalues λ_i of $\mathbf{B}$ and the respective eigenvectors γ_i. Naturally, the parameter m is bounded by the dimensionality of $\mathbf{B}$ and the number of positive eigenvalues. The (relative) difference between the sum of the m used eigenvectors and the trace of the matrix $\mathbf{B}$ (i. e., the sum of all its eigenvalues) is in the context of this thesis called *residual*. It captures the distance error induced by MDS.

4.4.1 *Landmark Multidimensional Scaling*

Landmark Multidimensional Scaling (LMDS) as described by SILVA and TENENBAUM [225] (with additional details in [226]) is a computationally efficient approximation to classical MDS. The general idea of this approach is as follows: Given a sample set of *landmark* or *pivot objects* selected by some heuristic, an embedding into a low-dimensional

space is computed for these objects using classical MDS. Each remaining object can then be located within the output space according to its distances to the landmarks. Obviously, the quality of the projection depends on the choice of the landmarks – especially if the landmark sample set is small compared to the size of the whole dataset. If the landmarks lie close to a low-dimensional subspace (e. g., a line), there is the chance of systematic errors in the projection. SILVA and TENENBAUM [226] describe a *random* and a *MaxMin* heuristic with the goal to approximate the covariance matrix of the dataset with the sample set defined by the landmarks. The *MaxMin* approach greedily searches for extreme, well-separated landmarks but has the tendency to select outliers in the data. The authors state that accuracy increases with the number of landmarks (but at higher computational costs).

4.4.2 *Complexity*

According to SILVA and TENENBAUM [226], classical MDS has a computational complexity of $O(N^3)$ for the projection, where N is the number of objects in the dataset. Additionally, the $N \times N$ distance matrix needed as input requires $O(N^2)$ space and is computed in $O(CN^2)$, where C are the costs of computing the distance between two objects. By limiting the number of landmark objects $k \ll N$, a LMDS projection can be computed in $O(k^3 + kmN)$, where m is the number of output dimensions, which is usually 2 for visualizations. The first part refers to the computation of the classical MDS for the k landmarks and the second to the projection of the remaining objects with respect to the landmarks. Further, LMDS requires only the distances of each object to the landmarks, i. e., only a $m \times N$ distance matrix has to be computed resulting in $O(kN)$ space and $O(CkN)$ computational complexity. This way, LMDS becomes feasible for application on large datasets as it scales linearly with the dataset size.

Part II

APPROACHES TO ADAPTIVE MUSIC RETRIEVAL

Wir haben Gitarren,
das Klavier und den Bass.
Wir haben das Schlagzeug,
den Gesang und all das
ist in guten Momenten für eine Weile
mehr als die Summe der einzelnen Teile.
"Die Summe der einzelnen Teile"
Kante

5 DATA-ADAPTIVE FEATURE EXTRACTION

Feature extraction is the first step in the general retrieval process described in Section 2.2.1 as illustrated by Figure 13. Thus, it is crucial for the performance of a retrieval system as a whole. Unfortunately, the extraction of content-based features from raw audio signals is largely error-prone. Here, adaptive methods can help to increase the robustness of the extraction method and consequently the quality of the features. Several examples have already been given in Section 3.3.1.

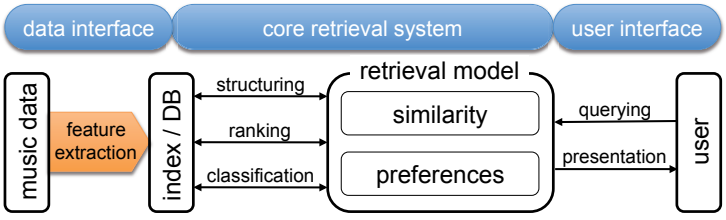


Figure 13: Feature extraction in the context of the general retrieval process described in Section 2.2.1.

In the underlying work of this thesis, the focus has not been on the feature extraction step. Most of the feature extraction has been "outsourced" to sophisticated libraries and web services such as *Marsyas* [239], *JAudio* [150], *Sonic Annotator* [url:42], and *EchoNest* [url:47]. However, two particular diploma theses that have been supervised in the context of this work deal with adaptive feature extraction. The relevant aspects are summarized in this chapter as examples for adaptive feature extraction.

In his diploma thesis "Query-by-Singing/Humming with Low-Level Feature Extraction" [stud:1], Alexander Duda applies an adaptive noise removal technique in order to separate the singing voice (which has to be further analyzed) from the background music. A summary of this work has been published in [pub:1]. Section 5.1 describes the particular filtering technique.

The diploma thesis "Akkorderkennung in Audiodateien"[1] [stud:2] of Johannes Reinhard deals with the problem of chord recognition in audio recordings (cf. Section 3.3.1.2). He proposes an adaptive "smoothing" technique that aims to correct chord misclassifications. A detailed description of this approach together with a survey of related work in the field of chord recognition (as of 2008) has been published in [pub:4]. The fundamental technique is described in Section 5.2.

1 English translation: "Chord Recognition in Audio Files"

5.1 AN ADAPTIVE NOISE REMOVAL TECHNIQUE FOR MELODY EXTRACTION

The work presented in this section was primarily done in the diploma thesis of Alexander Duda [stud:1] and has been published in [pub:1].

In the popular Query-by-Singing/Humming (QBSH) scenario, a retrieval system has to find relevant songs for a query which is either sung or hummed. Such a query usually relates to the melody sung by the lead voice or played by a solo instrument. Early QBSH approaches as, e. g., described by NISHIMURA et al. [169] or PAUWS [189] focused on symbolic representations where each instrument usually has its own track thus allowing a straightforward separation of the individual voices. In real audio recordings, however, audio information of all instruments and voices is mixed and not stored separately in the channels. In this more challenging scenario, the melody first needs to be extracted from the mix before further processing steps like feature extraction and indexing can be applied. This can be considered as a special case of the more general source separation problem of which BURRED [28] gives a broad overview.

In his diploma thesis, Alexander Duda [stud:1] describes a two-step filtering approach that aims to *reduce* the impact of backing instruments and voices in the audio recordings instead of pursuing the more challenging goal of *separation* mentioned above. The first step is a common *band-path filter*, i. e., a filter that only keeps frequencies in a specific range. Here, a range of 300Hz to 3000Hz is chosen which is motivated by the features to be extracted.

The other filtering step exploits the spatial arrangement of instruments and voices in the stereo signal in a way that could be described as "inverse karaoke" effect. It is inspired by a basic technique called *center pan removal* that is commonly used by karaoke machines to remove the lead voice from a typical rock/pop song: One of the stereo channels is phase-inverted and mixed together with the other one into a mono signal. The lead voice and solo instruments are usually in the center of the stereo mix whereas most instruments and backing vocals are arranged out of center as depicted in Figure 14. This means for a

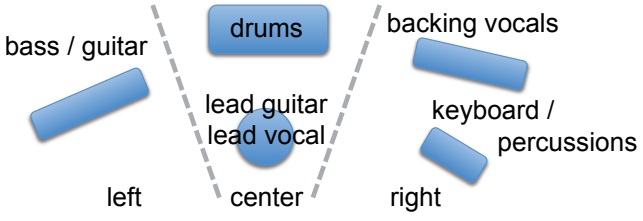


Figure 14: A typical stereo arrangement of a rock/pop band.

stereo recording that some components have nearly the same power level on both audio channels, which is called *center panned*. Applying the above transformation drastically reduces the power level of these centered signals.

The idea is to invert this effect, so that the filtered recording yields a high portion of the lead voice, while most other instruments are filtered out. Unfortunately, there is no simple way to keep the center pan. Because the karaoke signal is a linear combination of the stereo channels, mixing it (either phase-inverted or not phase-inverted) with the original does not work. The result would just be a new mix of the center panned and non-center signals. Hence, a different approach is taken that is based on the "noise removal" effect of the *Audacity* audio editor [url:2]. This filter is basically an adaptable multi-band noise gate – i. e., the frequency range of the input signal is divided into multiple intervals called "bands" and for each band, signals below a band-specific threshold are blocked. The noise thresholds for the bands are derived from a noise sample that has to be provided in advance. This algorithm works well for removing constant noise, e. g., white noise or growling. However, using the karaoke signal of the whole recording as noise sample, the noise profile becomes too imprecise and thus does not lead to the desired effect of removing the background music. Most notably, this causes warbling artifacts which result from removing important frequencies from the center as well.

This problem can to some extent be overcome by confining the signal to be filtered (and the respective noise sample from the karaoke version) to a small time window. With decreasing window size the background "noise" becomes more stable and the derived parameters for the multi-band noise gate more precise consequently. Instead of using a global setting for the whole recording, the filter adapts to the local context determined by the sliding window. This way, the warbling effect is reduced and the removal of background music becomes possible. Using overlapping windows furthermore minimizes effects at the edges.

Figure 15 illustrates the signal flow of the filter. The original im-

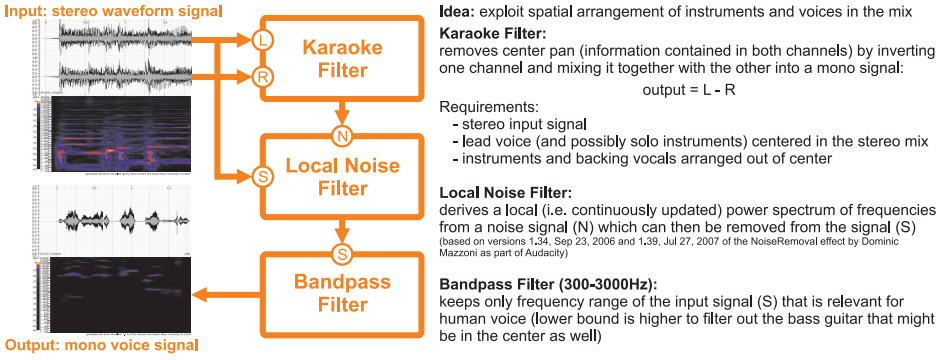


Figure 15: Part of the poster for the QBSH system presented at ISMIR 2007 [pub:1] illustrating the preprocessing step that aims to extract the melody using an adaptive noise filtering technique. (The images of the waveforms and the melodic range spectrograms have been generated with *Sonic Visualiser* [31].)

plementation described in [stud:1] used the internal batch processing functionality of *Audacity* to do the filtering within the editor. The length of the window was set to two seconds, while the hop size for the window was one second. Later, a refined version of the filter has been implemented based on the *Audacity* filter code.[2] It allows even finer adjustments with window sizes down to the actual frame size.

The refined version of the filter was demonstrated together with the QBSH system at the ISMIR'07 conference [pub:1].

Audio Clips 1-4

As an example to demonstrate the difference, the filter output obtained with the global and the adaptive local noise profile is shown in Figure 16 for a 22 seconds clip from *"Have a little faith in me"* by Joe Cocker. Especially the waveform plots reveal that using the local profile results in a stronger emphasis on the actual lead voice. Using the global profile of the whole song further increases the discrepancy.

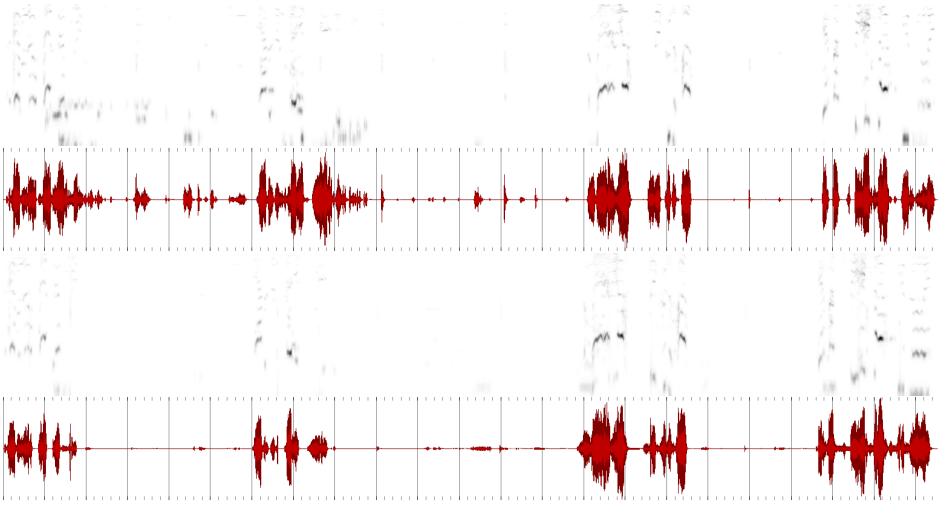


Figure 16: Frequency spectrum (300-3000Hz) and waveform for a 22 seconds clip from *"Have a little faith in me"* by Joe Cocker processed with the "inverse karaoke" filter using a global (top) and local (bottom) noise profile. (The images of the waveforms and the spectrograms have been generated with *Sonic Visualiser* [31].)

Apart from empirical testing where better results were obtained with the adaptive local profiles, the "inverse karaoke" filter has been evaluated implicitly within the QBSH system as a whole. This evaluation has been published in [pub:1]. For the majority of the rock and pop tracks considered in the evaluation, the voice extraction method works reasonably well. In general, the quality of the filter result largely depends on the karaoke signal because the whole technique works under the assumption that this signal contains only irrelevant information that should be removed from the mix. Consequently, whichever part of the lead vocal remains in the karaoke signal might be removed as well. For instance, recordings where reverberation effects (e. g.,

2 Specifically, versions 1.34 and 1.39 of the *Audacity* noise removal filter written by Dominic Mazzoni were used as code base. The older and much simpler version results in a clearer filtering.

echo) are applied are problematic. Especially stereo reverberation spreads the sound sources such that a centered lead vocal contributes differently to the two stereo channels. Another recording technique that causes a poor karaoke signal is *voice doubling* or *double tracking*. Here, the same voice is recorded twice (or more times) and the resulting slightly differing versions are placed in different positions in the stereo mix to create a "denser" or "bigger" sound. Commonly, one version is panned hard left and the other hard right resulting in a karaoke track where neither is removed. As a further limitation, the filtering technique obviously does not work for mono recordings. Mixing approaches that differ significantly from the one illustrated in Figure 14 are also not considered. For instance, in recordings of classical music, the position of instruments in the mix usually corresponds to the actual spatial arrangement of the instruments in the concert hall. Using a more sophisticated technique to derive the karaoke signal might solve some of these problems which is, however, beyond the scope of this thesis.

5.2 ADAPTIVE CORRECTION OF MISCLASSIFICATIONS IN CHORD DETECTION

The work presented in this section was primarily done in the diploma thesis of Johannes Reinhard [stud:2] and has been published in [pub:4].

The harmonic chord progression of a song is an important high-level feature which enables indexing as well as deeper analysis of musical recordings. Commonly, a chord is defined as a set of tones played at the same time. However, a looser definition also allows for tones not played simultaneously to form a chord, provided that they are to be interpreted as *somehow belonging together*. This makes chord determination a challenge even if music is available in symbolic notation. The problem is much harder, if the music is only available in raw audio format, as the single tones must be extracted from the musical signal. Different approaches to chord recognition have been suggested in the past. Based on a comprehensive survey presented in [pub:4], Johannes Reinhard roughly divides the general approach to chord detection into three steps as shown in Figure 17: Feature extraction, chord classification, and post-processing. In the first step, suitable features need to be extracted from the raw data – commonly this includes *pitch-class profiles* which are also referred to as "chromagrams" or just "chroma"-vectors (cf. Table 2 on page 24). These features can then be used by a classifier to predict a chord, while using more robust features can significantly improve the accuracy of the classification. The third step tries to correct unavoidable misclassifications caused, e. g., by the presence of percussive sounds or harmonics.

An overview of related chord detection approaches is given in Section 3.3.1.2.

In his diploma thesis, Johannes Reinhard proposes a data-adaptive "smoothing" technique for post-processing that exploits (uncertain) knowledge about the chord-distribution in a chord's neighborhood and that can be applied independently of the features and classifier

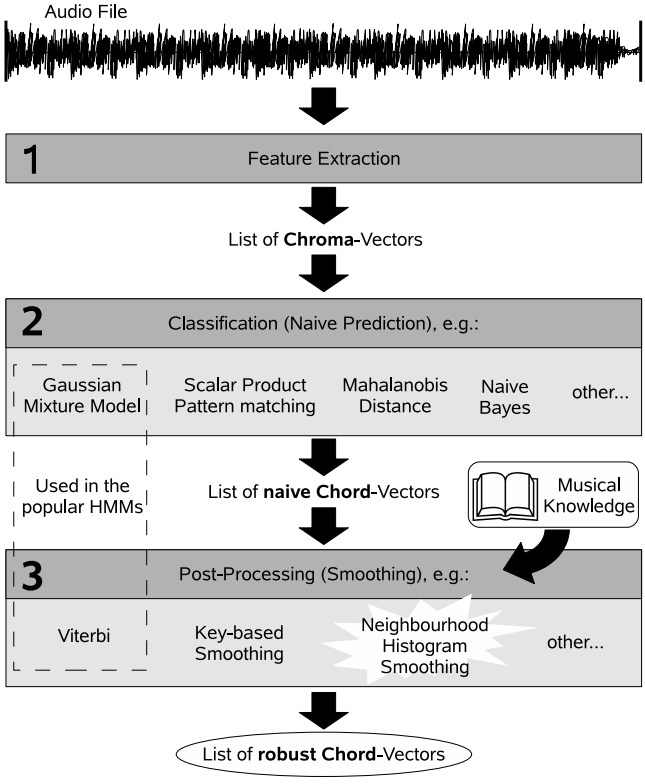


Figure 17: Illustration of the generalized chord recognition process (from [pub:4]).

used. The underlying assumption is that the pool of chords used in a song is limited and that strong oscillations of chords are uncommon. Section 5.2.1 and Section 5.2.2 briefly describe the feature extraction step and classification step respectively to outline the context in which the post-processing method is applied. The post-processing step is explained in detail in Section 5.2.3. Subsequently, Section 5.2.4 presents the results of an evaluation using three baseline classifiers on two early *Beatles* albums. Finally, Section 5.2.5 concludes with a discussion.

5.2.1 *Feature Extraction*

The first step of the three-step approach is the extraction of the chromagram feature from an audio recording. This is accomplished using the *Sonic Visualiser* [31]. Here, only a small frequency band of three octaves between 65,41Hz and 523,25Hz is considered. Although this discards low bass tones and some high voice or guitar tones, most important pitches for chord recognition are contained in this frequency band. As only the lowest tones will have their misleading third or fifth or even higher harmonic in this pitch range, this reduces the risk

of wrong classification due to the presence of *misleading* harmonics. The second or fourth harmonics do not hamper chord recognition that much as they fall into the same pitch class as the fundamental.

Instrument tunings that differ from the standard concert pitch of 440Hz are accounted for in a similar way as described by HARTE and SANDLER [87]. This step is quite important, especially for the early *Beatles* albums used in the evaluation as deviant tunings are common for these records. Using a beat detection system included in the *Sonic Visualiser*, the beat times of a song are extracted. The chromagrams located between two beats are averaged to form new chromagrams with larger and more meaningful window size. This common practice approach is based on the assumption that chords often change at beat times or at times with strong note onsets. As the time between two beats is longer than a chromagram window, averaging the windows between two beats also results in some smoothing.

5.2.2 *Naïve Prediction*

In the second step, a classifier is used to predict the chord solely based on the chromagram feature extracted in the previous step. This is in the following called *naïve prediction* because this prediction might differ from the final prediction which additionally incorporates knowledge of musical principles. The post-processing technique applied later requires that the classifier does not only predict the most probable chord, but returns a probability or confidence for every possible chord considering the chromagram observed. The chord alphabet considered comprises the 12 major and 12 minor chords. Other chords like diminished, augmented, seventh or other complex chords are mapped to major and minor depending on their third. For instance, an E diminished seventh chord would be mapped on an E minor chord as it contains a minor third. Three commonly used classifiers are tested in order to demonstrate that the post-processing technique described in the following section can be applied in combination with arbitrary classifiers as long as they meet the above stated requirement:

The first classifier returns a similarity score calculated as the scalar product between the chromagram vector and a chord template. The chord template contains a 1 if the tone is part of the chord and a 0 if it is not. So, if the order of tones in a chromagram vector is C,C♯,D,...,A♯,B, a C-Major chord template has the format $(1,0,0,0,1,0,0,1,0,0,0,0)$ as shown in Figure 18. For the C♯-Major chord, the template looks similar with every number shifted to the right by one. Likewise a C-Minor chord template is $(1,0,0,1,0,0,0,1,0,0,0,0)$ with a minor third – contrast to the major third in the major chord template. This classifier which is also used in [87] is simple but yields quite good results. Moreover, it does not only predict one chord but additionally gives a score for every chord. Here, each score is divided by the sum

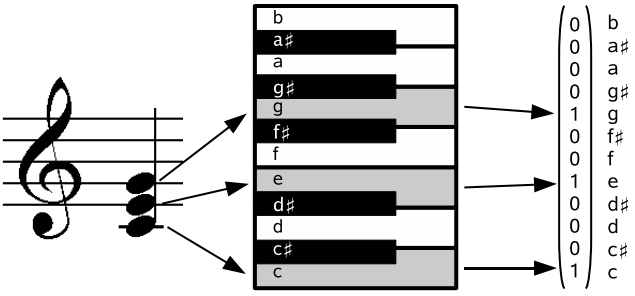


Figure 18: Illustration of the template for the C-Major chord (from [pub:4]).

of all scores, so that it can be considered as a probability. It is denoted as $P(\mathbf{chroma}|c_i) \quad i \in [1, 24]$ and describes the probability to observe the chroma-vector $\mathbf{chroma}$ given chord c_i.

The second classifier calculates its scores using the Mahalanobis distance between the chroma-vector $\mathbf{chroma}$ and a distribution, represented by a mean vector $\boldsymbol{\mu}$ and its associated covariance matrix $\boldsymbol{\Sigma}$. The Mahalanobis distance is defined as:

$$d(\mathbf{chroma}, \boldsymbol{\mu}) = \sqrt{(\mathbf{chroma} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{chroma} - \boldsymbol{\mu})} \qquad (5.1)$$

The mean vector $\boldsymbol{\mu}$ and the covariance matrix $\boldsymbol{\Sigma}$ are calculated from some training samples provided (cf. Section 5.2.4). This classifier is, e. g., used by Yoshioka et al. [251]. Furthermore, it is basically also applied in those HMM approaches that use a single multivariate Gaussian to model their output distribution (e. g., [14, 122, 186, 219]) as the calculated distance and the calculated value of the output distribution can be transformed into each other (i. e., the ranking is the same).

The third classifier is a Naïve Bayes classifier.[3] As required, this classifier returns a probability for every chord. However, the probabilities from this classifier tend to be rather extreme. Often, many chords have a probability close to 0, whereas only a small set of chords has high probabilities. This nature of the probability distribution makes this classifier less suitable. Nevertheless, for sensibly chosen parameters, an improvement of recognition rates can still be achieved.

5.2.3 *Post-Processing (Smoothing)*

This step attempts to correct mistakes in the naïve predictions obtained by the classifiers described in the preceding section. It is motivated by the fact that music does not have too many shifts to sound harmonically and repetition of musical patterns is more likely than steady change. Furthermore, most songs are written in a certain key which constrains the choice of chords and can stabilize chord recognition.

3 included in the Information Miner software [url:18]

For a certain unknown chord, it is more likely to be one of the chords out of the pool of neighboring chords than to be just any arbitrary chord. This can be modeled as follows:

Every chord is regarded as the center of a sliding window containing n chords which form his neighborhood. From these chords, a histogram of the chord distribution is created including only the most probable chord for each chroma-vector. Figure 19 (top and middle) illustrates this for a window of $n = 16$. The bin-frequencies of the histogram are divided by n so that the histogram can be considered as the probability distribution of the chords in the neighborhood, which are denoted as $P(c_i)$. The new probability of each chord c_i given the respective chromagram $\boldsymbol{chroma}$ is calculated according to the Bayesian theorem as:

$$P(c_i|\boldsymbol{chroma}) = \frac{P(\boldsymbol{chroma}|c_i) \cdot P(c_i)}{P(\boldsymbol{chroma})} \tag{5.2}$$

The information about $P(\boldsymbol{chroma})$ is not explicitly required as it is only a normalization constant to obtain values that sum up to 1. If only the rank but not the probability of a chord is interesting, it can simply be discarded, as it does not have any influence on the ranking of the chords. In case a probability value is required, it is sufficient to divide all values obtained without normalization by their sum.

In this approach, the probability $P(c_i|\boldsymbol{chroma})$ for chords that are not predicted at least once in the window is 0, because they have a marginal probability $P(c_i)$ of 0. Especially for small windows this appears to be too restrictive. To avoid marginal probabilities which are 0, one possibility is to add a constant number of *virtual appearance* to all bins which is calculated as $n \cdot \mathrm{virtAppFactor}$ (Figure 19, bottom).

Alternatively, apart from the most probable chord, the next r most probable chords for each chroma vector could be added to the histogram as well. This, however, raises the question of how to weight the different chords. Here, the value v_i to be added to the histogram for a chord ranked at position i is defined as:

$$v_i = \frac{P_{\mathrm{Rank}=i} - P_{\mathrm{Rank}=r+1}}{P_{\mathrm{Rank}=1} - P_{\mathrm{Rank}=r+1}} \tag{5.3}$$

where $P_{\mathrm{Rank}=i}$ denotes the probability of the chord that is ranked at position i. If, for example, the 3 most probable chords have probabilities of 0.06, 0.052 and 0.05 and $r = 2$, then the bin frequency of the most probable chord is increased by 1 and the bin frequency of the second most probable chord is increased by 0.2.

Furthermore, a measure of *reliability* is defined which is high if the prediction of the most probable chord is relatively certain and low if the difference to the second most probable chord is just marginal and thus the prediction is doubtful:

$$\mathrm{Rel} = P_{\mathrm{Rank}=1} \cdot (P_{\mathrm{Rank}=1} - P_{\mathrm{Rank}=2}) \tag{5.4}$$

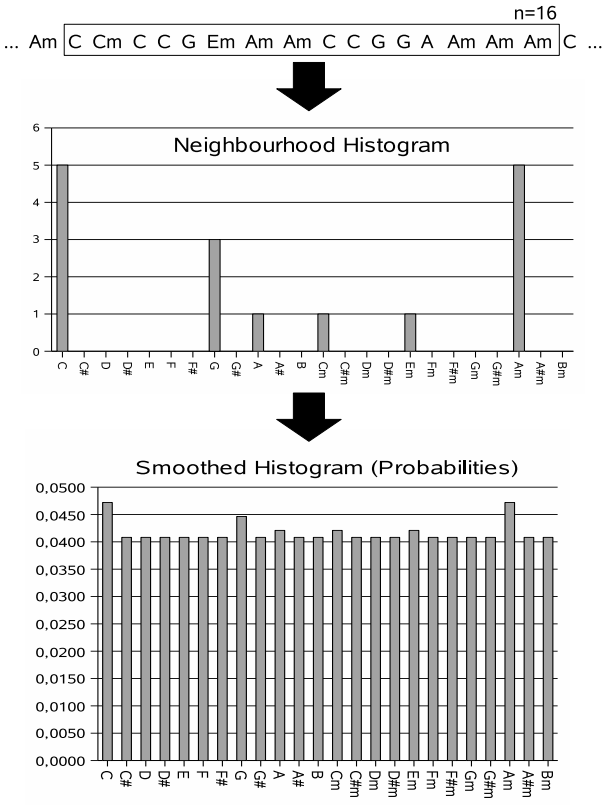


Figure 19: Illustration of the histogram-based smoothing approach (from [pub:4]) showing the Naïve predictions (top), the corresponding neighborhood histogram (middle) and the resulting smoothed histogram (bottom) obtained for $r = 1$, $\mathrm{virtAppFactor} = 2$ and $\mathrm{relBonus} = 0$.

All the reliability values of the predicted chords in a window are compared and a reliability bonus *relBon* added to the histogram bin of the most reliable chord. The bonus for the other chords decreases in the same manner as for ranked histogram probabilities, such that for the least reliable chord a bonus of 0 remains.

Finally, the result of the post-processing step can be smoothed further as it can just be interpreted as the output of a generic classifier. As possible classification errors may have been corrected, the updated histograms may be more reliable. Thus further iterations of the smoothing step might further improve the results. The number of additional iterations is denoted by the parameter *k*.

All the above parameters influence the smoothing. The impact of different parameter combinations on the quality of the chord predictions has been analyzed in an evaluation. Results are discussed in the next section.

5.2.4 *Evaluation*

The proposed post-processing technique has been tested on the two early *Beatles* albums *Please Please Me* and *Beatles For Sale* which have also been used for evaluation by BELLO and PICKENS [14], LEE and SLANEY [121] and HARTE and SANDLER [87].

The ground truth chord annotations were kindly provided by Christopher Harte.

For the evaluation with the **scalar product similarity classifier**, the following parameters were considered which resulted in an overall amount of 900 parameter combinations to be tested:

- the window size $n \in \{4, 8, 16, 32\}$,
- the factor multiplied by n to determine the number of virtual appearances in a histogram $virtAppFactor \in \{2, 5, 10\}$,
- the maximum rank of a chord to be included in the histogram $r \in \{1, 2, 3, 12, 24\}$,
- the bonus value to be added to the bin of the most reliable chord $relBonus \in \{0, 1, 3\}$, and
- the number of additional iterations $k \in \{0, 1, 2, 3, 4\}$.

The results are shown in Table 5. Throughout the experiments, small window sizes yielded better results than larger ones. The best results were obtained with a window containing only 4 chromagrams. In general the post-processing led to an improvement of the detection accuracy. Only in 4% of all tested parameter combinations the accuracy decreased. On average there was a relative increase by 10.72%. For the best parameter combination $(4, 5, 3, 1, 4)$, it was 19.05%.

	Baseline accuracy	Average accuracy (all parameter combinations)	Accuracy (best parameters)
Please Please Me	56.42%	64.12%	69.05%
Beatles For Sale	63.91%	69.10%	74.02%
Overall	60.16%	66.61%	71.62%

Table 5: Accuracies before and after the smoothing step using the scalar product classifier.

For the **Mahalanobis distance classifier**, the same parameter combinations were tested. The results are shown in Table 6. In contrast to the scalar product classifier, the Mahalanobis distance classifier as well as the Naïve Bayes classifier need to be trained on training data. To this end, training samples from two other *Beatles* albums – *With The Beatles* and *A Hard Day's Night* – were used. While the increase of accuracy for some promising parameter combinations is again large, there is a higher risk of over-smoothing and thus decreasing the accuracy. This happened in 18% of all cases. However, for small window sizes of 4 or 8, the histogram smoothing performed generally well. The best accuracy with a relative increase of 26,13% was reached with parameters $(4, 10, 12, 1, 4)$.

Table 6: Accuracies before and after the smoothing step using the classifier based on the Mahalanobis distance.

	Baseline accuracy	Average accuracy (all parameter combinations)	Accuracy (best parameters)
Please Please Me	54.60%	62.64%	72.52%
Beatles For Sale	62.81%	65.87%	75.56%
Overall	58.70%	64.25%	74.04%

As already stated in Section 5.2.2, different parameter values had to be chosen for the **Naïve Bayes classifier**. Here, just one or very few chords dominate the probability distribution and have high distance to the following chords. Hence, it is unlikely that any correction will occur if the classifier probabilities are multiplied with a highly smoothed histogram. Therefore, it is important not to donate too much initial appearance to the empty bins in the histogram by choosing a low $\mathit{virtAppFactor}$. This was approved by the tests where a $\mathit{virtAppFactor}$ of 0 performed best. The parameters tested for this classifier are:

- $n \in \{8, 16, 32, 128, 1000\}$,
- $\mathit{virtAppFactor} \in \{0, 0.5\}$,
- $r \in \{1, 3, 12\}$,
- $\mathit{relBonus} \in \{0, 2\}$, and
- $k \in \{0, 1, 2, 3, 4\}$.

The best results were obtained independently of choosing r and $\mathit{relBonus}$, with the other parameters set to $n = 1000$ (i. e., the whole song is considered as neighborhood), $\mathit{virtAppFactor} = 0$ and $k = 2$ or $k = 3$. As Table 7 shows, the results for using the post-processing step with this classifier were not as good as for the other classifiers. This may be due to the worse baseline accuracy, so that the enhancements relied on less accurate data.

Table 7: Accuracies before and after the smoothing step using the Naïve Bayes classifier.

	Baseline accuracy	Average accuracy (all parameter combinations)	Accuracy (best parameters)
Please Please Me	49.85%	52.49%	54.25%
Beatles For Sale	60.81%	63.98%	66.50%
Overall	55.02%	58.24%	60.37%

5.2.5 *Discussion*

The proposed post-processing step uses a probabilistic model to smooth the chord predictions. The marginal chord probabilities are derived from the distribution of the chords in the neighborhood defined by a sliding window. Hence, the approach is data-adaptive w.r.t. the context given by the sliding window.

As can be seen from the results, the post-processing step significantly improves recognition accuracy. This holds for all three different classifiers tested. A direct comparison with other post-processing approaches – especially with the Viterbi decoding step of the very popular HMMs – is difficult as this requires the use of the identical baseline classifier, chord alphabet and test set of songs. The classifier described by BELLO and PICKENS [14] (though not identical) is similar to the Mahalanobis classifier used here and the test set is the same. The results of both approaches only differ by 1% (they report 75,04% compared to 74,04%), so both methods seem to be nearly equally good, where the proposed method is notably less complex than HMMs. For further comparisons, the methods proposed by PAPADOPOULOS and PEETERS [186] and NOLAND and SANDLER [170] have been reimplemented. Both apply the Viterbi decoding for post-processing. The former reached 74,68% using a manually set transition matrix based on the double nested circle of fifth. Using a transition matrix based on the correlation between key profiles, the latter performed with 70,54% considerably worse – but still better than its double nested circle of fifth variant in the experiments in [186].

More details on the comparison can be found in [pub:4].

These comparable results proved that the approach based on the relatively simple idea of using the uncertain information of the chord distribution around a chord to aid a chord's prediction was able to compete with state-of-the-art techniques at the time of its publication in early 2008. Proposals for future directions of research comprised to integrate further information for a more complex probabilistic context-model – most importantly [pub:4]:

- knowledge about the roughly extracted harmonic movement of the piece,
- static knowledge about chord changes as stated by music theory,
- (uncertain) information obtained by automatic detection of measure and key, and
- special treatment of bass tone intensities

where the first two points correspond to the information exploited by the HMM-based approaches mentioned above. Indeed, the chord detection approach later described by MAUCH and DIXON [148] uses a very sophisticated probabilistic model – the dynamic Bayesian network shown in Figure 6 on page 35 – including metric position, key, chord, and bass which allows to differentiate between 109 different chords with superior accuracy.

5.3 SUMMARY

This chapter presented two examples that demonstrate how adaptive techniques can be incorporated into the feature extraction process in

order to increase the robustness and quality of the extracted information:

- an adaptive noise removal technique which can be used to remove background music from a stereo recording as a preprocessing step for extracting features that capture information about the melody, and
- an adaptive smoothing technique which can be applied as a post-processing step for chord recognition in order to correct unavoidable misclassifications caused, e. g., by the presence of percussive sounds or harmonics.

Both techniques have been primarily developed by diploma students who have been supervised in the context of this PhD thesis. Hence, most credit for the work goes to them.

When you try your best
but you don't succeed,
When you get what you want
but not what you need, ...

"Fix You"
Coldplay

6 USER-ADAPTIVE GENRES

While the preceding chapter covered data-adaptive feature extraction approaches, this chapter addresses the problem of genre classification. It can be considered as one of the most common MIR tasks belonging to the classification/prediction retrieval scenario which Figure 20 highlights as part of the general retrieval process described in Section 2.2.1. Here, it is considered as an internal process of the core retrieval system with the goal to predict *meaningful* genre annotations that can, e. g., later be used for collection structuring.

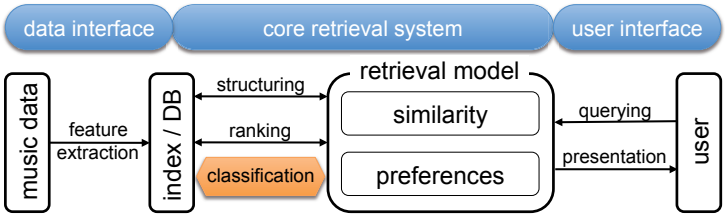


Figure 20: Genre classification in the context of the general retrieval process described in Section 2.2.1.

Objective genre classification appears to be a hard task and a general applicable classification scheme which holds for everybody has not yet been agreed upon. As observed in experiments by Mitri, Uitdenbogerd, and Ciesielski [160], there is a high disagreement between people on the genre assignment of musical pieces as well as the categories to be used. This problem has been bothering the MIR research community for about a decade and the question has come up, whether it is really worth *"trying to teach a computer a marketing construct"* (Brian Whitman, co-founder of the *EchoNest* [url:47]). For instance, McKay and Fujinaga [155] argue for continuing genre classification research but Sordo et al. [228] conclude that experts and "wisdom of the crowd" agree only for specific, well defined genres (hip-hop, blues) whilst in other cases (e. g., rock) there is no or little correlation. An alternative to dealing with a supposed marketing construct could be to promote the idea of user-specific "idiosyncratic genres".

Several studies indicate that there might be meaningful user-specific genres emerging from usage patterns that people consciously or unconsciously use when they access music collections or describe music: In a user study, Jones, Cunningham, and Jones [104] analyzed organization and access techniques for personal music collections. Several "idiosyncratic genres", i. e., "genres" peculiar to the individual, could be identified that users tend to use to classify and organize their music.

These idiosyncratic genres comply largely with the usage context. Typical examples could be "music for driving (and keeping me awake)", "music for programming" or "music to relax in the evening after a long working day". Further, Bainbridge, Cunningham, and Downie [9] revealed in an analysis of requests at the answering service *Google Answers* [url:14] in the category "music" that such descriptions were also used in this public setting. In a larger survey on search strategies for public MIR systems by Lee and Downie [119], more than 40% stated that they would query or browse by usage context if this would be supported by the system. Strong correlations between genre, artist, album and the usage context also became apparent in a more recent study by Hu, Downie, and Ehmann [96].

Even for the construction of an objective taxonomy of 378 music genres, Pachet and Cazaly [177] included features referring to consumer context ("audience location") and usage ("danceability") as important criteria. Govaerts, Corthaut, and Duval [80] describe how such meta-data is already exploited in a commercial application to select music for a desired atmosphere in hotels, restaurants and cafés. However, the respective properties need to be assigned manually by experts and if necessary can only be adapted by hand. It would be very desirable to have at least a semi-automatic context assignment from automatically retrievable or measurable data. According to the definition by Dey [52] (which has already been considered for defining the context of an adaptive system in Section 3.2.1), any information that can be used to characterize the situation of a person, place or object of consideration makes up its context. He differentiates four types of primary context: location, identity, time, and activity [53]. In the MIR domain, there exists already a variety of systems that capture time, (user) identity and location, e. g., the popular *Audioscrobbler* [url:3] plug-in from *Last.fm* [url:22]. As shown in Section 3.3.3.2, such information can be exploited to describe the usage context in context-aware music recommender systems. However, to the author's knowledge, it has so far not been used for personalized access to music collections.

Recalling the phenomenon of idiosyncratic genres observed in user studies, genres based on listening habits yield a high potential for supporting an individual user in maintaining and using his personal music collection. Adapting not only the *process* of genre classification but also the *classes* (i. e., the genres) could lead to a higher degree of personalization and user satisfaction. Alternatively, information about the usage context could be used directly to browse a music collection or to enrich a similarity-based structuring as orientation aid or as separate content-describing facet.

From a pilot study described in Section 6.1, several possibilities to automatically record a large variety of information about the listening context emerged which are addressed in Section 6.2. These could be used in order to enrich MIR applications with information about a

user's listening habits. However, recording such information could violate the user's privacy. Therefore, a subsequent survey has been conducted to assess the general acceptance of listening context logging amongst potential users. The results are presented and discussed in Section 6.3. Finally, Section 6.4 summarizes this chapter.

6.1 PILOT STUDY

In order to motivate and demonstrate how widely available environmental data can be exploited to allow organization, structuring and exploration of music collections by personal listening contexts, a small pilot study was conducted. To this end, a logging plug-in for music players that automatically records data about the listening context was developed and used to collect data in a small user experiment as described in Section 6.1.1. Several standard data mining techniques were applied to reveal common usage patterns in the collected data. This is covered by Section 6.1.2. Further, a prototype user interface based on elastic lists for browsing by listening context has been developed which is presented in Section 6.1.3.

The work described in this section has been published in [pub:5].

The logging plug-in and the context browser prototype have been implemented as part of the diploma thesis of Valentin Laube [stud:3] who also took care of the data acquisition.

6.1.1 *Data Acquisition*

For logging the context information together with the played songs, a plug-in for the *foobar2000*, *Winamp* and *iTunes* music players was developed. Whenever a song is played, the plug-in records its ID3 meta-data together with a time stamp and the "end reason", i. e., whether the song played till the end, was skipped or the player was closed before the song ended. Further, information about the local weather conditions is gathered from online services. The location of the user is estimated by resolving the IP address of the computer. If the computer is offline, data is gathered once it is re-connected with the internet. The recorded data is cached in a local *SQLite* database [url:43] and transferred in constant intervals via HTTP to a central server that collects the data for analysis.

In a small test experiment with 8 participants, 15325 played songs were logged between February and April 2008. The data comprises the following 14 dimensions: user id, artist, title, album, genre, date (from ID3) end reason y, weekday, time of day t, weather quality w, temperature ϑ, humidity, air pressure p, pressure change (during the last four hours) Δp. However, because of initial problems with the logger, a large number of the records is not complete. Moreover, the data is biased towards bad weather and about half of the data has been contributed by a single user. This is important to keep in mind when assessing the data mining results presented in the following section.

Table 8: Discretized attribute values used for data analysis. (weather quality is discrete)

attribute	(discretized) values
time of day	morning (5-8), forenoon (8-11), noon (11-14), afternoon (14-17), evening (17-20), night (20-23), late night (23-5)
weather quality w	sunny, mostly sunny, partly sunny, clear, mostly clear, partly cloudy, mostly cloudy, cloudy, overcast, light fog, mist, light snow, snow shower (sleet), snow, drizzle, light rains shower, light rain, rain shower, rain
temperature ϑ	$<0\,^{\circ}$C, 0..5 $^{\circ}$C, 5..10 $^{\circ}$C, 10..15 $^{\circ}$C, 15..20 $^{\circ}$C, $\geqslant 20\,^{\circ}$C
pressure p	<900 hPa, 900..1000 hPa, 1000..1050 hPa, $\geqslant 1050$ hPa
pressure change Δp	neg. big ($<$-10 hPa), neg. medium (-10..-5 hPa), neg. small (-5..-2 hPa), zero (-2..2 hPa), pos. small (2..5 hPa), pos. medium (5..10 hPa), pos. big (>10 hPa)

6.1.2 *Data Mining*

The data mining was done in collaboration with Christian Moewes.

In order to find common listening patterns or useful information from the acquired music data, several data mining techniques from the data analysis platform *Information Miner* [url:18] were applied. The focus was on learning the dependency between the weather conditions $\mathcal{X}$ and the reason why a user ended a song $\mathcal{Y} = \{\texttt{finished}, \texttt{skipped}, \texttt{quitting}\}$. Rather than expecting new insights from the results of the analysis, the primary aim was to identify suitable techniques for finding patterns that capture listing behavior and to gather early feedback from the presentation of the results.

Formally, the problem can be described as finding a function $f : \mathcal{X} \mapsto \mathcal{Y}$. First, the data was projected to a subset of attributes, i. e., $\mathcal{Y}$ and $\mathcal{X} = \{t, w, \vartheta, p, \Delta p\}$. In the second preprocessing step, all records containing missing values were removed. After this step, only 2064 records remained for further analysis. As a final step before the data mining, all continuous variables were discretized as shown in Table 8. Using these attributes, the data was analyzed with several techniques from the *Information Miner* toolkit.

Figure 21 shows an induced graphical network structure [21] that was generated by applying the K2 metric [42] to all variables $\mathcal{X}$ and $\mathcal{Y}$. Edges indicate interdependencies of attributes. Not surprisingly, the interdependencies shown in the network match with common-sense knowledge. For instance, the time of day has an impact on the temperature, the air pressure and the weather quality. Analyzing the impact of the time of the day and the weather quality on the end reason, several rules can be generated from the model. They are shown as circles in Figure 22, plotted by lift and recall and colored with respect to the end reason. The most interesting rules are close to the top right corner.

Figure 23 shows a decision tree [201] learned on the same attributes as the graphical model with $\mathcal{Y}$ set as the class variable. It allows

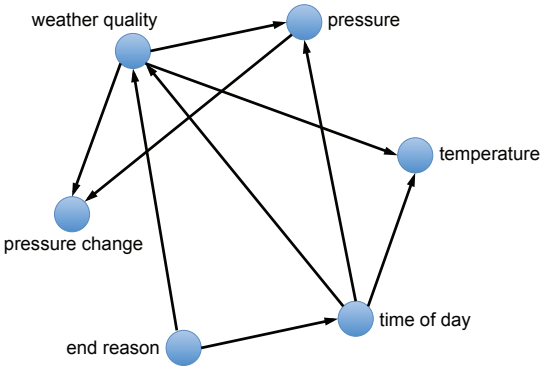


Figure 21:
The most probable graphical model given the data. Edge directions can be ignored in this case as only interdependencies are of interest. Note that the applied K2 metric tries to maximize the probability of a directed acyclic graph given a database of sample cases.

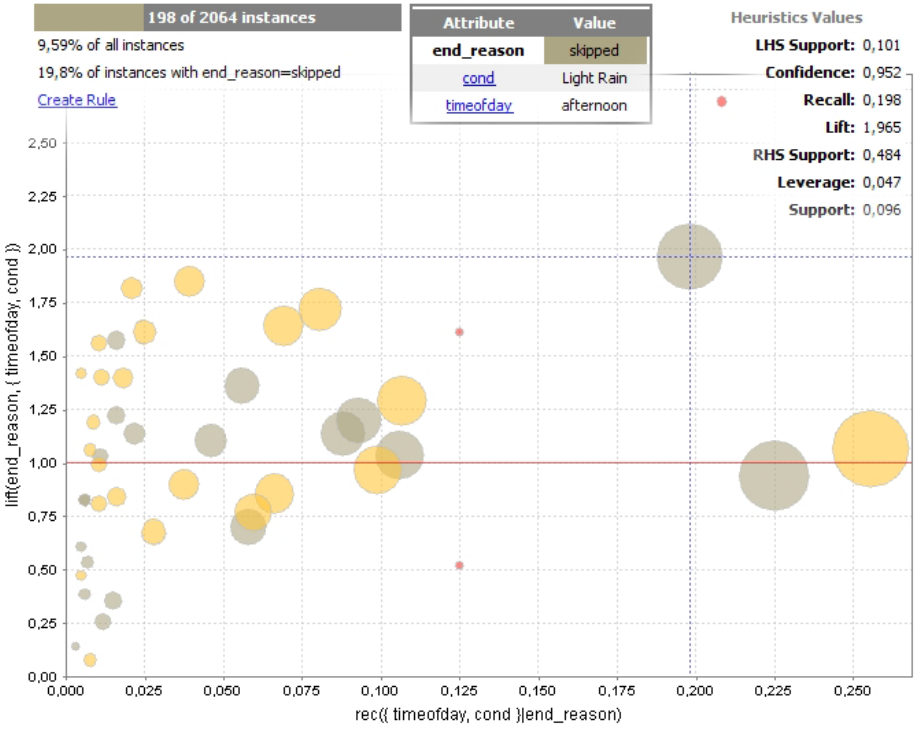


Figure 22: Some found association rules for the end reason plotted by their recall and lift. The colors represent the different end reasons, i. e., yellow corresponds to finished, gray to skipped and red to quitting. The selected rule (blue cross hairs) is as follows: In case of light rain in the afternoon there is a 10% chance of skipping a song. Note that the area of each circle is directly proportional to its rule's relative number of instances.

