Artist to
Business to Business
to Consumer
Audio Branding System

www.abcdj.eu

D4.7 Final Research Report on Auto-Tagging of Music

Project Reference	688122 — ABC_DJ — H2020-ICT-2015
Deliverable/WP/Task	D4.7 / WP4 / T4.2
Delivery Date	31/12/2018
Main Author(s)	Diemo Schwarz, IRCAM,schwarz@ircam.fr Geoffroy Peeters, IRCAM, peeters@ircam.fr
Co-Author(s)	Alice Cohen-Hadria, IRCAM, alice.cohen-hadria@ircam.fr Dominique Fourer, IRCAM, dominique.fourer@ircam.fr Ugo Marchand, IRCAM, ugo.marchand@ircam.fr Rémi Mignot, IRCAM, remi.mignot@ircam.fr Frédéric Cornu, IRCAM, cornu@ircam.fr Pierre Laffitte, IRCAM, pierre.laffitte@ircam.fr Daniel Schindler, HDIS, daniel.schindler@heardis.com Robin Hofmann, HDIS, robin.hoffmann@heardis.com Rino Spadaveccia, HDIS, rino.spadavecchia@heardis.com
Quality Assurance	Hauke Egermann, TUB, hauke.egermann@york.ac.uk Borja Torres, LMONK, borja@lovemonk.net Richard Wages, TUB, richard.wages@tu-berlin.de
Filename	D4.7_ABC_DJ_Final_Research_Report_on_Auto-Tagging_of_Music.pdf
Publication Level	PU

ABC_DJ - Artist-to-Business-to-Business-to-Consumer Audio Branding System
contact: www.abcdj.eu
email: info@abcdj.eu

Disclaimer

Neither the ABC_DJ consortium as a whole, nor a certain party of the ABC_DJ consortium warrant that the information contained in this document is capable of use, nor that use of the information is free from risk, and accepts no liability for loss or damage suffered by any person using this information.

Neither the European Commission, nor any person acting on behalf of the Commission, is responsible for any use which might be made of the information in this document.

The views expressed in this document are those of the authors and do not necessarily reflect the policies of the European Commission.

Project Information

Full project title	ABC_DJ — Artist-to-Business-to-Business-to-Consumer Audio Branding System
Project Coordinator	Stefan Weinzierl / TU Berlin
Project ID	688122 — ABC DJ — H2020-ICT-2015

Acknowledgements

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 688122.