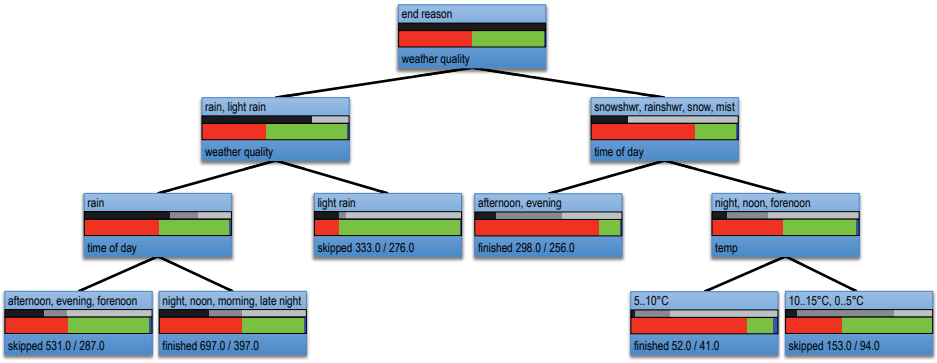


Figure 23: Decision tree for the end reason. Tree nodes consist of three rows: selected attribute values (top), value distribution of the end reason (middle with red=finished, green=skipped and blue=quit) and the final decision and its accuracy (bottom).

to predict the end reason from observed attributes by following the branches (which represent attribute combinations that with each level become more specific) top-down to the corresponding leaf (which represents a classification). Values of the same attribute were grouped into single nodes wherever possible to reduce the complexity of the tree [19]. The resulting tree was pruned to a maximum height of 4. Several selection measures were tested in order to find reasonable tree structures of which the sum of weighted differences showed the most promising results. In the induced tree, every path from the root to a leaf node corresponds to a rule that can be directly derived. For instance, if there is snow shower, snow, light rain shower or mist in the afternoon or evening, then there is a $256/298 = 86\%$ chance that the user will finish the song.

In order to apply frequent pattern mining [20] to the problem, the Apriori algorithm [2] for finding maximum item sets was used. The only difference between the previous tests and this one is the extended set of attributes. Here, the weekday was considered additionally. The identified frequent item sets with their relative frequency of occurrence in the dataset (called *support*) are listed in Table 9. From these item sets, rules can easily be constructed by putting all attribute values in a table row except for y into the antecedent (precondition). The consequent is simply determined by y.

Finally, naïve and full Bayesian classifiers [21] were trained on both, the discretized and the raw data. However, the results were not satisfactory and the induced rules were also harder to interpret. Therefore, detailed results are omitted here. Yet, Bayesian learning might still come up with useful information if there is more data with fewer missing values.

Table 9: Induced maximum item sets ordered by descending relative support. Only item sets containing an item from $\mathcal{Y}$ and a minimum support of 10% are shown.

$\mathcal{Y}$	t	w	ϑ (°C)	p (hPa)	Δp	weekday	rel. supp.
skip		rain	5..10	1000..1050	zero		16.0%
finished		rain	5..10	1000..1050	zero		15.2%
skip		light rain					13.4%
finished						Saturday	12.3%
skip	afternoon		0..5				11.8%
skip	afternoon					Saturday	11.3%
skip			0..5			Saturday	11.0%
skip	night	rain			zero		10.9%
finished	night	rain		1000..1050	zero		10.8%
skip	night		5..10	1000..1050	zero		10.7%
finished		light rain shower					10.6%
finished					zero	Monday	10.6%
skip			10..15				10.4%
finished			0..5		zero		10.3%
finished	evening			1000..1050	zero		10.1%
finished			10..15	1000..1050	zero		10.1%

6.1.3 *Context Browser Prototype*

For a first prototype user interface that allows browsing by listening context, the elastic list technique [231] was adopted that has been developed for browsing multi-faceted data structures.[1] This approach enhances traditional facet browsing interfaces such as presented by DACHSELT and FRISCH [47] for music collections that allow a user to explore a data set by filtering available meta-data information. In the scope of the pilot study, all available context meta-data was used as facets, i. e., user, time of day, day of week, weather quality, temperature, air pressure and air pressure change (during the last 4 hours). The discretized version of the attributes had to be used as only discrete attributes are supported. Further, the logged ID3 meta-data (artist, title, album and genre) could also be used naturally as facets. A screenshot of the interface is shown in Figure 24.

Additionally to basic facet browser filtering, elastic lists visualize relative proportions of values by size. For instance, the sizes of the blocks referring to the days of the week in the respective facet column reflect the distribution of the number of played songs on these days. Selecting some value of a facet as filter updates the proportions of the blocks in all other facet columns, now reflecting the distribution of

1 There is also an online demo of the original elastic lists user interface [url:9].

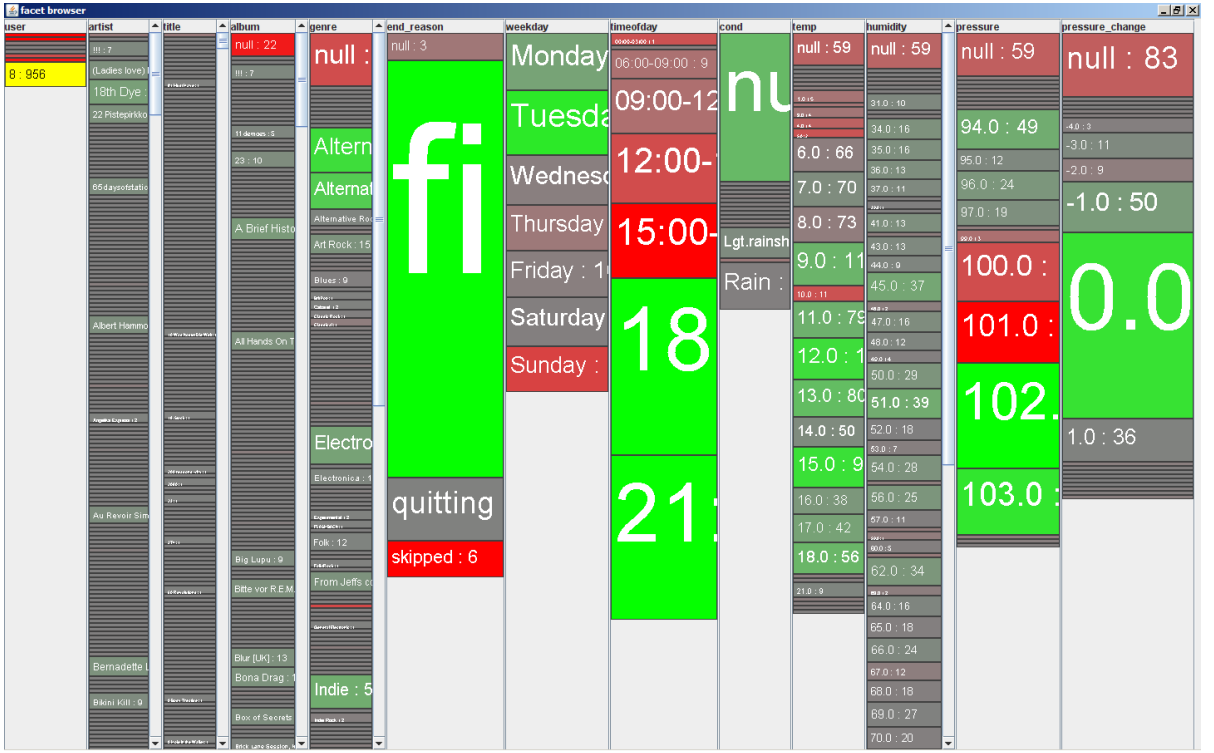


Figure 24: Context browser prototype. User #8 has been selected (yellow). He mostly lets the songs play till the end which is significantly more probable than for the average user (green). He usually listens to music in the evening and at night (green) and less often than other during the day (red).

the facet values under the given filter constraints. Furthermore, elastic lists visualize unusualness by brightness. This approach was slightly modified here to visualize negative and positive deviations. If, for instance, the end reason "skipped" is selected and for some value of a facet the number of skipped songs is significantly higher or lower than the expected average value, then the respective block is colored red or green respectively. Brighter colors indicate a stronger deviation from the expected value. For instance, in Figure 24 the selection of user #8 shows that in this context, significantly more songs are finished than usual.

6.1.4 *Conclusions from the Pilot Study*

The pilot study motivated and demonstrated how widely available environmental data can be exploited to allow organization, structuring and exploration of music collections by personal listening contexts. The data collected during the small user experiment was however too limited and also very biased (especially towards bad weather due to the recording period) to allow for interesting new insights on listening behavior from the analysis. For more significant results, data collected over a longer time period would be required. Nevertheless, from the pilot study and the discussion of its results at the 2nd International Workshop on Learning Semantics of Audio Signals (LSAS 2008) [pub:5], a variety of ideas emerged on how the automatically listening context logging could be further extended by more sophisticated techniques. A plug-in architecture was proposed that would allow to add or remove "sensors" for different data related to the listening context. The following section covers several of the proposed extensions for automatic listening context logging.

6.2 POSSIBILITIES FOR AUTOMATIC LISTENING CONTEXT LOGGING

DEY [52] considers any information as context that can be used to characterize the situation of a person, place or object of consideration. He differentiates four types of *primary context*: location, identity, time, and activity [53]. The activity is especially important because it is closely related to the idiosyncratic genres identified by JONES, CUNNINGHAM, and JONES [104] (cf. page 75, bottom for some examples). Moreover, knowledge about the user's current activity might enable a more sophisticated modeling of the listening context with regard to the listening modes defined by HURON [97].

From the ideas that emerged from the pilot study, the following options for automatic context logging were considered for implementation:

- *Music meta-data*
  Together with a timestamp, this is the minimal information required to build a user's listening profile. It usually comprises the title, artist, album of a song. In the MIR domain, there exists already a variety of systems that capture time, (user) identity and location, e. g., the *Audioscrobbler* [url:3] plug-in of the music community website *Last.fm* [url:22]. Similarly, *iTunes Genius* [url:21] sends information about a user's music collection and playlists to a central server. This information is then used to generate *Genius* playlists through collaborative filtering.

- *Ambient noise*
  Most devices that are capable of playing music also have a built-in microphone. This could be used to record short snippets (1 or 2 seconds) of the environmental soundscape in the gap between two consecutive songs. (Alternatively, signal processing methods could be applied to remove the known music signal from the recorded one. This way, information could even be collected when music is playing.) From the resulting sound snippets, noise profiles could be generated that have enough information to classify the soundscape into general categories like "*silence*", "*people talking*", "*nature sounds*", "*traffic sounds*" or "*party*" thus giving valuable information about location and activity. Using noise profiles instead of the actual recording would further not allow sensible information to be extracted.

- *GPS position*
  Many mobile devices nowadays have a GPS receiver. Periodically recording the position would provide valuable information about the location. Together with the inferred speed of travel this could be linked with specific activities or travel.

- *Keyboard and mouse events per minute*
  Assuming the listener is using a computer, further possibilities arise. Detecting whether and how much the mouse and keyboard are used in a sliding time window yields evidence about the user's activity. For instance, low keyboard and mouse activity may indicate reading or browsing whereas high keyboard activity may refer to writing a text or programming. It might not even be necessary to derive such higher-level activity description. The low-level information might be already sufficient to distinguish activity contexts. Note that no information about the actual keys would be recorded such that it would be impossible to reconstruct the data that was entered.

- *Currently running applications*
  Together with the previous one, this is probably the best source of information about the listener's current activity if a computer

is used but also comes close to surveillance. Through a client-side tracking software, it could be periodically logged which applications are currently running and which application has the focus. The social networking website *wakoopa* [url:52] already uses such a technique to monitor which programs and web applications are used by its members and build profiles.

- *Facial expression*
  Many notebooks and mobile phones have a built-in webcam. With such a camera, the facial expression of the user could be classified periodically using, e. g., image processing methods applied in the context of human- computer interaction [99]. Note that only the classifications and no actual images need to be logged.

- *Bio-information*
  Listening context information can also comprise direct information about the user's current condition such as the skin conductance, body temperature, breathing rate or heart pulse. For instance, the adaptive system for playlist generation called *PAPA* (Physiology and Purpose-Aware Automatic Playlist Generation) [174] as well as the already commercially available *BODiBEAT* music player [url:56] use sensors that measure certain bio-signals (such as the pulse) of the user as immediate feedback for the music currently played. This information is then used to learn which characteristics of music have a certain effect on the user. Based on this continuously adapting model playlists for certain purposes can be created. Alternatively, the sensor information could be used to derive listening contexts.

- *Ambient light*
  Recently, some notebooks are equipped with illuminance sensors to adapt the display brightness. Data from such sensors could be exploited, too. Whilst this might provide only little information about the listening context, logging this would result in almost no interference with the listener's privacy.

- *(Attentional) Status*
  There exists a variety of applications that allow a user to set his current status: instant messenger applications come with predefined states such as "*online*", "*away*" or "*occupied*" and the option to specify an additional custom status message. Further, many social networking websites or micro-blogs such as *twitter* [url:50] allow their members to specify what they are currently doing. Depending on how much effort a user puts into updating the status as a measure of communication, this information may be very valuable to describe the listening context.

Except for the bio-information, all data could be gathered at low costs by using only built-in hardware. Furthermore, it could be measured without distracting the user from his current activity. Clearly this is an advantage over simply asking the user, what he is currently doing. (The latter would require a user action without a directly recognizable benefit so that there is hardly any motivation for the user to cooperate.) However, for most of the logging options, privacy appears to be the most important issue as more sophisticated methods would come close to surveillance. So the question is rather not, what would be technically possible but how much information about his activities a user would be willing to share. Therefore, a survey has been conducted to assess the acceptance of the above options for automatic listening context logging.

6.3 A SURVEY OF THE ACCEPTANCE OF AUTOMATIC LISTENING CONTEXT LOGGING

The work described in this section has been published in [pub:9].

Section 6.3.1 briefly describes design and the context of the survey. The results of the survey are presented in Section 6.3.2. Possible correlations between the background of the participants and their acceptance of context logging are investigated in Section 6.3.3. Finally, Section 6.3.4 draws conclusions.

6.3.1 *Survey Design and Context*

The design of the survey emerged from pre-surveys with a small number of participants. The original key question targeted MIR applications in general but was considered as too abstract. It had to be rephrased so that it described a scenario that participants not familiar with the MIR domain could easily grasp. Therefore, the chosen motivation was to learn personalized genres for sorting a personal music collection:

> *Current music players allow to sort music according to genres. Unfortunately, genres are often either too general (e. g., rock/pop) or far too specific (e. g., "Scottish lo-fi post-rock" for the band "Mogwai") such that they are not very helpful for sorting. An alternative is currently investigated within the AUCOMA project of the DKE research group: It might be possible to learn individual "genres" that reflect a user's listening habits (e. g., "breakfast music", "car driving music", "party music"). These could be used to structure the music collection according to individual listening habits. For the identification of different listening situations, the player could record a variety of information.*
>
> *BUT: Recording such information may violate your privacy! Therefore, please tell us what information your music player may record about you!*

The survey contained 8 questions that can be categorized into 4 topics, each presented in detail with the results in a respective subsection of Section 6.3.2:

1. Demographic information comprising, gender, age and country of residence.
2. General relation to music.
3. Use of (web-) applications that collect, access and expose to some extent private data of their users.
4. Acceptance of logging information about the listening context.

The last topic represents the main question of the survey whereas the others were added to be able to estimate a possible bias of the participants. Further, correlations between the background of the participants (especially in relation to music) were expected that could be identified.

The survey was conducted in two parts: Between March 3rd and March 8th, 2009 a paper questionnaire in German was filled out by 156 fare visitors of the German *CeBIT 2009* fare [url:6]. To extend the scope of the survey, an online questionnaire was designed afterwards based on the paper version using the open source online survey application *LimeSurvey* [url:26]. It was open to the public from March 16th until June 15th, 2009. The questions of the online questionnaire were identical to those of the paper version. However, the questions were split across multiple (screen-) pages and an English translation was added for international participants. Both questionnaires can be found in Appendix C. 305 persons filled out the online questionnaire resulting in 461 participants in total.

6.3.2 *Survey Results*

6.3.2.1 *Demographic Background of the Participants*

From the 461 participants of the survey, 101 (i. e., 21.9%) were female and 354 (i. e., 76.8%) were male. 6 persons did not answer this question. The average age was 29.25 with a standard deviation of 10. Table 10 shows the countries with more than 5 participants in the survey. In total, persons from 34 countries participated where 4 persons did not state their country of residence.

The majority of the participants was from Germany. This is primarily due to the fact that part of the survey was conducted amongst visitors of the German *CeBIT* fare. Further, the survey was advertised at the *Otto-von-Guericke-University*. Thus, it can be assumed that there are many German students amongst the participants. Most of the international participants had probably been informed about the

country	number of participants	percentage
Germany	323	70.07%
USA	24	5.21%
Austria	14	3.04%
France	10	2.17%
Turkey	7	1.52%
Switzerland	6	1.30%
Spain	5	1.08%

Table 10: Countries with more than 5 participants in the survey.

survey by the announcements posted on popular MIR mailing lists [url:35] such as the *music-ir* list maintained by the *Institut de Recherche et Coordination Acoustique/Musique (IRCAM)* [url:19].

6.3.2.2 *General Relation to Music*

6 statements were given that described a person's general relation to music. The participants were asked to check all of these that were applicable to themselves (using tick boxes). Figure 25 shows the statements and how often each one was checked.

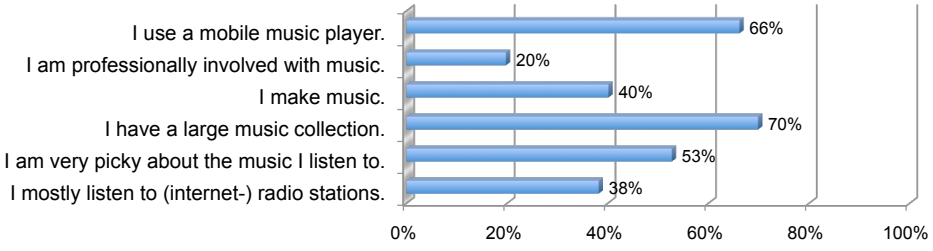


Figure 25: Statements selected that describe the person's general relation to music.

Further, the participants were asked, how frequently they listen to music. The possible answers for this question were deliberately formulated in a fuzzy way as participants of a pre-survey found this less complicated. The distribution of the answers is shown in Figure 26.

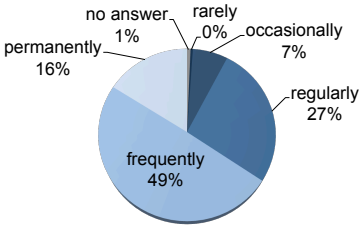


Figure 26: Answers for the question: *How frequently do you listen to music?*

6.3.2.3 *Use of (Web-) Applications that Collect, Access and Expose to some Extent Private Data of their Users*

For 10 categories, one or two popular representatives were chosen exemplarily and depicted by their logos:

1. *last.fm* [url:22] - a music community website, that builds a detailed profile of each user's musical taste by recording details of all the songs the user listens to via a plugin installed into the user's music player.
2. *flickr* [url:11] – an image and video community website that allows users to share personal photographs.
3. *delicious* [url:8] – a social bookmarking website for storing, sharing, and discovering web bookmarks.
4. *wakoopa* [url:52] – a social networking website that monitors which programs and web application are used by its members through a client-side tracking software.
5. *wordpress* [url:54] / *blogger* [url:4] – two popular blog publishing services.
6. *twitter* [url:50] – a micro-blogging service that enables its users to send and read short text-based messages of up to 140 characters.
7. *LinkedIn* [url:27] / *XING* [url:55] – two business-oriented social networking websites mainly used for professional networking.
8. *facebook* [url:10] / *studiVZ* [url:45] – two networking websites, the latter being rather popular amongst german-speaking students.
9. *Gmail / Google Mail* [url:16] – a webmail service.
10. *Google Docs* [url:15] - a web-based word processor, spreadsheet, presentation, and form application.

Participants were asked whether and how often they use each of these 10 applications. Figure 27 shows the distribution of the answers for the different applications. For better readability, the applications were sorted by decreasing number of users (i. e., participants that answered either *"yes, frequently"* or *"yes, occasionally"*).

6.3.2.4 *Acceptance of Logging Listening Context Information*

Here, answers were mandatory in contrast to the preceding questions as this was the key point of the survey. Figure 28 shows the distribution of the answers for this question sorted by the decreasing number of persons that answered *"yes (unconditionally)"*. Furthermore, there was the possibility to leave a comment, if for some case the answer was *"maybe"* or *"no"*.

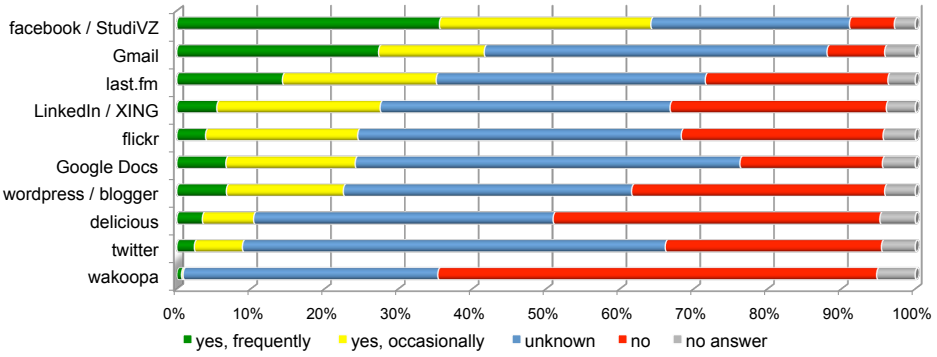


Figure 27: Answers to the question: *Do you use the following (or comparable) applications?*

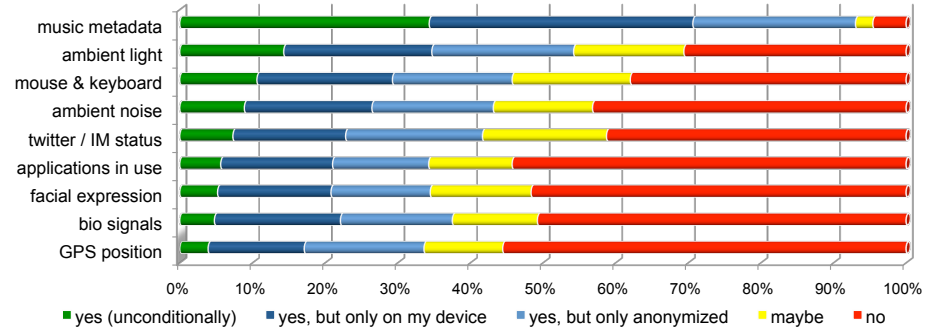


Figure 28: Answers for the question: *Would you allow your music player (as software or as a self-contained device) to log the following information in order to enable it to learn personalized genres for sorting your music collection? (It is assumed, that you can pause the logging anytime you find it inappropriate.)*

Clearly, logging the music metadata was the variant with the highest acceptance. Given that about 34% of the participants used *Last.fm*, an unconditional acceptance of at least 34% could be expected. Surprisingly, however, the number is not higher than this. It is also a surprise that only less than 15% would allow logging of the ambient light – supposedly the least sensible information of all. Logging mouse and keyboard events per minute, ambient noise or the status via twitter or instant messaging had medium acceptance. For the latter option, a much higher acceptance was anticipated given that the twitter and instant messaging status is by default visible to anyone and thus not an information that people tend to keep private. On the other end of the spectrum, logging the GPS position was the least popular option. Surprisingly, it was even slightly less accepted than logging bio signal which is definitely most obtrusive. Taking additionally into account the conditional acceptance (i. e., either local or anonymized logging), the participants would rather agree with logging of bio signals than facial expression or applications in use.

In the comments, many participants stated that they are concerned about their privacy. Some were against any kind of data collection whereas others did not intend to log data unless they are convinced of the benefit, i. e., an actual improvement of the MIR system. Some even expressed their doubts that such context information could actually be relevant to learn personalized genres. On the other hand, there was fear that users could be patronized by their "intelligent music-player" and would no longer be in control of the music selection. Another fear was that the collected information could be used for marketing purposes. A few participants remarked that the additional logging functions would require more storage and processing power or that the development in the end would increase the costs for the hard- or software. Furthermore, some wrote that they would be worried that data once recorded could leak out of the system, e. g., if someone hacked the server or even their computer.

6.3.3 *Analysis of Factors that Influence the Acceptance*

This analysis was done in collaboration with Matthias Steinbrecher and has been published in [pub:9].

While the findings presented so far allowed to create a coarse picture of the idiosyncrasies of the survey participants, this section addresses possible dependencies in greater detail. The general question of interest was how the acceptance of the amount of context logging varies under certain conditions such as gender, age and other attributes specified in the survey. For this, the possible answers to the 9 different logging options discussed in Section 6.2 were assigned weights from 0 to 2 in the following way: Unconditionally accepting the logging was weighted by a factor of 2. Acceptance of local context information storage on the user's device got assigned the weight 1 while both, the claim for anonymized storage and the indecisiveness (indicated by a *"maybe"* answer), amounted 0.5 to the score. Finally, a *"no"* led to zero weight. A new "logging" attribute was introduced by summing up the respective weights of all 9 possible logging contexts (yielding scores between 0 and 18) and subsequently binning it into six equidistant groups which served as indicators for the general acceptance of logging context information.

The remainder of this section investigates selected conditional distributions of this new logging attribute. The applied visualization is explained along with the answer to the question how the logging acceptance varies among the country of residence of the respective survey participant. Figure 29 depicts a sequence of stacked bar charts, one for each distinct condition instance, here: country of residence. Every bar chart represents the relative frequency of the values of the logging attribute, starting with the lowest acceptance at the bottom and increasing acceptance vertically. The widths of the bar charts correspond to the probability of the condition instance itself. Since the majority of participants were Germans, the majority of 70% from

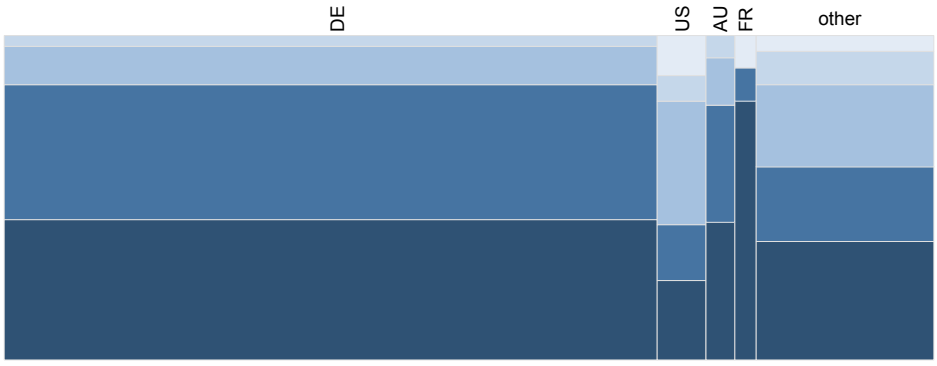


Figure 29: Distributions of logging acceptance given the country of residence. Brighter areas belong to higher acceptance.

Table 10 on page 88 is reflected by the broad column in the left of Figure 29. Interestingly, the next smaller population (USA) has considerable fewer reservations against collecting context information as can be seen by the respective column which shows a large portion of participants that belong the highest acceptance group.

Figure 30 shows the logging acceptance distribution conditioned on both, gender and age group. The majority of survey participants depicted by the third bar chart were male persons between 20 and 30 years old. It was unexpected that this subgroup showed a lesser

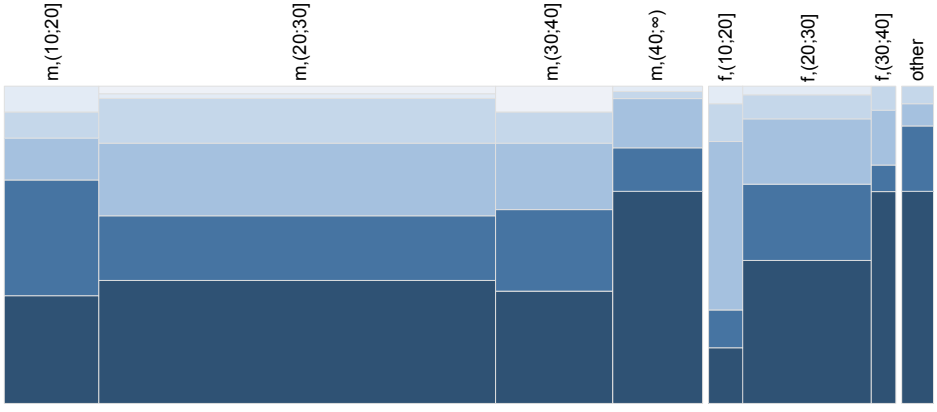


Figure 30: Distributions of logging acceptance given the age group and gender of the survey participant. Brighter areas belong to higher acceptance.

affinity to context logging in general than the next older subgroup represented by the first bar chart in the row. Another noteworthy subgroup comprises the persons under 20 years of age. First, they show the most drastic difference in the distribution when further conditioned on gender. And second, this subgroup seems to be quite

reluctant to context logging as there is no participant that belongs to the highest acceptance group. This is insofar interesting as a rather airy handling of private information was anticipated here.

In order to analyze the impact of the personal relation to music (Section 6.3.2.2) on the logging acceptance, the six different attributes were aggregated into one quantifying the overall affinity to music. This new attribute has six values with 0 representing no relation to music at all and 6 denoting strongest relation (by having ticked all possible relations in the survey). Here, no significantly differing subgroups could be identified, neither for the aggregated affinity to music nor for the individual relations.

The remaining question to be answered was how the involvement and participation in the several web applications and online communities (cf. Section 6.3.2.3) influences the logging acceptance. To assess the overall involvement, the respective attributes were again aggregated into a new one with a domain of four possible values: "low" if the participant used up to 2 web applications, "medium" if he used 3 or 4, "high" if he used 5 or 6 and "very high" of he used more than 6. Brighter segments in the chart of Figure 31 correspond to a higher usage. A correlation between the level of logging acceptance and the intensity of using online communities and web applications is clearly observable, respectively: The more a participant was involved into online communities (and thus: the more he was used to give away personal information), the higher the acceptance of logging.

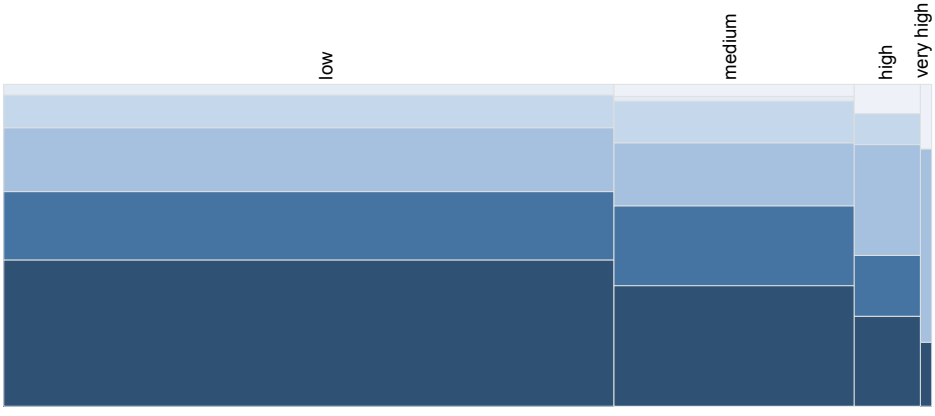


Figure 31: Distributions of logging acceptance given the usage intensity of online communities and web applications. Brighter areas belong to higher acceptance.

6.3.4 *Conclusions of the Survey*

As expected, privacy was an important issue for most of the participants but it was more important than anticipated. For many par-

ticipants of the survey the sophisticated logging methods addressed came close to surveillance. Thus, they were reluctant and demanded to be fully informed about the extent of the logged data and in full control of whether they want this data to be logged or not. Many participants were skeptical and first need to be convinced of the benefits from providing their context information. Furthermore, even if the logged context information might indeed prove to be useful for MIR applications, the users still want to be in full control of the music selection. This should be considered as guideline for developing future personalized MIR applications.

6.4 SUMMARY AND OUTLOOK

This chapter presented findings from a pilot study and a subsequent survey investigating the potential of automatically recorded listening context information for learning idiosyncratic genres adapted to the user. Such genres would make the results of genre classification more meaningful as users can directly relate to them. As result of the pilot study, several possibilities for automatically gathering context information at low costs and with mostly little hardware requirements have been proposed. However, the survey showed that the question should rather not be, what would be technically possible, but what and how much information about his activities a user would be willing to share. The more sophisticated logging methods were rather perceived as means for surveillance – presumably because an immediate benefit for providing this information could not be seen - and thus privacy issues were raised. The survey further emphasized the need for keeping the user in control of both, the recorded information about his listening habits as well as whether and how it is used for adaptation within personalized MIR applications – e. g., for generating recommendations. This guideline should be paramount for designing future personalized MIR applications.

With well-designed applications that intelligently make use of listening context information to improve the user experience, users might be more eager to share such information. By the time of writing this chapter (September 2010), the *Last.fm listening clock* [url:25] shown in Figure 32 has been introduced (for subscribed users) based on a technique described by HERRERA, RESA, and SORDO [89]. It roughly resembles some of the visualization drafts (Figure 33) created in 2008 during the pilot study.

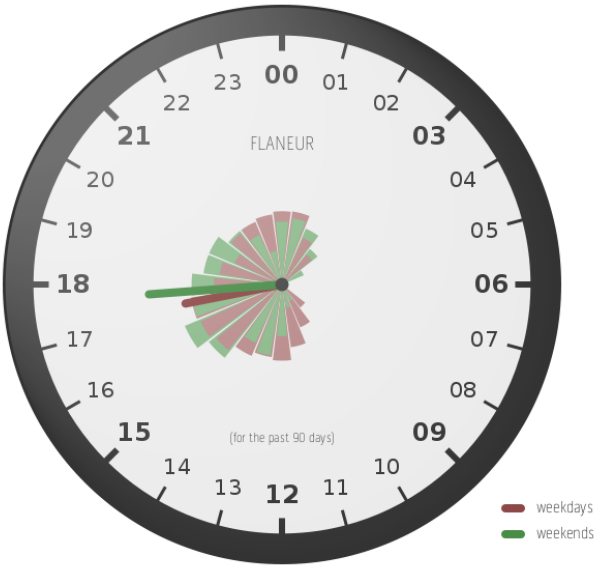


Figure 32: Illustration from [url:25] showing a *Last.fm listening clock* for the last 90 days of a user's listening history.

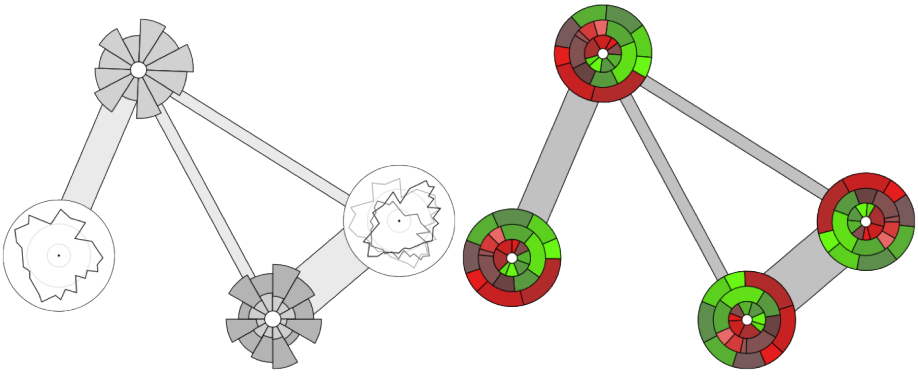


Figure 33: Two concept drawings by Valentin Laube [stud:3] of a "facet ring graph" that visualizes emerging listening contexts of a user and transitions between them.

Do not try and bend the spoon... that's impossible.
Instead only try to realize the truth...
Then you'll see, that it is not the spoon that bends,
it is only yourself.

"The Matrix"

7

FOCUS-ADAPTIVE VISUALIZATION

The work described in this chapter has been published in [pub:19].

So far, two of the general retrieval scenarios described in Section 2.2.1 have been addressed in the preceding chapters: classification and search by query. This chapter focuses on exploratory search instead. Here, users may not be able to pose a query because the search goal is not clearly defined. For instance, a user might look for background music for a photo slide show but does not know where to start. All he knows is that he can tell if it is the right music the moment he hears it. In such a case, exploratory retrieval systems can help by providing an overview of the collection and letting the user decide which regions to explore further. In the context of the general retrieval process described in Section 2.2.1, this primarily relates to the user interface and the presentation in particular which is highlighted in Figure 34.

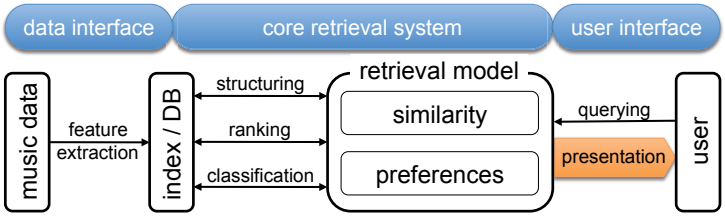


Figure 34: Visualization in the context of the general retrieval process described in Section 2.2.1.

When it comes to getting an overview of a music collection, *neighborhood-preserving* projection techniques have become increasingly popular. The general objective of such techniques can then be paraphrased as follows: Arrange the objects in two or three dimensions (on the display) in such a way that neighboring objects are very similar and the similarity decreases with increasing object distance (on the display). As the feature space of the objects to be projected usually has far more dimensions than the display space, the projection inevitably causes some loss of information – irrespective of which dimensionality reduction technique is applied. Consequently, this leads to a distorted display of the neighborhoods such that some objects will appear closer than they actually are (type I error), and on the other hand some objects that are distant in the projection may in fact be neighbors in feature space (type II error). Such neighborhood distortions are depicted in Figure 35. These "projection errors" cannot be fixed on a global scale without introducing new ones elsewhere as the projection is already optimal w.r.t. some criteria (depending on the technique used). In this sense, they should not be considered as errors made by

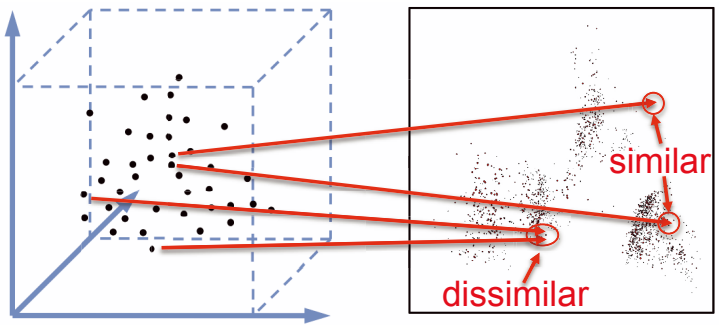


Figure 35: Possible problems caused by projecting objects represented in a high-dimensional feature space (left) onto a low-dimensional space for display (right).

the projection technique but of the resulting (displayed) arrangement. When a user explores a projected collection, type I errors increase the number of dissimilar (i. e., irrelevant) objects displayed in a region of interest. While this might become annoying, it is much less problematic than type II errors. They result in similar (i. e., relevant) objects to be displayed away from the region of interest – the neighborhood they actually belong to. In the worst case they could even be off-screen if the display is limited to the currently explored region. This way, a user could miss objects he is actually looking for.

The interactive visualization technique described in this chapter exploits these distorted neighborhood relations during user interaction. Instead of trying to globally repair errors in the projection, the general idea is to *temporarily* fix the neighborhood in focus. The approach is based on a multi-focus fish-eye lens that allows a user to enlarge and explore a region of interest while at the same time adaptively distorting the remaining collection to reveal distant regions with similar tracks. It can therefore be considered as a focus-adaptive distortion technique.

Another problem that arises when working with similarity-based neighborhoods is that music similarity is highly subjective and may depend on a person's background. Consequently, there is more than one way to look at a music collection – or more specifically to compare two tracks based on their features. The user interface presented in this chapter therefore allows the user to modify the underlying distance measure by adapting weights for different aspects of similarity. Approaches to *automatically* adapt the distance / similarity measure are covered separately by Chapter 8 in detail.

The remainder of this chapter is structured as follows: Section 7.1 gives an overview of related approaches that aim to visualize a music collection. Subsequently, Section 7.2 outlines the approach developed in this work. The underlying techniques are addressed in Section 7.3 and Section 7.4 explains how a user can interact with the proposed visualization. In order to evaluate the approach, a user study has been conducted which is described in Section 7.5. Finally, Section 7.6 concludes this chapter with a brief summary.

7.1 RELATED WORK

There exists a variety of approaches that in some way give an overview of a music collection. For the task of music discovery which is closely related to collection exploration, a very broad survey of approaches has been given by DONALDSON and LAMERE [57]. Generally, there are several possible levels of granularity that can be supported, the most common being: track, album, artist and genre. Though a system may cover more than one granularity level (e. g., in [236] visualized as disc or TreeMap [222]), usually a single one is chosen. The user interface presented in this chapter focuses on the track level as do most of the related approaches. (However, like most of the other techniques, it may as well be applied on other levels such as for albums or artist. All that is required is an appropriate feature representation of the objects of interest.) Those approaches focusing on a single level can roughly be categorized into graph-based and similarity-based overviews.

Graphs facilitate a natural navigation along relationship edges. They are especially well-suited for the artist level as social relations can be directly visualized (e. g., the *Last.fm Artist Map* [url:24] or the *Relational Artist Map RAMA* [212]). However, building a graph requires relations between the objects – either from domain knowledge or artificially introduced. For instance, there are some graphs that use similarity relations obtained from external sources (such as the Application Programming Interfaces (APIs) of *Last.fm* [url:23] or *EchoNest* [url:48]) and not from an analysis of the objects themselves. Either way, this results in a very strong dependency and may quickly become problematic for less mainstream music where such information might not be available. This is why a similarity-based approach is chosen here instead.

Similarity-based approaches require the objects to be represented by one or more features. They are in general better suited for track level overviews due to the vast variety of content-based features that can be extracted from tracks. For albums and artists, either some means for aggregating the features of the individual tracks are needed or non-content-based features, e. g., extracted from knowledge resources like *MusicBrainz* [url:36] and *Wikipedia* [url:53] or cultural meta-data [244], have to be used. In most cases, the overview is then generated using some metric defined on these features which leads to proximity of similar objects in the feature space. This neighborhood should be preserved in the collection overview which usually has only two dimensions. Popular approaches for dimensionality reduction are SOMs (Section 4.3), Principal Component Analysis (PCA) [103] and MDS techniques (Section 4.4).

In the field of MIR, SOMs are widely used as already pointed out in Section 3.3.4. SOMs are prototype-based and thus there has to be a way to initially generate random prototypes and to modify them gradually when objects are assigned. This poses special requirements regarding

the underlying feature space and distance metric – a problem which is addressed in Appendix A. Moreover, the result depends on the random initialization and the neural network gradient descent algorithm may get stuck in a local minimum and thus not produce an optimal result. Further, there are several parameters that need to be tweaked according to the data set such as the learning rate, the termination criterion for iteration, the initial network structure, and (if applicable) the rules by which the structure should grow. However, there are also some advantages of SOMs: Growing versions of SOMs can adapt incrementally to changes in the data collection (Section 4.3.1) whereas other approaches may always need to generate a new overview from scratch. Section 7.3.1.2 will address this point more specifically for the approach taken here. For the interactive task at hand, which requires a real-time response, the disadvantages of SOMs outweigh their advantages. Therefore, the approach taken here is based on MDS.

Given a set of data points, MDS finds an embedding in the target space that maintains their distances (or dissimilarities) as far as possible – without having to know their actual values. This way, it is also well suited to compute a layout for spring- or force-based approaches. PCA identifies the axes of highest variance termed principal components for a set of data points in high-dimensional space. To obtain a dimensionality reduction to two-dimensional space, the data points are simply projected onto the two principal component axes with the highest variance. PCA and MDS are closely related [247]. In contrast to SOMs, both are non-parametric approaches that compute an optimal solution (with respect to data variance maximization and distance preservation respectively) in fixed polynomial time. Systems that apply PCA, MDS or similar force-based approaches comprise those described in [32, 74, 84] as well as the *fm4 Soundpark* [url:12, 72], *MusicBox* [128], and *SoundBite* [131].

All of the above approaches use some kind of projection technique to visualize the collection but only a small number tries to additionally visualize properties of the projection itself. Here, mountain ranges have become a popular metaphor as shown in Figure 36. The *MusicMiner*

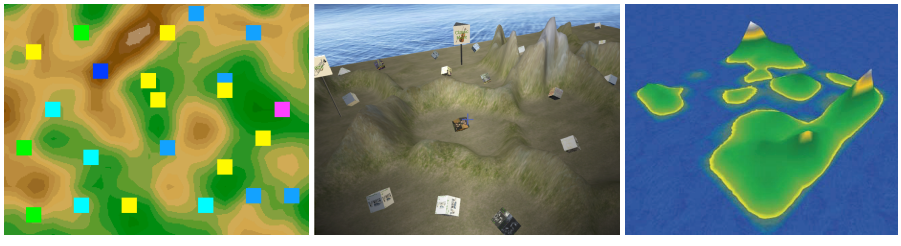


Figure 36: Screenshots of related approaches that use mountain ranges to separate dissimilar regions (left: *MusicMiner* [163], middle: *SoniXplorer* [135]) or to visualize regions with a high density of similar songs (right: *nepTune* [108], a variant of *Islands of Music*). Illustrations from the respective publications.

[163] draws mountain ranges between songs that are displayed close to each other but dissimilar. The *SoniXplorer* [134, 135] uses the same geographical metaphor but in a 3D virtual environment that the user can navigate with a game pad. The *Islands of Music* [180, 185] and its related approaches [72, 108, 165] use the third dimension the other way around: Here, islands or mountains refer to regions of similar songs (with high density). Both ways, local properties of the projection are visualized – neighborhoods of either dissimilar or similar songs. In contrast (and possibly as a supplementation) to this, the technique proposed in this chapter aims to visualize properties of the projection that are not locally confined: As visualized in [172], there may be distant regions in a projection that contain very similar objects (Figure 37; right).

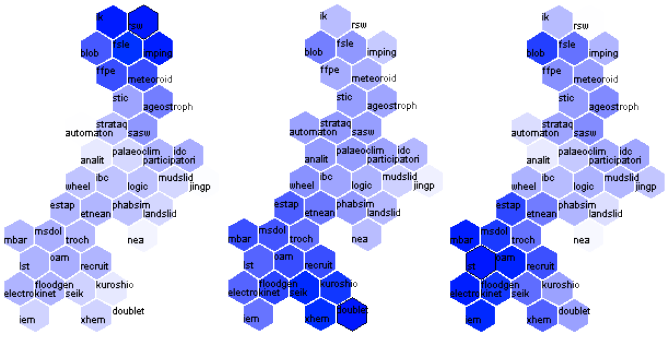


Figure 37: Illustrations from [172] of hexagonal GSOMs colored according to similarity w.r.t. a sample document (dark color indicates high similarity).

This is much like a "wormhole" connecting both regions through the high-dimensional feature space. To the author's knowledge, the only other attempt so far to visualize such distortions caused by the projection is described by Lloyd [131]. His approach is to draw lines that connect a selected seed track (highlighted with a circle) with its neighbors as shown in Figure 38.

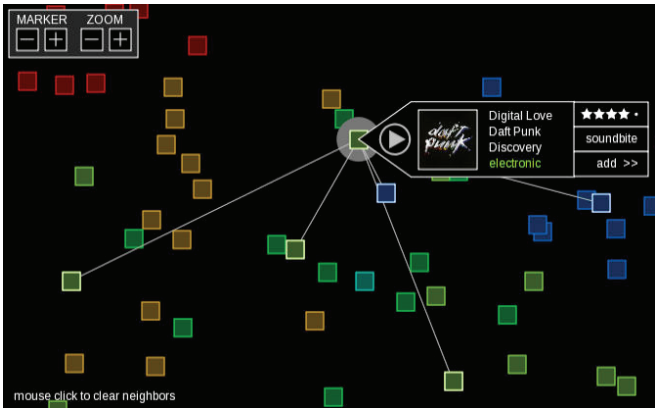


Figure 38: Illustration from [131] of *SoundBite* that connects a seed song and its nearest neighbors by lines.

7.2 OUTLINE

The goal is to provide a user with an interactive way of exploring a music collection that takes into account the above described inevitable limitations of a low-dimensional projection of a collection. Further, it should be applicable for realistic music collections containing several thousands of tracks. The approach taken can be outlined as follows:

- An overview of the collection is given, where all tracks are displayed as points at any time. For a limited number of tracks that are chosen to be spatially well distributed and representative, an album cover thumbnail is shown for orientation. The view on the collection is generated by a neighborhood-preserving projection (e. g., MDS, SOM, PCA) from some high-dimensional feature space onto two dimensions, i. e., tracks that are close in feature space will likely appear as neighbors in the projection.
- Users can adapt the projection by choosing weights for several aspects of music (dis-) similarity. This gives them the possibility to look at a collection from different perspectives.[1] In order to allow immediate visual feedback in case of similarity adaptation, the projection technique needs to guarantee near real-time performance – even for large collections. The quality of the produced projection is only secondary – in any case, perfect projections that correctly preserve all distances are extremely unlikely.
- The projection will inevitably contain distortions of the actual distances of the tracks. Instead of trying to improve the quality of the projection method and trying to fix heavily distorted distances, they are exploited during interaction with the projection: The user can zoom into a region of interest. The space for this region is increased, thus allowing to display more details. At the same time, the surrounding space is compacted but not hidden from view. This way, there remains some context for orientation. To accomplish such a behavior, the zoom is based on a non-linear distortion similar to so-called "fish-eye" lenses. At this point the original (type II) projection errors come into play: Instead of putting a single lens focus on the region of interest, additional focuses are introduced in regions that contain tracks similar to those in primary focus. The resulting distortion brings original neighbors back closer to each other. This gives the user another option for interactive exploration.