Table of Contents

History ... **6**

Glossary ... **7**

Executive Summary ... **8**

1 Introduction ... **9**

1.1 "Soft" and "Hard" Features ... **9**

1.2 Organization of the Deliverable ... **11**

2 Auto-Tagging System for "Soft" Features ... **12**

2.1 New Audio Features for Rhythm Description ... **12**

2.1.1 Introduction ... 12

2.1.2 The 2D Modulation Scale Spectrum ... 12

2.1.3 The Modulation Scale Spectrum with Auditory Statistics (MASSS) ... 12

2.1.4 Experiments ... 13

2.1.5 Conclusion and Future Work ... 14

2.1.6 References ... 14

2.2 Extension and Adaptation of ircamclass for "Soft" Features ... **15**

2.2.1 Description of the Given Tasks ... 15

2.2.2 Description of the Algorithms Used ... 17

2.2.3 Audio Augmentation ... 17

2.2.4 Audio Segmentation ... 17

2.2.5 Results of Classification ... 18

2.2.6 Conclusion ... 21

3 Auto-Tagging System for "Hard" Features ... **22**

3.1 Auto-Tagging System for HearDis! "Hard" Features ... **22**

3.2 Auto-Tagging System for TU Berlin "Hard" Features ... **22**

3.2.1 Structure Features ... 23

3.2.2 Rhythm Features ... 23

3.2.3 Key Features ... 23

3.2.4 Chord Features ... 24

3.2.5 Melody Features ... 25

3.2.6 Timbre Set-A Features ... 27

3.2.7 Timbre Set-B Features ... 27

3.2.8 Timbre Set-C Features ... 31

3.3 Main Melody Estimation with Recurrent Neural Networks ... **33**

3.3.1 State of the Art ... 33

3.3.2 Method: Harmonic CQT ... 33

3.3.3 Deep Salience Representation with Convolutional Networks 34
3.3.4 Temporal Modelling with Recurrent Neural Networks 34
3.3.5 Modelling Long-Dependencies with Stateful RNN 35
3.3.6 RNN Results 35
3.3.7 Stateful RNN 38
3.4 References 39
4 Audio Quality Characterization 41
4.1 Introduction 41
4.2 Overview of the Proposed Method 41
4.3 Considered Audio Signal Alteration Effects 42
4.4 Audio Quality Features 42
4.5 Numerical Results 43
4.5.1 Audio Signal Alteration Prediction 43
4.5.2 Music Decade Prediction 44
4.6 Conclusions 44
4.7 References 44
5 Specific Audio Content Based Features in Order to Improve Music Mixes 46
5.1 Blind Audio Source Separation (BASS) for Singing Voice Characterization 46
5.1.1 Introduction 46
5.1.2 Approach 1: Source Separation by AM-FM Time-Frequency Clustering 47
5.1.2.1 Audio signal modelling and local modulation estimation 48
5.1.2.2 Blind source separation of synthetic signals 49
5.1.2.3 Blind source separation of real-world music mixture 50
5.1.2.4 Application to harmonic/percussive source separation (HPSS) 50
5.1.2.5 Conclusion and Future Work 51
5.1.3 Approach 2: Source Separation through Time-Frequency Representation Morphological Filtering 51
5.1.3.1 Towards a training method for supervised KAM-based source separation 52
5.1.3.2 Source separation results through morphological clustering 54
5.1.3.3 Conclusion and Future Work 54
5.1.3.4 References 55
5.1.4 Approach 3: Blind Source Separation of Stereo Mixtures Using Synchrosqueezing 56
5.1.4.1 Synchrosqueezing transform 56
5.1.4.2 Numerical results 57
5.1.4.3 Conclusion and future works 58
5.1.4.4 References 58
5.2 Convolutional Neural Network (ConvNet) for Singing Voice 59
5.2.1 Unsupervised Method based on Blind Audio Source Separation 59
5.2.2 Supervised Method based on ConvNet 59
5.2.3 Results 60
5.2.4 Conclusions and Future Work 61
5.2.5 References 61

5.3 Convolutional Neural Network (ConvNet) for Music Structure Boundaries 62
5.3.1 Introduction 62
5.3.2 Proposed Method 63
5.3.3 Evaluation of the Proposed Method 64
5.3.4 References 65
5.4 Cue Point Estimation 66
5.4.1 Human Expert Rules 66
5.4.2 Ground Truth Database of Cue Points 67
5.4.3 Comparison of Ground Truth Cue Points with Music Structure Analysis 68
5.4.4 Cue Point Estimation Heuristic Algorithm 70
5.4.5 First Evaluation of Cue Point Detection 71
5.4.6 Second Iteration 72
5.4.7 Implementation and Data Format for Cue Points 74
5.4.8 Conclusions of Cue Point Estimation 75
5.4.9 References 75
6 Summary and Conclusion 76

History

Version	Name	Date	Remark
V0.1	Diemo Schwarz	2018-12-12	Updated from D4.3
V0.2	Diemo Schwarz	2018-12-16	Integrate melody estimation
V0.3	Diemo Schwarz	2018-12-18	Integrate cue region estimation from D4.6
V0.4	Diemo Schwarz	2018-12-19	Update audio quality and references
V0.5	Dominique Fourer	2018-12-23	Update blind source separation
V0.6	Diemo Schwarz	2018-12-28	Integrate first revisions
V0.7	Diemo Schwarz	2018-12-29	Integrate second revisions, normalise formatting
V1.0	Diemo Schwarz	2018-12-31	Submitted to EC

Glossary

Acronym/Abbreviation	Full Name/Description
ABC_DJ	Artist-to-Business-to-Business-to-Consumer Audio Branding System
BASS	Blind Audio Source Separation
CNN	Convolutional Neural Network
HPSS	Harmonic/Percussion Source Separation
ISP	In-Store Player
KAM	Kernel Additive Modelling
MASSS	Modulation Scale Spectrum with Auditory Statistics
MFCCs	Mel Frequency Cepstral Coefficients
MLM	Music Library Manager
RNN	Recurrent Neural Network
AM	Amplitude Modulation
FM	Frequency Modulation
SSM	Self-similarity Matrix
PLG	Playlist Generator

Executive Summary

The deliverable D4.7 concerns the work achieved by IRCAM until M36 for the "auto-tagging of music". The deliverable is a research report. The software libraries resulting from the research have been integrated into Fincons/HearDis! Music Library Manager or are used by TU Berlin. The final software libraries are described in D4.5.

The research work on auto-tagging has concentrated on four aspects:

1) Further improving IRCAM's machine-learning system ircamclass. This has been done by developing the new MASSS audio features, including audio augmentation and audio segmentation into ircamclass. The system has then been applied to train HearDis! "soft" features (Vocals-1, Vocals-2, Pop-Appeal, Intensity, Instrumentation, Timbre, Genre, Style). This is described in Part 3.
2) Developing two sets of "hard" features (i.e. related to musical or musicological concepts) as specified by HearDis! (for integration into Fincons/HearDis! Music Library Manager) and TU Berlin (as input for the prediction model of the GMBI attributes). Such features are either derived from previously estimated higher-level concepts (such as structure, key or succession of chords) or by developing new signal processing algorithm (such as HPSS) or main melody estimation. This is described in Part 4.
3) Developing audio features to characterize the audio quality of a music track. The goal is to describe the quality of the audio independently of its apparent encoding. This is then used to estimate audio degradation or music decade. This is to be used to ensure that playlists contain tracks with similar audio quality. This is described in Part 5.
4) Developing innovative algorithms to extract specific audio features to improve music mixes. So far, innovative techniques (based on various Blind Audio Source Separation algorithms and Convolutional Neural Network) have been developed for singing voice separation, singing voice segmentation, music structure boundaries estimation, and DJ cue-region estimation. This is described in Part 6.

1 Introduction

The deliverable D4.7 concerns the work achieved by IRCAM until M36 for the task 4.2 “auto-tagging of music”. According to Annex I, the objectives of this task are the following:

1) Development of innovative algorithms for audio content-based auto-tagging for automatic recognition of music features as defined in T2.4 (‘hard’ and ‘soft’: instrumentation, tonality, bpm, genre, emotion), to be defined in WP3 and to be exemplified in T4.1.
2) Algorithms for measuring the audio quality (real wav or up-cycled, stereo-spread, compression, bandwidth) are developed.
3) Finally, specific audio content based features are developed in order to improve music transitions, such as estimating the whole metrical structure, beat, downbeat and pattern level, large scale music structure, intro/outro (cue regions) for DJ mixing, and vocal positions). The resulting algorithms are developed as feature modules for integration in T4.4.

In this deliverable D4.7, we describe the research work related to these tasks achieved since the start of the project until M36.