Figure 39 depicts the outline of the approach. The following sections

1 This adaptation is purely manual, i. e., the visualization as described in this chapter is only *adaptable* w.r.t. music similarity. Techniques to further enable adaptive music similarity are discussed in Chapter 8.

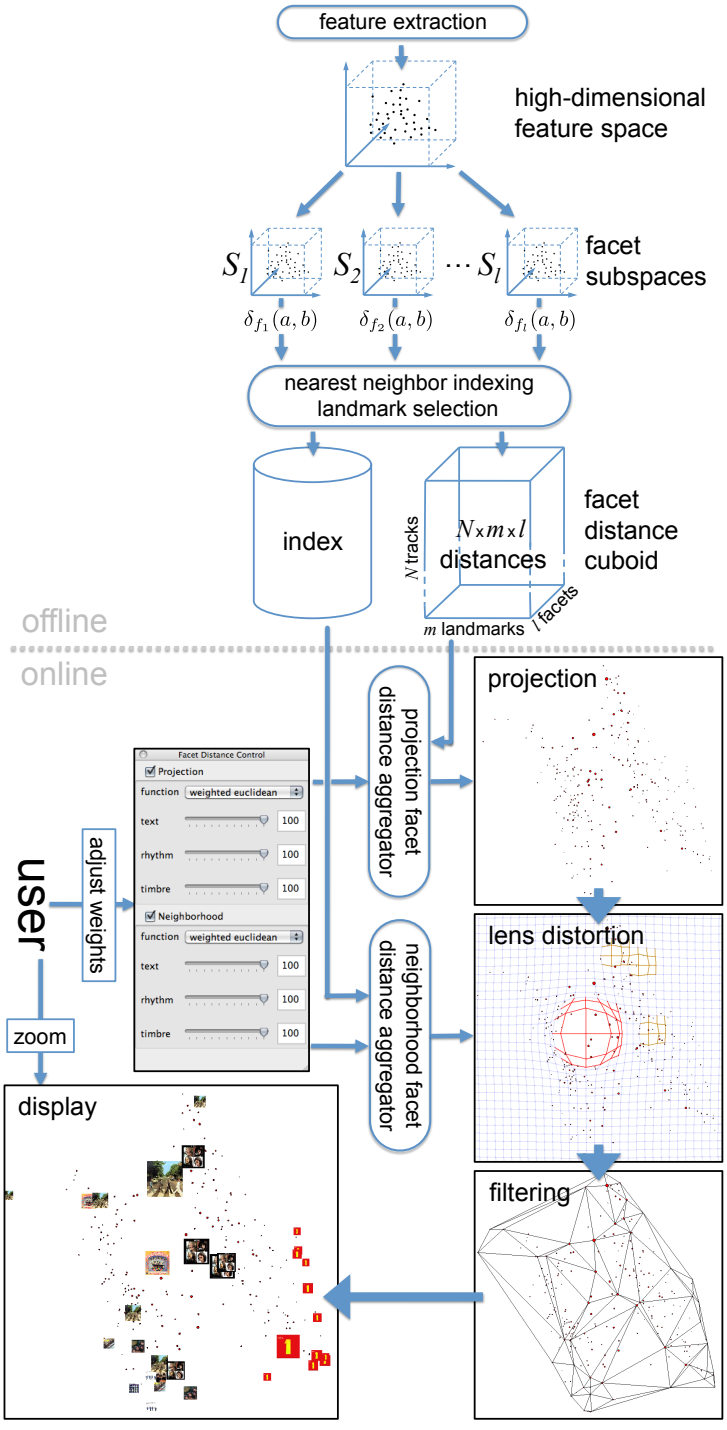


Figure 39: Outline of the approach showing the important processing steps and data structures. Top: preprocessing. Bottom: interaction with the user.

cover the underlying techniques (Section 7.3) and the user-interaction (Section 7.4) in detail.

7.3 UNDERLYING TECHNIQUES

The underlying techniques have been published in [pub:12].

As a prerequisite, it is assumed that the tracks are represented by some descriptive features that can, e. g., be extracted, manually annotated or obtained from external sources. Based on the features associated with the tracks, *facets* are defined (on subspaces of the feature space) that refer to different aspects of music (dis-) similarity. This is depicted in Figure 39 (top). For each facet, a distance value of two tracks can be computed. The actual distance between tracks is determined through aggregation of the facet distance values. The concept of facets plays an important role for modeling adaptive music similarity which is the subject of Chapter 8. It is therefore explained in detail in Section 8.1.

7.3.1 *Projection*

In the projection step shown in Figure 39 (bottom), the position of all tracks on the display is computed according to their (aggregated) distances in the high-dimensional feature space. Naturally, this projection should be neighborhood-preserving such that tracks close to each other in feature space are also close in the projection. To this end, the LMDS algorithm described in Section 4.4.1 has been implemented. The representative sample of tracks – called "landmarks" – is drawn randomly from the whole collection.[2] For this landmark sample, an embedding into two-dimensional space is computed using classical MDS. The remaining objects can then be located within this space according to their distances to the landmarks. For constant landmark sample size (and display space dimensionality), LMDS scales linearly with the number of tracks in the collection – both, in terms of computational complexity and memory requirements for the distance matrix.

7.3.1.1 *Facet Distance Caching*

The computation of the distance matrix that is required for LMDS can be very time consuming – not only depending on the size of the collection and landmark sample but also on the number of facets and the complexity of the respective facet distance measures. Caching can reduce the amount of information that has to be recomputed.

2 Alternatively, the MaxMin heuristic [226] could be used – with the optional modification to replace landmarks with a predefined probability by randomly chosen objects (similar to a mutation operator in genetic programming). Neither alternative seems to produce less distorted projections while having much higher computational complexity. However, there is possibly some room for improvement here but this is out of the scope of this thesis.

Assuming a fixed collection, the distance matrix only needs to be recomputed if the facet weights or the facet aggregation function change. Moreover, even a change of the aggregation parameters has no impact on the facet distances. This allows to pre-compute for each track the distance values to all landmarks for all facets offline and store them in the 3-dimensional data structure depicted in Figure 39 (top) called "facet distance cuboid". It is necessary to store the facet distance values separately as it is not clear at indexing time how these values are to be aggregated. During interaction with the user, where near real-time response is required, only the computational lightweight facet distance aggregation that produces the distance matrix from the cuboid and the actual projection need to be done.

If N is the number of songs, m the number of landmarks and l the number of facets, the cuboid has the dimension $N \times m \times l$ and holds as many distance values. Note that m and l are fixed small values of $O(100)$ and $O(10)$ respectively. Thus, the space requirement effectively scales linearly with N and even for large N the data structure should fit into memory. To further reduce the memory requirements of this data structure, the distance values are discretized to the byte range ($[0 \dots 255]$) after normalization as described in Section 8.1.

7.3.1.2 *Incremental Collection Updates*

Re-computation becomes also necessary once the collection changes. Some neighborhood-preserving projection techniques such as GSOMs (Section 4.3.1) are inherently incremental as, e. g., described by NÜRNBERGER and KLOSE [173]: Adding or removing objects from the collection only gradually changes the way the data is projected. This is a nice characteristic because too abrupt changes in the projection caused by adding or removing some tracks might irritate the user if he has gotten used to a specific projection. Unfortunately, LMDS does not allow for incremental changes of the projection. However, it still allows objects to be added or removed from the data set to some extent without the need to compute a new projection: If a new track is added to the collection, an additional "layer" has to be appended to the facet distance cuboid (Section 7.3.1.1) containing the facet distances of the new track with all landmarks. The new track can then be projected according to these distances. If a track is removed, the respective "layer" of the cuboid can be deleted. Neither operation does further alter the projection.[3] Adding or removing many objects may however alter the distribution of the data (and thus the covariances) in such a way that the landmark sample may no longer be representative. In this case, a new projection based on a modified landmark sample should

3 In case a landmark track is removed from the collection, its feature representation has to be kept to be able to compute facet distances for new tracks. However, the corresponding "layer" in the cuboid can be removed as for any ordinary track.

be computed. However, for the scope of this thesis, a stable landmark set is assumed and this point is left for further work.

7.3.2 *Lens Distortion*

Once the position of all tracks in two-dimensional space is computed by the projection technique, the collection could already be displayed. However, an intermediate distortion step is introduced as depicted in Figure 39 (bottom) on page 103. It serves as the basis for the interaction techniques described later.

7.3.2.1 *Lens Modeling*

The distortion technique is based on an approach originally developed to model complex nonlinear distortions of images called "*SpringLens*" [73]. A *SpringLens* consists of a mesh of mass particles and interconnecting springs that form a rectangular grid with fixed resolution. Through the springs, forces are exerted between neighboring particles affecting their motion. By changing the rest-length of selected springs, the mesh can be distorted as depicted in Figure 40. (Furthermore, Figure 39 (bottom) on page 103 and Figure 43 on page 113 show larger meshes simulating lenses.) The deformation is calculated by an iterative physical simulation over time using a simple Euler integration [73].

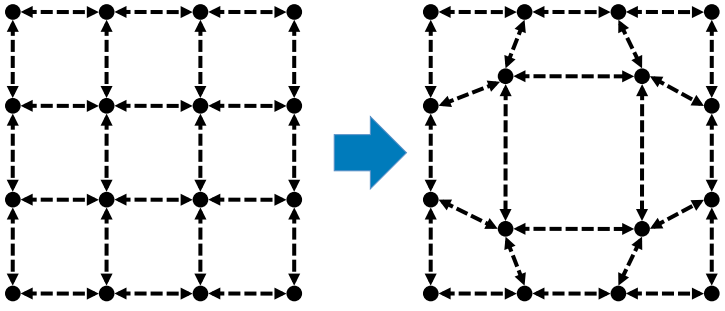


Figure 40: The *SpringLens* particle mesh is distorted by changing the rest-length of selected springs.

In the context of this work, the *SpringLens* technique is applied to simulate a complex superimposition of multiple fish-eye lenses. A moderate resolution is chosen with a maximum of 50 cells in each dimension for the overlay mesh which yields sufficient distortion accuracy while real-time capability is maintained. The distorted position of the projection points is obtained by barycentric coordinate transformation [221, Chapter 2.11] with respect to the particle points of the mesh. Additionally, z-values are derived from the rest-lengths that are used in the visualization to decide whether an object has to be drawn below or above another one.

7.3.2.2 *Nearest Neighbor Indexing*

For the adaptation of the lens distortion, the nearest neighbors of a track need to be retrieved. Here, the two major challenges are:

1. The facet weights are not known at indexing time and thus the index can only be built using the facet distances.
2. The choice of an appropriate indexing method for each facet depends on the respective distance measure and the nature of the underlying features.

As the focus lies here on the visualization and not the indexing, only a very basic approach is taken and further developments are left for future work: A limited list of nearest neighbors is pre-computed for each track. This way, nearest neighbors can be retrieved by simple lookup in constant time ($O(1)$). However, updating the lists after a change of the facet weights is computationally expensive. While the resulting delay of the display update is still acceptable for collections with a few thousands tracks, it becomes infeasible for larger N.

For more efficient index structures, it may be possible to apply generic multimedia indexing techniques such as space partition trees [48] or approximate approaches based on locality sensitive hashing [98] that may even be kernelized [113] to allow for more complex distance metrics. Another option is to generate multiple nearest neighbor indexes – each for a different setting of the facet weights – and interpolate the retrieved result lists w.r.t. to the actual facet weights.

7.3.3 *Visualization Metaphor*

The music collection is visualized as a galaxy. Each track is displayed as a star or as its album cover. The brightness and (to some extent) the hue of stars depend on a predefined importance measure. The currently used measure of importance is the track play count obtained from the *Last.fm* API and normalized to $[0, 1]$ (by dividing by the maximum value). However, this could also be substituted by a more sophisticated measure, e. g., based on (user) ratings, chart positions or general popularity. The size and the z-order (i. e., the order of objects along the z-axis) of the objects depend on their distortion z-values. Optionally, the *SpringLens* mesh overlay can be displayed. The visualization then resembles the space-time distortions well known from gravitational and relativistic physics.

7.3.4 *Filtering*

In order to reduce the amount of information displayed at a time, an additional filtering step is introduced as depicted in Figure 39 (bottom)

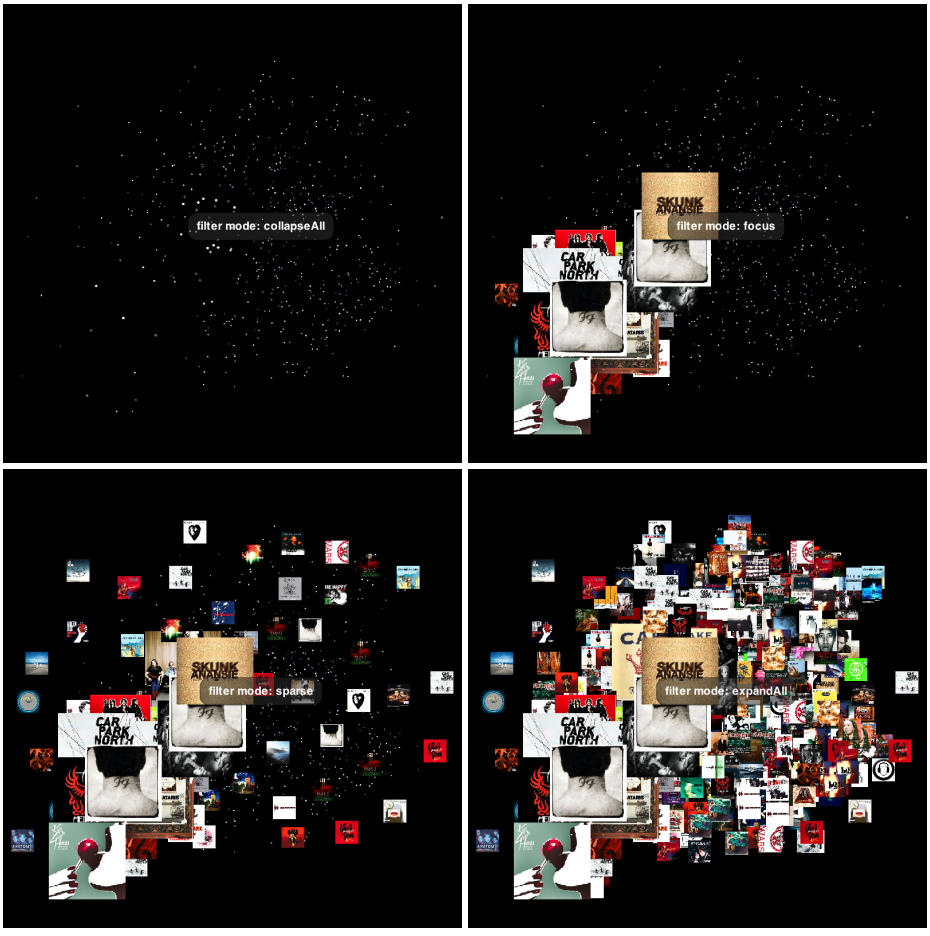


Figure 41: Available filter modes: *collapseAll* (top left), *focus* (top right), *sparse* (bottom left), *expandAll* (bottom right). The *SpringLens* mesh overlay is hidden.

on page 103. The user can choose between different filters that decide whether a track is displayed collapsed or expanded – i. e., as a star or album cover respectively. While album covers help for orientation, the displayed stars give information about the data distribution. Trivial filters are those displaying no album covers (*collapseAll*) or all (*expandAll*). Apart from collapsing or expanding all tracks, it is possible to expand only those tracks in magnified regions (i. e., with a z-level above a predefined threshold) or to apply a *sparser filter*. The results of using these filter modes are shown in Figure 41.

A sparser filter selects only a subset of the collection to be expanded that is both, sparse (well distributed) and representative. Representative tracks are those with a high importance (described in Section 7.3.3).

The first sparser version used a Delaunay triangulation (Section 7.3.4.1) and was later substituted by a raster-based approach (Section 7.3.4.2) that produces more appealing results in terms of the spatial distribution of displayed covers.

Originally, the set of expanded tracks was updated after any position changes caused by the distortion overlay. However, this was considered irritating during early user tests and the sparser strategy was changed to update only if the projection or the displayed region changes.

7.3.4.1 *Delaunay Sparser Filter*

The Delaunay sparser filter has been implemented by Sebastian Loose as a student research assistant and later refined during his diploma thesis [stud:4]

This sparser filter constructs a Delaunay triangulation [48, Chapter 9] incrementally, starting with the track with the highest importance and some virtual points at the corners of the display area. Next, the size of all resulting triangles given by the radius of their circumcircle is compared with a predefined threshold $size_{min}$. If the size of a triangle exceeds this threshold, the most important track within this triangle is chosen for display and added as a point for the triangulation. This process continues recursively until no triangle that exceeds $size_{min}$ contains any more tracks that could be added. All tracks belonging to the triangulation are then expanded (i. e., displayed as album thumbnail).

The Delaunay triangulation can be computed in $O(n \log n)$ and the number of triangles is at most $O(n)$ with $n \ll N$ being the number of actually displayed album cover thumbnails. To reduce lookup time, projected points are stored in a quadtree data structure [48, Chapter 14] and sorted by importance within the tree's quadrants. A triangle's size may change through distortion caused by the multi-focal zoom. This change may trigger an expansion of the triangle or a removal of the point that caused its creation originally. Both operations are propagated recursively until all triangles meet the size condition again. Figure 39 (bottom) on page 103 shows a triangulation and the resulting display for a (distorted) projection of a collection.

7.3.4.2 *Raster Sparser Filter*

The raster sparser filter divides the display into a uniform grid of quadratic cells. The size of the cells depends on the screen resolution and the minimal display size of the album covers. Further, it maintains a list of the tracks ranked by importance that is precomputed and only needs to be updated when the importance values change. If an update is triggered by some user action, the sparser runs through its ranked list. For each track it determines the respective grid cell. If the cell and the surrounding cells are empty, the track is expanded and its cell blocked. (Checking surrounding cells avoids image overlap. The necessary radius for the surrounding can be derived from

the cell and cover sizes.) The computational complexity of a filter update is $O(Ns)$ where $s \ll N$ is the number of surrounding cells that need to be checked for collision. In contrast to the filter based on the Delaunay triangulation, no incremental updates are supported. Thus, this approach requires more computation for each update. In practice, however, it is not noticeably slower. In particular, as the most important objects are checked first, their state can be updated very early, whereas less important objects which most likely are not expanded might be skipped in case of more recent updates.

7.4 INTERACTION

While the previous section covered the underlying techniques, this section describes how users can interact with the user interface that is built on top of them. Figure 42 shows a screenshot of the "*MusicGalaxy*" prototype. It allows several ways of interacting with the visualization:


Video Clip 1

A description of the interaction techniques has been published in [pub:14].

Users can explore the collection through common panning & zooming (Section 7.4.1). Alternatively, they can use the adaptive multi-focus technique introduced with this prototype (Section 7.4.2). Further, they can change the facet aggregation function parameters and this way adapt the view on the collection according to their preferences (Section 7.4.3). Hovering over a track displays its title and a double-click starts the playback that can be controlled by the player widget at the bottom of the interface. Apart from this, several display parameters can be changed such as the filtering mode (Section 7.3.4), the size of the displayed album covers or the visibility of the *SpringLens* overlay mesh.

7.4.1 *Panning & Zooming*

These are very common interaction techniques that can, e. g., be found in programs for geo-data visualization or image editing that make use of the map metaphor. Panning shifts the displayed region whereas zooming decreases or increases it. (This does not affect the size of the thumbnails which can be controlled separately using the PageUp and PageDn keys.) Using the keyboard, the user can pan with the cursor keys and zoom in and out with + and − respectively. Alternatively, the mouse can be used: Clicking and holding the left button while moving the mouse pans the display. The mouse wheel controls the zoom level. If not the whole collection can be displayed, an overview window indicating the current section is shown in the top left corner, otherwise it is hidden. Clicking into the overview window centers the display around the respective point. Further, the user can drag the section indicator around which also results in panning.

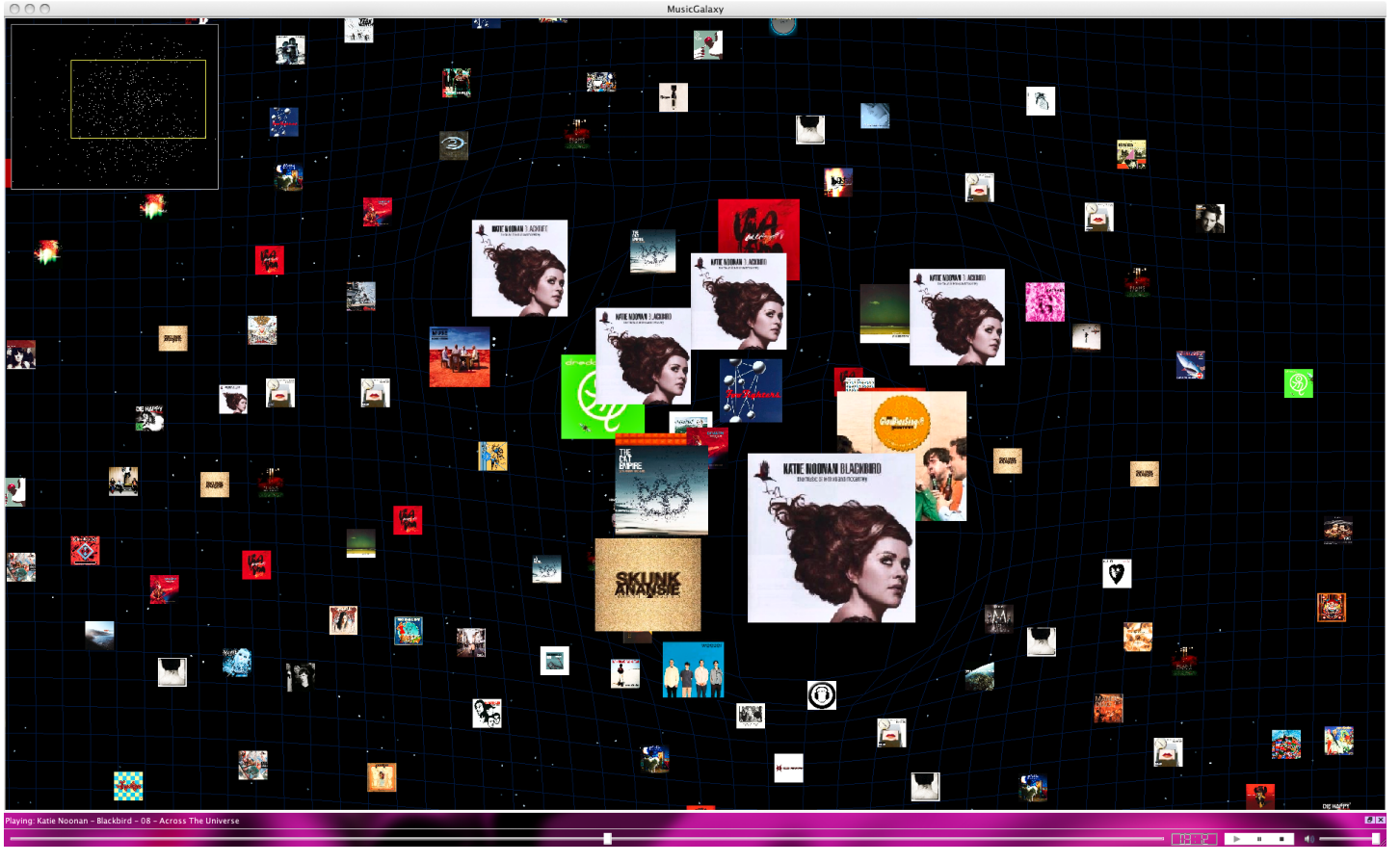


Figure 42: Screenshot of the *MusicGalaxy* prototype with visible overview window (top left), player (bottom) and *SpringLens* mesh overlay (blue). In this example, a strong album effect can be observed as for the track in primary focus, four tracks of the same album are nearest neighbors in secondary focus.

7.4.2 *Focusing*

This interaction technique allows to visualize – and to some extent alleviate – the neighborhood distortions introduced by the dimensionality reduction during the projection. The approach is based on a multi-focus fish-eye lens that is implemented using the *SpringLens* distortion technique (Section 7.3.2). It consists of a user-controlled *primary focus* and a neighborhood-driven *secondary focus*.

The primary focus is a common fish-eye lens. By moving this lens around (holding the right mouse button), the user can zoom into regions of interest. In contrast to the basic linear zooming function described in Section 7.4.1, this leads to a nonlinear distortion of the projection. As a result, the region of interest is enlarged making more space to display details. At the same time, less interesting regions are compacted. This way, the user can inspect closely the region of interest without loosing the overview as the field of view is not narrowed (as opposed to the linear zoom). The magnification factor of the lens can be changed using the mouse wheel while holding the right mouse button. The visual effect produced by the primary zoom resembles a two-dimensional version of the popular "cover flow" effect.

The secondary focus consists of multiple such fish-eye lenses. These lenses are smaller and cannot be controlled by the user but are automatically adapted depending on the primary focus. When the primary focus changes, the neighbor index (Section 7.3.2.2) is queried with the track closest to the center of focus. If nearest neighbors are returned that are not in the primary focus, secondary lenses are added at the respective positions. As a result, the overall distortion of the projection brings the distant nearest neighbors back closer to the focused region of interest. Figure 43 shows the primary and secondary focus with visible *SpringLens* mesh overlay.

As it can become very tiring to hold the right mouse button while moving the focus around, the latest prototype introduces a focus lock mode (toggled with the return key). In this mode, the user clicks once to start a focus change and a second time to freeze the focus. To indicate that the focus is currently being changed (i. e., mouse movement will affect the focus), an icon showing a magnifying glass is displayed in the lower left corner. The secondary focus is by default always updated instantly when the primary focus changes. This behavior can be disabled resulting only in an update of the secondary focus once the primary focus does not change anymore.

7.4.3 *Adapting the Aggregation Functions*

Two facet control panels allow to adapt two facet distance aggregation functions by choosing one of several function types from a drop-down menu and adjusting weights for the individual facets (through sliders).

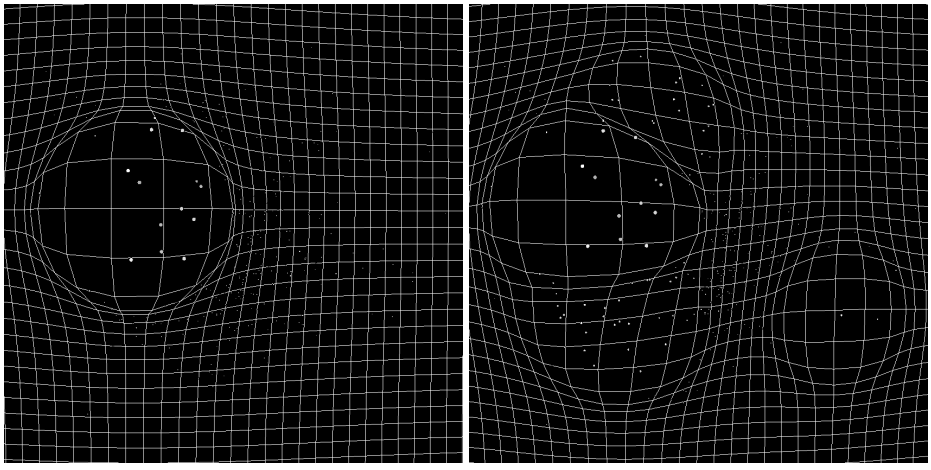


Figure 43: *SpringLens* distortion with only primary focus (left) and additional secondary focus (right).

The control panels are hidden in the screenshot Figure 42) but shown in Figure 39 (bottom) on page 103 that depicts the user interaction. The first facet distance aggregation function is applied to derive the track-landmark distances from the facet distance cuboid (Section 7.3.1.1). These distances are then used to compute the projection of the collection. The second facet distance aggregation function is applied to identify the nearest neighbors of a track and thus indirectly controls the secondary focus.

Changing the aggregation parameters results in a near real-time update of the display so that the impact of the change becomes immediately visible: In case of the parameters for the nearest neighbor search, some secondary focus region may disappear while somewhere else a new one appears with tracks now considered more similar. Here, the transitions are visualized smoothly due to the underlying physical simulation of the *SpringLens* grid. In contrast to this, a change of the projection similarity parameters has a more drastic impact on the visualization possibly resulting in a complete re-arrangement of all tracks. This is because the LMDS projection technique produces solutions that are unique only up to translation, rotation, and reflection and thus, even a small parameter change may, e. g., flip the visualization. As this may confuse users, one direction of future research is to investigate how the position of the landmarks can be constrained during the projection to produce more gradual changes. This might be done, e. g., using Procrustes analysis [81] – a form of statistical shape analysis.

The two facet distance aggregation functions are linked by default as it is most natural to use the same distance measure for projection and neighbor retrieval. However, unlinking them and using, e. g., orthogonal distance measures can lead to interesting effects: For

instance, one may choose to compute the collection based solely on acoustic facets and find nearest neighbors for the secondary focus through lyrics similarity. Such a setting would help to uncover tracks with a similar topic that (most likely) sound very different. This idea is further elaborated in Chapter 9.

7.5 EVALUATION

The development of the focus-adaptive *SpringLens* technique and the *MusicGalaxy* application followed a user-driven design approach [166] by iteratively alternating between development and evaluation phases. The first prototype was presented at the *CeBIT 2010* [url:6], a German trade fair specialized on information technology, in early March 2010. During the fair, feedback was collected from a total of 112 visitors aged between 16 and 63 years. The general reception was very positive. The projection-based visualization was generally welcomed as an alternative to common list views. However, some remarked that additional semantics of the two display axes would greatly improve orientation. Young visitors particularly liked the interactivity of the visualization whereas older ones tended to have problems with this. They stated that the reason lay in the amount of information displayed which could still be overwhelming. To address the problem, they proposed to expand only tracks in focus, increase the size of objects in focus (compared to the others) and hide the mesh overlay as the focus would be already visualized by the expanded and enlarged objects. All of these proposals have been integrated into the second prototype.

The early prototype has been described in [pub:12].

The second prototype was tested thoroughly by three testers. During these tests, the eye movements of the users were recorded with a *Tobii T60* eye tracker that can capture where and how long the gaze of the participants rests for some time (referred to as "fixation points"). Using the adaptive *SpringLens* focus, the mouse generally followed the gaze that scans the border of the focus in order to decide on the direction to explore further. This resulted in a much smoother gaze-trajectory than the one observed during usage of panning and zooming where the gaze frequently switched between the overview window and the objects of interest – as not to lose orientation. This indicates that the proposed approach is less tiring for the eyes. However, the testers criticized the controls used to change the focus – especially having to hold the right mouse button all the time. This lead to the introduction of the focus lock mode and several minor interface improvements (not explicitly covered here).

An outline of the prototype presented at the CeBIT 2010 and a summary of the user feedback have been given in [pub:13].

The final prototype was presented as a late breaking demo at ISMIR'10 [pub:16].

The remainder of this section describes a user study, which aimed to proof the usefulness of the focus-adaptive *SpringLens* technique for exploratory search based on a projection of a document collection. In particular, the following four questions were addressed:

1. How does the lens-based user interface compare in terms of usability to common panning & zooming techniques that are very popular in interfaces using a map metaphor (such as *Google Maps* [url:17])?
2. How much do users actually use the secondary focus or would a common fish-eye distortion (i. e., only the primary focus) be sufficient?
3. What interaction patterns do emerge?
4. What can be improved to further support the user and increase user satisfaction?

To this end, screencasts of 30 participants solving an exploratory retrieval task were recorded together with eye tracking data (again using a *Tobii T60* eye tracker) and web cam video streams. This data was used to identify emerging interaction strategies among all users and to analyze to what extent the primary and secondary focus was used. Moreover, first-hand impressions of the usability of the interface were gathered by letting the participants say aloud whatever they think, feel or remark as they go about their task (think-aloud protocol).

The user study has been conducted in cooperation with Christian Hentschel. The results have been published in [pub:18].

Instead of letting the participants explore a music collection with the *MusicGalaxy* user interface prototype described earlier, a different scenario was constructed for this experiment using a modified version that can handle photo collections. It relies on widely used *MPEG-7* visual descriptors (*EdgeHistogram*, *ScalableColor* and *ColorLayout*) [136, 146] to compute the visual similarity – replacing the originally used music features and respective similarity facets. This allowed to reduce the impact of other factors that were deliberately not addressed by the study. In particular, it was not the goal to evaluate projection-based visualizations. Neither should the problem be addressed of how to effectively visualize individual music tracks, which still remains to be solved satisfactorily and is out of the scope of this thesis. Visualizing photos with thumbnails is straightforward and does not raise such interfering issues. Furthermore, using photos is also advantageous from a pragmatic perspective as similarity and relevance of photos can be assessed in an instant. This is much harder for music tracks and requires additional time for listening – especially if the tracks are previously unknown. Consequently, using photos, the participants would actually spend more time interacting with the collection than with individual objects, which also served the purpose of the study. As an additional advantage, it could easily be assured that none of the participants knew any of the photos in advance what could otherwise have introduced some bias. Apart from these considerations, the questions to be answered in the study addressed the focus-adaptive *SpringLens* as a general technique irrespective of the content to be explored.

The user interface prototype for exploration of photo collections has been developed in cooperation with Christian Hentschel. An early version is described in [pub:11].

To answer the first question, participants compared a purely *SpringLens*-based user interface with a common panning & zooming and additionally a combination of both. For questions 2 and 3, the recorded interaction of the participants with the system was analyzed in detail. Answers to question 4 were collected by asking the users directly for missing functionality. Section 7.5.1 addresses the experimental setup in detail and Section 7.5.2 discusses the results.

7.5.1 *Experimental Setup*

At the beginning of the experiment, the participants were asked several questions to gather general information about their background. Afterwards, they were presented the four image collections described in Section 7.5.1.1 in fixed order. On the first collection, a survey supervisor gave a guided introduction to the interface and the possible user actions. Each participant could spend as much time as needed to get used to the interface. Once the participant was familiar with the controls, she or he continued with the other collections for which a retrieval task (Section 7.5.1.2) had to be solved without the help of the supervisor. At this point, the participants were divided into two groups. The first group used only panning & zooming (P&Z) as described in Section 7.4.1 on the second collection and only the *SpringLens* functionality (SL) described in Section 7.4.2 on the third one. The other group started with SL and then used P&Z. The order of the datasets stayed the same for both groups. This way, effects caused by the order of the approaches and slightly varying difficulties among the collections were avoided. The fourth collection could then be explored by using both, P&Z and SL. The functionality for adapting the facet distance aggregation functions described in Section 7.4.3 was deactivated for the whole experiment. After the completion of the last task, the participants were asked to assess the usability of the different approaches. Furthermore, feedback was collected pointing out, e. g., missing functionality. Specifically, the questionnaire contained the following questions[4]:

1. How helpful was the interface to accomplish your task?
2. Did you miss any functionality that could have eased your task? If yes, please tell us which!
3. How easy or complicated was the use of the interface?
4. How intuitive was the interface? – How quick did you get used to it?

For the questions 1, 3, and 4, the answers were coded on 7-point scale where 1 was worst, 4 neutral and 7 best.

4 The questionnaire was in German. These are English translations of the questions.

7.5.1.1 *Test Collections*

Four image collections were used during the study. They were drawn from a personal photo collection of the author.[5] Each collection comprises 350 images – except the first collection (used for the introduction of the user interface) which only contains 250 images. All images were scaled down to fit 600x600 pixels. For each of the collections 2 to 4, five non-overlapping topics were chosen and the images annotated accordingly. These annotations served as ground truth and were not shown to the participants. Table 11 shows the topics for each collection. In total, 264 of the 1050 images belong to one of the 15 topics.

Collection	Topics (number of images)
Melbourne & Victoria	–
Barcelona	Tibidabo (12), Sagrada Família (31), Stone Hallway in Park Güell (13), Beach & Sea (29), Casa Milà (16)
Japan	Owls (10), Torii (8), Paintings (8), Osaka Aquarium (19), Traditional Clothing (35)
Western Australia	Lizards (17), Aboriginal Art (9), Plants (Macro) (17), Birds (21), Ningaloo Reef (19)

Table 11: Photo collections and topics used during the user study.

7.5.1.2 *Retrieval Task*

For the collections 2 to 4, the participants had to find five (or more) representative images for each of the topics listed in Table 11. As guidance, handouts were prepared that showed the topics – each one printed in a different color –, an optional brief description and two or three sample images giving an impression what to look for. Images representing a topic had to be marked with the topic's color. This was done by double clicking on the thumbnail what opened a floating dialog window presenting the image at big scale and allowing the participant to classify the image to a predefined topic by clicking a corresponding button. As a result, the image was marked with the color representing the topic. Further, the complete collection could be filtered by highlighting all thumbnails classified to one topic. This was done by pressing the numeric key (1 to 5) for the respective topic number. Highlighting was done by focusing a fish-eye lens on every marked topic member and thus enlarging the corresponding thumbnails as shown in Figure 44.


Video Clip 2

It was pointed out that the decision whether an image was representative for a group was solely up to the participant and not judged otherwise. There was no time limit for the task. However, the participants

5 The collections and topic annotations are publicly available under the Creative Commons Attribution-Noncommercial-Share Alike license, http://creativecommons.org/licenses/by-nc-sa/3.0/ Please contact the author.

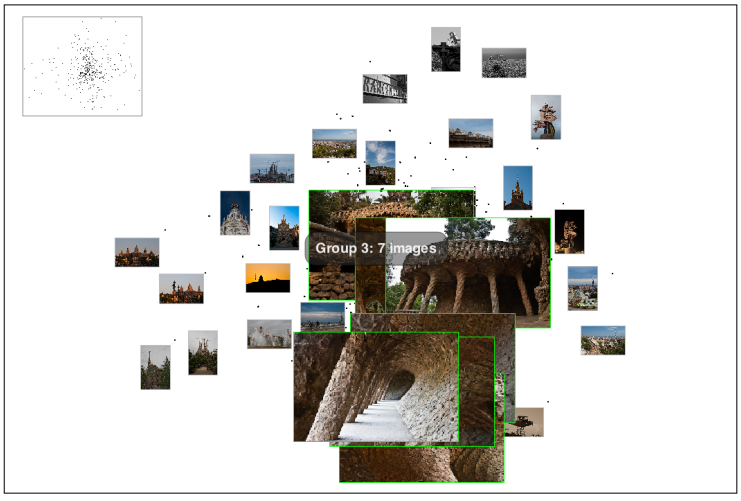


Figure 44:
The evaluated user interface showing the Barcelona collection with group 3 (green) in focus.

were encouraged to skip to the next collection after approximately five minutes as during this time already enough information would have been collected.

7.5.1.3 *Tweaking the Nearest Neighbor Index*

In the original implementation, at most five nearest neighbors are retrieved with the additional constraint that their distance to the query object has to be in the 1-percentile of all distances in the collection. (This avoids returning nearest neighbors that are not really close.) 264 of the 1050 images belonging to collections 2 to 4 have a ground truth topic label. For only 61 of these images, one or more of the five nearest neighbors belonged to the same topic and only in these cases, the secondary focus would have displayed something helpful for the given retrieval task. This let us conclude that the feature descriptors used were not sophisticated enough to capture the visual intra-topic similarity. A lot more work would have been involved to improve the features – but this would have been beyond the scope of the study that aimed to evaluate the user interface and most specifically the secondary focus which differentiates this approach from the common fish-eye techniques. In order not to have the user evaluate the underlying feature representation and the respective similarity metric, the index was modified for the experiment: Every time, the index was queried with an image with a ground truth annotation, the two most similar images from the respective topics were injected into the returned list of nearest neighbors. This ensured that the secondary would contain some relevant images.

7.5.2 *Results*

The user study was conducted with 30 participants – all of them graduate or post-graduate students. Their age was between 19 and 32 years (mean 25.5) and 40% were female. Most of the test persons (70%) were computer science students, with half of them having a background in computer vision or user interface design. 43% of the participants stated that they take photos on a regular basis and 30% use software for archiving and sorting their photo collection. The majority (77%) declared that they are open to new user interface concepts.

7.5.2.1 *Usability Comparison*

Figure 45 shows the results from the questionnaire comparing the usability and helpfulness of the SL approach with baseline P&Z. What

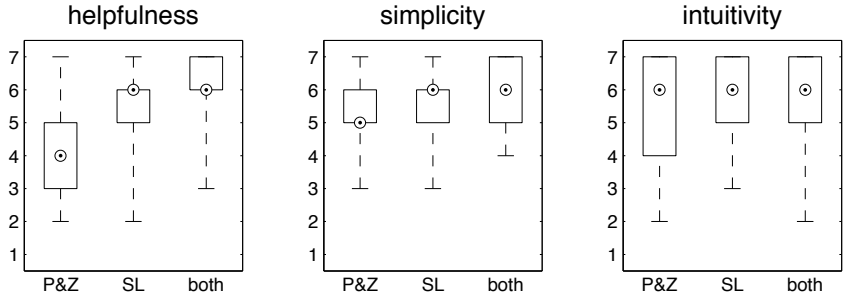


Figure 45: Usability comparison of common panning & zooming (P&Z), focus-adaptive *SpringLens* and the combination of both. Ratings were on a 7-point-scale where 7 is best. The box plots show minimum, maximum, median and quartiles for N = 30.

becomes immediately evident is that half of the participants rated the SL interface as being significantly more helpful than the simple P&Z interface while being equally complicated in use. The intuitiveness of the SL was surprisingly rated slightly better than for the P&Z interface, which is an interesting outcome since it was expected that users would be more familiar with P&Z as it is more common in today's user interfaces (e. g., *Google Maps* [url:17]). This, however, suggests that interacting with a fish-eye lens can be regarded as intuitive for humans when interacting with large collections. The combination of both got even better ratings but has to be considered noncompetitive here, as it could have had an advantage by always being the last interface used. Participants have had more time for getting used to the handling of the two complementary interfaces. Moreover, since the collection did not change as for P&Z and SL, the combined interface might have had the advantage of being applied to a possibly easier collection – with

topics being better distributed or a slightly better working similarity measure so that images of the same topic are found more easily.

7.5.2.2 *Usage of Secondary Focus*

For this part, the analysis was confined to the interaction with the last photo collection where both, P&Z and the lens, could be used and the participants had had plenty of time (approximately 15 to 30 minutes depending on the user) for practice. The question to be answered is, how much the users actually make use of the secondary focus which always contains some relevant images if the image in primary focus has a ground truth annotation.[6] For each image marked by a participant, the location of the image at the time of marking was determined. There are four possible regions:

1. primary focus (only the central image),
2. extended primary focus (region covered by primary lens except primary focus image),
3. secondary focus, and
4. the remaining region (i. e., no focus).

Furthermore, there are up to three cases for each region with respect to the (user-annotated or ground truth) topic of the image in primary focus. Table 12 shows the frequencies of the resulting eight possible cases. (Some combinations are impossible. For instance, the existence of a secondary focus implies some image in primary focus.) The

Table 12: Percentage of marked images (N = 914) categorized by focus region and topic of the image in primary focus at the time of marking.

focus region	primary	ext. primary	**secondary**	none
same topic	37.75	4.27	**30.74**	4.38
other topic		4.49	13.24	2.08
no focus				3.06
total	37.75	8.75	43.98	9.52

most interesting number is the one referring to images in secondary focus that belong to the same topic as the primary because this is what the secondary focus is supposed to bring up. It comes close to the percentage of the primary focus that – not surprisingly – is the highest. Ignoring the topic, (extended) primary and secondary almost contribute equally and only less than 10% of the marked images were not in focus – i. e., discovered only through P&Z.

6 Ground truth annotations were never visible to the users.

7.5.2.3 *Emerging Search Strategies*

For this part, again only interaction with the combined interface was analyzed. A small group of participants excessively used P&Z. They increased the initial thumbnail size in order to perceive the depicted contents and chose to display all images as thumbnails. To reduce the overlap of thumbnails, they operated on a deeper zoom level and therefore had to pan a lot. The gaze data shows a tendency for systematic sequential scans which were however difficult due to the scattered and irregular arrangement of the thumbnails. Furthermore, some participants occasionally marked images not in focus because of being attracted by dominant colors (e. g., for the aquarium topic).

Another typical strategy was to quickly scan through the collection by moving the primary focus – typically with small thumbnail size and at a zoom level that showed most of the collection but the outer regions. In this case the attention was mostly at the (extended) primary focus region with the gaze scanning in which direction to explore further and little to moderate attention at the secondary focus. Occasionally, participants would freeze the focus or slow down for some time to scan the whole display.

In contrast to this rather continuous change of the primary focus, there was a group of participants that browsed the collection mostly by moving (in a single click) the primary focus to some secondary focus region – much like navigating an invisible neighborhood graph. Here, the attention was concentrated onto the secondary focus regions.

7.5.2.4 *User Feedback*

Many participants had problems with an overcrowded primary fish-eye in dense regions. This was alleviated by temporarily zooming into the region which lets the images drift further apart. However, there are possibilities that require less interaction such as automatically spreading the thumbnails in focus with force-based layout techniques.

Working on deeper zoom levels where only a small part of the collection is visible, the secondary focus was considered mostly useless as it was usually out of view. Further work could therefore investigate off-screen visualization techniques to facilitate awareness of and quick navigation to secondary focus regions out of view and better integrate P&Z and SL. The increasing "empty space" at deep zoom levels should be avoided – e. g., by automatically increasing the thumbnail size as soon as all thumbnails can be displayed without overlap. An optional re-arrangement of the images in view into a grid layout may ease sequential scanning as preferred by some users.

Another proposal was to visualize which regions have already been explored similar to the (optionally time-restricted) "fog of war" used in strategy computer games. Some participants would welcome advanced filtering options such as a prominent color filter. An undo

function or reverse playback of focus movement would be desirable and could easily be implemented by maintaining a list of the last images in primary focus.

Finally, some participants remarked that it would be nice to generate the secondary focus for a set of images (belonging to the same topic). In fact, it is even possible to adapt the similarity measure used for the nearest neighbor queries automatically to the task of finding more images of the same to topic. This is further investigated in Section 8.6.

7.6 SUMMARY

A common approach for exploratory retrieval scenarios is to start with an overview from where the user can decide which regions to explore further. The focus-adaptive *SpringLens* visualization technique described in this chapter addresses the following three major problems that arise in this context:

1. Approaches that rely on dimensionality reduction techniques to project the collection from high-dimensional feature space onto two dimensions inevitably face projection errors: Some tracks will appear closer than they actually are and on the other side, some tracks that are distant in the projection may in fact be neighbors in the original space.

2. Displaying all tracks at once becomes infeasible for large collections because of limited display space and the risk of overwhelming the user with the amount of information displayed.

3. There is more than one way to look at a music collection – or more specifically to compare two music pieces based on their features. Each user may have a different way and a retrieval system should account for this.

The first problem is addressed by introducing a complex distortion of the visualization that adapts to the user's current region of interest and temporarily alleviates possible projection errors in the focused neighborhood. The amount of displayed information can be adapted by the application of several sparser filters. Concerning the third problem, the proposed user interface allows users to (manually) adapt the underlying similarity measure used to compute the arrangement of the tracks in the projection of the collection. To this end, weights can be specified that control the importance of different facets of music similarity and further an aggregation function can be chosen to combine the facets.

Following a user-centered design approach with focus on usability, a prototype system has been created by iteratively alternating between development and evaluation phases. For the final evaluation of the focus-adaptive *SpringLens* technique, an extensive user study with 30

participants has been conducted, including gaze analysis using an eye tracker. The results prove that the proposed technique is helpful while at the same time being easy and intuitive to use. Based on user feedback, several directions for further development have been identified of which some are addressed in the following chapters:

- Chapter 8 and in particular Section 8.6 focuses on the problem of automatically adapting the similarity measures.
- Chapter 9 proposes a way how the focus-adaptive *SpringLens* visualization technique could be used to facilitate serendipitous music discoveries.
- Chapter 10 investigates gaze-supported interaction techniques for the *MusicGalaxy* user interface prototype.

Everything we hear
is an opinion, not a fact.
Everything we see
is a perspective, not the truth.

Marcus Aurelius

8

CONTEXT-ADAPTIVE MUSIC SIMILARITY

This last chapter of Part ii finally turns to a key challenge of many MIR applications which also has been the main motivation for this thesis. As pointed out in Section 2.1, music information has many facets as well as many possible forms of representation. At the same time, users of MIR systems may have a varying (musical) background and experience music in different ways. Consequently, when comparing musical pieces with each other, opinions may diverge. A musician, for instance, might especially look after structures, harmonics or instrumentation (possibly paying – conscious- or unconsciously – special attention to his own instrument). Non-musicians will perhaps focus more on overall timbre or general mood. Others, in turn, may have a high interest in the lyrics as long as they are able to understand the particular language. But how musical pieces are to be compared also depends very much on the retrieval task at hand (Section 2.2.4). For instance, when looking for cover versions of a song, the timbre may be less interesting than the lyrics. In order to support individual user perspectives and different retrieval tasks, an adaptable model of music (dis-) similarity is required. Being at the very center of the general retrieval process as illustrated by Figure 46, it is the key to many MIR applications.

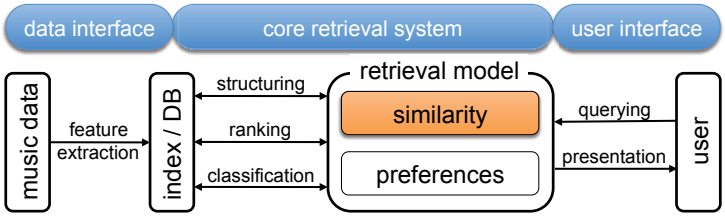


Figure 46: Music similarity in the context of the general retrieval process described in Section 2.2.1.

Given these considerations, the general idea of the approach presented in this chapter is as follows: Starting from various features that cover different aspects of music information, a complex multi-facet distance (or dissimilarity) measure is constructed. For each facet, distances can be computed *objectively* whereas the aggregation of the facet distances is *adaptable* by weighting the individual facets. A weighting scheme then represents a user's preference or task-specific model for grouping similar songs together and can be applied for any similarity-based structuring, ranking or classification approach. A user could explicitly adjust the facet weighting to fit his needs – but this is most likely a very difficult thing to do and some users might not even be

aware of their own preferences. Hence, ways for automatic adaptation of the facet weights based on user interaction with a collection are proposed.

Part of this overview has been published in [pub:22].

This chapter brings together work in this field from the recent years [pub:8, pub:10, pub:17], introducing a generalized view based on generic relative distance constraints (Section 8.1) which is also consistent with various related works (Section 8.2). This formalization provides a unified model for several adaptation approaches (Section 8.3) which have been applied in three different scenarios with varying objectives (Section 8.4 – Section 8.6). Covering all retrieval scenarios described in Section 2.2.1, this exemplifies how the proposed method can be employed in various contexts by deriving distance constraints either from domain-specific expert information or user actions in an interactive setting. For each application, the problem specific modeling of the learning problem is described together with the essential findings from experiments. Finally, the different approaches are compared against each other in an experiment based on the *Magnatagatune* benchmark dataset (Section 8.7). Section 8.8 concludes this chapter.

8.1 FORMALIZATION

To begin with, the concept of facet distances needs to be formalized assuming a feature-based representation of the objects of interest (which are generally music pieces in the scope of this paper):

Definition Given a set of features F, let S be the space determined by the feature values for a set of objects O. A *facet* f is defined by a *facet distance measure* δ_f on a subspace $S_f \subseteq S$ of the feature space, where δ_f satisfies the following conditions for any $a, b \in O$:

- $\delta_f(a,b) \geqslant 0$ and $\delta_f(a,b) = 0$ if and only if $a = b$
- $\delta_f(a,b) = \delta_f(b,a)$ (symmetry)

Furthermore, δ_f is a distance metric if it additionally obeys the triangle inequality for any $a, b, c \in O$:

- $\delta_f(a,c) \leqslant \delta(a,b) + \delta(b,c)$ (triangle inequality)

In order to avoid a bias when aggregating several facet distance measures, the values need to be normalized. The following normalization is applied for all distance values $\delta_f(a,b)$ of a facet f:

$$\delta'_f(a,b) = \frac{\delta_f(a,b)}{\mu_f} \tag{8.1}$$

where μ_f is the mean facet distance with respect to f:

$$\mu_f = \frac{1}{|\{(a,b) \in O^2\}|} \sum_{(a,b)\in O^2} \delta_f(a,b) \tag{8.2}$$

As a result, all facet distances have a mean value of 1.0. Special care has to be taken, if extremely high facet distance values are present that express "infinite dissimilarity" or "no similarity at all". Such values introduce a strong bias for the mean of the facet distance and thus should be ignored during its computation.

The actual distance between objects $a, b \in O$ w.r.t. the facets $f_1, \ldots, f_l$ is computed as weighted sum of the individual facet distances $\delta_{f_1}(a, b), \ldots, \delta_{f_l}(a, b)$:

$$d(a, b) = \sum_{i=1}^{l} w_i \delta_{f_i}(a, b) \tag{8.3}$$

This way, facet weights $w_1, \ldots, w_l \in \mathbb{R}$ are introduced that allow to adapt the importance of each facet according to user preferences or for a specific retrieval task. These weights obviously have to be non-negative and should also have an upper bound, thus:

$$w_i \geqslant 0 \qquad \forall 1 \leqslant i \leqslant l \tag{8.4}$$

$$\sum_{i=1}^{l} w_i = l \tag{8.5}$$

They can either be specified manually or learned from preference information. In the scope of this work, all preference information is reduced to *relative distance constraints*.

Definition A *relative distance constraint* (s, a, b) demands that object a is closer to the seed object s than object b, i. e.:

$$d(s, a) < d(s, b) \tag{8.6}$$

With Equation 8.3 this can be rewritten as:

$$\sum_{i=1}^{l} w_i(\delta_{f_i}(s, b) - \delta_{f_i}(s, a)) = \sum_{i=1}^{l} w_i x_i = \boldsymbol{w}^T \mathbf{x} > 0 \tag{8.7}$$

substituting $x_i = \delta_{f_i}(s, b) - \delta_{f_i}(s, a)$. Such basic constraints can directly be used to guide an optimization algorithm that aims to identify weights that violate as few constraints as possible [pub:2]. Alternatively, the positive examples $(\mathbf{x}, +1)$ and the negative examples $(-\mathbf{x}, -1)$ can be used to train a binary classifier in which case the weights $w_1, \ldots, w_l$ define the model (separating hyperplane) of the classification as pointed out by Cheng and Hüllermeier [38]. This is illustrated in Figure 47. Furthermore, the relative distance constraints are still rich enough to cover more complex forms of preference as shown in the following sections. Also note that such relative statements are usually much easier to formulate than absolute ones.

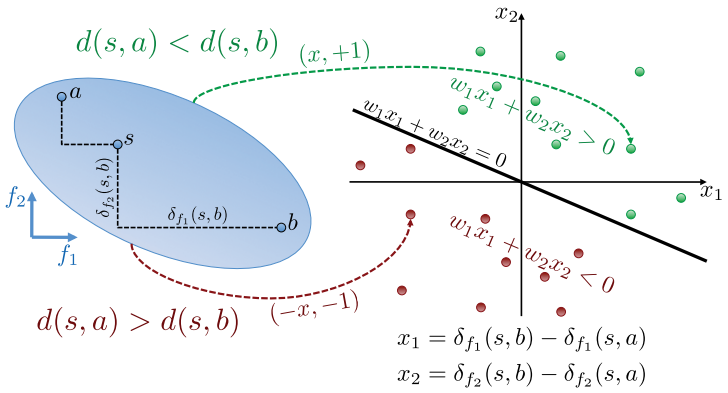


Figure 47: Transformation of a relative distance constraint into two training instances of the corresponding binary classification problem as described in [38].

8.2 RELATION TO OTHER APPROACHES

The approach presented here is based on prior work of NÜRNBERGER and KLOSE [173] and further inspired by the closely related formalizations of BADE [6] and CHENG and HÜLLERMEIER [38].

NÜRNBERGER and KLOSE [173] describe a SOM-based prototype for structuring text and image collections that automatically adapts its underlying weighted Euclidean distance according to user feedback (changing the location of objects in the map). For the adaptation, they do not explicitly use distance constraints as described by Equation 8.6. Instead, they apply weight update rules based on the feature differences which correspond to the x_i in Equation 8.7.

BADE [6] applies metric learning for personalized hierarchical structuring of (text) collections where each document is represented by a vector of term weights [210]. To this end, structuring preferences are modeled by so-called *"must-link-before constraints"* – each referring to a triplet (a, b, c) of documents. Such a constraint expresses a relative relationship according to hierarchy levels, namely that a and b should be linked on a lower hierarchy level than a and c. In hierarchical clustering, this means nothing else but that a and b are more similar than a and c. Consequently, *must-link-before constraints* can be considered as a domain-specific interpretation of the more generic relative distance constraints (Equation 8.6).

CHENG and HÜLLERMEIER [38] approach the metric learning problem from a case-based reasoning perspective and thus call the relative distance constraints *"case triples"*. In contrast to the previously outlined works, they address object representations beyond plain feature vectors: They model similarity as a weighted linear combination of "local distance measures" which corresponds to the facet concept and the aggregation function used here. As a major contribution, they also show how this formulation of the metric learning problem can be interpreted as a binary classification problem (Figure 47) that can

be solved by efficient learning algorithms and furthermore allows non-linear extensions by kernels.

Several other metric learning approaches with focus on music similarity have already been covered by Section 3.3.5. The work of McFee, Barrington, and Lanckriet [151, 152] is related in that their learning methods are also guided by relative distance constraints (which they call "partial order constraints"). They also combine features from different domains (acoustic, auto-tags and tags given by users) which could be interpreted as facets with the respective kernels corresponding to facet similarity measures. However, their approaches differ from the one proposed in this thesis in that they aim to learn an embedding of the features into an Euclidean space and to this end apply complex non-linear transformations using kernels as illustrated in Figure 48. Whilst their techniques are more powerful in the sense that they allow to model complex correlations, this comes at a high price: The high complexity is problematic when users want to understand or even manually adapt a learned distance measure. Here, the simplicity of the linear combination approach used in this thesis is highly beneficial.

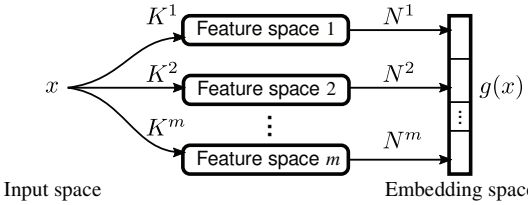


Figure 48: Illustration from [152] of the multi-kernel partial order embedding approach. An object representation x is first mapped into m different non-linear spaces (using m different kernel matrices $K^1, \ldots, K^m$). Then, for each space, a projection is learned ($N^1, \ldots, N^m$). The resulting vector representation $g(x)$ in the Euclidean embedding space is obtained by concatenation of the projection results.

Slaney, Weinberger, and White [227] state that they *"use labels* [artist, label and blog] *to tune the Mahalanobis matrix so that similar songs are likely to be close to each other in the metric space."* Although this is not explicitly stated in their description, this also implies either relative distance constraints (as used here) or absolute constraints of the form *"Songs* a *and* b *have to be in the same cluster."* However, using the Mahalanobis distance, they require more restrictive plain vector representations as input. Furthermore, the resulting music similarity model – i. e., the covariance matrix of the Mahalanobis distance – is still harder to interpret and adapt manually.

Finally, the *SoniXplorer* [135] is in many aspects similar to the (much earlier released) *BeatlesExplorer* prototype described in Section 8.5: As Figure 49 illustrates, the system covers multiple facets and uses a weighted linear aggregation for the underlying similarity measure. However, the adaptation is not guided by relative distance constraints. Instead, the system allows the users to specify distance information

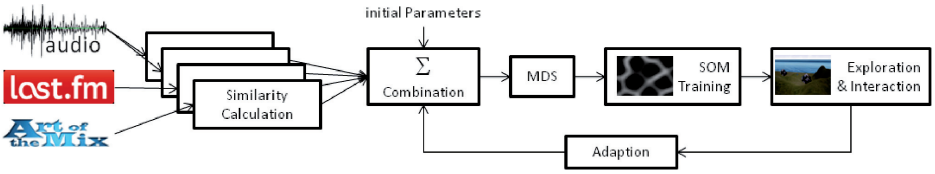


Figure 49: Illustration from [135] of the *SoniXplorer* data transformation and adaptation workflow.

by manipulation of the terrain, i. e., the formation of new separating hills or their removal respectively. By numerical integration over the height profile, a target distance matrix for the learning algorithm is derived that contains *absolute* (quantitative) distance information.

8.3 OPTIMIZATION APPROACHES

It is possible to look at the optimization problem introduced in Section 8.1 from different perspectives: tolerance w.r.t. constraint inconsistencies, stability, continuity and responsiveness.

Inconsistency-tolerance: In some scenarios, it might happen that constraints or combinations thereof contradict each other. Reasons for this can be manifold. For instance, the collected preferences might stem from various users or reflect the preference of a single user at different points in time. It is also possible that the user's comparison of the objects is based on features that are not covered by the distance facets. In any such case where inconsistencies occur, no weighting can be learned that satisfies all constraints. One way to deal with this problem is to detect and remove inconsistencies – e. g., by building a directed constraint graph, checking for cycles and removing appropriate edges as described in Section 8.3.4. Alternatively, an optimizer may use soft constraints as opposed to hard constraints that have to be satisfied.

Stability: If there is a weighting that satisfies all constraints, it is most likely not a unique solution. I. e., for the binary classification multiple hyperplanes might exist that separate the positive from the negative training examples (cf. Section 4.2). Choosing the maximal margin separating hyperplane – i. e., the one with the largest distance to the training examples – reduces the likelihood that small variances in the data lead to a different outcome.

Continuity: In an incremental learning setting where constraints are added over time, a gradual change of the weights might be desired. Here, the weighting with the smallest difference to the previous values is preferred rather than the one that results in the maximal margin.

Responsiveness: In interactive scenarios, the response time of the user-interface plays an important role. Here, the learning algorithm might need to guarantee that an acceptable solution is computed in restricted time bounds. So-called *anytime algorithms* are especially

suited for such tasks as they are able to return an approximate answer with the quality depending on the time available for computation.

The following sections describe three different optimization approaches that have been taken in the context of this thesis – each one for a different application: a gradient descent approach (Section 8.3.1), a Quadratic Programming (QP) approach (Section 8.3.2), and a maximum margin approach (Section 8.3.3). Furthermore, Section 8.3.4 describes a generic way of dealing with constraint inconsistencies and Section 8.3.5 proposes several alternative QP problem formulations.

8.3.1 *Gradient Descent*

For the folk song classification experiments described in Section 8.4, weights are learned by a gradient descent approach (as introduced in Section 4.1) similar to the one described by BADE and NÜRNBERGER [7]. During learning, all constraint triples (s, a, b) are presented to the algorithm several times until convergence is reached. If a constraint is violated by the current distance measure, the weighting is updated by trying to maximize

$$obj(s, a, b) = \sum_{i=1}^{l} w_i(\delta_{f_i}(s, b) - \delta_{f_i}(s, a)) \tag{8.8}$$

which can be directly derived from Equation 8.7. This leads to the update rule for the individual weights:

$$w_i = w_i + \eta \Delta w_i, \quad \text{with} \tag{8.9}$$

$$\Delta w_i = \frac{\partial obj(s, a, b)}{\partial w_i} = \delta_{f_i}(s, b) - \delta_{f_i}(s, a) \tag{8.10}$$

where the learning rate η defines the step width of each iteration.[1] To enforce the bounds on w_i given by Equation 8.4 and Equation 8.5, an additional step is necessary after the update, in which all negative weights are set to 0 and then the weights are normalized which yields a constant weight sum of l.

This algorithm can compute a weighting, even if not all constraints can be satisfied due to inconsistencies. However, it is not guaranteed to find a globally optimal solution and no maximum margin is enforced. Using the current weights as initial values in combination with a small learning rate allows for some continuity but there may still be solutions with less change required. It is possible to limit the number of iterations to increase responsiveness but this may result in some unsatisfied constraints.

1 Interestingly, approaching the weight learning problem from the classification perspective using a perceptron for classification as described by CHENG and HÜLLERMEIER [38] eventually leads to the same update rule.

8.3.2 *Quadratic Programming: Minimizing Weight Change*

For maximum continuity which is considered most important in the application described in Section 8.5, the weights should change only as little as necessary to satisfy all constraints. This can directly be modeled as a Quadratic Programming (QP) problem demanding in the objective function that the sum over all (quadratic) deviations of the weights from their previous values should be minimal (with initial values 1):

$$\min_{(w_1,\ldots,w_l)\in\mathbb{R}^l} \sum_{i=1}^{l} \left(w_i - w_i^{(old)}\right)^2 \tag{8.11}$$

subject to the constraints that enforce the weight bounds (Equation 8.4 and Equation 8.5) and the distance constraints (Equation 8.7) which can be used directly. The problem can be solved using the Goldfarb and Idnani dual QP algorithm for convex QP problems subject to general linear equality/inequality constraints [75]. For this formalization of the weight learning problem, there is only a solution if all constraints are consistent. In order to allow constraint violations, the distance constraints need to be turned into soft constraints by introducing slack variables as described in Section 8.3.5.

8.3.3 *Maximum Margin Classifier*

If stability is more important than continuity, the primary objective is to maximize the margin between the separating hyperplane and the positive and negative training samples (generated from the distance constraints as described in Equation 8.7). For the application described in Section 8.6, the linear SVM algorithm as provided by *LIBLINEAR* [66] is used. The fundamental idea of this technique is described in Section 4.2. With this approach, a valid value range for the weights cannot be enforced. Specifically, weights can become negative. To reduce the chance of negative weights, artificial training examples are added that require positive weights (setting a single x_i to 1 at a time and the others to 0). As these constraints are also soft constraints, they may still be violated in favor of a larger margin or in case of general constraint inconsistencies.

8.3.4 *Dealing with Inconsistent Constraint Sets*

Sometimes the set of constraints to be used for learning may be inconsistent because there are constraints that contradict each other. Reasons for this may be manifold – e. g., a user may have changed his mind or the constraints may be from different users or contexts in general. In such cases, it is impossible to learn a facet weighting that satisfies all constraints – regardless of the learning algorithm or

the facets used. In order to obtain a consistent set of constraints, the constraint filtering approach described by McFee and Lanckriet [152] can be applied as follows:

1. A directed multigraph (i. e., a graph that may have multiple directed edges between two nodes) is constructed with pairs of objects as nodes and the distance constraints expressed by directed edges. For instance, for the distance constraint $d(b,c) < d(a,c)$, a directed edge from the node (b,c) to the node (a,c) would be inserted.

2. All cycles of length 2 are removed, i. e., all directly contradicting constraints. (This can be done very efficiently by checking the graph's adjacency matrix.)

3. The resulting multigraph is further reduced to a Directed Acyclic Graph (DAG) in a randomized fashion: Starting with an empty DAG, the edges of the multigraph are added in random order omitting those edges that would create cycles.

4. The corresponding distance constraints of the remaining edges in the DAG form a consistent constraints set.

This can be repeated multiple times as the resulting consistent set of constraints may not be maximal because of the randomized greedy approach taken in step 3. However, an exhaustive search for a maximum acyclic subgraph would be NP-hard.

8.3.5 *Quadratic Programming Approaches with Soft Constraints*

In order to derive alternative formalizations of the QP problem outlined in Section 8.3.2, a closer look at the optimization problem is required. The underlying algorithm [75] solves convex QP problems of the form

$$\min_{\mathbf{x}\in\mathbb{R}^n} \mathbf{a}^T\mathbf{x} + \frac{1}{2}\mathbf{x}^T\mathbf{G}\mathbf{x} \tag{8.12}$$

subject to linear equality and inequality constraints

$$\mathbf{x}^T\mathbf{C}_e = \mathbf{b}_e \tag{8.13}$$

$$\mathbf{x}^T\mathbf{C}_i \geqslant \mathbf{b}_i \tag{8.14}$$

given the vectors $\mathbf{b}_e$ of dimension m_e, $\mathbf{b}_i$ of dimension m_i, and $\mathbf{a}$ of dimension n, and the matrices $\mathbf{G}$ of of dimension $n \times n$, $\mathbf{C}_e$ of dimension $m_e \times n$, and $\mathbf{C}_i$ of dimension $m_i \times n$. The matrix $\mathbf{G}$ has to be symmetric positive definite. In this case, a unique $\mathbf{x}$ solves the problem or the constraints are inconsistent.

8.3.5.1 *Original Problem Formulation*

In the original modeling (Section 8.3.2), the objective is to minimize the weight change under some constraints given the previous weights $w_1^{(old)}, \ldots, w_l^{(old)}$. In particular, this approach can be used to determine facet weights that are closest to a uniform weighting and feasible under the given constraints by simply setting all previous weights to 1. Here, the elements of the vector $\mathbf{x}$ in the QP problem description correspond to the facet weights $w_1, \ldots, w_l$ (where l is the number of facets), and therefore n equals l. Figure 50 shows the problem description for the QP solver which is explained in the following:

$$\mathbf{G} = \begin{bmatrix} 2 & 0 & \cdots & 0 \\ 0 & 2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 2 \end{bmatrix} \qquad \mathbf{a} = \begin{bmatrix} -2w_1^{(old)} \\ -2w_2^{(old)} \\ \vdots \\ -2w_l^{(old)} \end{bmatrix}$$

$$\mathbf{C}_e = \begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix} \qquad \mathbf{b}_e = \begin{bmatrix} l \end{bmatrix}$$

$$\mathbf{C}_i = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ c_{1,1} & c_{1,2} & \cdots & c_{1,l} \\ c_{2,1} & c_{2,2} & \cdots & c_{2,l} \\ \vdots & \vdots & \ddots & \vdots \\ c_{k,1} & c_{k,2} & \cdots & c_{k,l} \end{bmatrix} \qquad \mathbf{b}_i = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \epsilon \\ \epsilon \\ \vdots \\ \epsilon \end{bmatrix}$$

Figure 50: QP problem description for the minimization of the change of l non-negative and bounded distance facet weights subject to k distance constraints.

The objective function given in Equation 8.11 can be transformed into:

$$\min_{(w_1,\ldots,w_l)\in\mathbb{R}^l} \sum_{i=1}^{l} w_i^2 - 2\sum_{i=1}^{l} w_i w_i^{(old)} + \sum_{i=1}^{l} w_i^{(old)^2} \tag{8.15}$$

With respect to Equation 8.12, the first sum is captured by $\frac{1}{2}\mathbf{x}^T\mathbf{G}\mathbf{x}$, the second sum is expressed by $\mathbf{a}^T\mathbf{x}$, and the third sum results in a constant value independent of the w_i and thus can be omitted. A single equality constraint is required to model the bound on the weight sum (Equation 8.5) by setting the respective coefficients in $\mathbf{C}_e$ to 1 and the value in $\mathbf{b}_e$ to l. Setting the i-th element of a row vector of $\mathbf{C}_i$ to 1 and the other elements and the corresponding value in $\mathbf{b}_i$ to 0 enforces the non-negativity of w_i. Thus, l inequality constraints are needed to express Equation 8.4. Finally, each distance constraint is represented by a single row vector of $\mathbf{C}_i$ where the value $c_{i,j}$ is the x_i

from Equation 8.7 for the j-th distance constraint. The respective value in $\mathbf{b}_i$ has to be a value $\epsilon > 0$ because 0 would also allow the equality in Equation 8.6. Naturally, this value should be as small as possible w.r.t. machine precision. Greater values increase the stability of the solution but may also result in an inconsistent system if the solution space is trimmed too rigorously such that feasible solution no longer exists.

8.3.5.2 *Introducing Slack Dimensions*

In order to allow a distance constraint as formulated in Equation 8.7 to be violated, a slack variable $\xi \geqslant 0$ needs to be introduced such that:

$$\sum_{i=1}^{l} w_i(\delta_{f_i}(s,b) - \delta_{f_i}(s,a)) + \xi > 0 \tag{8.16}$$

This has to be done individually for all k distance constraints of the QP problem resulting in the respective slack variables $\xi_1, \ldots, \xi_k$. They are modeled as k additional dimensions of the vector $\mathbf{x}$ which is now $(w_1, \ldots, w_l, \xi_1, \ldots, \xi_k)$. (Consequently, the dimensionality of the modified QP problem is $l + k$ and as the number of distance constraints k can become quite big, this has a significant impact on the performance of the optimization algorithm.) Figure 51 highlights the modifications of the equality and inequality constraints.

$$\mathbf{C}_e = \Big[\overbrace{1 \quad 1 \quad \cdots \quad 1}^{l} \quad \overbrace{0 \quad 0 \quad \cdots \quad 0}^{k}\Big] \qquad \mathbf{b}_e = \begin{bmatrix} l \end{bmatrix}$$

$$\mathbf{C}_i = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 & 0 & \cdots & 0 \\ c_{1,1} & c_{1,2} & \cdots & c_{1,l} & 1 & 0 & \cdots & 0 \\ c_{2,1} & c_{2,2} & \cdots & c_{2,l} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{k,1} & c_{k,2} & \cdots & c_{k,l} & 0 & 0 & \cdots & 1 \end{bmatrix} \qquad \mathbf{b}_i = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \epsilon \\ \epsilon \\ \vdots \\ \epsilon \end{bmatrix}$$

Figure 51: Modified equality and inequality constraints with one added slack dimension for each of the k distance constraints.

For the constraints that ensure the weight bounds and the non-negativity of the weights, the added matrix columns are filled with zeros. For each of the k distance constraints, only the value in the column of the respective slack dimension is set to 1 while all others remain 0.

Slack values other than 0 have to result in a penalty. To this end, the objective function needs to be extended. There are two possibilities

to incorporate a slack penalty: either in the linear or the quadratic part of Equation 8.12. In the first case, the sum of the slack values is minimized which results in an objective function similar to the maximum margin objective (Equation 4.10). In the second case, it is the sum of the squared slack values. The constant κ allows to balance importance between the (initial) objective function and the minimization of the aggregated slack. Figure 52 shows the respective changes in the formulation of the objective function for the solver.

$$\mathbf{G}' = \begin{bmatrix} & & & 0 & \cdots & 0 \\ & \mathbf{G} & & \vdots & \ddots & \vdots \\ & & & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 0 \end{bmatrix} \quad \mathbf{a}' = \begin{bmatrix} a_1 \\ \vdots \\ a_l \\ \kappa \\ \vdots \\ \kappa \end{bmatrix}$$

$$\mathbf{G}' = \begin{bmatrix} & & & 0 & \cdots & 0 \\ & \mathbf{G} & & \vdots & \ddots & \vdots \\ & & & 0 & \cdots & 0 \\ 0 & \cdots & 0 & 2\kappa & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 0 & \cdots & 2\kappa \end{bmatrix} \quad \mathbf{a}' = \begin{bmatrix} a_1 \\ \vdots \\ a_l \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Figure 52: Modified objective functions with k added slack dimensions aggregated as sum (top) or sum of squared values (bottom) and weighted by a constant $\kappa > 0$. Values of $\mathbf{G}$ and $\mathbf{a}$ are determined by the primary (initial) objective function.

In case the sum of the squared slack values is used, both, negative and positive slack values are penalized. This way, the solver naturally avoids negative slack values. For the (linear) sum, however, the QP approach does not work if negative slack values are allowed because this introduces the problem that a single big negative slack value (of a constraint which is not violated) can compensate many small positive slack values of constraints that are violated. This results in a bias towards solutions with more violated constraints than necessary. Therefore, k inequality constraints that explicitly demand non-negative slack values have to be added to the scheme shown in Figure 51.

Both approaches for incorporating a slack penalty into the objective function of the QP solver can be combined without interference (in the modeling) with both primary objectives – either minimizing the weight change or maximizing a single facet weight. Furthermore, it is possible to have no primary objective and just minimize the slack penalty. The performance of these combinations is analyzed in Section 8.7. Afore, however, the following sections describe the application of the discussed optimization approach in different scenarios.

8.4 APPLICATION I: FOLK SONG ANALYSIS

Folk song researchers detect and document relations between folk songs and their performances in order to understand oral transmission. At the *Meertens Institute*, they classify folk song variants into so called *melody norms*. This traditional classification captures both, aspects of musical similarity and historical relationships. There is only one class per tune and each class is represented by a *reference tune*. The *WITCHCRAFT* project supports the musicologists with a system for browsing tunes by musical content (in a symbolic representation). Given a query tune, the system ranks the tunes in the database according to a chosen similarity measure. Such a *tune ranking list* can be filtered by tune classification. This results in a *class ranking list* that contains for each melody norm only the most similar tune of that class. The latter is especially helpful when new tunes need to be classified. Several distance and similarity measures can be chosen for generating the ranked lists. However, a particular musicological way of classifying songs is usually not directly reflected by just a single one of these measures. Therefore, the multi-facet approach introduced in Section 8.1 suggests itself for modeling tune similarity based on a range of basic distance and similarity measures.

A detailed description of this application has been published as joint work with Korinna Bade and Jörg Garbers in [pub:8].

8.4.1 *Modeling the Learning Problem*

The challenge is to find an optimal facet weighting that produces tune and class ranking lists with tunes belonging to the same melody norm ranked first. Two scenarios are considered: Retrieving similar tunes for already classified tunes and querying with unclassified tunes.

Given a *classified* query tune q, it is known from the expert classification, which other tunes t_r belong to the same (relevant) class and which tunes t_i are irrelevant. As tunes of the same class should be ranked first and thus should be more similar, relative distance constraints (q, t_r, t_i) can directly be derived. Depending on the set of queries (and the resulting set of constraints) used, weighting for different levels of specificity can be learned:

- *individual tune weightings* $\boldsymbol{w}^{\mathrm{tune}}$ – for a single tune (most specific and thus with the risk to overfit),
- *class weightings* $\boldsymbol{w}^{\mathrm{class}}$ – for the set of tunes belonging to the same class (more general), and an
- *overall weighting* $\boldsymbol{w}^{\mathrm{all}}$ – for all tunes (most general).

In case of an *unclassified* query tune without a specific individual- or class weighting, either the overall weighting has to be used or a case-based approach is taken to select a weighting in the following two steps – assuming that similar tunes have similar optimal weightings:

1. Find the closest (classified) tune t_{best}.
   (Here, the overall weighting is applied or the specific individual- or class weighting of each tune.[2])

2. Use the individual or class weighting of t_{best} to compute the ranking for the query.

In the following, $\boldsymbol{w}^b \circ \boldsymbol{w}^a$ denotes that $\boldsymbol{w}^a$ is used for case selection (step 1) and $\boldsymbol{w}^b$ for ranking (step 2).

8.4.2 *Experiments*

A dataset was provided that comprises 360 tunes – all well-understood single melodic strophes – and their classification into one of 26 melody norms. Instead of the actual tunes, the dataset contains only the pairwise facet distances / similarities. 14 measures are considered of which 11 are taken from the *Simile* package [url:33]. These are *rawEd*, *diffEd*, *nGrSumCo*, *nGrUkkon*, *harmCorE*, *rhytFuzz*, *rhytGaus*, *opti1*, *opti3*, *accents_opti1* and *accents_opti2*. Note that the last four measures are themselves linear combinations of basic similarity measures in the sense of *Simile*. Two distance measures are based on the spectra of Laplacian and adjacency graphs [196] and one is an unpublished pitch sequence edit distance, implemented at *Meertens Institute*.

8.4.2.1 *Querying with Classified Tunes*

Appendix B provides a brief explanation of the evaluation measures.

Figure 53 (top) contains the precision/recall curves for the three learned weightings and the two best basic measures, `rawEd` and `opti1`. Table 13 (top) shows the corresponding evaluation of the class ranking lists, ordered by best performance w.r.t. *average rank* of correct class (smaller is better), *average inverse rank* (larger is better), and classification *precision at 1st rank*. Not surprisingly, higher specificity leads to better performance in this scenario but is also more vulnerable to overfitting. The overall weighting performs worse than the best basic similarity measure (*rawEd*) in most precision/recall regions (although only slightly) but *rawEd* produces the worst class ranking lists. All this indicates that there might not be a single perfect overall measure that can be used in general but rather data/problem specific measures instead.

8.4.2.2 *Querying with Unclassified Tunes*

In this more realistic scenario, two situations are possible: In the first case (Figure 53; middle), all but the query tune are used for learning

2 Note that in this case different weightings are used to compute the distance to different database tunes, which leads to local distortions of the distance space around each case. While such a locally distorted metric is unsuitable for the computation of the entire ranking, it may still be useful to retrieve only t_{best}.

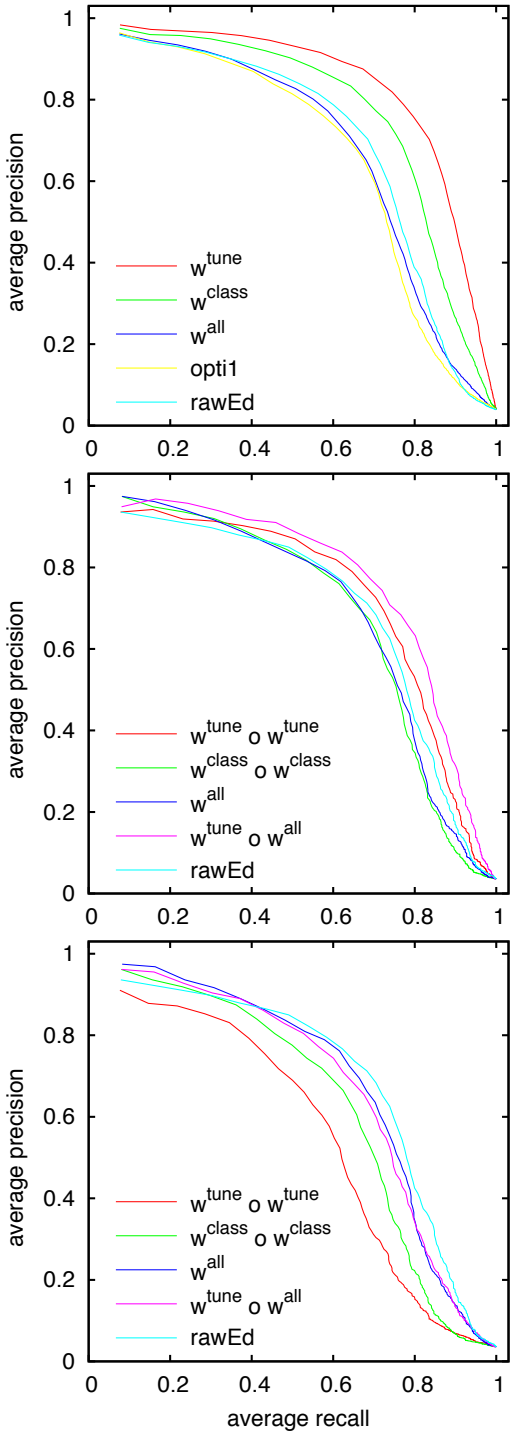


Figure 53: Precision / Recall plots for the tune ranking lists using classified tunes (top), unclassified tunes of a known class (middle), and un- classified tunes of an unknown class (bottom).

measure	average rank	avg. inv. rank	precision @ 1
w^{tune}	1.042	0.989	0.983
w^{class}	1.083	0.985	0.975
opti1	1.169	0.975	0.961
w^{all}	1.172	0.974	0.961
rawEd	1.233	0.967	0.956
$w^{tune} \circ w^{all}$	1.218	0.969	0.949
w^{all}	1.231	0.981	0.973
$w^{tune} \circ w^{tune}$	1.244	0.957	0.936
$w^{class} \circ w^{class}$	1.346	0.976	0.973
rawEd	1.410	0.946	0.936
w^{all}	1.218	0.982	0.973
$w^{tune} \circ w^{all}$	1.244	0.971	0.961
$w^{tune} \circ w^{tune}$	1.282	0.942	0.910
$w^{class} \circ w^{class}$	1.359	0.970	0.961
rawEd	1.410	0.946	0.936

Table 13: Evaluation of the class ranking lists for classified tunes (top), unclassified tunes of a known class (middle), and unclassified tunes of an unknown class (bottom).

– including other tunes of the same class. But of course, the system does not know during ranking which ones these are. In the other case (Figure 53; bottom) none of the tunes from the query tune's class are used for learning, simulating an entirely new class that shall be added to the database. This is of course a much harder case. For computation of the precision and recall values, *all* tunes were ranked according to the query tune, including the songs of the unknown class.[3]

Comparing the plots for both cases and taking the *rawEd* measure as baseline for reference shows that the performance of the learned measures is lower for the harder case – most notably for $w^{tune} \circ w^{tune}$. This implies that the case-based weight selection approach successfully chooses members of the same tune family if such are already in the database and otherwise resorts to less appropriate weightings from other tune families. The overall weighting w^{all} turns out as most suitable for selecting the tune whose weighting is then used for ranking and even as best choice for the hardest case in general. For the very specific weightings there seems to be too much overfitting which makes them less applicable for other tunes. The baseline *rawEd* is better at the end of the ranking, while it is worse at the beginning. This is also reflected in the class ranking lists evaluation (Table 13) where *rawEd* performs worst. For automatic classification, w^{all} produces the fewest errors at the first rank.

3 This methodology of simulating new tunes is very time-consuming as for each query the learning has to be redone without the respective information. Thus, only the reference melody and two random tunes are considered for each class resulting in 78 out of 360 queries.

8.5 APPLICATION II: BEATLESEXPLORER

With the *BeatlesExplorer*, a first user-interface prototype for adaptive structuring and exploration of music collections has been developed and presented in [pub:10]. The system uses a dataset containing 282 songs of The Beatles and a multi-facet distance measure with about 20 facets covering sound, harmonics, lyrics and information about the production process. The feature information is extracted (semi-automatically) from Wikipedia [url:53], LyricWiki [url:28], Alan W. Pollack's notes on The Beatles [url:41], manual chord annotations [88] and from the audio recordings using the frameworks CoMIRVA [213] and JAudio [150]. A detailed description is given in [pub:10].

Using initially uniform facet weights, a GSOM is induced as described in Section 4.3.1, clustering similar songs of the music collection into hexagonal cluster cells. The result is a two-dimensional topology that preserves the neighborhood relations of the originally high-dimensional feature space, i. e., not only songs within the same cluster cell are similar to each other but also songs of cells in the neighborhood are expected to be more similar than those in more distant cells. Using a growing approach as opposed to the common static approaches as, e. g., [108, 179] ensures that only as many cells are created as are actually needed. Further, the approach is incremental: Songs may be added to the collection without having to relearn the whole map from scratch – instead, it is only extended. A screenshot of the prototype user interface is shown in Figure 54.

Video Clip 3

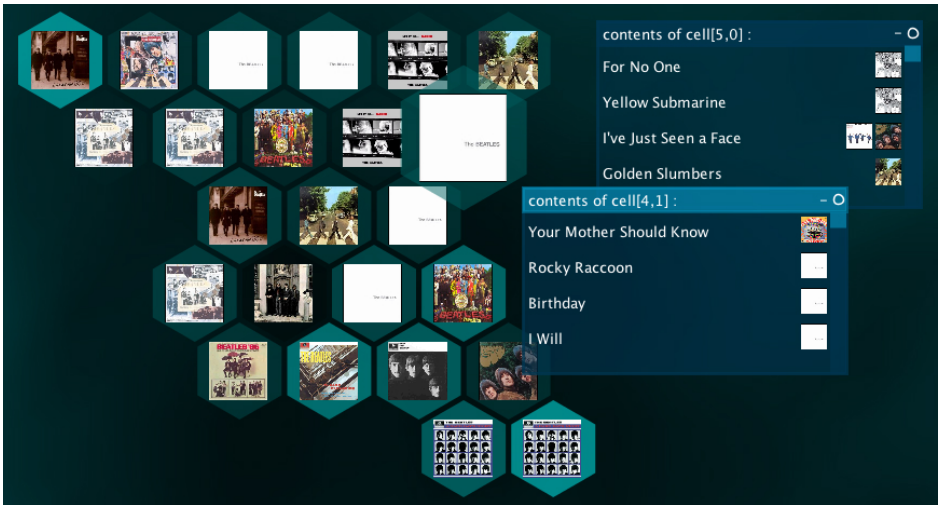


Figure 54: Screenshot of the *BeatlesExplorer* with the grid on the left (colored according to the number of songs in each cell) and two cell content windows on the right.

The cells of the generated hexagonal grid act as "virtual folders", each one containing a set of similar songs. Cells are labeled with the album cover(s) that are most frequently linked with the songs contained in the cell. Clicking on a cell opens a window that displays its content. For each song, the title and (if available) the album covers are listed.

8.5.1 *Vectorization*

Self-Organizing Maps (SOMs) as described in Section 4.3 belong to the category of prototype-based clustering approaches, i. e., each cluster is represented by a prototype. They require that the objects to be clustered and the cluster prototypes are vectors, i. e., elements of a vector space. The multi-facet feature representation used here does not adhere to this rather severe restriction and thus, some means of vectorization is required. An established vectorization approach that can be considered as common practice in related work (e. g., [108, 179]), is to simply interpret each row of the distance matrix (containing the pairwise distances of all objects in the dataset) as a feature vector. I. e., the objects are described by the distances between each other. This approach has also been used originally in the BeatlesExplorer. However, recent investigations of the impact of this transformation on the distance relations (cf. Appendix A) suggest an alternative approach based on MDS that is especially suited for multi-facet representations like the one at hand.

8.5.2 *Modeling the Learning Problem*

The initial clustering is done in an unsupervised manner. If the user does not agree on the cluster assignment for a specific song, he may change its location on the map by simple drag-and-drop actions. Furthermore, he can query the system for the ten most similar songs given a seed song and change the order of the resulting list.[4] Any of these actions, moving a song to another cell or modifying a ranking, results immediately in an adaptation of the distance measure and ultimately in a reassignment of all songs to the grid. This way, single manual actions may cause several automatic changes. While the user is interacting with the system in an iterative process of user action and resulting adaptation, the facet weights step-by-step converge towards to the (possibly only subconsciously existing) preferences of the user.

The problem of dragging songs as described above can be mapped to a problem of cluster reassignment: Let P be the set of prototypes representing all the cluster cells of the SOM. Following the "winner

4 Such an action, e. g., through drag & drop on the ranked list, has not yet been incorporated into the graphical user interface. Currently, it can only be carried out through the simulation interface for evaluation.

takes all" principle, each object is assigned to the cluster with the most similar prototype. If the user moves a song o to a target cluster t, a *position constraint* is generated. Such a constraint can be expressed by multiple distance constraints as the distance of o to the respective prototype p_t of t has to be smaller than to any other prototype:

$$d(o, p_t) < d(o, p) \qquad \forall p \in P \setminus \{p_t\} \tag{8.17}$$

If the user corrects a retrieved top-10 list $o_1, \ldots, o_{10}$ for a seed song s by moving the song at rank i to rank j, a ranking constraint $(s, o_1, \ldots, o_{10}, i, j)$ is generated. Such a constraint is interpreted as follows: If j is less than i, it can be concluded that $o_j, \ldots, o_{i-1}$ are less similar and that o_{j-1} (if such exists) is more similar than o_i w.r.t. the seed song s. In the other case, i. e., if j is greater than i, the songs with rank $i < r < j$ are more similar than o_i and o_j is less similar w.r.t. the seed song s. This translates to the following distance constraints:

$$\begin{aligned} & d(s, o_i) > d(s, o_{j-1}) \\ \wedge\ & d(s, o_i) < d(s, o_r) \quad \forall j \leqslant r < i \quad \text{if } j < i \\ & d(s, o_i) < d(s, o_j) \\ \wedge\ & d(s, o_i) > d(s, o_r) \quad \forall i < r < j \quad \text{if } i < j \end{aligned} \tag{8.18}$$

This way, the set of constraints increases with each user action. In this incremental learning scenario, where a gradual change of the weights is desired to avoid too abrupt changes of the cluster assignments, the Quadratic Programming (QP) approach introduced in Section 8.3.2 has been applied.

8.5.3 *Experiments*

Additional details have been published in [pub:10].

By simulating the user interaction with the system, it is possible to objectively evaluate the usefulness of the adaptation approach. This section outlines the experimental setup and the most important findings. The basic idea is to assume for each simulation a fixed random facet weighting that the simulated user has "in mind". The following three scenarios are considered: fully random weights, random binary weights (with additional normalization to satisfy Equation 8.5), and random single facet only (the weights for all facets except for a single random facet are zero). The resulting distance measure (which is unknown to the system) is used to identify misplaced songs or the correct order of a ranking respectively. During a simulation, the user either only rearranges misplaced songs or corrects rankings. In each simulation step, one randomly selected misplaced song is moved or the top-10 ranking for a random seed song is corrected respectively (only considering the result with the highest rank discrepancy). Afterwards, the weights are adapted incorporating the new constraints. A simulation terminates when no misplaced object can be moved anymore or

no more top-10 rankings can be corrected respectively. Each scenario is tested on multiple SOMs (learned on the same data but with different random initializations) for multiple users (different facet weights) and multiple simulations each (different songs chosen to be moved or as seeds for ranked lists). Two measures are computed for evaluation: the average top-10 precision (Prec@10) which measures how much the nearest neighbors agree, and the Object Position Error (OPE). The latter is defined as the aggregated Euclidean distance on the cell grid between the current and the correct cell coordinates for each object in the dataset [pub:10]. It thus allows a much finer assessment of the object misplacement than the bare number of misplaced objects.

8.5.4 *Results*

Of the three scenarios, the fully random weighting is closest to the uniform initialization. Only few iterations for adaptation are required and the OPE is surprisingly small. Scenario 3 is the hardest with an initial Prec@10 of about only 10% and nearly all songs misplaced.

Using position constraints, the adaptation converges much quicker and requires far fewer iterations than for ranking constraints. A possible reason is that more distance constraints can be derived from a position constraint and thus the adaptation algorithm has more information in fewer steps. Furthermore, the termination criterion is met earlier what becomes clear when looking at the final Prec@10: For rearranging misplaced songs, the value would never come even close to 100% – even though all songs finally are at their correct position – and is much smaller than what can be achieved by using ranking constraints. The explanation for this "glass ceiling" effect lies in the nature of the SOM. All objects are assigned to the correct cluster, as long as there is no other cluster that is more similar. If this criterion is met, the simulation stops because no more position constraints can be generated that would add information for the optimizer. At this point, however, there may still be many valid weighting schemes. The system has just not enough evidence to decide which one is the right one, and chooses the one that minimizes the objective function (Equation 8.11). Especially with a large solution space, the chosen weighting may significantly differ from the real one which results in the "glass ceiling" for the Prec@10.

Using the ranking constraints directly optimizes the Prec@10 and only indirectly affects the OPE through the adapted distance measure. Consequently, a significantly higher number of iterations is necessary in these simulations until all songs are correctly arranged, independently from the scenario of user weight initialization. Apart from the fact that less information can be derived from a ranking constraint, the resulting distance constraints may also be not as restrictive because they mostly refer to songs that are considered similar. In contrast,

the variance amongst the cluster prototypes (which are considered for each position constraint) is high because they cover the whole collection.

Consequently, the optimal strategy for a user to minimize manual effort would be a combination of both, rearranging and ranking. Moving misplaced songs can be used for a rough adaptation of the system in only a few steps. Afterwards, the correction of the ranking provides a means for fine-tuning.

8.6 APPLICATION III: MUSICGALAXY

The user-interface prototype presented in the previous section has several limitations: Apart from the required vectorization (Section 8.5.1), there are several parameters that have to be tuned carefully to obtain a good SOM such as the initialization of the prototypes, the learning rate, the termination criterion for iteration, the initial structure, and the rules by which the structure should grow. Moreover, if the distance measure is changed drastically during user adaptation, the structure of the SOM which was learned using the initial facet weights may no longer be appropriate. In the worst case, all songs might end up in the same cluster cell. Another problem is the scalability w.r.t. the collection size: The initial generation of the SOM already takes several seconds for the rather small Beatles corpus. Furthermore, for large collections, the SOM will consist of many cells or cells that contain many songs – and both limits the usefulness of the visualization. Finally, there is also a fundamental problem of approaches that (like SOMs) project a dataset onto two dimensions while trying to preserve the topology as discussed in Chapter 7.

The *MusicGalaxy* prototype described in Chapter 7 therefore takes a different, focus-adaptive visualization approach. As described in Section 7.4.3, it also provides means for adapting the distance facet weighting for the projection and the distortion. Hence, it is also an *adaptable* system according to the definition given in Section 3.2.1 w.r.t. the underlying distance measures. One way to turn it into an *adaptive* system is to incorporate the techniques demonstrated for the *BeatlesExplorer* in the preceding section, i. e., letting the user re-arrange tracks and/or re-rank tracks according to the perceived similarity to a seed track. While the latter is very straightforward to implement, a solution for the former is not obvious because of the lack of clusters and respective prototypes that are required to formulate position constraints as described in Section 8.5.2. Thus, a somewhat different interpretation of object movements which ultimately leads to distance constraints would be required here to guide the learning algorithm for the projection distance measure.