The developments achieved for the software libraries of Task 4.2 are described in D4.3 *Research Report and Software Libraries for Auto-Tagging of Music*, chapter 7, and D4.5 *Final Software Libraries for Auto-Tagging of Music*. Two development paths have been followed:

1) Development for integration into the Music Library Manager *imdABCDJ*
2) Development for computing the specific features of TU Berlin *imdABCDJhardfeatures.py*

1.1 “Soft” and “Hard” Features

In ABC_DJ, two terms are used to describe audio features:

- “Soft” features refer to features that can be considered as partly subjective (such as pop-appeal, intensity, genre, style). In ABC_DJ, these features are exemplified by a collection of audio files that allow the use of machine-learning to train algorithms.
- “Hard” features are supposed to be non-subjective features and to relate to musical or musicological concepts

An **initial list** of “Soft” and “Hard” features has been defined by HearDis! in D2.4 at M6. From this initial list, two lists of features have been derived.

The **first list** of features is to be used directly by **HearDis!** within the **Music Library Manager**. This list contains

- “Soft” features; which are exemplified by a dataset as provided by task 4.1.
- Direct “hard” features (such as BPM, key or mode) as indicated in the DOW.

The whole list of HearDis! features is indicated in the following table. In this table, green cells indicate features provided by IRCAM.

The final status of development is the following: all HearDis! features have been developed by IRCAM.

Feature Group	Name	Unit	Source
Hard Features	BPM	Float	foobar/IRCAM
	Duration/Length	Integer	foobar (automatic)
	Encoding/Codec	String	foobar (automatic)
	Low quality source (File history)	Boolean	IRCAM
	Key	String	IRCAM: ICK_Key_KN
	Mode		IRCAM: ICK_Key_KM
	Mono compatibility	Boolean	IRCAM: MonoCompatibility_Mean
	Resolution	Integer	foobar (automatic)
	Sampling Rate	Integer	foobar (automatic)
	Loudness	Float	IRCAM
(Basic) Soft Features	Artist	String	foobar (manual/plugin)
	Composer	String	MLM (manual)
	Orchestra	String	MLM (manual)
	Conductor	String	MLM (manual)
	Title	String	foobar (plugin)
	ISRC	String	foobar (plugin?)
	Album	String	foobar (plugin)
	Tracknumber	Integer	foobar (plugin)
	Totaltracks	Integer	foobar (plugin)
	Release Year	Integer	foobar (plugin)
	Source Media	String	foobar (manual)
	Created	String	foobar (automatic)
	Encoded By	String	foobar (automatic)
	Website Artist	String	MLM (manual)
	Replay Gain	Float	foobar (plugin)
	Label	String	foobar (plugin)
	Publisher	String	MLM (manual)
	Terms of use	String	MLM (manual)
(Advanced) Soft Features	Instrumentation	String	IRCAM
	Vocals	Boolean	IRCAM
	Vocal Style	String	IRCAM or MLM (manual)
	Language	String	MLM (manual)
	Lyrics	String	MLM (automatic?)
	Explicit Lyrics	Boolean	MLM (manual)
	Intensity	Integer	IRCAM
	Conventionality	Integer	IRCAM
	Pop Hit	Integer	MLM (manual)
	Timbre	String	IRCAM
	Genre	String	IRCAM
	Main-Style	String	IRCAM
	Sub-Styles	String	IRCAM
	Time Reference	String	MLM (manual)
	Comment	String	MLM (manual)
	GMBI (level 2)	36 dimensions	TUB
	Target-Group	?	TUB

The **second list** of features is to be used by **TU Berlin**. These features are to be used as numerical values for the prediction of the GMBI attributes within WP3. This list contains “indirect” Hard features (i.e. features indirectly derived from Hard features, such as counting the number of II-V-I chord transitions, or minor second interval in the dominant melody) and Low-level audio features (such as statistics of MFCCs or features obtained using the TimbreToolbox). No training or testing dataset is provided for these "indirect" Hard features.

The final status of development at M36 is the following: Not all of TU Berlin features have been developed by IRCAM.

This is due to several factors:

- These features were defined after the start of the project (they were not indicated in the DOW);
- Some of these features require very high-skill to be developed (such as the features derived from the dominant melody);
- No training or testing dataset from the project can be used to develop them.

However, some additional features have been added after the first feedback from TUB, that partly replace the missing features.

1.2 Organization of the Deliverable

The deliverable is organized as follows:

- **Part 3** describes the work achieved related to the estimation of **"soft" features**.
- **Part 4** describes the work achieved related to the estimation of **"hard" features** (as defined by HearDis! in part 4.1 and as defined by TU Berlin in part 4.2)
- **Part 5** describes the work related to the measurement of the **audio quality**
- **Part 6** describes the work related to the estimation of **specific audio content** based features in order to improve music mixes (including transitions)

2 Auto-Tagging System for “Soft” Features

2.1 New Audio Features for Rhythm Description

Executive Summary: We have developed new audio features dedicated to the description of the rhythm content of an audio track. These features have been evaluated on several reference datasets (Ballroom, Cretan dances) and allow to obtain the best results today for these tasks. A new dataset has been created for the purpose of our research on rhythm description, the Extended Ballroom dataset. Two papers [Marchand16A] [Marchand16B] have been published and presented describing these works.

Within ABC_DJ, the MASSS rhythm feature is used within our machine-learning recognition system to estimate “soft” feature. It allows to increase its recognition rate.

2.1.1 Introduction

We propose two novel scale and shift-invariant time-frequency representations of the audio content. Scale-invariance is a desired property to describe the rhythm of an audio signal, as it will allow to obtain the same representations for same rhythms played at different tempi. This property can be achieved by expressing the time-axis in log-scale, for example using the Scale Transform (ST) [Cohen93]. Since the frequency locations of the audio content are also important, we previously extended the ST to the Modulation Scale Spectrum (MSS) [Marchand14]. However, this MSS does not allow the representation of the inter-relationship between the audio content existing in various frequency bands. To solve this issue, we propose two novel representations. The first one is based on the 2D Scale Transform, the second on statistics that represent the inter-relationship between the various frequency bands. We apply both representations to a task of rhythm class recognition and demonstrate their benefits. We show that the introduction of auditory statistics allows a large increase of the recognition results.

2.1.2 The 2D Modulation Scale Spectrum

In this method, we extend the idea of the MSS but represent the inter-relationship between the frequency bands using the 2D-Scale Transform of the modulus of the 2D-Fourier Transform instead of the independent 1D-Scale Transforms of independent auto-correlation functions. The flowchart of the computation process of the 2DMSS is given in the Figure below.

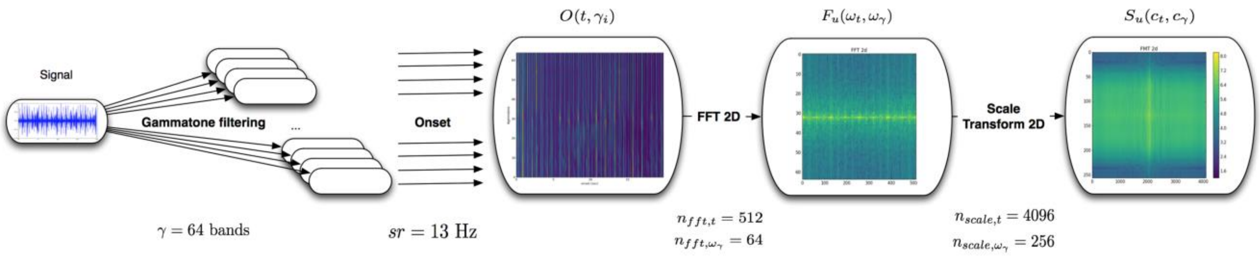


Figure: Computation of the 2D Modulation Scale Spectrum

2.1.3 The Modulation Scale Spectrum with Auditory Statistics (MASSS)

The previous 2DMSS representation provides a scale and shift invariant representation of the audio content and allows representing the inter-relationship between the various frequency bands. However, it also produces shift-invariance over frequencies, including shift-invariance when circularly rotating the frequency axis. This property is not desired since it will correspond to consider as equivalent low (kick) and high (hi-hat) patterns. This is the reason why we propose here a second representation, which uses statistics (inspired by the auditory experiments of McDermott, cf. [McDermott11]) to represent the inter-relationship between the various frequency bands.