However, there is also another way of making *MusicGalaxy* adaptive – by learning a distance measure that identifies neighbors for the sec-

ondary focus. As described in Section 7.5, the galaxy user-interface has been evaluated in a user study where participants had to solve an exploratory image retrieval task: finding and tagging representative images for several topics. The evaluation showed that the participants indeed frequently used the secondary focus to find other photos belonging to the same topic as the one in primary focus. However, some photos in secondary focus did not belong to the same topic. Follow-up experiments investigated whether it is possible to automatically adapt the neighbor index during the exploratory search process to return more relevant neighbors for the primary focus topic. The approach and findings are summarized in the following.

A detailed report of the experiments has been published in [pub:17].

8.6.1 *Modeling the Learning Problem*

The required preference information is deduced from the annotations already made by the user: Given a tagged object in primary focus all other objects with the same tag are considered as relevant. Assuming that they share some common feature(s), the distance measure of the neighbor index that is queried for the distortion has to be adapted accordingly, such that more relevant objects are amongst the nearest neighbors. This *distortion distance measure* can be adapted independently of the *projection distance measure* which is used by the MDS and left untouched here to not confuse the user by a changing arrangement. The distance constraints for the adaptation can be derived much like for the folk song classification (Section 8.4): A pair of objects, a and b, annotated with the same tag T should be more similar to each other than to any other object c not belonging to T:

$$d(a,b) < d(a,c) \;\wedge\; d(a,b) < d(b,c) \qquad \forall (a,b,c) | a,b \in T \wedge c \notin T \tag{8.19}$$

As for the folk songs, facet weightings with different levels of specificity can be learned depending on the scope of the constraints: individual, per-tag or overall. Here, the problem of learning to tag is modeled as classification task using the maximum margin approach (Section 8.3.3) for a robust classification model that can deal with constraint inconsistencies. Continuity is less important here, because the adaptation has an impact on the position of the objects in the galaxy.

8.6.2 *Observations & Outlook*

The experiments with the photo collection show that the adaptation of the distortion distance measure indeed increases the number of relevant (and still untagged) objects retrieved by the neighbor index. – In *MusicGalaxy*, this would not only be helpful for tagging but, e. g., also for generating playlists by navigating the secondary focus.

However, the initial number of only three facets (in the image tagging task) did not provide enough degrees of freedom for the adaptation algorithm. This problem was solved by decomposing one facet (based on a color histogram) into 64 sub-facets (one per bin) which allowed a finer level of adaptation. At the same time, this increased the risk of overfitting as well – especially for the individual weightings. Similar problems are likely to occur for *MusicGalaxy* which currently also considers only a small number of facets. Therefore, the variety of facets needs to be increased before further experiments with real users are conducted.

8.7 EXPERIMENTAL COMPARISON

This section has been published in [pub:23].

In order to compare the different adaptation approaches covered in this chapter in a fully controlled environment, an experiment has been conducted using the publicly available *Magnatagatune* benchmark dataset [117, url:29]. The experimental setup is explained in Section 8.7.1 including the dataset and its pre-processing, the evaluation methodology and the adaptation algorithms. Results are presented and discussed in Section 8.7.2.

8.7.1 *Experimental Setup*

8.7.1.1 *The Magnatagatune Dataset*

The *Magnatagatune* dataset comprises 25863 clips – each one 29 seconds long – generated from 5405 source MP3s provided by the American independent record label *Magnatune* [url:30] for research purposes. The clips are annotated with a combination of 188 unique tags that have been collected through the *TagATune* game [117, url:46]. Additionally, the dataset contains a detailed analysis of each clip computed using the *EchoNest* API. The features comprise musical events, beats, structure, harmony, and various global attributes such as key, mode, loudness, tempo and time signature.

Most importantly for the purpose of this evaluation, there is also a set of music similarity judgments. This information has been collected by showing a triple of clips and asking the player to choose the most different one. 533 such triples have been presented to multiple players resulting in 7650 similarity judgments.

8.7.1.2 *Features and Facets*

In total, 110 facets are used to describe the distances between the clips in the experiment. An overview with brief explanations is given in Table 14. The facets comprise seven globally extracted features of which two – dancability and energy – are not contained in the original clip analysis information of the dataset but have become available with

a newer version of the *EchoNest* API. Furthermore, the segment-based features describing pitch ("chroma") and timbre have been aggregated (per dimension) resulting in 12-dimensional vectors with the mean and standard deviation values. This has been done according to the approach described in [57] for the same dataset. A detailed description of the extracted features can be found in the documentation of the *EchoNest Track Analyze* API [url:49].

feature	dim	value description	distance measure
key	1	0 to 11 (one of the 12 keys) or −1 (none)	binary (exact match)
mode	1	0 (minor), 1 (major) or −1 (none)	binary (exact match)
loudness	1	overall value in decibel (dB)	absolute difference
tempo	1	in beats per minute (bpm)	absolute difference (taking tempo doubling into account)
time signature	1	3 to 7 ($\frac{3}{4}$ to $\frac{7}{4}$), 1 (complex), or −1 (none)	exact match and $\delta(3,6) = 0.5$
danceability	1	between 0 (low) and 1 (high)	absolute difference
energy	1	between 0 (low) and 1 (high)	absolute difference
pitch mean	12	dimensions correspond to pitch classes	Euclidean distance
pitch std. dev.	12	dimensions correspond to pitch classes	Euclidean distance
timbre mean	12	normalized timbre PCA coefficients	Euclidean distance
timbre std. dev.	12	normalized timbre PCA coefficients	Euclidean distance
tags (99 facets)	1	binary, one facet per tag, very sparse	binary (exact match)

Table 14: Facet definition for the *Magnatagatune* dataset used in the experiment. Top rows: Globally extracted features. Middle rows: Aggregation of features extracted per segment. Bottom row: Manual annotations from *TagATune* game.

The 188 unique tags used in the manual annotations have been preprocessed as follows:

1. Singular and plural forms have been merged.
   e. g., "guitar" and "guitars"
2. Misspellings have been corrected.
   e. g., "harpsicord" → "harpsichord"
3. Semantically identical tags have been combined.
   e. g., "funk" and "funky"
4. Meta-tags have been created for groups of tags that express the same concept.
   e. g., "instrumental" = "instrumental" or "no vocal(s)" or "no voice(s)" or "no singer(s)" or "no singing"
5. Unused tags (w.r.t. the relevant subset of *Magnatagatune*) have been removed.

The resulting 99 tags are interpreted as one (binary) facet each.[5]

8.7.1.3 *Deriving Distance Constraints*

Two distance constraints can be derived from a single judgment that clip c is the most different of a triple (a, b, c), namely $d(a, b) < d(a, c)$ and $d(a, b) < d(b, c)$. However, the resulting set of constraints is inconsistent because there are constraints that contradict each other. This is most likely because the similarity judgments stem from multiple players of the *TagATune* game. Applying the filtering technique described in Section 8.3.4, a constraint graph with 15300 edges of which 1598 are unique is constructed. After the removal of length-2 cycles, 860 unique edges remain (6898 in total). The randomized filtering finally results in a DAG with 674 unique edges (6007 in total). Thus, the filtered consistent set contains 674 constraints of which each is backed by 8.9 judgments on average. In the following, this set is referred to as *all constraints* set.

Even for the consistent *all constraints* set, it is impossible to learn a facet weighting that violates none of the constraints. This is either because the information captured by the facets is insufficient – i. e., players may have considered features in their judgments that are not covered by the features – or the adaptable distance model is too simple. From the classification perspective of Equation 8.7 this means that there is no hyperplane that clearly separates the positive from the negative examples. Or from the QP perspective, the system of equality and inequality constraints (cf. Figure 50) is inconsistent – i. e., has no solution. Therefore, another set – in the following referred to as *selected constraints* – has been constructed by further filtering the *all constraints* set. To this end, the randomized approach of Section 8.3.4, step 3 has been applied again but this time constraints are only added to the set if the resulting QP problem has a solution. The *selected constraints* set obtained this way contains 521 constraints.

At first sight, the two sets of constraints seem to be quite large. After all, which user would like to answer several hundred questions of the form *"Which one of these three objects do you think is the most distinct one from the others?"* However, these distance constraints are in fact only the very atomic pieces of information used to guide the adaptation. As the example applications described earlier in this chapter show, usually multiple such distance constraints are derived from a single action like moving an object (Equation 8.17), correcting a ranking (Equation 8.18) or adding a tag annotation (Equation 8.19).

5 Alternatively, it is possible to combine all annotations into a single facet or define facets for groups of related tags (e. g., all tags related to instrumentation) which significantly reduces the number of facets. However, this would also drastically reduce the size of the *selected constraints* set described in the next section.

8.7.1.4 *Considered Algorithms*

Table 15 lists the considered algorithms and their parameters. They comprise the three algorithms used in the applications described in Abschnitte 8.4 bis 8.6 and furthermore several variants of the QP approach with added slack dimensions that allow the violation of distance constraints (Section 8.3.5.2).

abbreviation	algorithm	parameters
GradDesc	Gradient Descent	50 repetitions with random permutations of training samples, dynamic learning rate
QPmin(Δw)	Quadratic Programming	minimal weight change, no slack
LibLinear	Maximal Margin Classifier (Java *LIBLINEAR* v1.5)	L2-regularized L2-loss SVC, $C = 10^7$, $\epsilon = 10^{-6}$, no bias term
QPmin(ξ)	Quadratic Programming	no primary objective, linear slack, $\kappa = 1$
QPmin(ξ^2)	Quadratic Programming	no primary objective, quadratic slack, $\kappa = 1$
QPmin(Δw+ξ)	Quadratic Programming	minimal weight change, linear slack, $\kappa = 1$
QPmin(Δw+ξ^2)	Quadratic Programming	minimal weight change, quadratic slack, $\kappa = 1$
QPmin(Δw+$10^5\xi^2$)	Quadratic Programming	minimal weight change, quadratic slack, $\kappa = 10^5$

Table 15: Algorithms covered in the comparison. Top: Algorithms used in the applications described in Abschnitte 8.4 bis 8.6. Bottom: Alternative QP problem formulations with added slack dimensions (Section 8.3.5.2).

As the GradDesc learner may get stuck in a local optimum, the computation is repeated up to 50 times if no solution could be found that satisfies all training constraints. Each run uses a different random order of the same training constraints. Finally, the solution which results in the lowest number of constraint violations is chosen.

8.7.1.5 *Evaluation Methodology*

The evaluation aims to answer the following questions:

- How good is the obtained adaptation (in terms of constraint violations)?
- How fast (with how much user effort) can it be learned?
- How stable is the quality of an adaptation if new constraints are added to the set?

The number of violated distance constraints serves as a performance measure for how well the algorithm has adapted to the similarity preferences given some training constraints. All algorithms except QPmin(Δw) that cannot deal with inconsistencies are tested on both sets of constraints described in Section 8.7.1.3. For the *selected constraints* set, a solution satisfying all constraints is expected. Whereas for the *all constraints* set, the behavior of the algorithms under constraints that cannot all be satisfied is tested. As the size difference

between the two sets is 153, the optimal performance value for the *all constraints* set is expected to be close to 150.

For each of the two sets, 100 random permutations of the distance constraints are generated. Each permutation is presented to the adaptation algorithm – one constraint at a time (i. e., stepwise) – until all constraints are used for training. After each step, the number of violated constraints in the whole set is determined. The values are averaged per step over the 100 permutations to reduce ordering effects.

8.7.2 *Results*

All detail plots in this section (Abbildungen 55 bis 57) show all performance values obtained for the 100 random permutations of the constraints as points (in light gray). This gives an impression of the variance between the different runs. Furthermore, the average values are displayed as colored curves. Each diagram combines the results of a single algorithm on both constraint sets – *all constraints* (top, red curve) and *selected constraints* (bottom, blue curve) – as these do not overlap. The two gray dotted horizontal lines indicate the baseline performance value obtained by the uniform facet weighting on *all constraints* (upper line) and the subset of *selected constraints* (lower line). The scaling of all plots is identical for better comparability.

8.7.2.1 *Approaches used in the Applications*

Figure 55 shows the performance of the algorithms used in the applications described in Abschnitte 8.4 bis 8.6 and listed in Table 15 (top rows). The plots for the *selected constraints* set, where a weighting can be found that satisfies all constraints, are almost identical. For LibLinear the mean curve is a bit steeper, indicating slightly better early solutions. However, a little more variance can be observed – especially between 50 and 80 training constraints. Furthermore, it has to be noted that GradDesc and LibLinear converge on a solution that leaves a small number of constraints violated whereas QPmin(Δw) finds a weighting without constraint violations. The GradDesc learner still gets stuck in a local optimum possibly close to the global one. The problem of LibLinear is that it favors a large margin over small constraint violations, e. g., caused by small negative facet weights.

For the larger *all constraints* set, QPmin(Δw) does not return a solution because the derived QP system to be solved is inconsistent. Comparing GradDesc and LibLinear which both can deal with constraint violations, the latter again shows faster convergence but slightly higher variance. Most notably, LibLinear leads to a solution that violates approximately 30 constraints less than GradDesc. The main reason for this is that it trades a few (slightly) violated weight bounds constraints

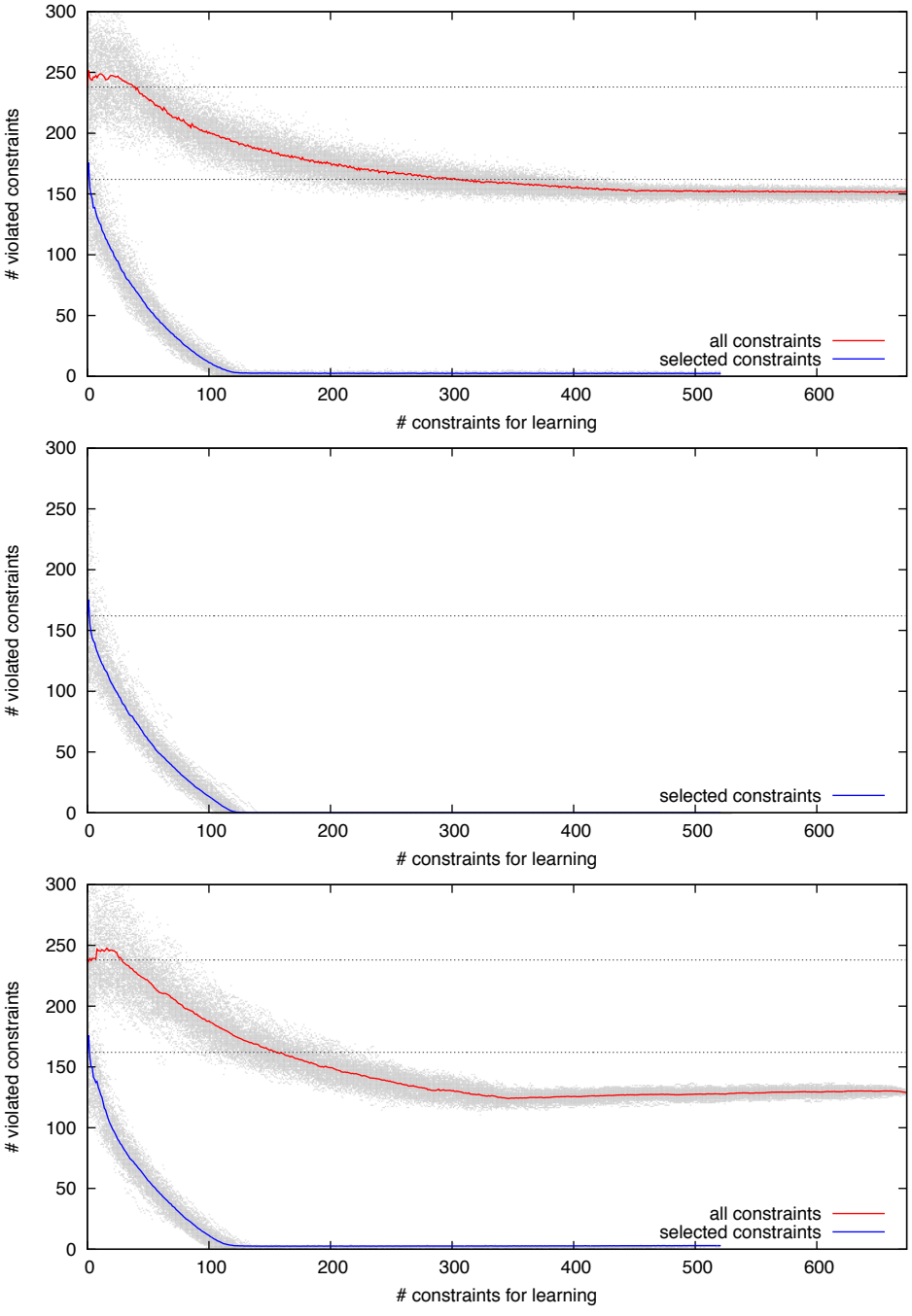


Figure 55: Performance of the algorithms applied in the previously described applications and experiments. Values for 100 random permutations. Top: Gradient Descent (GradDesc). Middle: Quadratic Programming (QPmin(Δw)). Bottom: Maximal Margin Classifier (LibLinear).

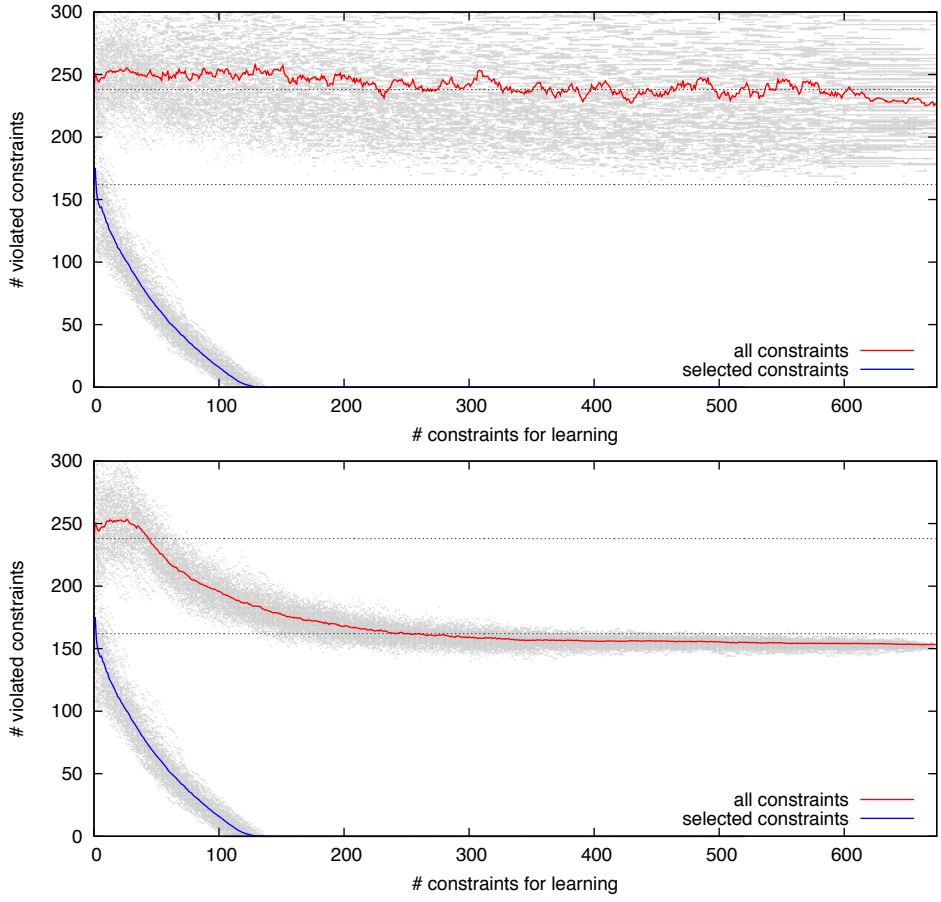


Figure 56: Performance of the Quadratic Programming approaches minimizing only the slack weights without a primary objective function. Values for 100 random permutations. Top: $\mathsf{QPmin}(\xi)$ (linear slack). Bottom: $\mathsf{QPmin}(\xi^2)$ (quadratic slack).

against a larger number of distance constraints that are not violated. This results however in an invalid weighting.

8.7.2.2 *Slack-Only Quadratic Programming Approaches*

Figure 56 compares the performance of the alternative QP approaches that aim to minimize only the slack as described in Section 8.3.5.2 without a primary objective. The plots for the *selected constraints* set look almost identical. They do not differ much from those seen before in Figure 55. Therefore, it can be concluded that both approaches work well if there is a solution that violates no constraints. However, the performance could not look much more different for the *all constraints* set: $\mathsf{QPmin}(\xi)$, i. e., modeling the slack in the linear part of the QP

objective function (and leaving the quadratic part constant), seems not to work at all. There is almost no improvement compared to the baseline. At the same time, the variance increases which is much in contrast to all other approaches. However, QPmin(ξ^2), i. e., modeling the slack in the quadratic part of the objective function, produces a better solution than GradDesc. At the beginning, for up to 30 training constraints, there is no improvement. In fact, both plots look here very much alike. But then the number of violations drops quickly and for 100 training constraints, it is already lower than for GradDesc.

8.7.2.3 *Combined Quadratic Programming Approaches*

Figure 57 shows the performance for the alternative QP approaches that minimize a combination of both, the change of the facet weights and the slack penalty. Here, QPmin($\Delta w+\xi$), the combination of the weight change minimization objective with the linear slack penalty, has the best performance for both sets of constraints. This is surprising considering that QPmin(ξ) does not work at all for the *all constraints* set as seen in the preceding section. Much in contrast, minimizing the quadratic slack penalty works only well without the primary objective (QPmin(ξ^2)). For the combination, QPmin($\Delta w+\xi^2$), there seems to be a conflict between both objectives. This results in an unsatisfactory adaptation for the *selected constraints* set with more than 40 constraint remaining violated.

The QPmin($\Delta w+\xi^2$) plot (Figure 57; middle) for the *all constraints* set is very remarkable. It can be divided into three sections: In the first section up to roughly 440 constraints, there is high variance between the permutations and no significant improvement. Then, however, the values converge and until about 525 constraints, no variance can be observed. This point coincides with the size of the *selected constraints* set which is close to the maximal number of constraints that can be satisfied. Afterwards, in the last section, the number of violated constraints quickly decreases to a final value that is comparable to the other working approaches. This late adaptation suggests that the primary objective (minimizing the weight change) suppresses the minimization of the slack until the last section. Indeed, the facet weights have converged to 1 at the beginning of the second section which explains the performance close to the baseline (uniform facet weights). Only afterwards, the importance of the slack gains the upper hand – most likely because of the high number of slack dimensions caused by the many training constraints in this section. Choosing a high slack weight results in an earlier adaptation as shown for QPmin($\Delta w+10^5\xi^2$) with $\kappa = 10^5$. However, the variance is very high and the performance is still inferior to QPmin($\Delta w+\xi$). Even higher values of κ result in no significant improvement.

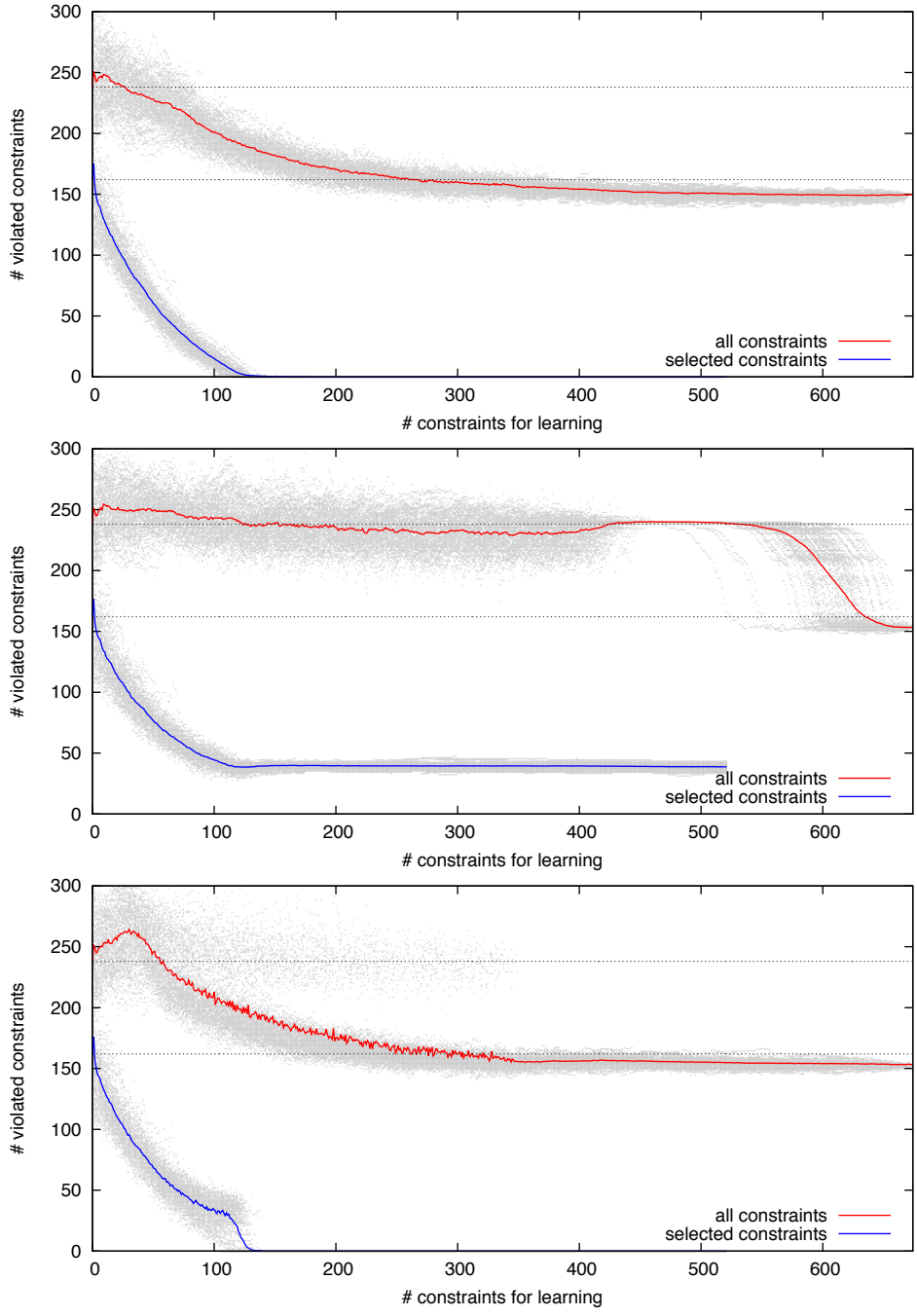


Figure 57: Performance of the Quadratic Programming approaches minimizing a combination of both, the change of the facet weights and the slack penalty. Values for 100 random permutations. Top: $\mathrm{QPmin}(\Delta w + \xi)$ (minimal weight change, linear slack, $\kappa = 1$). Middle: $\mathrm{QPmin}(\Delta w + \xi^2)$ (minimal weight change, quadratic slack, $\kappa = 1$). Bottom: $\mathrm{QPmin}(\Delta w + 10^5 \xi^2)$ (minimal weight change, quadratic slack, $\kappa = 10^5$).

algorithm	selected constraints			all constraints		
(abbreviation)	10	100	521	10	100	674
GradDesc	<0.01	3.34	13.29	<0.01	5.41	8.45
QPmin(Δw)	<0.01	0.02	1.08			
LibLinear	0.13	1.21	2.54	0.38	0.46	1.19
QPmin(ξ)	<0.01	0.06	5.84	<0.01	0.19	19.62
QPmin(ξ^2)	<0.01	0.03	1.15	<0.01	0.05	5.95
QPmin($\Delta w+\xi$)	<0.01	0.06	6.59	<0.01	0.06	27.73
QPmin($\Delta w+\xi^2$)	<0.01	0.02	0.82	<0.01	0.03	3.05
QPmin($\Delta w+10^5\xi^2$)	<0.01	0.02	1.12	<0.01	0.04	4.59

Table 16: Processing times (in seconds) for the adaptation depending on the number of training constraints measured on both constraint sets. Values averaged over 100 repetitions on a consumer notebook (2.4 GHz Intel Core 2 Duo, 4GB RAM). Algorithms that did not produce satisfying adaptations in the evaluation are grayed out.

8.7.2.4 *Processing Time*

Table 16 lists some empirically determined values for the processing time of the different algorithms. For GradDesc, the times refer only to a single repetition. Generally, these measurements can only give an impression of the processing time for the adaptation as no special preparations of the testing system, the runtime environment or the compiler have been made. Especially in interactive settings, a short response time is important. The values are averaged over 100 random permutations and have been measured for 10, 100 and all available training constraints of the two sets. Values for the *all constraints* set are expected to be higher because of the unavoidable constraint violations that occur here. For GradDesc and LibLinear, this is surprisingly not the case. Possibly, finding a solution for the *selected constraints* set is harder for these algorithms. However, it has to be noted that LibLinear is the only approach that in the current implementation of the library interface requires slow hard disk access to read the problem description from a temporary file. The actual processing times for LibLinear are therefore much lower. In the adaption experiment described in Section 8.6 with much larger constraint sets, LibLinear has already shown that its runtime scales well. The QP approaches could run into problems here – especially QPmin($\Delta w+\xi$) which requires even more constraints for the non-negativity of the slack variables. In contrast to the other algorithms, GradDesc, which is rather slow (but also the only algorithm in the evaluation that does not rely on highly optimized library code) could be interrupted during computation and still return a satisfying adaptation.

8.7.2.5 *Overall Comparison*

A direct comparison of all tested approaches is shown in Figure 58. For the *selected constraints* set, all approaches except those using the square slack penalty work almost equally well. I. e., if a solution exists that satisfies all constraints, one is found. Only GradDesc gets stuck a little too early and LibLinear favors the larger margin. Of the approaches with square slack penalty, QPmin($\Delta w+\xi^2$) does not work well leaving roughly 40 unsatisfied constraints and QPmin($\Delta w+10^5\xi^2$) converges too slowly.

For the harder *all constraints* set, LibLinear can be considered as the "disqualified winner" of the competition. It shows the overall quickest convergence requiring fewer steps than the other approaches for the adaptation and returns weightings that violate significantly fewer constraints. However, the latter is only possible because of "cheating" as the weights violate the essential non-negativity constraint (Equation 8.4) and thus cannot be interpreted as intended. Given these results and the good scalability for large problems, an internal modification of *LIBLINEAR* that ensures non-negative weights looks promising. However, this is not a trivial task and beyond the scope of this thesis.

QPmin($\Delta w+\xi$) has the best final performance value for a valid adaptation on *all constraints* which is even slightly below 150. However, its adaptation is a bit slow in the beginning. For the first 70 steps, GradDesc would be a better choice and in the middle section, QPmin(ξ^2) does slightly better. In the end, the performance difference of these three approaches is only very small.

Finally, QPmin(ξ) does not work at all for *all constraints* and QPmin($\Delta w+\xi^2$) converges only in the end which is not acceptable either. These combinations should therefore not be used.

8.8 CONCLUSIONS

In this chapter, a generalized approach has been presented that allows to model and learn individual distance measures for comparing music pieces based on multiple facets that can be weighted. The described technique for adapting distance computation is very generic and can be applied in various contexts, most importantly:

- Only little restriction is put on the underlying distance facets.
- The adaptation model based on a linear combination of distance facets is intuitively understandable to users. It can easily be visualized, e. g., by standard Graphical User Interface (GUI) elements like sliders that would also allow to override automatic adaptations.

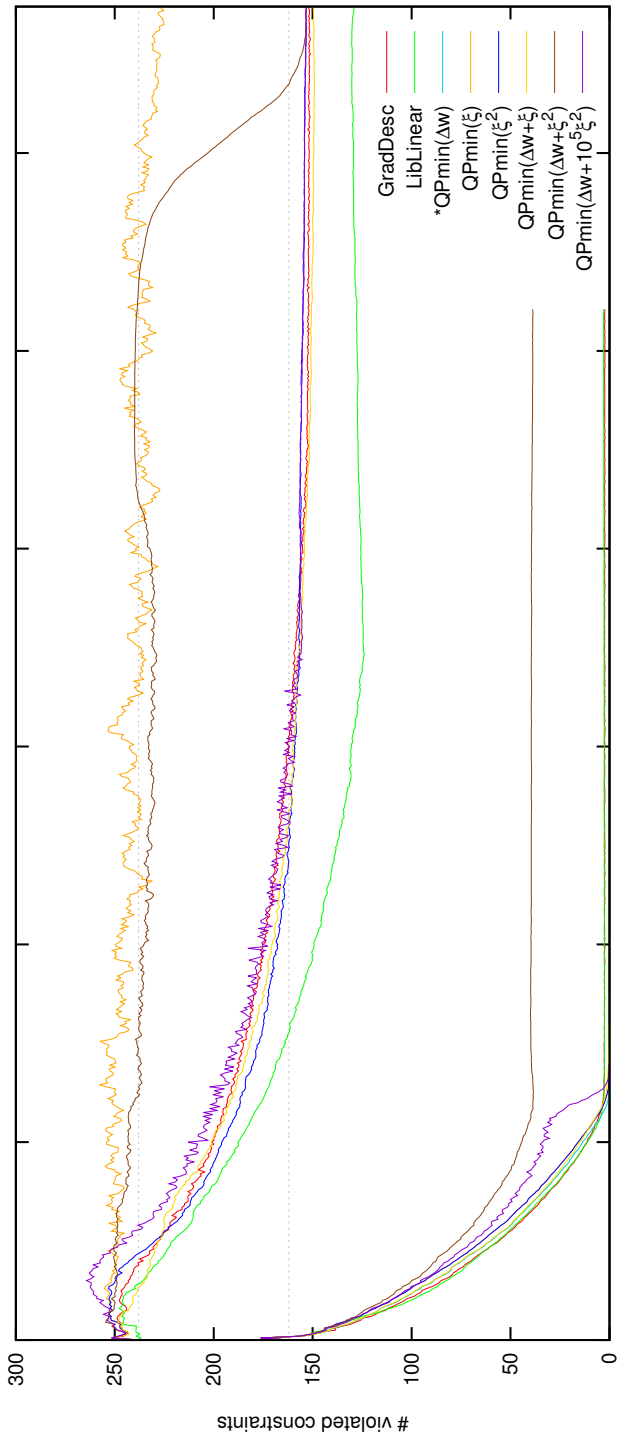


Figure 58: Direct comparison of all approaches tested in the experiment. Average values for 100 random permutations. (*QPmin(Δw) not applicable on *all constraints* set.)

- The adaptation process is formulated as common constrained optimization or classification problem and thus, it is possible to employ a wide range of generic solvers and classifiers with different objectives that fit the application scenario.
- Possibilities for dealing with contradicting and inconsistent constraints have been shown.
- As demonstrated by the presented applications, the distance constraints that guide the adaptation can easily be derived from a wide range of information – either provided by an expert or inferred from the actions of a user in an interactive setting.
- Already having been applied in the image [pub:17] and text retrieval [pub:2] domain, the possible application areas go far beyond the scope of this thesis.

Furthermore, multiple approaches for solving the generic adaptation problem with different objectives have been proposed and evaluated using the public *Magnatagatune* benchmark dataset. This way, it is possible to decide which of the presented approaches is the most suitable in a specific setting or to compare them against novel ones.

Part III

OUTLOOK

The most exciting phrase to hear in science,
the one that heralds new discoveries,
is not "Eureka!" (I found it!) but "That's funny..."
Isaac Asimov

9

BISOCIATIVE MUSIC DISCOVERY

The work described in this chapter is based on a collaboration with Stefan Haun as part of the BISON project and has been published in [pub:24].

Surprising a user with unexpected and fortunate recommendations is a big challenge for recommender systems. Motivated by the concept of bisociations, this chapter proposes ways to create an environment where such serendipitous recommendations become more likely. To this end, the focus-adaptive *SpringLens* visualization technique which has been presented in Chapter 7 is utilized. It has been developed to alleviate the impact of inevitable projection errors caused by dimensionality reduction. This chapter gives an outlook on how the multi-focus distortion technique of the focus-adaptive *SpringLens* can be used beyond its originally intended purpose to adapt the distortions of the visualization in a way that facilitates bisociative music discovery.

9.1 INTRODUCTION

Music recommender systems aim to help users cope with the large amount of music available today and find new interesting music or rediscover once loved pieces users have forgotten about – a task also called "recomindation" [197]. One common problem that many recommender systems face is that their recommendations are often too obvious and thus not particularly useful when it comes to discovering new music. Especially, collaborative filtering approaches are prone to a strong popularity bias [36]. In fact, McNee, Riedl, and Konstan [156] argue that there is too much focus on improving the accuracy of recommender systems. They identify several important aspects of human-recommender interaction of which serendipity is specifically related to the above phenomenon [157]. A serendipitous recommendation is unexpected and fortunate – something that is particularly hard to grasp and evaluate.

As described in Section 7.5, a user study has been conducted to assess the usability and usefulness of the focus-adaptive *SpringLens* visualization technique for the exploration of large multimedia collections. The user-interface was supposed to support the participants by pointing out possibly relevant photos for a seed photo. As it happened, one of the participants encountered a funny incident: While looking for photographs showing a lizard, he selected an image of a *monitor lizard* as seed (i. e., primary focus, cf. Section 7.4.2). To his surprise, the system retrieved an image showing the rock painting of a lizard like shown in Figure 59. Interestingly, rock paintings were

Figure 59: Serendipitous encounter with a rock painting of a lizard when looking for photographs of a lizard.

actually another topic to find photos for and the relevant photos were a lot harder to make out in the collection than the lizards. Bearing in mind that according to Isaac Asimov *"the most exciting phrase to hear in science, the one that heralds new discoveries, is not 'Eureka!' (I found it!) but 'That's funny ...'"*, this phenomenon called for further investigation.

What the participant encountered is called a *bisociation* – a bridging element between the two distinct domains: animals and rock paintings. As it turns out, many scientific discoveries are in some way bisociations [110]. Admittedly, no one expects scientific discoveries from a music recommender application. However, the question persists whether the effect of bisociations can be leveraged to create an environment where serendipitous recommendations become more likely.

The remaining chapter is structured as follows: Section 9.2 points out related work in the field of exploratory music discovery and recommendation. The concept of bisociation is formalized and explained in Section 9.3. Based on this foundation, Section 9.4 describes how the *MusicGalaxy* user-interface can be turned into an environment that supports bisociative music discovery. Finally, Section 9.5 discusses early findings and Section 9.6 concludes the chapter.

9.2 RELATED WORK

As already discussed in Section 7.1, there exists a variety of approaches to music discovery and recommendation that rely on some way of collection exploration. Further related approaches are sketched in the following.

MusicRainbow [182] is an interface to explore music collections at the artist level. Using a traveling salesman algorithm, similar artists are mapped near each other on a circular rainbow where the colors of the rainbow reflect the genres. Audio-based similarity is combined with words extracted from web pages related to the artists. The words are used to label the rainbow and describe the artists.

MusicSun [183] applies a similar concept to discover artists. Recommendations are based on one or more artists that are selected by the user and displayed in the center of a sun. The sun rays (triangles) represent words that describe these seed artists. The size of a ray's base reflects how well the respective word fits the artist and its length is proportional to the number of artists in the collection that can also be described by that word. Selecting a ray, a list of recommended artists is generated. Similarly to the work presented in this paper, users can also adapt the impact of three different aspects of music similarity that are combined.

Musicream [78] facilitates active, flexible, and unexpected encounters with musical pieces by extending the common concept of query by example: Several tubs provide streams of music pieces (visualized as discs) that the user can grab and drop into the playback region of the interface or use as a magnet to filter similar pieces from the streams. The interface also provides enhanced playback functions such as building playlists of playlists or going back to any previous point in the play history.

The *MusicExplorer FX* takes a different approach: Built upon the *EchoNest* API [url:48], it displays a local similarity graph, connecting an artist with the most similar ones. The interface also shows a navigation history containing the previously visited artists. A similar approach is taken by the *Relational Artist Map RAMA*[212] that additionally displays labels as graph overlay. However, both lack a global overview of the whole artist space and users need to specify a seed artist to start with. In contrast to this, the Last.fm *Artist Map* [url:24] displays the whole graph (based on the *Last.fm* API [url:23]). As this results in a lot of clutter caused by crossing edges, it is hard to navigate and explore the graph. Consequently, it is rather suited to map a user's listening preferences.

9.3 THE CONCEPT OF BISOCIATIONS

Two concepts being in some relation to each other are often referred to as being *associated*, i. e., there is some reasonable connection leading from one concept to the other. However, there is no mention about the path itself, especially its length and the steps between both concepts.

While most associations are found between concepts of one domain, there are certain paths which either bridge two different domains or connect concepts by incorporating another domain. Arthur Köstler,

an Austrian publisher, coined the term *bisociation* for these types of associations. In his book *The Act of Creation*, he defines a bisociation as

> *"the perceiving of a situation or idea,* L, *in two self-consistent but habitually incompatible frames of reference,* M_1 *and* M_2. *The event* L, *in which the two intersect, is made to vibrate simultaneously on two different wavelengths, as it were. While this unusual situation lasts,* L *is not merely linked to one associative context but bisociated with two."* ([110], p. 36)

Although describing the idea, this definition is quite informal and unspecific regarding frames of reference, the type of link and the situation or idea. A translation to a formalism is necessary, where in the field of mathematics and computer science the concepts of *graphs* is perfectly capable to describe concepts as vertices and connections between them as edges [111].

In the further elaboration, the frames of reference will be called *domains*, keeping in mind that these domains need not necessarily be clearly defined, but are rather induced by a user perceiving the bisociation. Köstler requires these domains to be self-consistent and habitually incompatible, i. e., there must be some clear distinction between them. However, the difference may result from a situation specific view and is not necessarily related to distinctions made in taxonomies or other common domain definitions.

Köstler sets a very strong emphasis on establishing an environment which supports bisociative acts. While propositions like *"entering a dream-like state to suspend rational thinking"* are not viable for a user interface, the environment can be changed to present unusual setups or provide different views on the same conceptual space. A tool that supports the user at integrating different frames of reference leads him to finding bisociations.

9.4 BISOCIATIVE SPRINGLENS

This section shows how different domains (frames of reference) can be incorporated into the *MusicGalaxy* user interface to establish an environment that supports bisociative music discovery. In the basic *MusicGalaxy* interface described in Chapter 7, the underlying (personalized) music similarity measure is relevant in two stages: First, it is used to compute the distances for the MDS projection and thus has an impact on the arrangement of the tracks in the visualization. Later, during user interaction, it is also used to identify the nearest neighbors to be focused by the secondary fish-eye lenses. The straightforward and originally intended setting is obviously to use the same similarity to identify nearest neighbors as to compute the arrangement of the tracks in the projection. Changing this setting, however, opens up the possibility for bisociative exploration.

The general idea is to use separate domains for projection and distortion. The projection domain is directly visible to the user and contains the displayed tracks connected by neighborhood relations that are implicitly induced between each track and its neighbors in the projection.[1] The other domain is used to identify nearest neighbors for the secondary focus distortion. It will hence be referred to as "secondary domain" in the following. In contrast to the projection domain (which can be considered the "primary domain"), it is not directly visible to the user. A bisociation occurs in this setting, if two tracks are not neighbors in the projection domain (i. e., they are not close to each other in the display) but are connected in the secondary domain. In this case, the secondary focus will highlight this connection by focusing on the bisociated track.

9.4.1 *Orthogonal Similarity Measures*

The simplest way to create such a setting is to use orthogonal similarity measures (i. e., defined on non-overlapping facet sets) for the two domains by adapting the facet weights accordingly. As described in Section 7.4.3, this is already supported by the *MusicGalaxy* user interface. For instance, Figure 60 shows a *SpringLens* setting that only considers the *rhythm* facet for the projection whereas the other facets are relevant to find the nearest neighbors for the secondary lens distortion. The tracks highlighted by the secondary focus are thus very similar to the one in primary focus w.r.t. timbre and dynamics but differ significantly in rhythm (all the more with increasing distance in the projection). Just as well, tracks with similar lyrics that sound very differently (e. g., w.r.t. instrumentation, rhythm or harmonics) could be discovered – provided respective facets.


Video Clip 1

However, the focus-adaptive *SpringLens* visualization could also be used in different applications. To illustrate the possibilities, imagine a user wants to explore a collection of world music as, e. g., addressed by *mHashup* [139, url:31] and the *globalmusic2one* project [url:13, 112]. In such an application, a straightforward way for the arrangement of the tracks would be according to their geographical origin, i. e., mapping the tracks on a common world map. Using this primary domain instantly gives the user an overview of the geographic distribution of the tracks in the collection. With the primary fish-eye lens, the user could magnify a region he is interested in. This would allow to display the local distribution of tracks in more detail and differentiate smaller sub-regions. Note that in this special case, the arrangement of the tracks is perfect in the sense that all distances

1 Note that this is rather an artificial mental model that a user perceives when looking at the projection as no connections are visualized explicitly. Because of possible distortion introduced by a dimensionality reduction, this model may differ to some extent from the one that can directly be derived from the actual distances in the original space.

Figure 60: Using a bisociative *SpringLens* setting to explore a music collection. Top: *MusicGalaxy* visualization (inverted color scheme for print). Bottom right: corresponding lens distortion resulting from (user-controlled) primary focus (red) and (adaptive) secondary lenses (blue). Bottom left: facet weights for the projection and distortion distance measures (values adaptable).

can be displayed distortion-free (except for the neglectible mapping of the earth's surface to a plane) because there is no dimensionality reduction involved. The secondary focus in its original setting would be unnecessary here anyway. Hence, it could be freely used to highlight regions with nearest neighbors w.r.t. other aspects addressed by the secondary domain – e. g., acoustic similarity as a combination of several respective facets. Furthermore, analyzing the interaction with the customer, the system could over time learn which (acoustic) facets (of the secondary domain) are particularly important for the user and personalize the similarity measure for nearest neighbor retrieval accordingly as described in Section 8.6.

9.4.2 *Generalization to Domain Graphs*

The above example uses an orthogonal similarity measure for the secondary domain. This is however only a very special case. Generally, the secondary domain might be any graph that contains at least the tracks as concepts (nodes) and allows to find neighboring tracks by some way of traversing relations between the concepts. An orthogonal

similarity measure as described above induces such a graph: In this case, the graph contains only the tracks as concepts and relations between tracks that are nearest neighbors and finding nearest neighbors for a track means simply returning all directly related tracks. As the analysis of the search strategies described in Section 7.5.2.3 showed, some of the participants of the study intuitively navigated this graph by the secondary focus. An alternative way to construct a sparse neighborhood graph for the secondary domain is to use any (black-box) system that recommends similar tracks for a seed track or even a combination of several such systems.

However, the graph does not need to be confined to tracks. In fact, it may be arbitrarily complex – e. g., containing also artists, releases and respective relations and possibly allowing multiple paths between tracks. For instance, from the freely available data from *MusicBrainz* [url:36], a community-maintained music metadatabase, a large graph can be constructed containing more than 10M tracks, 740K albums, 600K artists and 48K labels.[2] Between these entities, common relationships exist that, e. g., link tracks to artists and albums as well as albums to artists and labels. Apart from this, a large variety of Advanced Relationship Links (ARLs) exists. They are particularly interesting as they go beyond trivial information that could, e. g., be automatically derived from ID3-tags. For instance, information covered by the advanced relationships comprises links from tracks and albums to mastering and recording engineers, producers and studios (in total more than 281K artist-album and 786K artist-recording ARLs), how artists are related with each other (more than 135K ARLs), or which tracks contain samples of others (more than 44K recording-recording ARLs).[3]

The graph is stored in a graph engine built upon the *Neo4j* graph database [url:39] that especially facilitates high-performance relationship traversal. The tracks of the music collection to be explored are matched against the track entities in the *MusicBrainz* graph through their Portable Unique IDentifiers (PUIDs). A PUID is an identifier returned by the proprietary *MusicDNS* audio fingerprinting service provided by *AmpliFIND Music Services* (formerly *MusicIP*) [url:1].

In order to identify nearest neighbors for a track in primary focus, the *MusicBrainz* graph is traversed in breadth-first order collecting paths to other tracks. Graph traversal stops when either the traversal depth or the number of reached track nodes exceeds a predefined threshold. As only the most relevant tracks can be highlighted by the secondary focus, some relevance measure is required to rank the retrieved tracks. Because increasing serendipity is the main objective, the relevance measure should capture how likely a track will be a

2 Figures as of January 2011 when the graph was created.
3 A full list of all advanced relationship types is available at [url:37]

lucky surprise for the user. This is however all but trivial. Possible simple heuristics are:

- Prefer tracks that are projected far away from the primary focus (and thus most likely sound very different).
- Prefer tracks that the user has not listened to a lot or for a long time (and probably is no longer aware of).
- Prefer tracks of a different artist and/or album.

The result of using either heuristic or a combination thereof will most likely surprise the user but at the same time the risk is high that the connection to the primary focus is too far fetched. Therefore, information about how two tracks are connected should be taken into account as well.