In [McDermott11], the authors show evidence that "the auditory system summarizes temporal details of sounds using time-averaged statistics". They show that, in order to resynthesize sound textures, these statistics should include the statistics of each individual frequency band but should also include the cross-correlations between the temporal energy profiles within each frequency band.

We therefore propose to add to the MSS the correlations between the onset-energy-functions of the various frequency bands. The flowchart of the computation process of the MASSS descriptor is given in the Figure below.

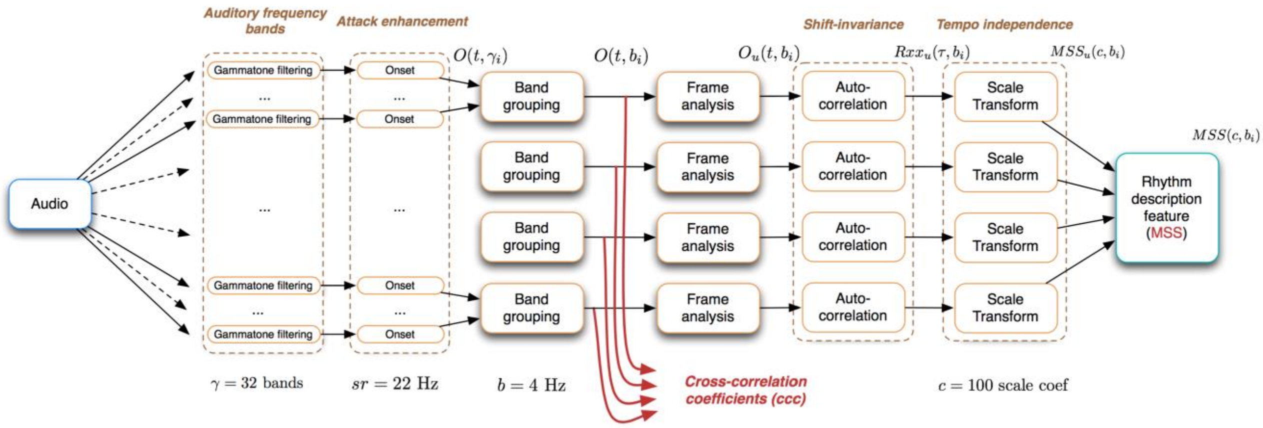


Figure: Computation of the Modulation Scale Spectrum with Auditory Statistics

2.1.4 Experiments

We compare the ability of the proposed descriptors to represent rhythm. For this we evaluate their performances for a task of rhythm class recognition.

The task consists in correctly recognizing the rhythm class of an audio track. For this we use datasets annotated into rhythm classes. We evaluate the performances of the 2DMSS, the MSS alone, the cross-correlation coefficients ccc alone, and finally both together (MASSS=MSS+ccc). We compare them to the best results published in [2] [Holzapfel11] and [3] [Marchand14]. For all classification tasks, we use Support Vector Machines (SVM) with a radial basis function kernel. Parameters of the SVM are found using grid-search. The results are presented in terms of mean-over-classes recall using 10-fold cross-validation.