In most cases, more than one path to another track will be identified by the breadth-first traversal. These paths somehow need to be judged according to their interestingness. PLATT [198] defines discrete edge distances depending on the type of relationships for a similar graph created on a dataset from the *All Music Guide* [65]. A similar weighting could be applied here. Alternatively, weights could be assigned to common path patterns instead – possibly penalizing longer paths. For instance, some path patterns are straightforward such as *track-artist-track* (same artist) or *track-album-track* (same album) where the latter is more interesting in terms of serendipity because it could be a compilation that also contains tracks of other artists. (Compilations are linked to an artificial "various artists" node.) However, both weighting approaches require empirical tuning of the respective weights. Another option is to count the frequencies of occurring path patterns and boost infrequent and thus remarkable patters which could be interpreted as analogy to the *idf* weights used in text retrieval [210]. Such an approach would favor patterns containing ARLs. Further, some means of aggregating the weights from multiple possible paths is needed. For instance, the maximum, minimum or average could be used. These and possibly more sophisticated methods as currently developed to facilitate bisociations on text collections [217] could further increase the chances of bisociative recommendation from complex domain graphs. However, this has to be studied more thoroughly as the impact of the different heuristics and the values of their respective parameters are currently not fully clear.

9.5 DISCUSSION

This research in the field of bisociative music collection exploration is still in an early stage and clearly leaves several options for elaboration. For instance, it would be possible to extend the domain graph beyond *MusicBrainz* by incorporating information from other sources such as

Last.fm [url:22], *EchoNest* [url:47] or *Myspace* [url:38] (see Section 9.2 for some graphs created from artist-similarity relations that can be obtained from these resources).

The user interface needs to better integrate the graph information – possibly displaying (single) interesting connections. It could also be important to point out why a specific track is highlighted by the secondary focus. Such explanations would make the recommendation more understandable and less ambiguous. Currently, a user can only recognize tracks of the same album (because of the same cover) and to some extent tracks of the same artists (given he can associate the album covers with the respective artists). Looking at the screenshot of *MusicGalaxy* shown in Figure 60, two tracks from the same album can be seen in secondary focus. This is most likely due to some "album effect" (e. g., caused by the production process or common typical characteristics of vocals and instrumentation) captured entirely only by acoustic facets and without knowledge of track-album or track-artist relations. However, a similar result could have been produced by using the *MusicBrainz* graph as secondary domain. There is currently no visual clue to differentiate one from the other.

Furthermore, a deeper analysis of the relationship graph could lead to more sophisticated ways to judge the interestingness of paths to related tracks. In order to personalize recommendations and increase the chance of surprises, additional information from a user profile could be incorporated. Finally, it is necessary to test the proposed approach in another user study. However, it still remains an open question how to objectively judge the quality of recommendations in terms of serendipity.

9.6 SUMMARY

This chapter outlined an approach to increase the chance of serendipitous recommendations in an exploratory music retrieval scenario. Instead of addressing serendipity directly, the related concept of bisociations has been exploited that can be formalized by means of graph theory. To this end, the focus-adaptive *SpringLens* visualization of the *MusicGalaxy* user interface can be utilized beyond its original intended purpose by separating the underlying similarity measures for projection and distortion, introducing an abstract graph model for the latter. This way, it becomes possible to link two distinct reference frames (domains) of the tracks in a music collection with each other and bring together both worlds, similarity- and graph-based approaches for exploration. Such a setting promotes bisociations and lets serendipitous recommendations become more likely. Still, much work needs to be done here, leaving this as a promising direction for future research.

The ability to focus attention on important things is a defining characteristic of intelligence.

"Irrational Exuberance"
Robert J. Shiller

10

GAZE-CONTROLLED ADAPTIVE FOCUS

This chapter describes joint work with Sophie Stellmach who has to be credited in the first place. It has been published as [pub:20] with additional details on the user studies and a thorough discussion of the results.

Like the preceding chapter, this chapter also builds upon the focus-adaptive *SpringLens* visualization technique which has been presented in Chapter 7 and the prototype application *MusicGalaxy* which allows users to explore a collection of music tracks visualized as galaxy with a multi-focus fish-eye lens. This user interface entirely relies on mouse and keyboard for interaction. In particular, the primary focus is controlled with the mouse. However, it would be more natural and intuitive to let the gaze control the focus. After all, eye tracking technology has already been used to evaluate the visualization technique as described in Section 7.5.

Indeed, eye tracking has recently gained more and more attention as a promising input channel. Several researchers have indicated the high potential of gaze-based interaction for efficient pointing tasks (e. g., [3, 101, 253]) as gaze often precedes a manual action. In this regard, fish-eye lenses are one solution to locally emphasize items of interest according to the user's visual attention while still maintaining context information. This is also beneficial for gaze-based selections as, e. g., reported in [3, 85, 159], because enlarged target items are much easier to hit via gaze. Solely gaze-controlled applications, however, face the so-called "*Midas Touch*" problem [102] named after King Midas from Greek mythology who turned everything he touched into gold – whether he wanted or not. Here, this refers to the problem of unintentionally issuing an action via gaze which poses one of the major challenges for gaze-based interaction [100, 101]. One approach to overcome this problem is a suitable combination with additional input modalities as, e. g., described by Castellina and Corno [35].

This chapter describes the development of a user interface for gaze-supported exploration of large media collections based on *MusicGalaxy* in a user-centered design approach. The actual target scenario is the interaction with a large remote display as illustrated in Figure 61. In order to avoid the *Midas Touch* problem and otherwise necessary dwell-time activations that slow down interaction, combinations of gaze with (1) a keyboard and (2) a smartphone are investigated. Here, the keyboard can be seen as a representative modality for other input devices that have distinct physical buttons, such as gaming controllers or remote controls for television sets. Smartphones, on the other hand, provide particular interesting features for interacting with multimedia content, such as accelerometers for tilt and throw gestures and touch-

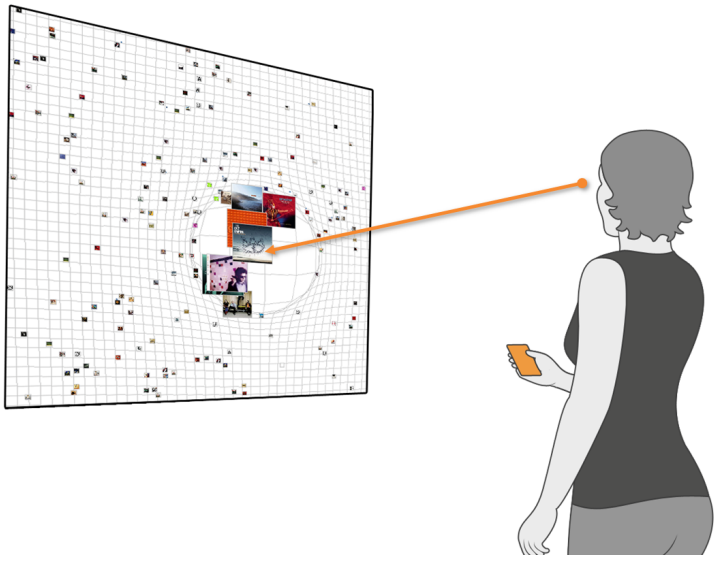


Figure 61: Illustration from [pub:20] of the envisioned setting for gaze-supported interaction with a large remote display.

sensitive screens [46]. In the following, such devices are referred to as *touch-and-tilt* devices.

The remaining chapter is structured as follows: Section 10.1 discusses how gaze has been applied for the interaction with fish-eye lenses in related work. Section 10.2 describes the conceptual user-centered design process and the resulting conflict-free interaction techniques for keyboard & gaze as well as touch-and-tilt & gaze. Details on the actual implementation of the gaze-supported exploratory retrieval system are given in Section 10.3. Section 10.4 summarizes the findings from a formative user study and points out directions for further improvement of the gaze-supported interaction techniques. Finally, Section 10.5 concludes this chapter.

10.1 RELATED WORK

Miniotas, Špakov, and MacKenzie [159] and Ashmore, Duchowski, and Shoemaker [3] argue that eye pointing speed and accuracy can be improved by target expansions. For this purpose, Ashmore, Duchowski, and Shoemaker [3] describe gaze-based fish-eye lenses to locally magnify the display at the *point-of-regard* which allows preserving the peripheral resolution. Another approach proposed by Fono and Vertegaal [68] is to decrease the size of peripheral windows (*minification*) to preserve the focal window at the original resolution.

Ashmore, Duchowski, and Shoemaker [3] point out that hiding the fish-eye lens during visual search helps the user to get a better overview before making a selection. In addition, they also claim that a localized target expansion has the advantage of maintaining detail and

context. MINIOTAS, ŠPAKOV, and MACKENZIE [159], however, express that the benefits of *dynamic* target expansions are arguable due to inaccurate and jittering eye movements. As they point out themselves, this can be compensated by specialized algorithms to stabilize the eye cursor [159, 254].

COCKBURN, KARLSON, and BEDERSON [41] provide a comprehensive review about focus-and-context techniques including fish-eye views. As an example, ASHMORE, DUCHOWSKI, and SHOEMAKER [3] use an underlying elastic mesh for the fish-eye deformations with a flat lens top (with a constant zooming level). They use a single lens with a gaze dwell-based activation. SHOEMAKER and GUTWIN [224] describe fish-eye lenses for multi-point interactions, however, they only use single-mouse input. Interestingly, they apply a dwell-time (via mouse) to trigger the fish-eye lens instead of using an additional button. Therefore, this approach could be of interest for the adaptation to a gaze-only interaction.

Facilitating a touch-and-tilt device for the gaze-supported exploration of large media collections on a remote display has not been investigated so far. However, several works present gaze-supported interaction with large displays for target selections [17] or, for example, in combination with freehand pointing [241] and hand gestures [250].

In a nutshell, gaze-controlled fish-eye lenses have not been combined with additional input devices yet. Also, they have not been used for the exploration of large media collections. However, previous work provides a good foundation and leads on what to consider for gaze-supported fish-eye lenses, such as hiding the lens if not explicitly required [3, 224].

10.2 DESIGN OF GAZE-SUPPORTED INTERACTIONS

As described in Section 7.4, the *MusicGalaxy* prototype supports various common interaction tasks for the exploration of information spaces [223], such as *overview*, *zoom + pan*, and *details on demand*. The interaction tasks, which are in the following further investigated, are listed in Table 17 with the originally implemented functionality mappings shown in the column "keyboard and mouse". Additional actions, e. g., to toggle filter modes, are not covered here as the focus lies mainly on the lens interaction and on panning and zooming in the visualization, as these are crucial tasks for various application contexts. Some of the considered tasks are mapped to different alternative input variants. For instance, panning can be performed by either dragging the workspace with a mouse or by using the cursor buttons on the keyboard. This gives users a clear and non-conflicting variety to choose the technique that best fits their individual interaction style.

Aiming for a user-centered development of the gaze-supported interactions, interviews were conducted with several potential users

Table 17: Main interaction tasks available in *GazeGalaxy* and possible functionality mappings to different multimodal input combinations. ("keyboard & mouse" corresponds to the original control setting in *MusicGalaxy*.)

change...	**keyboard & mouse**	**keyboard & eye tracker**	**touch-and-tilt device & eye tracker**
lens position	right-click + drag	look + hold key	look + touch
lens magnification	right-click + mouse-wheel	press keys (e. g., 8 and 2)	touch slide gesture
pan	cursor keys or left-click + drag	cursor keys or look at screen borders	relative panning on touchscreen or look at screen borders
zoom	mouse-wheel or press +/− keys	look + press +/− keys	look + touch + tilt
thumbnail size	press PageUp/PageDown keys	press PageUp/PageDown keys	touch slide gesture (+ mode switch)

already at an early stage of the design process – similar to the procedure described by NIELSEN et al. [167] for the development of natural interaction interfaces. This helped to find out how users would spontaneously use eye gaze with either a keyboard or a touch-and-tilt device to interact with the application. Based on the received user feedback, individual interaction sets have been elaborated that are free of ambiguous mappings (i. e., conflict-free): one set for condition (1) *keyboard and gaze* (Section 10.2.1) and one for condition (2) *touch-and-tilt and gaze* (Section 10.2.2). In general, the interaction technique that was most frequently mentioned for a certain task was selected (given that this technique has not already been assigned to another task). This approach worked well except for condition (2) as no clear overall user preferences could be identified for panning and zooming. An overview of the elaborated interaction sets is listed in Table 17. In the following, the individual sets are explained in more detail.

10.2.1 *Keyboard & Gaze*

Here, the gaze is used to indicate where to position the fish-eye lens and where to zoom in. These actions are, however, only carried out, if an additional key is pressed on the keyboard (e. g., pressing the PageUp key to zoom in or holding the Ctrl key for lens positioning). The other tasks are mapped to different buttons on the keyboard as listed in Table 17.

10.2.2 *Touch-and-Tilt & Gaze*

The interviews revealed that holding the touch-and-tilt device in one hand and ideally only using the thumb to interact is preferred to using multi-touch input such as a pinch gesture. Consequently, the first interface prototype of the touch-and-tilt device relies entirely on single-touch gestures as illustrated by Figure 62. Furthermore, in

order to distinguish commands from unintended tilting or gaze-input (the *Midas Touch* problem), a certain area on the touchscreen has to be triggered additionally to issue a command. Finally, the need to shift the visual attention between mobile and remote display should be kept to a minimum. Thus, regions on the touchscreen need to be large enough and arranged in a way that allows *blind interaction* – i. e., interacting without having to look at the mobile screen.

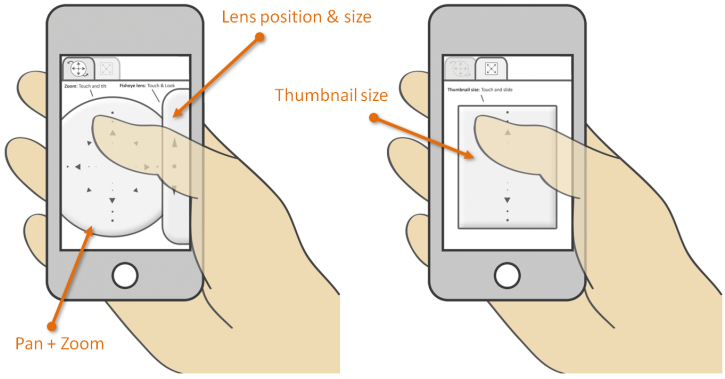


Figure 62: Illustration from [pub:20] of the user interface prototype for the touch-and-tilt device.

The interface prototype for the touch-and-tilt device shown in Figure 62 distinguishes three different modes: (a) pan+zoom, (b) fish-eye lens, and (c) thumbnail size. The interface uses two tabs which can be switched at the top of the touchscreen. Modes (a) and (b) are combined on one screen whereas the thumbnail size can be altered on an additional screen. This was motivated by the idea to integrate new tasks (e. g., for filtering the displayed content) in the future that would hardly fit on one single screen while still providing the possibility for a blind interaction.

The first screen is divided into two active areas: a large region for the pan+zoom mode on the left and a smaller area for the lens mode on the right. As soon as one of the two areas is touched, the corresponding mode is activated. Thus, no actions (whether by gaze or tilt) will be performed if there is no touch event. The fish-eye lens is positioned by looking at a location on the remote display while putting a finger on the lens area at the right of the first screen. If the finger slides up from the initial touch position, the lens size will increase (and vice versa). The pan+zoom mode is activated by touching the pan area. Once this mode is active, panning can either be performed by looking at rectangular active border regions of the remote display (as described in [1]) or by panning gestures on the touchscreen (as described in [45]). While touching the pan area, the user can also zoom by tilting the device forward and backward as also proposed by Dachselt and Buchholz [45]. For this, the orientation of the mobile device when activating the pan+zoom mode is used as a starting position relying on relative positioning data instead of

defining absolute positions for how to hold the device. DACHSELT and BUCHHOLZ [45] use absolute positions which may cause problems due to differing physical constraints (i. e., some people cannot bend their wrists as much as others). The second screen on the mobile device can be reached by touching the second tab at the top. Here, the thumbnail size can be altered by performing a slide gesture – i. e., moving the finger up results in larger and down in smaller thumbnail sizes.

Both elaborated interaction sets – for keyboard & gaze as well as touch-and-tilt & gaze – do not require any gaze dwell-time activations and thus should allow for a more fluent and quick gaze-based interaction. Most importantly, the *Midas Touch* problem is addressed by accompanying gaze-based and tilt interactions with an additional explicit action such as pressing a button or touching the mobile screen.

10.3 PROTOTYPE IMPLEMENTATION

Based on the elaborated interaction sets, a prototype system has been implemented. Figure 63 illustrates the overall system setup. The *GazeGalaxy* application is a modified version of the *MusicGalaxy* prototype described in Chapter 7 that supports additional input modalities: Besides ordinary control input from mouse and keyboard, it also allows control by touch-and-tilt devices and gaze via an additional device communication handler. The communication with connected devices is handled by a Virtual Reality Peripheral Network (VRPN) interface that has been extended to support a variety of input devices. In particular, the following device setup is used:

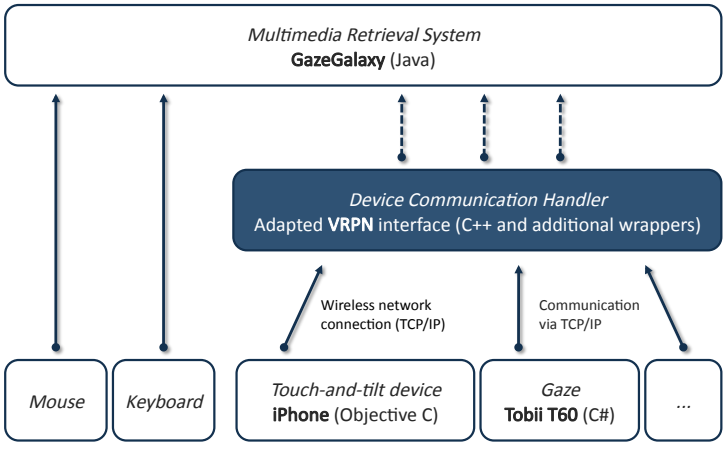


Figure 63: Illustration from [pub:20] of the overall system setup for the gaze-supported multimodal interaction with *GazeGalaxy*.

Gaze data is gathered with a Tobii T60 table-mounted eye tracker which is connected through the local area network. It can determine screen gaze positions at a frame rate of 60 Hz based on corneal-reflections that are identified in streamed video data. For stabilizing

the gaze cursor, the *speed reduction* technique described by ZHANG, REN, and ZHA [254] is applied. This means that raw gaze data are partially integrated with the previous gaze position. In addition, a minimal threshold distance (30 pixels) has to be traveled via gaze to assign the current value as new gaze cursor position. For the gaze panning, rectangular pan regions extending to 100 pixels at each screen border are defined (at a 1280x1024 screen resolution which is the maximum resolution supported by the eye tracker).

An iPod Touch (2nd generation) is used as touch-and-tilt device (but the system could easily be adapted to other multi-touch smartphone devices as well). This device allows multi-touch interaction on a mobile screen and provides a three-axis accelerometer for tilting. The graphical user interface for the iPod has been designed according to the screen prototype illustrated in Figure 62 on page 177. Communication is established via a wireless network.

10.4 DISCUSSION

In order to obtain first indications on the usability of the elaborated prototype for the combination of gaze and touch-and-tilt input, a formative qualitative user study with 6 participants was conducted. During the study, the eye tracker was positioned on an elevated rack as shown in Figure 64 so that the participants could comfortably stand in front of it. This allowed them a better feeling for the remote interaction with a distant display and provided insights on how users would hold the mobile device in such a situation.

Figure 64: Photograph from [pub:20] of a participant standing in front of the Tobii T60 eye tracker to interact via gaze and an iPod Touch with *GazeGalaxy*.

In general, the combination of gaze, touch, and tilt input for interacting with a remote display was perceived as very promising. At first sight, such a combination appears to be a step backwards with

respect to intuitive interaction design. Although using gaze for pointing is considered very intuitive, having to use another input channel simultaneously increases the effort and complexity of the interaction. However, as participants from both studies reported, this is accepted, because this allows for a more relaxed gaze-based interaction for the following reasons:

- The *Midas Touch* problem is avoided as users can indicate via an additional input channel whether an action is really intended.
- For the same reason, there is no need for dwell-time activations which otherwise would slow down the interaction.
- The different input modalities complement well for supporting multiple tasks simultaneously (such as panning and zooming), which is difficult for gaze-only interaction.

The importance of a well-designed system feedback was confirmed as users want to be assured that the system understood their intentions correctly and that the intended mode has been activated. Visual feedback about the current mode could, for example, be indicated by adapting the cursor's shape and color as done by Istance et al. [100] for different gaze interaction conditions. Feedback also plays an important role for identifying tracking problems (either of the gaze or touch and tilt data) – e. g., if the connection to the devices is temporarily lost. At the current stage, the *GazeGalaxy* application's only direct feedback is the focus visualization by the fish-eye lens. Further possibilities need to be investigated that also incorporate haptic feedback (e. g., vibrations) and auditory feedback (e. g., beep sounds) for not overloading the visual input channel.

Recalling the original exploratory search scenario, the combination of a fish-eye-based visualization with gaze input for focus control indeed seems to be very promising according to received user feedback (which confirms findings from Ashmore, Duchowski, and Shoemaker [3]). The *GazeGalaxy* application could clearly be further elaborated – especially in terms of additional gaze-contingent visualizations [62] to emphasize objects of (visual) interest. Furthermore, the possibilities for blind interaction with the system should be extended: Here, using the orientation of the touch-and-tilt device (i. e., vertical or horizontal layout) as mode switch is an interesting alternative option to the tabs interface.

For the near future, porting the system to a large display as shown in Figure 61 on page 174 is pursued using a mobile eye tracker. Finally, the implemented novel interaction techniques in *GazeGalaxy* need to be compared against other input combinations in terms of retrieval performance and interaction efficiency for the originally intended setting (some exploratory task).

10.5 SUMMARY

This chapter described how the focus-adaptive visualization of the *MusicGalaxy* prototype presented in Chapter 7 can be linked to the actual gaze focus of the user through eye tracking technology. In order to avoid the *Midas Touch* problem, gaze input was combined with additional input modalities: (1) a keyboard and (2) a mobile touch-and-tilt device (smartphone). The integration of user feedback at an early stage of the user-centered design process allowed for the development of intuitive and natural gaze-supported interaction techniques. Based on user-elicited interaction techniques, an extended multimedia retrieval system, *GazeGalaxy*, has been developed, which can be controlled via gaze and touch-and-tilt input to explore large media collections. While gaze acts as a pointing modality in this application, the touch and tilt actions complement the interaction for a multi-modal interaction. First user impressions on the implemented interaction techniques were gathered and discussed. Results indicate that gaze input may indeed serve as a natural input channel and that using the gaze positions to control a fish-eye lens is considered intuitive as long as certain fundamental design considerations are taken into account: Firstly, gaze data are inherently inaccurate and thus interaction should not rely on precise positions. Secondly, users should be able to confirm actions with additional explicit commands to prevent unintentional actions.

There is a theory which states that if ever anyone discovers exactly what the Universe is for and why it is here, it will instantly disappear and be replaced by something even more bizarre and inexplicable. There is another theory which states that this has already happened.

"The Hitchhiker's Guide to the Galaxy"
Douglas Adams

11 CONCLUDING REMARKS

In this thesis, various adaptive approaches have been proposed and discussed that all – in one way or another – contribute a part for accomplishment of the overall main goal: a user-centered organization of music collections. The work is summarized in Section 11.1 and Section 11.2 points out the main contributions. But while arguably significant steps have been taken, the problem is still far from being solved. Section 11.3 discusses limitations of the proposed approaches and points out directions of future research.

11.1 SUMMARY

Part i served as a foundation for the work of this thesis: A general introduction of the MIR research field was given in Chapter 2. In particular, this overview pointed out the inherent challenges and tasks for dealing with music information. Especially, the multi-faceted challenge (Section 2.1.2) and the multi-disciplinarity challenge (Section 2.1.4) underlined the need for adaptive approaches in this field. Furthermore, common MIR approaches were explained as well as a model of the general retrieval process (Section 2.2.1). This model served as a contextual frame for the different approaches described in later chapters. Chapter 3 provided a systematic overview of the state of the art in adaptive music retrieval. In order to categorize the covered approaches, a pragmatic definition and model of an adaptive system was elaborated in Section 3.2.1. The overview furthermore identified promising subjects of research addressed in later chapters such as user-adaptive genres (Chapter 6) and adaptive presentation (Chapter 7). Finally, the fundamental techniques that were applied in the context of this thesis were explained in Chapter 4 as a prerequisite for a deeper understanding of the approaches presented later.

The main work of this thesis was covered by Part ii. In total, four different aspects of the general retrieval process were addressed. Table 18 gives an overview of the presented approaches categorized in the same way as the state of the art (cf. Table 4 on page 46). In Chapter 5, two approaches developed by supervised diploma students demonstrated how adaptive techniques can be incorporated into the feature extraction process in order to increase the robustness and quality of the extracted information.

Table 18: Overview of the approaches presented in this thesis grouped by primary application area (cf. Figure 3).

technique (adaptation logic)	application area (section)	references	source for adaptation (context information)	target of adaptation	guidance of adaptation
Feature Extraction					
noise profiling	melody extraction (5.1)	[stud:1, pub:1]	(approximated) karaoke signal (sliding window)	noise profile	internal objective function
histogram smoothing	chord recognition (5.2)	[stud:2, pub:4]	data distribution (sliding window)	a priori chord probabilities	internal objective function
Classification					
log analysis (proposed)	genre classification (6)	[stud:3, pub:5, pub:6, pub:9]	logged data	idiosyncratic genres (classification categories)	internal objective function
Structuring					
GSOM	collection structuring & exploration (8.5)	[pub:2, pub:3, pub:6, pub:10, pub:21, pub:22, 172]	data	structure (flat)	internal objective function
Similarity					
QP	collection structuring & exploration (8.5)	[pub:2, pub:10, pub:22, pub:23]	user interaction (re-arranging & re-ranking)	distance facet weights	derived constraints
QP (soft constraints)		[pub:23]		distance facet weights	derived constraints
gradient descent	ranking & classification (8.4)	[pub:7, pub:8, pub:22, pub:23]	expert classification	distance facet weights	derived constraints
linear SVM	classification (8.6)	[pub:17, pub:22, pub:23]	user interaction (tagging)	distance facet weights	derived constraints
*	*Magnatagatune* (8.7)	[pub:23]	dissimilarity votes (triples)	distance facet weights	derived constraints
Presentation					
neighbor search	exploratory search (7)	[stud:4, pub:11, pub:12, pub:13, pub:14, pub:16, pub:18, pub:19]	user focus (primary lens)	*SpringLens* distortion (secondary lenses)	internal objective function
graph traversal	music discovery & recommendation (9)	[pub:24]	user focus (primary lens)	*SpringLens* distortion (secondary lenses)	internal objective function
gaze analysis	exploratory search (10)	[pub:20]	eye tracking data	*SpringLens* distortion	internal objective function

Next, Chapter 6 investigated the idea of learning idiosyncratic genres adapted to the user from automatically recorded listening context information. Such genres would make the results of genre classification more meaningful as users can directly relate to them. Hence, they would be well-suited for the personalized organization of music collections. Several possibilities for automatically gathering listening context information and listening habits were identified in a pilot study. However, a subsequent survey on the acceptance of such logging techniques revealed strong privacy concerns of the potential users. From this, it could be concluded as a general guideline for designing future personalized MIR applications that users need to be in control of both, the recorded information about their listening habits as well as whether and how it is used for adaptation.

Chapter 7 turned toward a common problem of similarity-based structuring techniques that apply some form of dimensionality reduction to generate an overview map (projection) where neighboring tracks should be similar – a very popular approach for collection organization and exploration. Such visualizations suffer from inevitable projection errors such that some tracks will appear closer than they actually are and some distant tracks may in fact be neighbors in the original feature space. Addressing this problem, the focus-adaptive *SpringLens* technique was elaborated and evaluated. It can be applied on top of such a visualization to temporarily alleviate possible projection errors depending on the user's current region of interest. Based on this adaptive visualization technique, the *MusicGalaxy* prototype – a user interface for music collection exploration – was developed in a user-centered design process. *MusicGalaxy* also features an adaptable multi-facet model of music similarity and makes use of special data structures which facilitate real-time updates of the "galaxy" map for interactive collection exploration.

Chapter 8 focused on the underlying similarity model and described a general approach to automatically adapt it. Similarity was modeled as a simple weighted linear combination of (objective) facet distances allowing users to easily understand and manually adapt it if needed. This way, it was also possible to formalize the adaptation process as common constraints optimization or binary classification problem driven by relative distance constraints which make up the context model. Three applications demonstrated how such constraints can be derived in real-world interactive scenarios: Learning suitable similarity measures for folk song classification from expert annotations, organizing and exploring the work of the Beatles with the *BeatlesExplorer* user interface by re-arranging songs and correcting rankings, and tagging photographs with a modified version of *MusicGalaxy*. Furthermore, several approaches for the adaptation logic were proposed with differing objectives to fit the application scenario. Their performance was compared in an experiment using the public *Magnatagatune* dataset.

Finally, Part iii gave an outlook for future work based on the focus-adaptive *SpringLens* and *MusicGalaxy*. Chapter 9 described ongoing work on bisociative music discovery using *MusicGalaxy*. Here, the *SpringLens* was used to combine two different views on a music collection, making it possible to unite the similarity-based galaxy map with graph structures that control the *SpringLens* distortion. This way, two popular approaches for collection exploration were brought together which created an environment where serendipitous discoveries become more likely when browsing a music collection. Chapter 10 investigated ways to control the *SpringLens* focus through the gaze by means of an eye tracker. Based on user-elicited interaction techniques, the *GazaGalaxy* application was created – a modified version of *MusicGalaxy* that can be controlled via gaze and touch-and-tilt input to explore large media collections. While gaze acts as a pointing modality in this application, the touch and tilt actions complement the interaction for a multi-modal interaction.

11.2 CONTRIBUTIONS

In the following, the *major* contributions of this thesis are summarized:

1. **A systematic overview on the state of the art in adaptive music retrieval (cf. Task 1):** The overview given in Chapter 3 is the first attempt to systematically categorize different approaches from a wide range of application areas in MIR that share the common aspect of adaptivity. It is unique in its extent and focus. Although this survey is most likely far from being complete, the author hopes that it increases the awareness and promotes the further use and development of adaptive techniques in this field as he believes therein lies one key to success of future MIR systems.

2. **Pioneering work on automatic listening context logging for emerging idiosyncratic genres (cf. Task 2):** The still early work on listening context logging presented in Chapter 6 promotes the idea of using user-specific idiosyncratic genres in place of generic categories for more meaningful classification results by gathering evidence for its usefulness from various related surveys and user studies. The findings from the pilot study and the survey provide a valuable guideline for the actual implementation of context logging applications as well as context-adaptive systems that use such information.

3. **A general approach for adaptive multi-facet music similarity (cf. Task 3):** Chapter 8 describes the first generalized framework containing all required building blocks for incorporating adaptive music similarity into MIR applications. The similarity model at its core is intuitively understandable and thus also supports

manual adaptation – much in contrast to other works in this field that rely on more complex models. The learning process is formulated as common constraint optimization problem or as dual binary classification task. This allows a wide range of optimization approaches and classifiers to be used as adaptation logic with different objectives to fit the specific application scenario. Furthermore, a method for experimental comparison using the *Magnatagatune* dataset is described. Generic relative distance constraints are the atomic pieces of information of the context model that guides the adaptation process. Example applications demonstrate how such information can be derived from user actions in various interactive settings.

4. **The focus-adaptive *SpringLens* (cf. Task 4):** This visualization technique takes the original *SpringLens* distortion approach described by GERMER et al. [73] onto a new level by adding adaptivity and transferring it to an interactive setting. Its applications are versatile as demonstrated in various chapters of this thesis: It can be used to alleviate common projection errors caused by dimensionality reduction in similarity-based map visualizations as described in Chapter 7. Here, the distorted neighborhood of the region in focus is highlighted in secondary focus lenses of the *SpringLens*. Furthermore, it allows to overlay two different views on a collection. Here, the primary view is a similarity-based projection. The secondary view, which in this case controls the adaptation of the *SpringLens* distortion, can be based on a different (possibly orthogonal) similarity space or a graph representation of the collection. Bringing the two views together, this allows to create auspicious settings for the exploration of information spaces as described in Chapter 9. Finally, Chapter 10 demonstrates the applicability of the focus-adaptive *SpringLens* in gaze-supported interaction scenarios. While these diverse applications have proven the usefulness of this interactive visualization technique, its full potential is not yet fully investigated as pointed out in the next section.

5. **Two prototype applications as demonstrators of the proposed adaptive techniques (cf. Task 3 and Task 4):** The *BeatlesExplorer* (Section 8.5) and *MusicGalaxy* (Section 7.4) – together with its variant for photo collections (Section 7.5.1) and gaze-supported interaction (Section 10.3) – exemplify how the adaptive techniques can be employed in actual applications. They also prove that the techniques are fit to be used in real-world interactive scenarios. Especially, *MusicGalaxy*, which has already gone through multiple design iterations, is almost a mature application. With moderate effort in software development, it can be turned into an end-user application.

These contributions are not only relevant for the MIR research community. Especially, the second and third points address general problems and thus are of interest for a much wider audience.

11.3 DIRECTIONS FOR FUTURE RESEARCH

Two very promising directions for future research have already been outlined in Chapter 9 and Chapter 10. Apart from this, several other open questions and challenges remain which have not been addressed in this thesis.

1. **How can long-term adaptations be supported?** Building adaptive MIR systems that ideally could be life-long companions and continuously learn from the user raises new problems: For instance, the similarity adaptation approaches described in Chapter 8 assume that the user's notion of similarity is somewhat constant such that the system can gradually adapt towards it. While this might be a reasonable assumption for short time spans, it is questionable whether it still applies in the long run. The user's preferences might change over time and thus, distance constraints gathered earlier may no longer be appropriate. This could be taken into account by introducing importance weights for the constraints which decay over time but also may be confirmed by newer evidence. An adaptation algorithm should then prefer constraints with higher importance if not all constraints can be satisfied. While this would make the algorithm more complex, the similarity model would be as simple as before. Still, users could optionally be enabled to manually alter the constraint importance weights if needed. The evaluation of such approaches would be more challenging, too. Some performance measures might be obtained through simulation as done in this thesis. However, it is the user's satisfaction which should be paramount – and this is hard to measure in the long run.

2. **More complex models of similarity:** The similarity model described in Chapter 8 is arguably a very simple one. Facet distances are aggregated by a weighted linear sum where the weights can be adapted. This has for instance the advantage that it can easily be understood and manually adapted if needed – e. g., through a simple slider interface. However, the model may be too simple for some use cases. The potentially biggest problem is the implicit assumption that the facets are not correlated. More complex models like the Mahalanobis distance take correlations into account but are much harder to comprehend and by far not as easily adapted. So the question is whether a similarity/distance model can be found somewhere in between

which considers facet correlations but ideally still needs only the adaptation of a single weight per facet.

3. **Neighbor-indexes for adaptable similarity:** In Section 7.3.2.2, a scalability issue has been raised that concerns the retrieval of nearest neighbors by means of appropriate index structures in multi-facet similarity spaces with changing facet weights. As the facet weights are not known when the index is built, only the facet distances can be considered for indexing. Furthermore, each facet may require a different indexing method depending on the respective distance measure and the nature of the underlying features. This poses a great challenge for research in the field of multimedia indexing. Some ideas for possible approaches have already been given in Section 7.3.2.2.

4. **Complex focus-adaptive *SpringLens* shapes:** Currently, the primary and secondary focus of the adaptive *SpringLens* visualization consist of a set of round and discrete fish-eye lenses. However, more complex distortions are supported as well by the underlying *SpringLens* mesh. For instance, each focus lens could adapt its shape according to regions in the visualization as illustrated by Figure 65 for the original *SpringLens* distorting a map of Europe.

Figure 65: Illustration from [73] of a *SpringLens* distortion using data-driven lens shapes. Italy and Great Britain are enlarged, while Spain and Scandinavia are shrunk. The region-cursor is currently located above Scandinavia.

Furthermore, instead of the discrete lens focus points currently used, the *SpringLens* distortion could, for instance, be controlled by a (continuous) *heat map* [246]. Such a map could be generated w.r.t. properties of the visualized data. Alternatively, it may also stem from gaze data collected with an eye tracker, which is in

fact a very popular application of heat maps as, e. g., described by Wooding [249]. As an example, Figure 66 shows a heat map computed from gaze data for a screenshot of *MusicGalaxy*. This is particularly interesting for gaze-supported applications like *GazeGalaxy* (cf. Chapter 10).

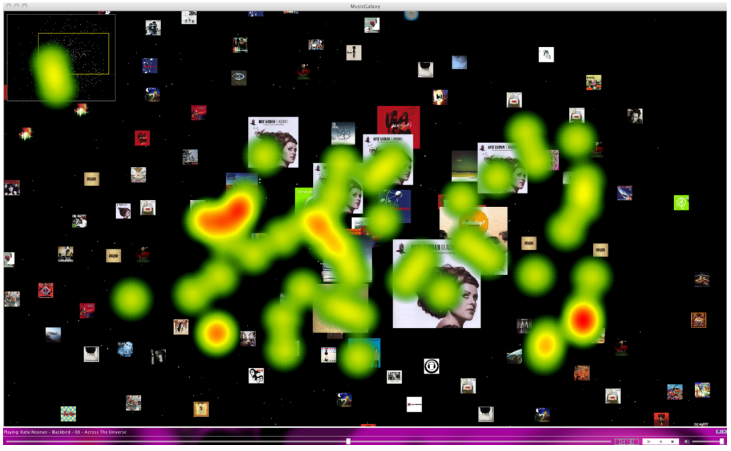


Figure 66:
An example heat map obtained from gaze data for a screenshot of *MusicGalaxy*.

Part IV

APPENDIX

Karma police, arrest this man
He talks in maths
"Karma Police"
Radiohead

A

ANALYZING THE IMPACT OF DATA VECTORIZATION ON DISTANCE RELATIONS

The work presented in this chapter has been published in [pub:21].

Some popular algorithms used in Music Information Retrieval (MIR) such as Self-Organizing Maps (SOMs) require the objects they process to be represented as vectors, i. e., elements of a vector space. This is a rather severe restriction and if the data does not adhere to it, some means of *vectorization* is required. As a common practice, the full distance matrix is computed and each row of the matrix interpreted as an artificial feature vector. This chapter empirically investigates the impact of this transformation. Further, an alternative approach for vectorization based on Multidimensional Scaling (MDS) is proposed that is able to better preserve the actual distance relations of the objects which is essential for obtaining a good retrieval performance.

A.1 INTRODUCTION

As explained in Section 2.1, music information can be described in many different ways: The loudness of a track can be captured in a single (continuous) value. The harmonic key may be a single value, too, but not from a continuous range. Other feature types comprise sets (e. g., instruments, tags), histograms (e. g., chord distribution), intervals (e. g., production period), vectors (e. g., lyrics in the common term frequency vector representation) up to complex descriptions of probability distributions commonly used to describe the timbre in terms of MFCCs. Moreover, usually more than one feature is used as description.

On the other hand, there are algorithms such as the popular SOMs – or prototype-based clustering approaches in general – that require input data as vectors. Here, the term *vector* is used in its strict mathematical sense, i. e., as an element of a (numerical) *vector space*. For many features, there is no straightforward way of transformation into a flat vector representation without losing semantics. For instance, writing a covariance matrix as a vector by concatenating the rows completely ignores the feature's semantics when applying vector operations. Sometimes, it may be possible to modify the algorithm such that it can cope with the different data representation as, e. g., described in [215] for Gaussian distributions in SOMs. For the remaining cases, an artificial vectorization step has to be introduced.

Vectorization – in the context of this thesis – is a generic preprocessing step that maps each object o of a dataset S onto a vector

representation $\mathbf{x} \in \mathbb{R}^n$. This mapping should preserve some characteristic of the data which is particular for the algorithm(s) to be applied afterwards. The focus of this chapter lies on the distance relations between the objects which are especially important when using SOMs. Consequently, the approaches covered here require a distance (or dissimilarity) matrix of the dataset to be vectorized as input. They are, however, not confined to any specific feature type and thus generally applicable.

An established vectorization approach that can be considered as common practice, is to simply interpret each row of the distance matrix (containing the pairwise distances of all objects in the dataset) as a feature vector. I. e., the objects are described by the distances amongst each other. This approach has been applied, e. g., for the SOM-based MIR applications *Islands of Music* [179], *nepTune* [108] and *BeatlesExplorer* (Section 8.5). The SOMs obtained using this vectorization technique seem satisfactory. However, to the knowledge of the authors, there has not been a formal proof why this vectorization approach works nor an evaluation how well it works so far.

This chapter does not aim to give a formal proof either but proposes a methodology for evaluation (Section A.2.2). Using a test collection (Section A.2.1) similar to the one described for the BeatlesExplorer (Section 8.5), several experiments motivated by real-world scenarios are conducted (Section A.2.3). Furthermore, an alternative vectorization approach based on MDS is described (Section A.3.2) and evaluated. MDS has already been applied successfully for vectorization by the *SoniXplorer* [135] which, however, simply chooses an output dimensionality of $d = 20$ without further analysis of the impact. Such an analysis is provided here with the additional extension to let the MDS automatically choose an appropriate value for the dimensionality parameter depending on a given boundary for the accuracy. Additionally, Section A.3.3 describes a vectorization meta-approach suitable for adaptive MIR applications that compute distances as weighted aggregation of several facets such as the *BeatlesExplorer* (Section 8.5) and the *SoniXplorer* [135]. Experimental results are discussed in Section A.4. Section A.5 concludes this chapter.

A.2 EXPERIMENTAL SETUP

A.2.1 *Test Collection*

The collection used for the experiments is similar to the one described in [pub:10] and contains 197 songs from The Beatles.[1] Each song is represented by 20 facets that cover different aspects of music similarity. As defined in Section 8.1, a facet refers to a (set of) feature(s) in combination with a respective (facet) distance measure. Distances

1 Of the original dataset, only those songs with an available recording were used.

between two songs are obtained by aggregating the respective facet distances as weighted sum. This way, facet weights are introduced that allow to adapt the importance of each facet according to user preferences. Table 20 contains a list of all facets. The information is extracted (semi-automatically) from Wikipedia [url:53], LyricWiki [url:28], Alan W. Pollack's notes on The Beatles [url:41], manual chord annotations [88] and from the audio recordings using the frameworks CoMIRVA [213] and JAudio [150]. A detailed description is given in [pub:10]. In order to avoid an aggregation bias towards facets with large distance values, the distances are normalized such that the mean distance per facet is 1.0 (cf. Equation 8.1).

A.2.2 *Evaluation Measures*

The following two straightforward measures are used to assess the quality of a vectorization:

Distance Triples Agreement: This evaluation measure aims to capture how well the distance relations are preserved by the vectorization. To this end, the distance matrix on the original objects $\mathbf{D} = (d_{ij})$ is compared with the distance matrix for the vectorized objects $\mathbf{D}' = (d'_{ij})$. Both matrices agree with respect to a triple (s, a, b) iff $\mathrm{sign}(d_{sa} - d_{sb}) = \mathrm{sign}(d'_{sa} - d'_{sb})$, i. e., the vectorized version of the object that was closer to the seed object o_s is closer to the vector x_s as well which means that the ranking order is maintained by the transformation. The distance triples agreement score of $\mathbf{D}$ and $\mathbf{D}'$ is the relative number of agreements w.r.t. the total number of possible triples.

Nearest Neighbor Agreement: The retrieval of nearest neighbors plays an important role in many MIR applications such as query-by-example, recommendation and clustering. For instance, for a SOM clustering of vectorized data, it is essential that nearest neighbors are reliably identified because otherwise objects may be assigned to less appropriate locations on the map. Two indicators for the preservation of neighborhoods are the agreement on the closest neighbor and on the ten nearest neighbors (top 10). The closest neighbor agreement is the relative number of objects that have the same nearest neighbor in both, original and vectorized space. Accordingly, the top 10 agreement compares the set of ten nearest neighbors in both spaces (without order taken into account).

A.2.3 *Test Scenarios*

Three test scenarios described in the following sections are considered which are motivated by common applications.

A.2.3.1 *Vectorizing a Fixed Dataset*

This scenario is by far the most common case in the literature: A dataset is vectorized once and afterwards neither the distances between the objects nor the dataset changes, i. e., no objects are added or removed. Here, the performance of the baseline is most interesting as a poor performance might question the common practice.

For the aggregation of the facet distances, the facets are weighted uniformly. Additionally, analyzing the performance for each single facet gives an impression of how well the vectorization approaches work for different kinds of data.

A.2.3.2 *Adapting Facet Weights*

What if the facet weights and thus the aggregated distances change after the initial vectorization? Applications like the BeatlesExplorer (Section 8.5) or the SoniXplorer [135] allow the automatic adaptation of facet weights based on user actions. Changing the facet weights, however, results in modified distances in the original space. One way to propagate this change to the vector space is to compute a new vectorization and all other steps done afterwards such as training a SOM for the vectorized dataset. This can have a severe effect on the visualization even for small weight changes and may confuse or even annoy the user. The SoniXplorer [135] tries to reduce this effect by choosing appropriate initialization values. However, the chain of necessary re-computations is very costly and makes real-time interaction impossible.

An alternative way that avoids the re-computation of the vectorization is proposed in Section A.3.3. For its evaluation, a preference-based adaptation approach as motivated in [38] is applied which can be briefly described as follows:

1. Random facet weights are generated that represent the user preferences (which are unknown to the system). Using these weights, the facet distances are aggregated into the user distance matrix $\mathbf{U} = (u_{ij})$.
2. Random triples (s, a, b) drawn from $\mathbf{U}$ with $u_{sa} < u_{sb}$ serve as training samples for a perceptron learner.
3. The learner uses gradient descent to find a facet weighting that complies with the sampled user preferences.
4. The learned facet weights are applied for aggregation in both, the original and the vector space, and the resulting distance matrices are evaluated against $\mathbf{U}$.

The performance measures are averaged over 100 random user weightings.

A.2.3.3 *Adding New Songs*

It might be sufficient for prototypes, demonstrations or retrieval experiments to assume a fixed dataset. However, in the real world this assumption seldom holds. Songs are added and removed from music collections as music taste changes and new interesting music is discovered. In such a case, computing a new vectorization of the changed collection may break existing structures. For instance, a SOM would need to be relearned and the result might look totally different. It would therefore be a welcomed property of a vectorization, if it (to some extent) supported changes of the dataset without severe degradation of the performance. The performance for added objects is of particular interest here as the respective distance information has not been available during the computation of the initial vectorization.

For the experiment, the test collection is randomly split into two parts. The first part is used as the initial dataset for the vectorization. The remaining part is added afterwards and vectorized with respect to the initial dataset. I. e., the vectorization of the new songs does not alter the vector representations of the initial songs. Performance is computed for each possible splitting ratio and averaged over 100 random splits to reduce effects caused by the random assignment.

A.3 VECTORIZATION APPROACHES

A.3.1 *Baseline*

Vectorization by interpreting the rows of the distance matrix as feature vectors is considered as baseline here because it is the common practice in the literature. Using this approach, the vectors can directly be read from the distance matrix without computation. Whilst this is very straightforward, it is not immediately clear how the distance between the resulting vectors should be computed. Therefore, the following three common distance metrics are considered here:

1. Euclidean distance: $d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i (x_i - y_i)^2}$
2. Manhattan distance: $d(\mathbf{x}, \mathbf{y}) = \sum_i |x_i - y_i|$
3. Cosine distance: $d(\mathbf{x}, \mathbf{y}) = 1 - \cos(\mathbf{x}, \mathbf{y})$

A.3.2 *Vectorization by Multidimensional Scaling*

MDS naturally suggests itself as a method for vectorization as it aims to retain the original object distances in the target Euclidean space as described in Section 4.4. Furthermore, the trade-off between the number of vector dimensions and the quality of the vectorization can be controlled directly by providing a threshold r_{max} for the residual instead of having to fix the target dimensionality, m, a priori. The

algorithm then automatically selects the smallest m that results in a mapping with a residual $r \leqslant r_{max}$. The best possible mapping is obtained for $r = 0$ which also results in the highest number of output dimensions.

A.3.3 *Vectorization per Facet*

In the second test scenario described in Section A.2.3.2, the facet weights and thus the aggregated distances between the objects (in the original space) change. This change has to be propagated somehow to the vector space. Re-computation of the vectorization can be avoided, if the adapted facet weights can also be applied in the vector space without losing their semantics. This is only possible, if there is a direct correlation of the vector space dimensions and the facets. To this end, the objects have to be vectorized with respect to each single facet. The resulting vectors are then concatenated for each object. This meta-technique can be applied for any basic vectorization approach.