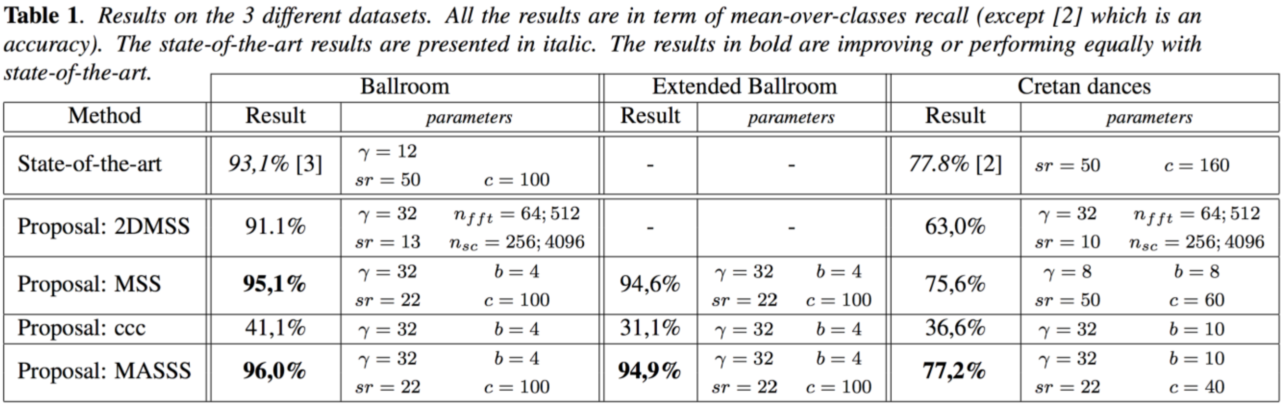


Table 1. *Results on the 3 different datasets. All the results are in term of mean-over-classes recall (except [2] which is an accuracy). The state-of-the-art results are presented in italic. The results in bold are improving or performing equally with state-of-the-art.*

	Ballroom		Extended Ballroom		Cretan dances	
Method	Result	*parameters*	Result	*parameters*	Result	*parameters*
State-of-the-art	*93,1%* [3]	$\gamma = 12$ $sr = 50$ $c = 100$	-	-	*77.8%* [2]	$sr = 50$ $c = 160$
Proposal: 2DMSS	91.1%	$\gamma = 32$ $n_{fft} = 64; 512$ $sr = 13$ $n_{sc} = 256; 4096$	-	-	63,0%	$\gamma = 32$ $n_{fft} = 64; 512$ $sr = 10$ $n_{sc} = 256; 4096$
Proposal: MSS	**95,1%**	$\gamma = 32$ $b = 4$ $sr = 22$ $c = 100$	94,6%	$\gamma = 32$ $b = 4$ $sr = 22$ $c = 100$	75,6%	$\gamma = 8$ $b = 8$ $sr = 50$ $c = 60$
Proposal: ccc	41,1%	$\gamma = 32$ $b = 4$	31,1%	$\gamma = 32$ $b = 4$	36,6%	$\gamma = 32$ $b = 10$
Proposal: MASSS	**96,0%**	$\gamma = 32$ $b = 4$ $sr = 22$ $c = 100$	**94,9%**	$\gamma = 32$ $b = 4$ $sr = 22$ $c = 100$	**77,2%**	$\gamma = 32$ $b = 10$ $sr = 22$ $c = 40$

Results are indicated in the Table above. First, it should be noted that the results of [Marchand14] on the Ballroom dataset (93.1%) were based on a MSS with many frequency bands. The new results presented as MSS (95.1%) are based on a reduced number of frequency bands, which seems beneficial. Over the two proposed new rhythm descriptors (2DMSS and MASSS), only the MASSS succeeded to outperform the MSS descriptor. For the Ballroom dataset, our MASSS descriptor outperforms (96,0%) state-of-the-art methods (93.1%) by 3%. On the Extended Ballroom dataset, our MASSS descriptor scores 94,9%. No comparison with state-of-the-art method is possible since this dataset is new for the research community. While the results obtained on this new dataset are slightly lower than those on the standard Ballroom, it should be considered that not only the number of files is 5 times larger but also the number of classes is larger (9 over 8). Therefore 94.9% on 9 classes is actually better than 96% on 8 classes. On the Cretan dances dataset, the MASSS descriptor has a mean-recall of 77,2% which is somewhat equivalent to state-of-the-art Holzapfel's accuracy of 77,8%.