For the computation of the distances between the resulting vectors, two possibilities are considered:

1. The distance measure to be used in the output vector space is applied to the concatenated vectors using the weight of the corresponding facet *for each dimension*. I. e., for the MDS this means to compute the weighted Euclidean distance:

$$d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i w_i (x_i - y_i)^2} \tag{A.1}$$

2. The distance measure to be used in the output vector space is applied to compute the distance *for each facet* individually. These facet distances are then aggregated in a weighted sum using the respective weights:

$$d(\mathbf{x}, \mathbf{y}) = \sum_f w_f d_f(\mathbf{x}, \mathbf{y}) \tag{A.2}$$

   where $d_f(\mathbf{x}, \mathbf{y})$ is the distance of the vectors $\mathbf{x}$ and $\mathbf{y}$ taking only into account the dimensions that correspond to facet f.

In the second case which will be referred to as "aggregated per facet", the same aggregation function (i. e., the weighted sum) as for the original object facets is used. Only the specific facet distance measures are substituted by the distance metric of the vector space that now computes each facet distance from the corresponding vector dimensions.

A.4 RESULTS

A.4.1 *Vectorizing a Fixed Dataset*

Table 19 shows the performance of the different vectorization approaches for this typical use case. Additionally to the evaluation measures, it contains the number of output dimensions. The baseline approach obviously produces vectors with 197 dimensions as this is the size of the dataset. The probably most important observation here is that the baseline vectorization approach does actually work reasonably well with an agreement of more than 75% for the triples and between 50% and 60% for the neighborhoods. The most suitable distance metric in the vector space is – quite surprisingly – the cosine distance which especially does much better preserve the neighborhoods than the Euclidean or Manhattan distance. It is therefore considered as baseline in further comparisons.

Table 19: Performance comparison of the different vectorization approaches. All values in percent except dimensions (dim). *Selected for further comparisons.

vectorization approach	dim	triples	nearest neighbors	
		agreement	closest	top 10
baseline (Euclidean distance)	197	76.8	45.7	54.4
baseline (Manhattan distance)	197	72.7	37.6	44.3
baseline (Cosine distance)*	197	78.9	54.8	60.7
baseline vectorized per facet (Euclidean distance)	3940	84.4	47.2	57.6
baseline vectorized per facet (Manhattan distance)	3940	85.9	52.8	63.6
baseline vectorized per facet (Cosine distance)	3940	82.2	46.2	55.7
MDS	58	96.1	78.2	87.2
MDS vectorized per facet	1051	92.5	71.6	80.7
MDS vectorized & aggregated per facet	1051	98.3	94.4	94.1

Combining the baseline with the meta-technique for per-facet vectorization described in Section A.3.3 mainly has an impact on the triples agreement. Surprisingly, the Cosine distance now performs worse whereas the Manhattan distance appears to be the best choice. However, the small improvement in performance hardly justifies the remarkable increase of vector dimensions which is now the dataset size times the number of facets. Applying Equation A.2 for distance computation in the vector space does not lead to an improvement for any of the considered distance metrics (omitted in Table 19).

Using the basic MDS approach increases performance by about 20% (absolute). Again, the agreement is significantly higher for the triples than for the neighborhoods. Most importantly, the number of output

dimensions is much less than for the baseline. As Figure 67 illustrates, MDS already outperforms the baseline performance using only 13 output dimensions. The plot also shows that using more than 40 dimensions which roughly corresponds to a residual of 10% does not lead to much performance improvement anymore.

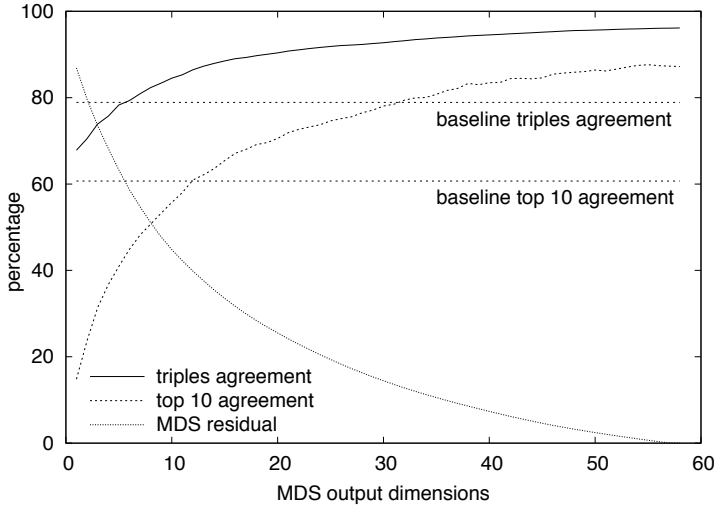


Figure 67: Performance of the MDS vectorization depending on the number of dimensions of the output space.

Looking at Table 20 confirms the superior performance of the MDS approach but also reveals that there is a high variance in the number of dimensions required per facet. The maximum number of dimension is needed for the distribution of MFCCs ("audio:mandelellis") which is a very complex feature. The worst performance is obtained for "production:date". This feature is a set of (time) intervals and for distance computation a rather complicated measure is used which does not conform to the triangle inequality and thus is not a metric. For such non-metric cases, special variants of MDS exist as described, e. g., in [86, Chapter 15.3] that could be applied but this is beyond the scope of this thesis. Interestingly, the (metric) MDS approach still performs significantly better than the baseline. This could be indicating that the baseline approach should be used only for distance metrics though there is not sufficient evidence to support this hypothesis. Table 20 also shows that there seems to be a problem with text-based features (title and lyrics). However, the reason for this is not clear and requires further investigation.

The concatenation of the vectors obtained separately for each facet results in 1051 dimensions – more than 5 times the value for the baseline but significantly less than the one for the baseline vectorized per facet. Compared to the basic MDS approach, there is only a small increase of the triples agreement (which was already close to the maximum anyways) but also a significant improvement of the neighborhood preservation close to maximum. This holds, however,

facet name	dim	triples agreement		nearest neighbors closest		nearest neighbors top 10	
title	143	19.2	(0.9)	51.3	(6.6)	54.0	(10.4)
lyrics:text	110	89.3	(18.6)	55.8	(23.9)	69.9	(34.2)
creators	28	100.0	(29.0)	100.0	(8.1)	100.0	(12.0)
year	7	100.0	(18.9)	100.0	(0.0)	100.0	(0.0)
producer / engineer	26	100.0	(37.4)	100.0	(6.1)	100.0	(14.3)
production:location	54	100.0	(36.5)	100.0	(18.8)	100.0	(28.5)
production:date	60	16.1	(8.4)	36.0	(-3.6)	71.3	(36.5)
instruments	109	99.4	(32.3)	95.4	(36.0)	96.1	(46.6)
musicians:all	55	100.0	(27.5)	100.0	(17.3)	100.0	(29.1)
musicians:lead vox	11	100.0	(25.7)	100.0	(1.5)	100.0	(4.1)
musicians:bg vox	16	100.0	(39.5)	100.0	(4.6)	100.0	(6.8)
musicians:guitars	10	100.0	(25.4)	100.0	(1.5)	100.0	(4.5)
musicians:bass	7	100.0	(3.3)	100.0	(0.5)	100.0	(5.1)
musicians:drums	8	100.0	(3.2)	100.0	(2.0)	100.0	(4.5)
audio:mandelellis	196	100.0	(17.9)	100.0	(7.6)	100.0	(9.7)
audio:fluctuation	101	100.0	(11.3)	100.0	(62.4)	100.0	(44.5)
audio:marsyas07	57	100.0	(23.0)	100.0	(67.0)	100.0	(52.6)
key	14	84.3	(7.4)	99.0	(0.5)	98.6	(5.9)
chords:variety	14	100.0	(7.6)	100.0	(0.0)	100.0	(1.3)
chords:distribution	25	100.0	(28.8)	100.0	(37.1)	100.0	(27.5)

Table 20: Single-facet MDS vectorization performance. (All values in percent except dim. Absolute improvement over baseline in brackets.)

only if Equation A.2 is used. In fact, Equation A.1 performs worse than the basic MDS approach.

A.4.2 *Adapting Facet Weights*

As pointed out in Section A.3.3, this is the intended application scenario of the per-facet vectorization approach. Table 21 shows that the approach indeed correctly propagates facet weight changes into the vector space for the MDS vectorization so that values close to the initial performance (cf. Table 19) are obtained. Again, distance computation using Equation A.2 is superior – especially in preserving the neighborhoods. Very surprisingly, the technique seems not to work for the baseline per-facet vectorization (i. e., with 3940 dimension). The learned weights[2] (as any weighting other than the uniform one) further degrade the performance. A reason for this could lie in the

2 Note that the weights are learned using the original object representation – i. e., independent of the vectorization – and thus are the same for all vectorization approaches as described in Section A.2.3.2. Therefore, an inappropriate weighting cannot be the reason for the poor performance of the baseline.

Table 21: Performance of the per-facet vectorization approaches in a scenario where initial distance facet weights (uniform) are adapted by a learning algorithm according to a user's distance judgments. Mean (and standard deviation in brackets) over 100 random user weightings. Top row: performance before and after adaptation in original space (i. e., expected best achievable value). Bottom rows: performance against uniform weighting (cf. Table 19) for comparison, and before and after propagation of the weight change.

vectorization approach	comparison	triples agreement		nearest neighbors closest		nearest neighbors top 10	
no vectorization	unadapted	82.3	(3.6)	42.2	(10.0)	55.5	(7.0)
	adapted	100.0	(0.0)	100.0	(0.0)	100.0	(0.0)
baseline per facet (Manhattan distance)	vs uniform	85.9		52.8		63.6	
	unadapted	77.1	(3.8)	31.9	(8.6)	44.9	(6.6)
	adapted	76.9	(3.9)	31.8	(8.6)	44.7	(6.6)
MDS per facet (Equation A.1)	vs uniform	92.5		71.6		80.7	
	unadapted	80.4	(3.9)	38.0	(8.4)	51.8	(6.9)
	adapted	92.5	(0.9)	70.8	(5.8)	80.1	(1.7)
MDS per facet (Equation A.2)	vs uniform	98.3		94.4		94.1	
	unadapted	82.2	(3.6)	41.7	(10.5)	55.3	(7.3)
	adapted	98.2	(1.3)	88.1	(8.3)	93.8	(4.4)

high dimensionality of the vector space for this case but this has to be investigated further.

A.4.3 *Adding New Songs*

As motivated in Section A.2.3.3, the distance relations and neighborhoods for the new songs are of interest here. The change of the distance triples agreement with increasing number of new songs (and simultaneously decreasing initial collection size) is shown in Figure 68 (top). The plot can be divided into two sections: Up to a ratio of roughly 4:1 (4 times more new songs than in the initial collection), there is almost no decrease in performance for all vectorization approaches. In this section, the agreement is close to what can be achieved by vectorizing the whole collection (cf. Section A.4.1). For higher ratios, the performance decreases – most significantly for the basic MDS vectorization. The other approaches are hardly affected up to a rather extreme 10:1 ratio.

Similar behavior can be observed for the top 10 agreement in Figure 68 (bottom). Here, only the very left region looks different. This is because the ranked lists considered for the top 10 agreement are very short due to the small number of new songs. Therefore, the performance values in this region cannot be considered significant.

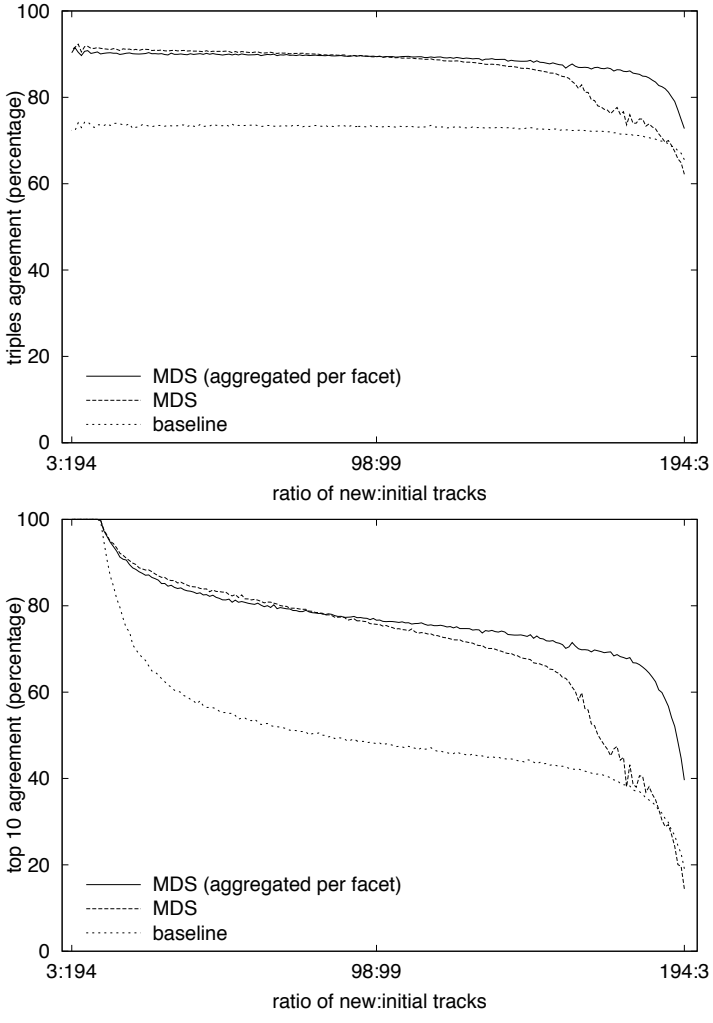


Figure 68: Performance degradation with increasing portion of new songs added after the vectorization of an initial collection (measured for the new songs). Mean values over 100 random splits for each ratio.

A.5 SUMMARY

In applications where data vectorization is unavoidable, it is important to be aware of the effect this transformation may have on the characteristics of the data. In this chapter, a general methodology that can be applied to arbitrary datasets has been proposed for how to assess changes in distance relations and neighborhoods – two important characteristics in retrieval applications. Using this methodology, the current common practice of vectorizing a dataset as the row vectors of a distance (or dissimilarity) matrix has been empirically evaluated on a multi-facet test collection. The experiments have been motivated by real-world applications: Apart from the common scenario of a fixed dataset, the adaptation of facet weights (and thus changes of distances) and changes of the dataset have been addressed. Furthermore, an alternative approach based on Multidimensional Scaling (MDS) has been proposed which shows significantly better performance while requiring far fewer vector dimensions.

Everything that can be counted
does not necessarily count;
everything that counts
cannot necessarily be counted.

ALBERT EINSTEIN

B

COMMON EVALUATION MEASURES IN INFORMATION RETRIEVAL

This appendix gives a brief overview of the essential measures that are commonly used to evaluate information retrieval approaches. More detailed explanations can, e. g., be found in [8].

If relevance judgments are given, objects retrieved by a system can be categorized into four types as shown in Table 22. False positives are also referred to as *type I errors* whereas false negatives are *type II errors*.

Table 22: Categorization of retrieved objects according to correct and predicted relevance/classification.

		correct	
		relevant	not relevant
predicted	relevant	true positive (TP)	false positive (FP)
	not relevant	false negative (FN)	true negative (TN)

Based on this categorization, the following evaluation measures can be defined:

- *Precision* – the fraction of retrieved documents that are actually relevant:

$$\text{precision} = \frac{|\text{TP}|}{|\text{TP}| + |\text{FP}|} \tag{B.1}$$

- *Recall* – the fraction of relevant documents that have been retrieved:

$$\text{recall} = \frac{|\text{TP}|}{|\text{TP}| + |\text{FN}|} \tag{B.2}$$

- *F-Measure* – the harmonic mean of precision and recall:

$$\text{F} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{B.3}$$

All these measure have values between 0 and 1 where larger values indicate better performance.

In binary classification scenarios, where a system has to predict whether an objects belongs to a class (=relevant) or not (=not relevant), the categories of Table 22 are used as well and the following additional measures are commonly applied:

- *Accuracy* – the proportion of correct classifications (both, true positives and true negatives):

$$\text{accuracy} = \frac{|\text{TP}| + |\text{TN}|}{|\text{TP}| + |\text{FP}| + |\text{FN}| + |\text{TN}|} \tag{B.4}$$

- *Specificity* – the fraction of negative results (objects not belonging to the class) correctly classified as such:

$$\text{specificity} = \frac{|\text{TN}|}{|\text{TN}| + |\text{FP}|} \tag{B.5}$$

- *Sensitivity* – the fraction of positive results (objects belonging to the class) correctly classified as such:

$$\text{sensitivity} = \frac{|\text{TP}|}{|\text{TP}| + |\text{FN}|} \tag{B.6}$$

Again, values are within the range $[0, 1]$ and 1 indicates the best possible performance.

All the above measures are set-based and thus particularly suitable for unordered sets of objects. If the system further returns the results as a ranked list (with the object at rank 1 considered as most relevant for the query), the following measures can take the order into account:

- *Precision/Recall at k* – defined as above but considering only results up to rank k.
- *Mean Average Precision* – the average of precisions computed at the point of each relevant document in the ranked list.
- *Reciprocal Rank* or *Inverse Rank* – the (multiplicative) inverse of the rank of the first relevant object, i.e.

$$\text{reciprocal rank} = \frac{1}{\text{rank}} \tag{B.7}$$

  If multiple queries are considered, the *Mean Reciprocal Rank* is computed as the average of the individual values. The value range is $(0, 1]$ where the best value, 1, is obtained if the first retrieved document is relevant.

Precision can also be plotted as a function of recall. An example is shown in Figure 69 (left). Alternatively, a ROC curve (Receiver Operating Characteristic) plots the sensitivity (true positive rate) against the false positive rate (1-specificity) as shown in Figure 69 (right). The area under the ROC curve (sometimes abbreviated as AUC) is also a prominent evaluation measure. Higher values indicate a better performance.

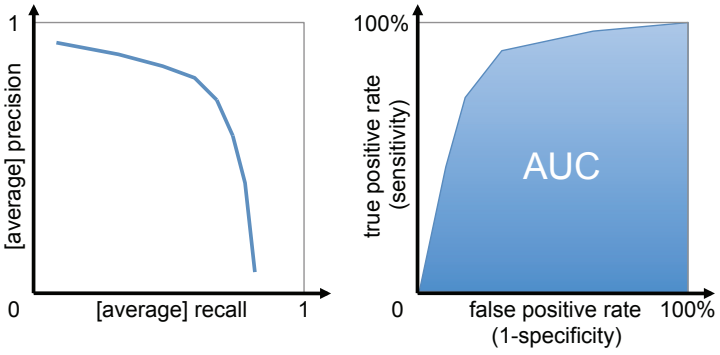


Figure 69: Common (interpolated) evaluation plots. Larger area under the curve indicates better performance. Left: precision / recall. Right: ROC curve.

A wise man can learn more from a foolish question than a fool can learn from a wise answer.

BRUCE LEE

C

QUESTIONNAIRES

This appendix contains the questionnaires used in the survey on the acceptance on listening context logging described in Section 6.3. Abbildungen 70 bis 75 show the English version of the web-based questionnaire. The paper questionnaire used at the *CeBIT 2009* fare is shown in Figure 76. For this, there was only a German version.

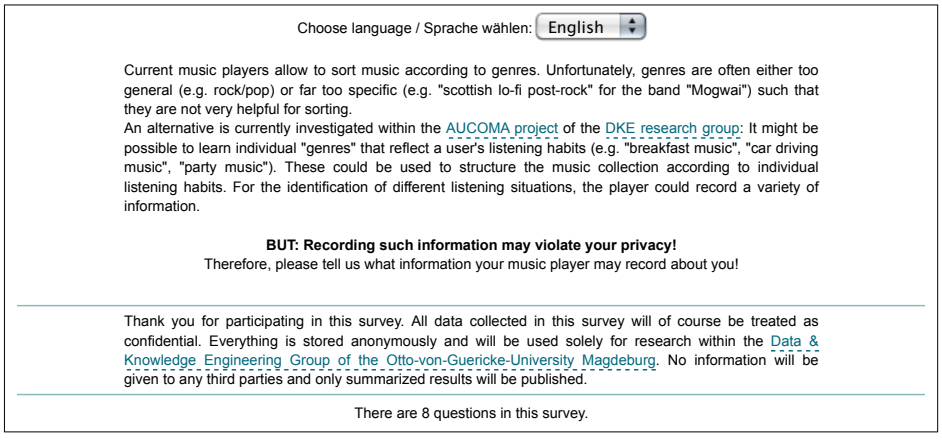


Figure 70: Introduction page of the web questionnaire.

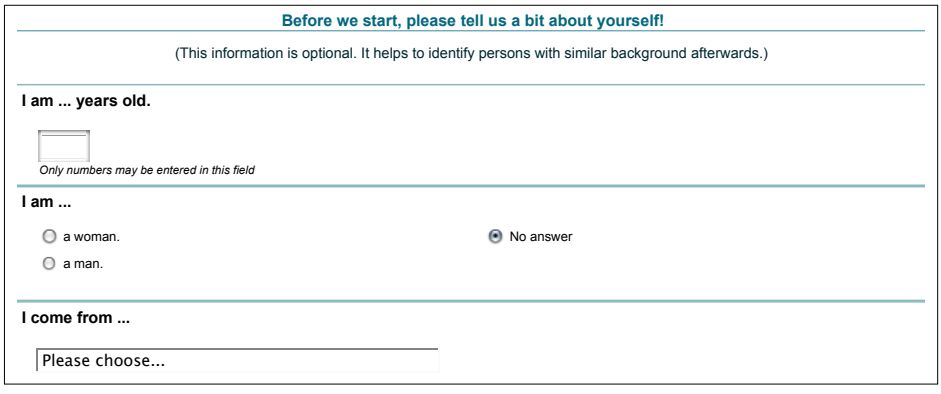


Figure 71: Second page of the web questionnaire, covering demographic information.

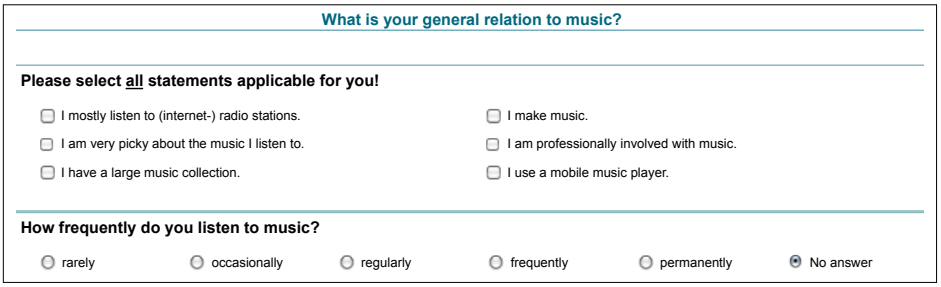


Figure 72: Third page of the web questionnaire, covering the general relation to music.

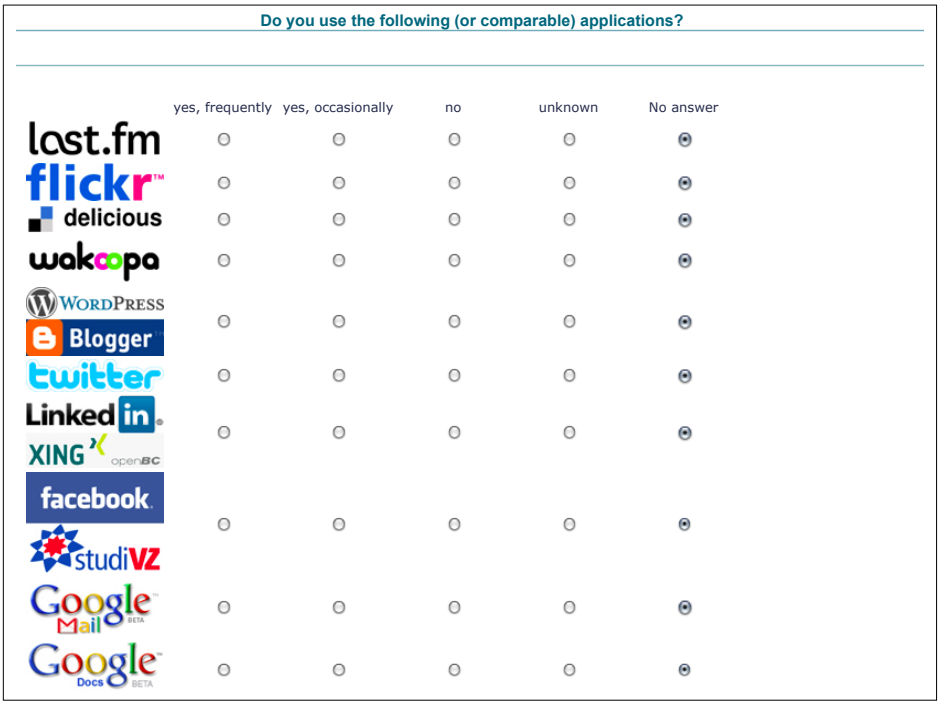
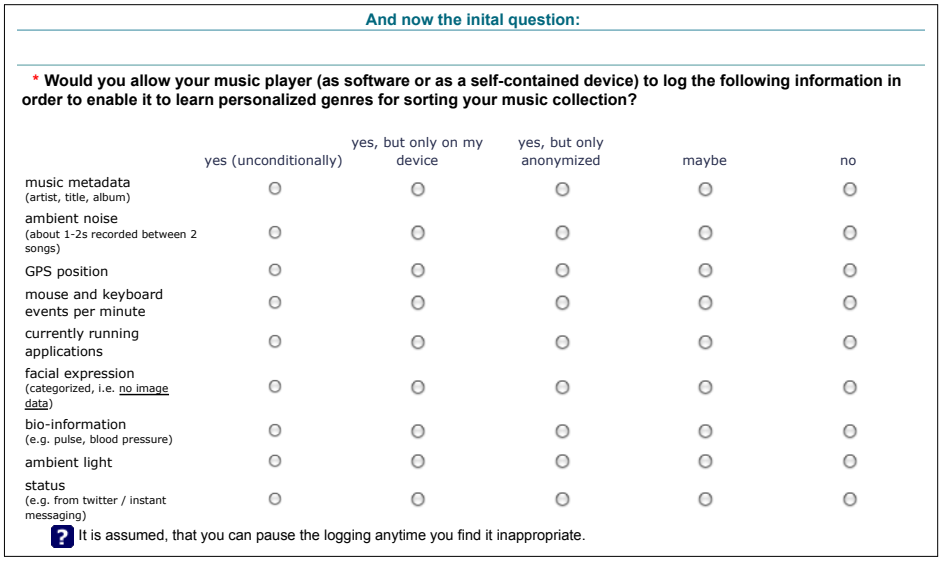


Figure 73: Forth page of the web questionnaire, covering the usage of (web-) applications that collect, access and expose to some extent private data of their users



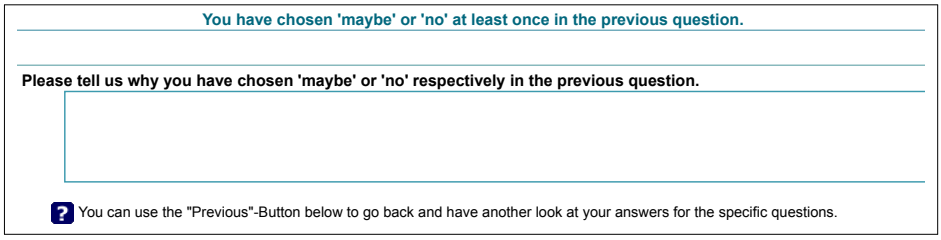
And now the inital question:

* **Would you allow your music player (as software or as a self-contained device) to log the following information in order to enable it to learn personalized genres for sorting your music collection?**

	yes (unconditionally)	yes, but only on my device	yes, but only anonymized	maybe	no
music metadata (artist, title, album)	○	○	○	○	○
ambient noise (about 1-2s recorded between 2 songs)	○	○	○	○	○
GPS position	○	○	○	○	○
mouse and keyboard events per minute	○	○	○	○	○
currently running applications	○	○	○	○	○
facial expression (categorized, i.e. no image data)	○	○	○	○	○
bio-information (e.g. pulse, blood pressure)	○	○	○	○	○
ambient light	○	○	○	○	○
status (e.g. from twitter / instant messaging)	○	○	○	○	○

? It is assumed, that you can pause the logging anytime you find it inappropriate.

Figure 74: Fifth page of the web questionnaire, covering the core question of the survey.



You have chosen 'maybe' or 'no' at least once in the previous question.

Please tell us why you have chosen 'maybe' or 'no' respectively in the previous question.

? You can use the "Previous"-Button below to go back and have another look at your answers for the specific questions.

Figure 75: Optional sixth page of the web questionnaire. This page was only shown, if one ore more answers of the preceding were “no” or “maybe”.

Wie viel darf Ihr Musikplayer über Ihre Hörgewohnheiten wissen?

Teilen Sie uns bitte zunächst ein wenig über Ihre Person mit!
(Diese Angaben sind freiwillig. Sie helfen, später Personengruppen mit ähnlichem Hintergrund zu identifizieren.)

Ich bin ☐ Jahre alt. Ich bin ☐ eine Frau. ☐ ein Mann. Ich komme ☐ aus Deutschland. ☐ aus ☐

In welcher Beziehung stehen Sie allgemein zu Musik? Kreuzen Sie bitte **alle** für Sie zutreffenden Aussagen an!

- ☐ Ich höre meist (Internet-) Radio.
- ☐ Ich bin bei der Musikauswahl sehr wählerisch.
- ☐ Ich habe eine große Musiksammlung.
- ☐ Ich mache Musik.
- ☐ Ich habe beruflich mit Musik zu tun.
- ☐ Ich benutze einen mobilen Musikplayer.

Wie oft hören Sie Musik?

☐ kaum ☐ gelegentlich ☐ regelmäßig ☐ häufig ☐ ständig

Nutzen Sie die folgenden (oder vergleichbare) Applikationen?

	ja, regelmäßig	ja, gelegentlich	nein	kenne ich nicht
last.fm	☐	☐	☐	☐
flickr	☐	☐	☐	☐
delicious	☐	☐	☐	☐
wakoopa	☐	☐	☐	☐
WordPress Blogger	☐	☐	☐	☐
twitter	☐	☐	☐	☐
LinkedIn XING openBC	☐	☐	☐	☐
facebook studiVZ	☐	☐	☐	☐
Google Mail	☐	☐	☐	☐
Google Docs	☐	☐	☐	☐

Nun zur Ausgangsfrage: Würden Sie Ihrem Musikplayer erlauben, die folgenden Daten aufzuzeichnen, damit dadurch beispielsweise für Sie personalisierte Wiedergabelisten erstellt werden können?

(Es wird vorausgesetzt, dass die Aufzeichnung ggf. unterbrochen werden kann, falls dies zwischenzeitlich nicht erwünscht sein sollte.)	ja (bedingungslos)	ja, aber nur auf meinem Gerät gespeichert	ja, aber nur anonymisiert	vielleicht	nein
Musikmetadaten (Künstler, Title, Album)	☐	☐	☐	☐	☐
Umgebungsgeräusche (jeweils 1-2s aufgenommen zwischen 2 Musikstücken)	☐	☐	☐	☐	☐
GPS Position	☐	☐	☐	☐	☐
Maus- und Tastaturaktionen pro Minute	☐	☐	☐	☐	☐
parallel laufende Programme	☐	☐	☐	☐	☐
Gesichtsausdruck (kategorisiert, d.h. keine Bilddaten)	☐	☐	☐	☐	☐
Bioinformationen (z.B. Puls, Blutdruck)	☐	☐	☐	☐	☐
Umgebungslichtstärke	☐	☐	☐	☐	☐
Status (z.B. aus Twitter/Instant Messaging)	☐	☐	☐	☐	☐

Falls Sie in der vorhergehenden Frage mindestens einmal 'nein' oder 'vielleicht' gewählt haben, begründen Sie Ihre Entscheidung bitte kurz:

Alle in dieser Umfrage erhobenen Daten werden streng vertraulich behandelt. Sie werden anonymisiert gespeichert und ausschließlich innerhalb der Arbeitsgruppe Data & Knowledge Engineering der Otto-von-Guericke-Universität Magdeburg zu Forschungszwecken verwendet. Es werden keine Daten an Dritte weitergegeben sondern lediglich zusammenfassende Ergebnisse publiziert.

Figure 76: Paper questionnaire that was filled out by 156 visitors of the *CeBIT 2009* fare.

[1] Adams, N., Witkowski, M., and Spence, R.: "The inspection of very large images by eye-gaze control." In: *Proceedings of the working Conference on Advanced Visual Interfaces (AVI'08)*. 2008, pp. 111–118 (cit. on p. 177).

[2] Agrawal, R., Imielienski, T., and Swami, A.: "Mining Association Rules between Sets of Items in Large Databases." In: *Proceedings of Conference on Management of Data*. 1993, pp. 207–216 (cit. on p. 80).

[3] Ashmore, M., Duchowski, A. T., and Shoemaker, G.: "Efficient eye pointing with a fisheye lens." In: *Proceedings of Graphics Interface (GI'05)*. 2005, pp. 203–210 (cit. on pp. 173–175, 180).

[4] Astrom, K. J. and Wittenmark, B.: *Adaptive Control*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1994 (cit. on p. 29).

[5] Baccigalupo, C. and Fields, B.: *Mining the Social Web for Music-Related Data*. Tutorial at ISMIR'09. 2009 (cit. on p. 14).

[6] Bade, K.: "Personalized Hierarchical Structuring." PhD thesis. Otto-von-Guericke-University Magdeburg, 2009 (cit. on p. 128).

[7] Bade, K. and Nürnberger, A.: "Creating a Cluster Hierarchy under Constraints of a Partially Known Hierarchy." In: *Proceedings of the 2008 SIAM International Conference on Data Mining*. 2008, pp. 13–24 (cit. on p. 131).

[8] Baeza-Yates, R., Ribeiro-Neto, B., et al.: *Modern information retrieval*. Addison-Wesley Reading, MA, 1999 (cit. on p. 205).

[9] Bainbridge, D., Cunningham, S. J., and Downie, J. S.: "How People Describe Their Music Information Needs: A Grounded Theory Analysis Of Music Queries." In: [95]. 2003 (cit. on p. 76).

[10] Baumann, S. and Halloran, J.: "An ecological approach to multimodal subjective music similarity perception." In: *Proceedings of 1st Conference on Interdisciplinary Musicology (CIM'04)*. 2004 (cit. on p. 41).

[11] Baumann, S., Burred, J. J., Nürnberger, A., and Stober, S., eds.: *Proceedings of the 3rd Workshop on Learning the Semantics of Audio Signals (LSAS'09)*. Graz, Austria, 2009 (cit. on pp. 229, 231).

[12] Belew, R. K.: "Adaptive information retrieval: using a connectionist representation to retrieve and learn about documents." In: *SIGIR Forum* 23.SI (1989), pp. 11–20 (cit. on p. 29).

[13] Bello, J. P., Chew, E., and Turnbull, D., eds.: *ISMIR 2008, 9th International Conference on Music Information Retrieval, Drexel University, Philadelphia, PA, USA, September 14-18, 2008, Proceedings*. 2008 (cit. on pp. 218, 225–227, 230, 232).

[14] Bello, J. P. and Pickens, J.: "A Robust Mid-Level Representation for Harmonic Content in Music Signals." In: [205]. 2005, pp. 304–311 (cit. on pp. 34, 46, 68, 71, 73).

[15] Berenzweig, A., Logan, B., Ellis, D., and Whitman, B.: "A large-scale evaluation of acoustic and subjective music-similarity measures." In: *Computer Music Journal* 28.2 (2004), pp. 63–76 (cit. on p. 41).

[16] Betser, M., Collen, P., and Rault, J.: "Audio identification using sinusoidal modeling and application to jingle detection." In: [55]. 2007, p. 139 (cit. on pp. 34, 46).

[17] Bieg, H.-J.: "Gaze-augmented manual interaction." In: *Proceedings of CHI'09 - Extended abstracts*. 2009, pp. 3121–3124 (cit. on p. 175).

[18] Bishop, C.: *Neural networks for pattern recognition*. Oxford University Press, 1995 (cit. on p. 50).

[19] Borgelt, C.: "A Decision Tree Plug-In for DataEngine." In: *Proceedings of the 6th European Congress on Intelligent Techniques and Soft Computing (EUFIT'98)*. Vol. 2. 1998, pp. 1299–1303 (cit. on p. 80).

[20] Borgelt, C.: "Efficient Implementations of Apriori and Eclat." In: *1st Workshop of Frequent Item Set Mining Implementations (FIMI'03)*. 2003 (cit. on p. 80).

[21] Borgelt, C. and Kruse, R.: *Graphical Models - Methods for Data Analysis and Mining*. Chichester, United Kingdom: J. Wiley and Sons, 2002 (cit. on pp. 78, 80).

[22] Borgelt, C., Klawonn, F., Kruse, R., and Nauck, D.: *Neuro-Fuzzy-Systeme: Von den Grundlagen künstlicher Neuronaler Netze zur Kopplung mit Fuzzy-Systemen*. Computational Intelligence. Vieweg, 2003 (cit. on pp. 50, 54).

[23] Brochu, E., De Freitas, N., and Bao, K.: "The sound of an album cover: Probabilistic multimedia and IR." In: *Workshop on Artificial Intelligence and Statistics*. 2003 (cit. on p. 14).

[24] Brossier, P.: "Fast Onset Detection Using Aubio." In: *2005 Music Information Retrieval Evaluation eXchange (MIREX)*. 2005 (cit. on p. 34).

[25] Broy, M., Leuxner, C., Sitou, W., Spanfelner, B., and Winter, S.: "Formalizing the notion of adaptive system behavior." In: *Proceedings of the 2009 ACM Symposium on Applied Computing (SAC'09)*. 2009, pp. 1029–1033 (cit. on p. 30).

[26] Brusilovsky, P.: "Methods and techniques of adaptive hypermedia." In: *User modeling and user-adapted interaction* 6.2 (1996), pp. 87–129 (cit. on pp. 29, 44).

[27] Brusilovsky, P.: "Adaptive Hypermedia." In: *User Modeling and User-Adapted Interaction* 11.1-2 (2001), pp. 87–110 (cit. on p. 29).

[28] Burred, J. J.: "Musical Source Separation: Principles and State of the Art." In: *Proceedings of the 2nd Workshop on Learning the Semantics of Audio Signals (LSAS'08)*. 2008 (cit. on p. 62).

[29] Byrd, D., ed.: *ISMIR 2000, 1st International Symposium on Music Information Retrieval, Plymouth, Massachusetts, USA, October 23-25, 2000, Proceedings*. 2000 (cit. on p. 7).

[30] Byrd, D. and Crawford, T.: "Problems of music information retrieval in the real world." In: *Information Processing and Management* 38.2 (2002), pp. 249–272 (cit. on pp. 7, 9, 16–17).

[31] Cannam, C., Landone, C., Sandler, M., and Bello, J.: "The Sonic Visualiser: A visualisation platform for semantic descriptors from musical signals." In: [124]. 2006, pp. 324–327 (cit. on pp. 63–64, 66).

[32] Cano, P., Kaltenbrunner, M., Gouyon, F., and Batlle, E.: "On the use of FastMap for Audio Retrieval and Browsing." In: [67]. 2002 (cit. on p. 100).

[33] Casey, M., Rhodes, C., and Slaney, M.: "Analysis of minimum distances in high-dimensional musical spaces." In: *Audio, Speech, and Language Processing, IEEE Transactions on* 16.5 (2008), pp. 1015–1028 (cit. on p. 25).

[34] Casey, M., Veltkamp, R., Goto, M., Leman, M., Rhodes, C., and Slaney, M.: "Content-based music information retrieval: current directions and future challenges." In: *Proceedings of IEEE* 96.4 (2008), pp. 668–696 (cit. on pp. 7, 9–10, 19, 25, 27, 29).

[35] Castellina, E. and Corno, F.: "Multimodal Gaze Interaction in 3D Virtual Environments." In: *Proceedings of the 4th Conference on Communication by Gaze Interaction (COGAIN'08)*. 2008, pp. 33–37 (cit. on p. 173).

[36] Celma, O. and Cano, P.: "From hits to niches?: or how popular artists can bias music recommendation and discovery." In: *Proceedings of the 2nd KDD Workshop on Large-Scale Recommender Systems and the Netflix Prize Competition (NETFLIX'08)*. 2008, pp. 1–8 (cit. on p. 163).

[37] Celma, O., Herrera, P., and Serra, X.: "Bridging the Music Semantic Gap." In: *1st International conference on Semantics And digital Media Technology (SAMT)*. 2006 (cit. on p. 26).

[38] Cheng, W. and Hüllermeier, E.: "Learning Similarity Functions from Qualitative Feedback." In: *Proceedings of the 9th European Conference on Advances in Case-Based Reasoning (ECCBR'08)*. 2008, pp. 120–134 (cit. on pp. 127–128, 131, 196).

[39] Chordia, P. and Rae, A.: "Raag recognition using pitch-class and pitch-class dyad distributions." In: [55]. 2007, pp. 431–436 (cit. on pp. 34, 46).

[40] Chuan, C.-H. and Chew, E.: "Fuzzy Analysis in Pitch-Class Determination for Polyphonic Audio Key Finding." In: [205]. 2005, pp. 296–303 (cit. on pp. 35, 46).

[41] Cockburn, A., Karlson, A., and Bederson, B. B.: "A review of overview+detail, zooming, and focus+context interfaces." In: *ACM Computing Surveys* 41 (1 2009), pp. 1–31 (cit. on p. 175).

[42] Cooper, G. and Herskovits, E.: "A Bayesian Method for the Induction of Probabilistic Networks from Data." In: *Machine Learning* 9 (1992), pp. 309–347 (cit. on p. 78).

[43] Csurka, G., Dance, C., Fan, L., Willamowski, J., and Bray, C.: "Visual categorization with bags of keypoints." In: *Workshop on Statistical Learning in Computer Vision, ECCV*. 2004, pp. 1–22 (cit. on p. 25).

[44] Cunningham, S., Caulder, S., and Grout, V.: "Saturday Night or Fever? Context Aware Music Playlists." In: *Proceedings of Audio Mostly 2008: Conference on Sound in Motion (AM'08)*. 2008 (cit. on p. 39).

[45] Dachselt, R. and Buchholz, R.: "Throw and Tilt – Seamless Interaction across Devices Using Mobile Phone Gestures." In: *2nd Workshop on Mobile and Embedded Interactive Systems (MEIS'08)*. 2008, pp. 272–278 (cit. on pp. 177–178).

[46] Dachselt, R. and Buchholz, R.: "Natural throw and tilt interaction between mobile phones and distant displays." In: *Proceedings of CHI'09 - Extended abstracts*. 2009, pp. 3253–3258 (cit. on p. 174).

[47] Dachselt, R. and Frisch, M.: "Mambo: a facet-based zoomable music browser." In: *Proceedings of the 6th International Conference on Mobile and Ubiquitous Multimedia (MUM'07)*. 2007, pp. 110–117 (cit. on p. 81).

[48] De Berg, M., Cheong, O., Van Kreveld, M., and Overmars, M.: *Computational geometry: algorithms and applications*. New York: Springer-Verlag, 2008 (cit. on pp. 107, 109).

[49] De Haas, W., Rohrmeier, M., Veltkamp, R., and Wiering, F.: "Modeling harmonic similarity using a generative grammar of tonal harmony." In: [91]. 2009, pp. 549–554 (cit. on p. 23).

[50] Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R.: "Indexing by latent semantic analysis." In: *Journal of the American society for information science* 41.6 (1990), pp. 391–407 (cit. on p. 26).

[51] Detyniecki, M., Leiner, U., and Nürnberger, A., eds.: *Proceedings of the 6th international workshop on Adaptive Multimedia Retrieval (AMR'08)*. Berlin, Germany, 2008 (cit. on pp. 230, 234).

[52] Dey, A. K.: "Understanding and Using Context." In: *Personal Ubiquitous Computing* 5.1 (2001), pp. 4–7 (cit. on pp. 31, 76, 83).

[53] Dey, A. K. and Abowd, G. D.: "Towards a better understanding of context and context-awareness." In: *Computer Human Interaction (CHI) 2000 Workshop on the What, Who, Where, When, Why and How of Context-Awareness*. 2000 (cit. on pp. 31, 76, 83).

[54] Diakopoulos, D., Vallis, O., Hochenbaum, J., Murphy, J., and Kapur, A.: "21st Century Electronica: MIR Techniques for Classification and Performance." In: [91]. 2009, pp. 465–469 (cit. on pp. 41, 46).

[55] Dixon, S., Bainbridge, D., and Typke, R., eds.: *ISMIR 2007, 8th International Conference on Music Information Retrieval, Vienna, Austria, September 23-27, 2007, Proceedings*. Vienna, AT: Österreichische Computer Gesellschaft, 2007 (cit. on pp. 214, 216, 221, 223–224, 228–231, 233–234).

[56] Dodge, C. and Jerse, T. A.: *Computer Music: Synthesis, Composition and Performance*. 2nd edition. Macmillan Library Reference, 1997 (cit. on p. 7).

[57] Donaldson, J. and Lamere, P.: *Using Visualizations for Music Discovery*. Tutorial at ISMIR'09. 2009 (cit. on pp. 99, 148).

[58] Dopler, M., Schedl, M., Pohle, T., and Knees, P.: "Accessing Music Collections Via Representative Cluster Prototypes in a Hierarchical Organization Scheme." In: [13]. 2008, pp. 179–184 (cit. on pp. 41, 46).

[59] Downie, J. S. and Veltkamp, R. C., eds.: *ISMIR 2010, 11th International Conference on Music Information Retrieval, Utrecht, Netherlands, August 9-13, 2010, Proceedings*. Utrecht, NL, 2010 (cit. on pp. 222–223, 225–226, 229, 231–232).

[60] Downie, J.: "Music information retrieval." In: *Annual review of information science and technology* 37.1 (2003), pp. 295–340 (cit. on pp. 7–13, 18, 20).

[61] Downie, S. and Bainbridge, D., eds.: *ISMIR 2001, 2nd International Symposium on Music Information Retrieval, Indiana University, Bloomington, Indiana, USA, October 15-17, 2001, Proceedings*. 2001 (cit. on pp. 227, 230).

[62] Duchowski, A. T., Cournia, N., and Murphy, H.: "Gaze-Contingent Displays: A Review." In: *CyberPsychology & Behavior* 7.6 (2004), pp. 621–634 (cit. on p. 180).

[63] Dunker, P., Nowak, S., Begau, A., and Lanz, C.: "Content-based mood classification for photos and music: a generic multi-modal classification framework and evaluation approach." In: *Proceeding of the 1st ACM international conference on Multimedia Information Retrieval (MIR'08)*. 2008, pp. 97–104 (cit. on p. 13).

[64] Elliott, G. T. and Tomlinson, B.: "PersonalSoundtrack: context-aware playlists that adapt to user pace." In: *Proceedings of CHI '06 - Extended abstracts*. 2006, pp. 736–741 (cit. on p. 40).

[65] Ellis, D. P. W., Whitman, B., Berenzweig, A., and Lawrence, S.: "The Quest for Ground Truth in Musical Artist Similarity." In: [67]. 2002 (cit. on pp. 41, 43, 170).

[66] Fan, R., Chang, K., Hsieh, C., Wang, X., and Lin, C.: "LIBLINEAR: A library for large linear classification." In: *The Journal of Machine Learning Research* 9 (2008), pp. 1871–1874 (cit. on p. 132).

[67] Fingerhut, M., ed.: *ISMIR 2002, 3rd International Conference on Music Information Retrieval, Paris, France, October 13-17, 2002, Proceedings*. Paris, FR: Ircam - Centre Pompidou, 2002 (cit. on pp. 215, 218–219, 225, 229–230).

[68] Fono, D. and Vertegaal, R.: "EyeWindows: evaluation of eye-controlled zooming windows for focus selection." In: *Proceedings of CHI'05*. 2005, pp. 151–160 (cit. on p. 174).

[69] Fujihara, H., Kitahara, T., Goto, M., Komatani, K., Ogata, T., and Okuno, H.: "Fo estimation method for singing voice in polyphonic audio signal based on statistical vocal model and viterbi search." In: *Proceedings IEEE international Conference Acoustics, Speech, and Signal Processing (ICASSP'06)*. Vol. 5. 2006 (cit. on pp. 36, 46).

[70] Futrelle, J. and Downie, J. S.: "Interdisciplinary Communities and Research Issues in Music Information Retrieval." In: [67]. 2002 (cit. on p. 7).

[71] Gabrys, B., Leiviska, K., and Strackeljan, J.: *Do Smart Adaptive Systems Exist?-Best Practice for Selection and Combination of Intelligent Methods*. Springer Berlin/Heidelberg, 2005 (cit. on p. 29).

[72] Gasser, M. and Flexer, A.: "FM4 Soundpark: Audio-Based Music Recommendation in Everyday Use." In: *Proceedings of the 6th Sound and Music Computing Conference (SMC'09)*. 2009 (cit. on pp. 100–101).

[73] Germer, T., Götzelmann, T., Spindler, M., and Strothotte, T.: "SpringLens: Distributed Nonlinear Magnifications." In: *Eurographics 2006 - Short Papers*. 2006, pp. 123–126 (cit. on pp. 106, 187, 189).

[74] Gleich, M. R. D., Zhukov, L., and Lang, K.: "The World of Music: SDP layout of high dimensional data." In: *Info Vis 2005*. 2005 (cit. on p. 100).

[75] Goldfarb, D. and Idnani, A.: "A numerically stable dual method for solving strictly convex quadratic programs." In: *Mathematical Programming* 27.1 (1983), pp. 1–33 (cit. on pp. 132–133).

[76] Good, M. et al.: "MusicXML: An internet-friendly format for sheet music." In: *XML Conference and Expo*. 2001, pp. 03–04 (cit. on p. 17).

[77] Goto, M.: "A predominant-Fo estimation method for CD recordings: MAP estimation using EM algorithm for adaptive tone models." In: *IEEE International Conference On Acoustics Speech And Signal Processing*. Vol. 5. 2001 (cit. on pp. 36, 46).

[78] Goto, M. and Goto, T.: "Musicream: New Music Playback Interface for Streaming, Sticking, Sorting, and Recalling Musical Pieces." In: [205]. 2005, pp. 404–411 (cit. on p. 165).

[79] Gouyon, F., Herrera, P., Gómez, E., Cano, P., Bonada, J., Loscos, A., Amatriain, X., and Serra, X.: "Content Processing of Music Audio Signals." In: *Sound to Sense, Sense to Sound: A State of the Art in Sound and Music Computing*.

Ed. by Polotti, P. and Rocchesso, D. Berlin: Logos Verlag Berlin GmbH, 2008. Chap. 3, pp. 83–160 (cit. on pp. 7, 24).

[80] Govaerts, S., Corthaut, N., and Duval, E.: "Moody Tunes: the Rockanango Project." In: [124]. 2006 (cit. on p. 76).

[81] Gower, J.: "Generalized procrustes analysis." In: *Psychometrika* 40 (1 1975), pp. 33–51 (cit. on p. 113).

[82] Grachten, M., Schedl, M., Pohle, T., and Widmer, G.: "The ISMIR Cloud: A Decade of ISMIR Conferences at Your Fingertips." In: [91]. 2009, pp. 63–68 (cit. on pp. 7, 29).

[83] Guan, D., Li, Q., Lee, S., and Lee, Y.: "A Context-Aware Music Recommendation Agent in Smart Office." In: *Fuzzy Systems and Knowledge Discovery* (2006), pp. 1201–1204 (cit. on pp. 40, 46).

[84] Gulik, R. van and Vignoli, F.: "Visual Playlist Generation on the Artist Map." In: [205]. 2005, pp. 520–523 (cit. on p. 100).

[85] Hansen, D. W., Skovsgaard, H. H. T., Hansen, J. P., and Møllenbach, E.: "Noise tolerant selection by gaze-controlled pan and zoom in 3D." In: *Proceedings of the 2008 symposium on Eye Tracking Research & Applications (ETRA'08)*. 2008, pp. 205–212 (cit. on p. 173).

[86] Härdle, W. and Simar, L.: *Applied multivariate statistical analysis*. Springer Verlag, 2007 (cit. on pp. 57, 200).

[87] Harte, C. and Sandler, M.: "Automatic chord identification using a quantised chromagram." In: *Proceedings of the 118th Audio Engineering Society's Convention*. 2005, pp. 291–301 (cit. on pp. 34, 67, 71).

[88] Harte, C., Sandler, M. B., Abdallah, S. A., and Gómez, E.: "Symbolic Representation of Musical Chords: A Proposed Syntax for Text Annotations." In: [205]. 2005, pp. 66–71 (cit. on pp. 141, 195).

[89] Herrera, P., Resa, Z., and Sordo, M.: "Rocking around the clock eight days a week: an exploration of temporal patterns of music listening." In: *1st Workshop On Music Recommendation And Discovery (WOMRAD), ACM RecSys*. 2010 (cit. on p. 94).

[90] Herrera, P., Serra, X., and Peeters, G.: "Audio descriptors and descriptor schemes in the context of MPEG-7." In: *International Computer Music Conference (ICMC'99)*. 1999 (cit. on p. 24).

[91] Hirata, K., Tzanetakis, G., and Yoshii, K., eds.: *ISMIR 2009, 10th International Conference on Music Information Retrieval, Kobe, Japan, October 26-30, 2009, Proceedings*. Kobe, Japan, 2009 (cit. on pp. 217, 220–227, 233).

[92] Hirjee, H. and Brown, D.: "Automatic detection of internal and imperfect rhymes in rap lyrics." In: [91]. 2009 (cit. on p. 27).

[93] Hitchner, S., Murdoch, J., and Tzanetakis, G.: "Music Browsing using a TableTop Display." In: [55]. 2007, pp. 175–176 (cit. on pp. 41, 46).

[94] Hoashi, K., Matsumoto, K., and Inoue, N.: "Personalization of user profiles for content-based music retrieval based on relevance feedback." In: *Proceedings of the 11th ACM international conference on Multimedia*. 2003, pp. 110–119 (cit. on pp. 37, 47).

[95] Hoos, H. H. and Bainbridge, D., eds.: *ISMIR 2003, 4th International Conference on Music Information Retrieval, Baltimore, Maryland, USA, October 27-30, 2003, Proceedings*. Baltimore (MD), US: The Johns Hopkins University, 2003 (cit. on pp. 213, 228, 231, 233).

[96] Hu, X., Downie, J. S., and Ehmann, A. F.: "Exploiting Recommended Usage Metadata: Exploratory Analyses." In: [124]. 2006, pp. 19–22 (cit. on p. 76).

[97] Huron, D.: *Listening Styles and Listening Strategies*. Society for Music Theory 2002 Conference. Columbus, Ohio, 2002 (cit. on pp. 20, 83).

[98] Indyk, P. and Motwani, R.: "Approximate nearest neighbors: towards removing the curse of dimensionality." In: *Proceedings of the 13th ACM Symposium on Theory of Computing (STOC '98)*. 1998, pp. 604–613 (cit. on p. 107).

[99] Ioannou, S., Caridakis, G., Karpouzis, K., and Kollias, S.: "Robust feature detection for facial expression recognition." In: *Journal on Image and Video Processing* 2007.2 (2007), pp. 5–5 (cit. on p. 85).

[100] Istance, H., Bates, R., Hyrskykari, A., and Vickers, S.: "Snap clutch, a moded approach to solving the Midas touch problem." In: *Proceedings of the 2008 symposium on Eye Tracking Research & Applications (ETRA'08)*. 2008, pp. 221–228 (cit. on pp. 173, 180).

[101] Jacob, R. J. K.: "What you look at is what you get: eye movement-based interaction techniques." In: *Proceedings of CHI'90*. 1990, pp. 11–18 (cit. on p. 173).

[102] Jacob, R. J. K.: "What you look at is what you get: eye movement-based interaction techniques." In: *Proceedings of the SIGCHI conference on Human factors in computing systems: Empowering people*. 1990, pp. 11–18 (cit. on p. 173).

[103] Jolliffe, I. T.: *Principal Component Analysis*. Springer Verlag, 2002 (cit. on p. 99).

[104] Jones, S., Cunningham, S. J., and Jones, M.: "Organizing digital music for use: an examination of personal music collections." In: [132]. 2004 (cit. on pp. 75, 83).

[105] Julia, C. F. and Jorda, S.: "SongExplorer: a tabletop application for exploring large collections of songs." In: [91]. 2009, pp. 675–680 (cit. on pp. 41, 46).

[106] Kassler, M.: "Toward musical information retrieval." In: *Perspectives of New Music* 4.2 (1966), pp. 59–67 (cit. on p. 7).

[107] Kim, Y. E., Schmidt, E. M., Migneco, R., Morton, B. G., Richardson, P., Scott, J., Speck, J. A., and Turnbull, D.: "Music Emotion Recognition: A State of the Art Review." In: [59]. 2010, pp. 255–266 (cit. on p. 18).

[108] Knees, P., Pohle, T., Schedl, M., and Widmer, G.: "Exploring Music Collections in Virtual Landscapes." In: *IEEE MultiMedia* 14.3 (2007), pp. 46–54 (cit. on pp. 41, 46, 100–101, 141–142, 194).

[109] Kohonen, T.: "Self-organized formation of topologically correct feature maps." In: *Biological cybernetics* 43.1 (1982), pp. 59–69 (cit. on p. 53).

[110] Köstler, A.: *The Act of Creation*. Macmillan, 1964 (cit. on pp. 164, 166).

[111] Kötter, T., Thiel, K., and Berthold, M.: "Domain Bridging Associations Support Creativity." In: *Proceedings 1st international Conference on Computational Creativity (ICCC'10)*. 2010, pp. 200–204 (cit. on p. 166).

[112] Kubek, M. and Nützel, J.: "Novel Interactive Music Search Techniques." In: *Proceedings of the 7th International Workshop for Technical, Economic and Legal Aspects of Business Models for Virtual Goods incorporating the 5th International ODRL Workshop (VG'09)*. 2009, pp. 49–59 (cit. on p. 167).

[113] Kulis, B. and Grauman, K.: "Kernelized Locality-Sensitive Hashing for Scalable Image Search." In: *Proceedings 12th international Conference on Computer Vision (ICCV'09)*. 2009 (cit. on p. 107).

[114] Lampinen, J. and Oja, E.: "Clustering properties of hierarchical self-organizing maps." In: *Journal of Mathematical Imaging and Vision* 2.2 (1992), pp. 261–272 (cit. on p. 55).

[115] Laplante, A.: "Users' Relevance Criteria in Music Retrieval in Everyday Life: An Exploratory Study." In: [59]. 2010, pp. 601–606 (cit. on p. 13).

[116] Lartillot, O. and Toiviainen, P.: "MIR in Matlab (II): A toolbox for musical feature extraction from audio." In: [55]. 2007, pp. 237–244 (cit. on p. 34).

[117] Law, E. and Ahn, L. von: "Input-agreement: a new mechanism for collecting data using human computation games." In: *Proceedings CHI '09*. 2009, pp. 1197–1206 (cit. on p. 147).

[118] Lee, J. and Lee, J.: "Music for My Mood: A Music Recommendation System Based on Context Reasoning." In: *Smart Sensing and Context* (2006), pp. 190–203 (cit. on pp. 39, 47).

[119] Lee, J. H. and Downie, J. S.: "Survey Of Music Information Needs, Uses, And Seeking Behaviours: Preliminary Findings." In: [132]. 2004 (cit. on pp. 14, 76).

[120] Lee, J. H., Jones, M. C., and Downie, J. S.: "An Analysis of ISMIR Proceedings: Patterns of Authorship, Topic, and Citation." In: [91]. 2009, pp. 57–62 (cit. on p. 7).

[121] Lee, K. and Slaney, M.: "A unified system for chord transcription and key extraction using hidden markov models." In: [55]. 2007, pp. 245–250 (cit. on pp. 34, 46, 71).

[122] Lee, K. and Slaney, M.: "Automatic chord recognition from audio using a supervised HMM trained with audio-from-symbolic data." In: *Proceedings of the 1st ACM workshop on Audio and Music Computing Multimedia (AMCMM'06)*. 2006, pp. 11–20 (cit. on p. 68).

[123] Leitich, S. and Topf, M.: "Globe of Music - Music library visualization using GeoSOM." In: [55]. 2007, pp. 167–170 (cit. on pp. 41, 46).

[124] Lemström, K., Tindale, A., and Dannenberg, R., eds.: *ISMIR 2006, 7th International Conference on Music Information Retrieval, Victoria, Canada, 8-12 October, 2006, Proceedings*. Victoria (BC), CA: University of Victoria, 2006 (cit. on pp. 215, 220–221, 223–224, 226–229).

[125] Lerch, A.: "On the Requirement of Automatic Tuning Frequency Estimation." In: [124]. 2006, pp. 212–215 (cit. on p. 34).

[126] Lesaffre, M., Leman, M., Tanghe, K., De Baets, B., De Meyer, H., and Martens, J.: "User-dependent taxonomy of musical features as a conceptual framework for musical audio-mining technology." In: *Proceedings of the Stockholm Music Acoustics Conference*. 2003, pp. 635–638 (cit. on p. 9).

[127] Lidy, T., Jr., C. N. S., Cornelis, O., Gouyon, F., Rauber, A., Kaestner, C. A. A., and Koerich, A. L.: "On the suitability of state-of-the-art music information retrieval methods for analyzing, categorizing and accessing non-Western and ethnic music collections." In: *Signal Processing* 90.4 (2010), pp. 1032–1048 (cit. on p. 8).

[128] Lillie, A. S.: "MusicBox: Navigating the space of your music." MA thesis. MIT, 2008 (cit. on p. 100).

[129] Little, D., Raffensperger, D., and Pardo, B.: "A Query by Humming System that Learns from Experience." In: [55]. 2007, pp. 335–338 (cit. on pp. 37, 47).

[130] Liu, T.-Y.: *Learning to Rank for Information Retrieval*. Springer, 2011 (cit. on p. 42).

[131] Lloyd, S.: "Automatic Playlist Generation and Music Library Visualisation with Timbral Similarity Measures." MA thesis. Queen Mary University of London, 2009 (cit. on pp. 100–101).

[132] Lomeli Buyoli, C. and Loureiro, R., eds.: *ISMIR 2004, 5th International Conference on Music Information Retrieval, Barcelona, Spain, October 10-14, 2004, Proceedings*. Barcelona, SP: Universitat Pompeu Fabra, 2004 (cit. on pp. 222–223, 233–234).

[133] Loscos, A., Wang, Y., and Boo, W. J. J.: "Low Level Descriptors for Automatic Violin Transcription." In: [124]. 2006, pp. 164–167 (cit. on pp. 36, 47).

[134] Lübbers, D.: "SoniXplorer: Combining Visualization and Auralization for Content-Based Exploration of Music Collections." In: [205]. 2005, pp. 590–593 (cit. on pp. 41, 101).

[135] Lübbers, D. and Jarke, M.: "Adaptive Multimodal Exploration of Music Collections." In: [91]. 2009, pp. 195–200 (cit. on pp. 41, 43–44, 46–47, 100–101, 129–130, 194, 196).

[136] Lux, M.: "Caliph & Emir: MPEG-7 photo annotation and retrieval." In: *Proceedings of the 17th ACM international conference on Multimedia*. 2009, pp. 925–926 (cit. on p. 115).

[137] Maddage, N. C., Xu, C., Kankanhalli, M. S., and Shao, X.: "Content-based music structure analysis with applications to music semantics understanding." In: *Proceedings of the 12th ACM international conference on Multimedia*. 2004, pp. 112–119 (cit. on p. 34).

[138] Madsen, K., Nielsen, H. B., and Tingleff, O.: *Methods for Non-Linear Least Squares Problems (2nd ed.)* 2004 (cit. on p. 50).

[139] Magas, M., Casey, M., and Rhodes, C.: "mHashup: fast visual music discovery via locality sensitive hashing." In: *SIGGRAPH '08: ACM SIGGRAPH 2008 new tech demos*. 2008, pp. 1–1 (cit. on p. 167).

[140] Mandel, M., Poliner, G., and Ellis, D.: "Support vector machine active learning for music retrieval." In: *Multimedia systems* 12.1 (2006), pp. 3–13 (cit. on pp. 37, 47).

[141] Mandel, M. and Ellis, D.: "Song-Level Features and Support Vector Machines for Music Classification." In: [205]. 2005, pp. 594–599 (cit. on p. 25).

[142] Mandl, T. and Womser-Hacker, C.: "Learning to cope with Diversity in Music Retrieval." In: [67]. 2002 (cit. on p. 42).

[143] Marks, R.: *Handbook of Fourier analysis & its applications*. Oxford University Press, USA, 2009 (cit. on p. 15).

[144] Marolt, M.: "Probabilistic Segmentation and Labeling of Ethnomusicological Field Recordings." In: [91]. 2009, pp. 75–80 (cit. on pp. 34, 46).

[145] Martín H., J. A., Lope, J., and Maravall, D.: "Adaptation, anticipation and rationality in natural and artificial systems: computational paradigms mimicking nature." In: *Natural Computing: an international journal* 8.4 (2009), pp. 757–775 (cit. on p. 29).

[146] Martinez, J., Koenen, R., and Pereira, F.: "MPEG-7: The Generic Multimedia Content Description Standard, Part 1." In: *IEEE MultiMedia* 9.2 (2002), pp. 78–87 (cit. on p. 115).

[147] Mauch, M. and Dixon, S.: "Approximate Note Transcription for the Improved Identification of Difficult Chords." In: [59]. 2010, pp. 135–140 (cit. on p. 35).

[148] Mauch, M. and Dixon, S.: "Simultaneous Estimation of Chords and Musical Context from Audio." In: *IEEE Transactions on Audio, Speech, and Language Processing* 18.6 (2010), pp. 1280–1289 (cit. on pp. 34, 73).

[149] Mayer, R., Neumayer, R., and Rauber, A.: "Rhyme and style features for musical genre classification by song lyrics." In: [13]. 2008, pp. 337–342 (cit. on p. 27).

[150] McEnnis, D., McKay, C., Fujinaga, I., and Depalle, P.: "jAudio: An Feature Extraction Library." In: [205]. 2005, pp. 600–603 (cit. on pp. 61, 141, 195).

[151] McFee, B., Barrington, L., and Lanckriet, G.: "Learning Similarity from Collaborative Filters." In: [59]. 2010, pp. 345–350 (cit. on pp. 26, 43, 47, 129).

[152] McFee, B. and Lanckriet, G.: "Heterogeneous Embedding for Subjective Artist Similarity." In: [91]. 2009, pp. 513–518 (cit. on pp. 43, 47, 129, 133).

[153] *McGraw-Hill Dictionary of Scientific & Technical Terms*. 6th Edition. The McGraw-Hill Companies, Inc., 2003 (cit. on p. 26).

[154] McKay, C., Burgoyne, J. A., Hockman, J., Smith, J. B. L., Vigliensoni, G., and Fujinaga, I.: "Evaluating the genre classification performance of lyrical features relative to audio, symbolic and cultural features." In: [59]. 2010, pp. 213–218 (cit. on p. 13).

[155] McKay, C. and Fujinaga, I.: "Musical genre classification: Is it worth pursuing and how can it be improved?" In: [124]. 2006, pp. 101–106 (cit. on p. 75).

[156] McNee, S. M., Riedl, J., and Konstan, J. A.: "Being accurate is not enough: how accuracy metrics have hurt recommender systems." In: *Proceedings of CHI '06 - Extended abstracts*. 2006, pp. 1097–1101 (cit. on p. 163).

[157] McNee, S. M., Riedl, J., and Konstan, J. A.: "Making recommendations better: an analytic model for human-recommender interaction." In: *Proceedings of CHI '06 - Extended abstracts*. 2006, pp. 1103–1108 (cit. on p. 163).

[158] Michels, K., Klawonn, F., Kruse, R., and Nürnberger, A.: *Fuzzy control: fundamentals, stability and design of fuzzy controllers*. Springer, 2006 (cit. on p. 29).

[159] Miniotas, D., Špakov, O., and MacKenzie, I. S.: "Eye gaze interaction with expanding targets." In: *Proceedings of CHI'04 - Extended abstracts*. 2004, pp. 1255–1258 (cit. on pp. 173–175).

[160] Mitri, G., Uitdenbogerd, A. L., and Ciesielski, V.: "Automatic Music Classification Problems." In: *27th Australasian Computer Science Conference (ACSC'04)*. Vol. 26. 2004, pp. 315–322 (cit. on p. 75).

[161] Moh, Y. and Buhmann, J. M.: "Kernel Expansion for Online Preference Tracking." In: [13]. 2008, pp. 167–172 (cit. on pp. 37, 47).

[162] Montecchio, N. and Orio, N.: "A Discrete Filter Bank Approach To Audio To Score Matching For Polyphonic Music." In: [91]. 2009, pp. 495–500 (cit. on pp. 34, 46).

[163] Mörchen, F., Ultsch, A., Nöcker, M., and Stamm, C.: "Databionic Visualization of Music Collections According to Perceptual Distance." In: [205]. 2005, pp. 396–403 (cit. on pp. 41, 46, 100–101).

[164] Müller, M., Konz, V., Scharfstein, A., Ewert, S., and Clausen, M.: "Towards Automated Extraction of Tempo Parameters from Expressive Music Recordings." In: [91]. 2009, pp. 69–74 (cit. on pp. 35, 46).

[165] Neumayer, R., Dittenbach, M., and Rauber, A.: "PlaySOM and PocketSOMPlayer, Alternative Interfaces to Large Music Collections." In: [205]. 2005, pp. 618–623 (cit. on pp. 41, 46, 101).

[166] Nielsen, J.: "Usability Engineering." In: *The Computer Science and Engineering Handbook*. Ed. by Tucker, A. B. CRC Press, 1997, pp. 1440–1460 (cit. on p. 114).

[167] Nielsen, M., Störring, M., Moeslund, T., and Granum, E.: "A Procedure for Developing Intuitive and Ergonomic Gesture Interfaces for HCI." In: *Gesture-Based Communication in Human-Computer Interaction*. Ed. by Camurri, A. and Volpe, G. Vol. 2915. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2004, pp. 105–106 (cit. on p. 176).

[168] Niitsuma, M., Takaesu, H., Demachi, H., Oono, M., and Saito, H.: "Development of an Automatic Music Selection System Based on Runner's Step Frequency." In: [13]. 2008, pp. 193–198 (cit. on p. 40).

[169] Nishimura, T., Hashiguchi, H., Takita, J., Zhang, J. X., Goto, M., and Oka, R.: "Music Signal Spotting Retrieval by a Humming Query Using Start Frame Feature Dependent Continuous Dynamic Programming." In: [61]. 2001, pp. 211–218 (cit. on p. 62).

[170] Noland, K. and Sandler, M. B.: "Key Estimation Using a Hidden Markov Model." In: [124]. 2006, pp. 121–126 (cit. on pp. 34, 46, 73).

[171] Nürnberger, A. and Detyniecki, M.: "Adaptive Multimedia Retrieval: From Data to User Interaction." In: *Do Smart Adaptive Systems Exist?* Ed. by Gabrys, B., Leiviskä, K., and Strackeljan, J. Vol. 173. Studies in Fuzziness and Soft Computing. Springer Berlin / Heidelberg, 2005, pp. 341–370 (cit. on p. 21).

[172] Nürnberger, A. and Detyniecki, M.: "Externally growing self-organizing maps and its application to e-mail database visualization and exploration." In: *Applied Soft Computing* 6 (4 2006), pp. 357–371 (cit. on pp. 55–56, 101, 184).

[173] Nürnberger, A. and Klose, A.: "Improving Clustering and Visualization of Multimedia Data Using Interactive User Feedback." In: *Proceedings of the 9th international Conference on Information Processing and Management of Uncertainty in*

Knowledge-Based Systems (IPMU'02). 2002, pp. 993–999 (cit. on pp. 105, 128).

[174] Oliver, N. and Kreger-Stickles, L.: "PAPA: Physiology and purpose-aware automatic playlist generation." In: [124]. 2006 (cit. on pp. 40, 85).

[175] Oliver, N. and Flores-Mangas, F.: "MPTrain: a mobile, music and physiology-based personal trainer." In: *Proceedings of the 8th conference on Human-Computer Interaction with mobile devices and services (MobileHCI'06)*. 2006, pp. 21–28 (cit. on p. 40).

[176] Orio, N.: "Music Retrieval: A Tutorial and Review." In: *Foundations and Trends in Information Retrieval* 1.1 (2006), pp. 1–90 (cit. on pp. 7, 9–10, 12, 16, 19, 29).

[177] Pachet, F. and Cazaly, D.: "A taxonomy of musical genres." In: *Proceedings of Content-Based Multimedia Information Access (RIAO)*. 2000 (cit. on p. 76).

[178] Pampalk, E., Flexer, A., and Widmer, G.: "Hierarchical organization and description of music collections at the artist level." In: *Proceedings of the 9th European Conference on Research and Advanced Technology for Digital Libraries (ECDL'05)*. 2005, pp. 37–49 (cit. on pp. 41, 46).

[179] Pampalk, E.: "Computational Models of Music Similarity and their Application in Music Information Retrieval." PhD thesis. Vienna, Austria: Vienna University of Technology, 2006 (cit. on pp. 141–142, 194).

[180] Pampalk, E., Dixon, S., and Widmer, G.: "Exploring music collections by browsing different views." In: [95]. 2003, pp. 201–208 (cit. on pp. 41, 46, 101).

[181] Pampalk, E. and Gasser, M.: "An Implementation of a Simple Playlist Generator Based on Audio Similarity Measures and User Feedback." In: [124]. 2006, pp. 389–390 (cit. on pp. 39, 47).

[182] Pampalk, E. and Goto, M.: "MusicRainbow: A New User Interface to Discover Artists Using Audio-based Similarity and Web-based Labeling." In: [124]. 2006, pp. 367–370 (cit. on p. 165).

[183] Pampalk, E. and Goto, M.: "Musicsun: A new approach to artist recommendation." In: [55]. 2007, pp. 101–104 (cit. on p. 165).

[184] Pampalk, E., Pohle, T., and Widmer, G.: "Dynamic Playlist Generation Based on Skipping Behavior." In: [205]. 2005, pp. 634–637 (cit. on pp. 39, 47).

[185] Pampalk, E., Rauber, A., and Merkl, D.: "Content-based organization and visualization of music archives." In: *Proceedings of the 10th ACM international Conference on Multimedia*. 2002, pp. 570–579 (cit. on pp. 41, 46, 101).

[186] Papadopoulos, H. and Peeters, G.: "Large-scale study of chord estimation algorithms based on chroma representation and hmm." In: *Proceedings of the International Workshop on Content-Based Multimedia Indexing (CBMI)*. 2007, pp. 53–60 (cit. on pp. 25, 34, 46, 68, 73).

[187] Park, H.-S., Yoo, J.-O., and Cho, S.-B.: "A Context-Aware Music Recommendation System Using Fuzzy Bayesian Networks with Utility Theory." In: *Fuzzy Systems and Knowledge Discovery* (2006), pp. 970–979 (cit. on pp. 40, 46).

[188] Paulus, J., Müller, M., and Klapuri, A.: "Audio-based Music Structure Analysis." In: [59]. 2010, pp. 625–636 (cit. on p. 12).

[189] Pauws, S.: "CubyHum: a fully operational "query by humming" system." In: [67]. 2002 (cit. on p. 62).

[190] Pauws, S. and Eggen, B.: "PATS: Realization and user evaluation of an automatic playlist generator." In: [67]. 2002 (cit. on pp. 38, 47).

[191] Pauws, S., Verhaegh, W., and Vossen, M.: "Fast Generation of Optimal Music Playlists using Local Search." In: [124]. 2006, pp. 138–143 (cit. on pp. 38, 47).

[192] Pauws, S. and Wijdeven, S. van de: "User Evaluation of a New Interactive Playlist Generation Concept." In: [205]. 2005, pp. 638–643 (cit. on pp. 38, 47).

[193] Peeters, G.: *A large set of audio features for sound description (similarity and classification) in the CUIDADO project*. 2004 (cit. on p. 24).

[194] Peeters, G.: "Musical key estimation of audio signal based on hidden Markov modeling of chroma vectors." In: *Proceedings of the international Conference on Digital Audio Effects (DAFx'06)*. 2006, pp. 127–131 (cit. on p. 34).

[195] Peeters, G. and Deruty, E.: "Is Music Structure Annotation Multi-Dimensional? A Proposal for Robust Local Music Annotation." In: [11]. 2009, pp. 75–90 (cit. on p. 12).

[196] Pinto, A., Leuken, R. H. van, Demirci, M. F., Wiering, F., and Veltkamp, R. C.: "Indexing Music Collections through Graph Spectra." In: [55]. 2007, pp. 153–156 (cit. on p. 138).

[197] Plate, C., Basselin, N., Kröner, A., Schneider, M., Baldes, S., Dimitrova, V., and Jameson, A.: "Recomindation: New Functions for Augmented Memories." In: *Proceedings of Adaptive Hypermedia and Adaptive Web-Based Systems (AH'06)*. 2006, pp. 141–150 (cit. on p. 163).

[198] Platt, J. C.: "Fast embedding of sparse music similarity graphs." In: *Advances in Neural Information Processing Systems (NIPS'03)*. Vol. 16. 2004 (cit. on p. 170).

[199] Pohle, T., Seyerlehner, K., and Widmer, G.: "An Approach to Automatically Tracking Music Preference on Mobile Players." In: [51]. 2008 (cit. on pp. 39, 47).

[200] Pugin, L., Burgoyne, J. A., and Fujinaga, I.: "MAP Adaptation to Improve Optical Music Recognition of Early Music Documents Using Hidden Markov Models." In: [55]. 2007, pp. 513–516 (cit. on pp. 36, 46).

[201] Quinlan, J. R.: "Induction of Decision Trees." In: *Machine Learning* 1 (1986), pp. 81–106 (cit. on p. 78).

[202] Rauber, A., Pampalk, E., and Merkl, D.: "Using Psycho-Acoustic Models and Self-Organizing Maps to Create a Hierarchical Structuring of Music by Musical Styles." In: [67]. 2002 (cit. on pp. 41, 46).

[203] Ravelli, E., Richard, G., and Daudet, L.: "Fast MIR in a Sparse Transform Domain." In: [13]. 2008, pp. 527–532 (cit. on pp. 34, 46).

[204] Rebelo, A., Capela, G., and Cardoso, J.: "Optical recognition of music symbols - A comparative study." In: *International Journal on Document Analysis and Recognition* 13 (1 2010), pp. 19–31 (cit. on p. 17).

[205] Reiss, J. D. and Wiggins, G. A., eds.: *ISMIR 2005, 6th International Conference on Music Information Retrieval, London, UK, 11-15 September, 2005, Proceedings*. London, UK: University of London, 2005 (cit. on pp. 214, 216, 219–220, 224–229, 233).

[206] Reynolds, G., Barry, D., Burke, T., and Coyle, E.: "Towards a Personal Automatic Music Playlist Generation Algorithm: The Need for Contextual Information." In: *Proceedings of 2nd Audio Mostly Conference: Interaction with Sound (AM'07)*. 2007, pp. 84–89 (cit. on p. 39).

[207] Rocchio, J. J.: "Relevance Feedback in Information Retrieval." In: *The SMART Retrieval System - Experiments in Automatic Document Processing*. 1971, pp. 313–323 (cit. on p. 37).

[208] Rolland, P.: "Adaptive user modeling in a content-based music retrieval system." In: [61]. 2001 (cit. on pp. 42, 45, 47).

[209] Rössl, C. and Theisel, H.: "Streamline Embedding for 3D Vector Field Exploration." In: *Visualization and Computer Graphics, IEEE Transactions on* (2011) (cit. on p. 57).

[210] Salton, G. and Buckley, C.: "Term Weighting Approaches in Automatic Text Retrieval." In: *Information Processing & Management* 24.5 (1988), pp. 513–523 (cit. on pp. 128, 170).

[211] Sandvold, V., Aussenac, T., Celma, Ò., and Herrera, P.: "Good Vibrations: Music Discovery through Personal Musical Concepts." In: [55]. 2006, pp. 322–323 (cit. on p. 37).

[212] Sarmento, L., Gouyon, F., Costa, B., and Oliveira, E.: "Visualizing Networks of Music Artists with RAMA." In: *Proceedings of the international Conference on Web Information Systems and Technologies (WEBIST'09)*. 2009 (cit. on pp. 99, 165).

[213] Schedl, M.: *The CoMIRVA Toolkit for Visualizing Music-Related Data*. Technical Report. Johannes Kepler University Linz, 2006 (cit. on pp. 141, 195).

[214] Schedl, M. and Knees, P.: "Context-based Music Similarity Estimation." In: [11]. 2009, pp. 59–74 (cit. on p. 14).

[215] Schnitzer, D., Flexer, A., Widmer, G., and Gasser, M.: "Islands of Gaussians: The Self Organizing Map and Gaussian Music Similarity Features." In: [59]. 2010, pp. 327–332 (cit. on p. 193).

[216] Schölkopf, B. and Smola, A. J.: *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002 (cit. on pp. 51, 53).

[217] Segond, M. and Borgelt: C.: "Selecting the Links in BisoNets Generated from Document Collections." In: *Advances in Intelligent Data Analysis IX* 6065 (2010), pp. 196–207 (cit. on p. 170).

[218] Selfridge-Field, E.: *Beyond MIDI: The Handbook of Musical Codes*. MIT Press, 1997 (cit. on p. 16).

[219] Sheh, A. and Ellis, D.: "Chord segmentation and recognition using EM-trained hidden Markov models." In: [95]. 2003, pp. 185–191 (cit. on pp. 34, 46, 68).

[220] Shenoy, A. and Wang, Y.: "Key, chord, and rhythm tracking of popular music recordings." In: *Computer Music Journal* 29.3 (2005), pp. 75–86 (cit. on p. 34).

[221] Shirley, P., Ashikhmin, M., Gleicher, M., Marschner, S., Reinhard, E., Sung, K., Thompson, W., and Willemsen, P.: *Fundamentals of computer graphics*. AK Peters, 2002 (cit. on p. 106).

[222] Shneiderman, B.: "Tree visualization with tree-maps: 2-d space-filling approach." In: *ACM Transactions on Graphics* 11.1 (1992), pp. 92–99 (cit. on p. 99).

[223] Shneiderman, B.: "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations." In: *Proceedings of the IEEE Symposium on Visual Languages*. 1996 (cit. on p. 175).

[224] Shoemaker, G. and Gutwin, C.: "Supporting multi-point interaction in visual workspaces." In: *Proceedings of CHI'07*. 2007, pp. 999–1008 (cit. on p. 175).

[225] Silva, V. de and Tenenbaum, J. B.: "Global Versus Local Methods in Nonlinear Dimensionality Reduction." In: *Advances in Neural Information Processing Systems 15 (NIPS'02)*. 2002, pp. 705–712 (cit. on p. 57).

[226] Silva, V. de and Tenenbaum, J. B.: *Sparse multidimensional scaling using landmark points*. Technical Report. Stanford University, 2004 (cit. on pp. 57–58, 104).

[227] Slaney, M., Weinberger, K. Q., and White, W.: "Learning a Metric for Music Similarity." In: [13]. 2008, pp. 313–318 (cit. on pp. 43, 47, 129).

[228] Sordo, M., Celma, Ò., Blech, M., and Guaus, E.: "The Quest for Musical Genres: Do the Experts and the Wisdom of Crowds Agree?" In: [13]. 2008, pp. 255–260 (cit. on p. 75).

[229] Sotiropoulos, D. N., Lampropoulos, A. S., and Tsihrintzis, G. A.: "MUSIPER: a system for modeling music similarity perception based on objective feature subset selection." In: *User Modeling and User-Adapted Interaction* 18.4 (2008), pp. 315–348 (cit. on pp. 42, 46).

[230] Stavness, I., Gluck, J., Vilhan, L., and Fels, S.: "The MUSICtable: A map-based ubiquitous system for social interaction with a digital music collection." In: *Proceedings of international Conference on Entertainment Computing (ICEC'05)*. 2005, p. 291 (cit. on pp. 41, 46).

[231] Stefaner, M. and Muller, B.: "Elastic lists for facet browsers." In: *Database and Expert Systems Applications, International Workshop on* 0 (2007), pp. 217–221 (cit. on p. 81).

[232] Su, M.-Y., Yang, Y.-H., Lin, Y.-C., and Chen, H.: "An Integrated Approach to Music Boundary Detection." In: [59]. 2010, pp. 705–710 (cit. on pp. 34, 46).

[233] Sumi, K., Itoyama, K., Yoshii, K., Komatani, K., Ogata, T., and Okuno, H. G.: "Automatic Chord Recognition Based on Probabilistic Integration of Chord Transition and Bass Pitch Estimation." In: [13]. 2008, pp. 39–44 (cit. on pp. 34, 46).

[234] Thayer, R.: *The biopsychology of mood and arousal*. Oxford University Press, USA, 1990 (cit. on p. 18).

[235] Thomas, V., Fremerey, C., Damm, D., and Clausen, M.: "SLAVE: A Score-Lyrics-Audio-Video-Explorer." In: [91]. 2009 (cit. on p. 15).

[236] Torrens, M., Hertzog, P., and Arcos, J. L.: "Visualizing and Exploring Personal Music Libraries." In: [132]. 2004 (cit. on p. 99).

[237] Typke, R., Wiering, F., and Veltkamp, R. C.: "A Survey of Music Information Retrieval Systems." In: [205]. 2005, pp. 153–160 (cit. on pp. 7, 18–19, 22, 27).

[238] Typke, R., Giannopoulos, P., Veltkamp, R. C., Wiering, F., and Oostrum, R. van: "Using transportation distances for measuring melodic similarity." In: [95]. 2003 (cit. on p. 23).

[239] Tzanetakis, G. and Cook, P.: "MARSYAS: a framework for audio analysis." In: *Organised Sound* 4.3 (1999), pp. 169–175 (cit. on p. 61).

[240] Vignoli, F. and Pauws, S.: "A Music Retrieval System Based on User Driven Similarity and Its Evaluation." In: [205]. 2005, pp. 272–279 (cit. on p. 42).

[241] Vogel, D. and Balakrishnan, R.: "Distant freehand pointing and clicking on very large, high resolution displays." In: *Proceedings of the 18th annual ACM symposium on User Interface Software and Technology (UIST'05)*. 2005, pp. 33–42 (cit. on p. 175).

[242] Volk, A., Garbers, J., Van Kranenburg, P., Wiering, F., Veltkamp, R., and Grijp, L.: "Applying rhythmic similarity based on inner metric analysis to folksong research." In: [55]. 2007, pp. 293–300 (cit. on p. 23).

[243] White, T.: "A Man out of Time Beats the Clock." In: *Musician magazine* 60 (1983), p. 52 (cit. on p. 7).

[244] Whitman, B. and Ellis, D.: "Automatic Record Reviews." In: [132]. 2004 (cit. on p. 99).

[245] Wiering, F.: "Can humans benefit from music information retrieval?" In: *Proceedings of the 4th international workshop on Adaptive Multimedia Retrieval (AMR'06)*. 2006, pp. 82–94 (cit. on p. 27).

[246] Wilkinson, L. and Friendly, M.: "The history of the cluster heat map." In: *The American Statistician* 63.2 (2009), pp. 179–184 (cit. on p. 189).

[247] Williams, C. K. I.: "On a Connection between Kernel PCA and Metric Multidimensional Scaling." In: *Machine Learning* 46.1-3 (2002), pp. 11–19 (cit. on p. 100).

[248] Wolter, K., Bastuck, C., and Gärtner, D.: "Adaptive User Modeling for Content-Based Music Retrieval." In: [51]. 2008 (cit. on pp. 38, 47).

[249] Wooding, D. S.: "Fixation maps: quantifying eye-movement traces." In: *Proceedings of the 2002 symposium on Eye Tracking Research & Applications (ETRA'02)*. 2002, pp. 31–36 (cit. on p. 190).

[250] Yoo, B., Han, J.-J., Choi, C., Yi, K., Suh, S., Park, D., and Kim, C.: "3D user interface combining gaze and hand gestures for large-scale display." In: *Proceedings of CHI'10 - Extended abstracts*. 2010, pp. 3709–3714 (cit. on p. 175).

[251] Yoshioka, T., Kitahara, T., Komatani, K., Ogata, T., and Okuno, H. G.: "Automatic Chord Transcription with Concurrent Recognition of Chord Symbols and Boundaries." In: [132]. 2004 (cit. on p. 68).

[252] You, W. and Dannenberg, R.: "Polyphonic music note onset detection using semi-supervised learning." In: [55]. 2007, pp. 279–282 (cit. on pp. 34, 46).

[253] Zhai, S., Morimoto, C., and Ihde, S.: "Manual and gaze input cascaded (MAGIC) pointing." In: *Proceedings of CHI'99*. 1999, pp. 246–253 (cit. on p. 173).

[254] Zhang, X., Ren, X., and Zha, H.: "Improving eye cursor's stability for eye pointing tasks." In: *Proceedings of CHI'08*. 2008, pp. 525–534 (cit. on pp. 175, 179).

[255] Zhu, Y. and Kankanhalli, M.: "Precise pitch profile feature extraction from musical audio for key detection." In: *IEEE Transactions on Multimedia* 8.3 (2006), pp. 575–584 (cit. on p. 34).

WEBSITES

All URLs have been verified on September 16, 2011.

[url:1] *AmpliFIND*. URL: http://www.amplifindmusicservices.com (cit. on p. 169).

[url:2] *Audacity – The Free, Cross-Platform Sound Editor*. URL: http://audacity.sourceforge.net (cit. on p. 63).

[url:3] *Audioscrobbler*. URL: http://www.audioscrobbler.net (cit. on pp. 76, 84).

[url:4] *blogger*. URL: http://www.blogger.com (cit. on p. 89).

[url:5] CASTAN, G.: *Musical Notation Codes*. URL: http://www.music-notation.info/en/compmus/notationformats.html (cit. on p. 16).

[url:6] *CeBIT*. URL: http://www.cebit.de (cit. on pp. 87, 114).

[url:7] *Cumulative ISMIR Proceedings*. URL: http://www.ismir.net/proceedings (cit. on p. 7).

[url:8] *delicious*. URL: http://www.delicious.com/ (cit. on p. 89).

[url:9] *Elastic Lists Online Demo*. URL: http://well-formed-data.net/experiments/elastic_lists (cit. on p. 81).

[url:10] *facebook*. URL: http://www.facebook.com (cit. on p. 89).

[url:11] *flickr*. URL: http://www.flickr.com (cit. on p. 89).

[url:12] *fm4 Soundpark*. URL: http://fm4.orf.at/soundpark (cit. on p. 100).

[url:13] *globalmusic2one*. URL: http://www.globalmusic2one.net (cit. on p. 167).

[url:14] *Google Answers*. URL: http://answers.google.com/answers (cit. on p. 76).

[url:15] *Google Docs*. URL: http://docs.google.com (cit. on p. 89).

[url:16] *Google Mail*. URL: http://mail.google.com (cit. on p. 89).

[url:17] *Google Maps*. URL: http://maps.google.com (cit. on pp. 115, 119).

[url:18] *Information Miner*. URL: http://fuzzy.cs.uni-magdeburg.de/wiki/pmwiki.php?n=Forschung.InformationMiner2 (cit. on pp. 68, 78).

[url:19] *Institut de Recherche et Coordination Acoustique/Musique (IRCAM)*. URL: http://www.ircam.fr (cit. on p. 88).

[url:20] *International Society for Music Information Retrieval*. URL: http://www.ismir.net (cit. on p. 7).

[url:21] *iTunes Genius*. URL: http://www.apple.com/itunes (cit. on p. 84).

[url:22] *Last.fm*. URL: http://www.last.fm (cit. on pp. 14, 76, 84, 89, 171).

[url:23] *Last.fm API*. URL: http://www.last.fm/api (cit. on pp. 99, 165).

[url:24] *Last.fm Artist Map*. URL: http://sixdegrees.hu/last.fm/interactive_map.html (cit. on pp. 99, 165).

[url:25] *Last.fm Listening Clock visualization*. URL: http://playground.last.fm/demo/clock (cit. on pp. 94–95).

[url:26] *LimeSurvey*. URL: http://www.limesurvey.org (cit. on p. 87).

[url:27] *LinkedIn*. URL: http://www.linkedin.com (cit. on p. 89).

[url:28] *LyricWiki*. URL: http://lyricwiki.org (cit. on pp. 141, 195).

[url:29] *Magnatagatune Dataset*. URL: http://tagatune.org/Magnatagatune.html (cit. on p. 147).

[url:30] *Magnatune*. URL: http://magnatune.com/ (cit. on p. 147).

[url:31] *mHashup*. URL: http://www.mhashup.com (cit. on p. 167).

[url:32] *MIDI Manufacturers Association Incorporated - Introduction to MIDI*. URL: http://www.midi.org/aboutmidi (cit. on p. 16).

[url:33] Müllensiefen, D. and Frieler, K.: *The Simile algorithms documentation 0.3*. URL: http://doc.gold.ac.uk/isms/mmm/SIMILE_algo_docs_0.3.pdf (cit. on p. 138).

[url:34] *Music Information Retrieval Evaluation eXchange (MIREX) Wiki*. URL: http://www.music-ir.org/mirex (cit. on p. 13).

[url:35] *music-ir mailing list*. URL: http://listes.ircam.fr/wws/info/music-ir (cit. on p. 88).

[url:36] *MusicBrainz*. URL: http://musicbrainz.org (cit. on pp. 99, 169).

[url:37] *MusicBrainz Advanced Relationship Types*. URL: http://wiki.musicbrainz.org/Category:Relationship_Type (cit. on p. 169).

[url:38] *Myspace*. URL: http://myspace.com (cit. on p. 171).

[url:39] *Neo4j graph database engine*. URL: http://neo4j.org (cit. on p. 169).

[url:40] *Nike+iPod Sport Kit*. URL: http://www.apple.com/ipod/nike (cit. on p. 40).

[url:41] Pollack, A. W.: *Notes on The Beatles*. URL: http://www.icce.rug.nl/~soundscapes/DATABASES/AWP/awp-notes_on.shtml (cit. on pp. 141, 195).

[url:42] *Sonic Annotator*. URL: http://omras2.org/SonicAnnotator (cit. on p. 61).

[url:43] *SQLite database engine*. URL: http://www.sqlite.org (cit. on p. 77).

[url:44] Sreedhar, S.: *IEEE Spectrum (Online) - The Future of Music, Part One: Tearing Down the Wall of Noise*. 2007. URL: http://spectrum.ieee.org/computing/software/the-future-of-music (cit. on p. 11).

[url:45] *StudiVZ*. URL: http://www.studivz.net (cit. on p. 89).

[url:46] *TagATune*. URL: http://tagatune.org/Home.html (cit. on p. 147).

[url:47] *The EchoNest*. URL: http://the.echonest.com (cit. on pp. 61, 75, 171).

[url:48] *The EchoNest API*. URL: http://developer.echonest.com (cit. on pp. 99, 165).

[url:49] *Track API – Analyze Documentation*. URL: http://developer.echonest.com/docs/v4/track.html#analyze (cit. on p. 148).

[url:50] *twitter*. URL: http://twitter.com (cit. on pp. 85, 89).

[url:51] *(Unofficial) Dynamic Range Database*. URL: http://www.dr.loudness-war.info (cit. on p. 11).

[url:52] *wakoopa*. URL: http://wakoopa.com (cit. on pp. 85, 89).

[url:53] *Wikipedia*. URL: http://www.wikipedia.org (cit. on pp. 7, 99, 141, 195).

[url:54] *WordPress*. URL: http://wordpress.com (cit. on p. 89).

[url:55] *XING*. URL: http://www.xing.com (cit. on p. 89).

[url:56] *Yamaha BODiBEAT music player*. URL: http://www.yamaha.com/bodibeat (cit. on pp. 40, 85).

Wir können alles schaffen
Genau wie die toll dressierten Affen
Wir müssen nur wollen

"Müssen nur wollen"
Wir sind Helden

SELBSTÄNDIGKEITSERKLÄRUNG

Ich erkläre hiermit, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Insbesondere habe ich nicht die Hilfe eines kommerziellen Promotionsberaters in Anspruch genommen. Dritte haben von mir weder unmittelbar noch mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form als Dissertation eingereicht und ist als Ganzes auch noch nicht veröffentlicht.

Magdeburg, November 2011

Sebastian Stober

Sebastian Stober

Music Information Retrieval (MIR) systems have to deal with multi-faceted music information and very heterogeneous users. Especially when the task is to organize a music collection, the diverse perspectives of users caused by their different level of expertise, musical background or taste pose a great challenge. This challenge is addressed in this book by proposing adaptive methods for several elements of MIR systems: Data-adaptive feature extraction techniques are described that aim to increase the quality and robustness of the information extracted from audio recordings. The classical genre classification problem is approached from a novel user-centric perspective – promoting the idea of idiosyncratic genres that better reflect a user's personal listening habits. An adaptive visualization technique for exploration and organization of music collections is elaborated that especially addresses the common and inevitable problem of projection errors introduced by dimensionality reduction approaches. Furthermore, it is outlined how this technique can be applied to facilitate serendipitous music discoveries in a recommendation scenario and to enable novel gaze-supported interaction techniques. Finally, a general approach for adaptive music similarity is presented which serves as the core of many adaptive MIR applications. Application prototypes demonstrate the usability of the described approaches.

OTTO VON GUERICKE
UNIVERSITÄT
MAGDEBURG

INF

FAKULTÄT FÜR
INFORMATIK