2.1.5 Conclusion and Future Work

We proposed two novel audio descriptors (2DMSS and MASSS) that allow representing in a shift and scale invariant way the time and frequency content of an audio signal and differ by the way they model the inter-relationship between the various frequency bands. The first one, named 2DMSS, is based on the application of the Scale Transform along the two dimensions of time and frequency. This method was not successful and led to lower scores than our initial results [Marchand14]. It can be explained as follows. While this 2D representation allows to represent the inter-relationship between the various frequency bands, it also produces shift-invariance over frequency, including invariance when circularly rotating the frequency axis. This means that low and high frequencies cannot be distinguished any more, which is not a desired property. For this reason, we proposed a second representation, which uses statistics to represent the inter-relationship between the various frequency bands [McDermott11]. This second descriptor, named MASSS, provides the new top-results for these datasets. We see that in each of the three experiments, adding the cross- correlation coefficients improves the classification result: 0,9% for the Ballroom, 0,3% for the Extended Ballroom and 1,6% for the Cretan dances dataset. These are promising scores and future work will concentrate on testing MASSS as input to other classification tasks.

2.1.6 References

[Cohen93] L. Cohen. The scale representation. IEEE Transactions on Signal Processing, 41(12):3275–3292, 1993.

[Holzapfel11] André Holzapfel and Yannis Stylianou, "Scale transform in rhythmic similarity of music," Audio, Speech, and Language Processing, IEEE Transactions on, vol. 19, no. 1, pp. 176–185, 2011.

[Marchand14] Ugo Marchand and Geoffroy Peeters, "The modulation scale spectrum and its application to rhythm-content description," in Proc. of the 17th Int. Conference on Digital Audio Effects (DAFx-14), September 2014.

[Marchand16A] U. Marchand and G. Peeters. The extended ballroom dataset. In ISMIR (International Society for Music Information Retrieval) / Late-Breaking News, New York, USA, 2016.

[Marchand16B] U. Marchand and G. Peeters. Scale and shift invariant time/frequency representation using auditory statistics: Application to rhythm description. In Proc. of IEEE MSLP (International Workshop on Machine Learning for Signal Processing), Salerno, Italy, 2016.

[McDermott11] J. McDermott and E. Simoncelli. Sound texture perception via statistics of the auditory periphery: evidence from sound synthesis. Neuron, 71(5):926–940, 2011.

2.2 Extension and Adaptation of ircamclass for “Soft” Features

Executive Summary: The IRCAM machine-learning system to perform auto-tagging is named ircamclassification. In the ABC_DJ project, we have extended it by including three new modules: the new audio features MASSS, the audio augmentation and the audio segmentation modules. Training datasets (i.e. the set of audio files with the corresponding “soft” tags annotations) corresponding to 8 different tag-families have been provided by HearDis! late October 2016. In this part, we present the results obtained by applying ircamclassification to automatically predict the “soft” tags annotations within each of the 8 tag-families.

Within ABC_DJ, our machine-learning recognition system is used to predict the “soft” features of HearDis! The system is integrated into the MLM. The estimated “soft” tags are then used for the creation of playlists.

2.2.1 Description of the Given Tasks

ircamclassification allows to deal with three types of problems:

- Single-label (an audio file is associated to a single tag/class within a tag family; tags/classes are therefore considered as mutually exclusive within a family),
- Multi-label (an audio file is associated to several tags/classes within a tag family; tags/classes are therefore not mutually exclusive within a family),
- Regression (tags/classes represent ranking. An example of this is the “pop-appeal” tag family with tags/classes 1-2-3-4-5).

When receiving the training data set from HearDis!, the first task has been to determine the type of problem corresponding to each tag family. The data set represents 8 different tag families.

Tag Family/ Dataset	Description
(a) Vocals-1	This dataset contains 2 tags/classes: vocal and mixed (voice + instruments). They are relatively balanced (same number of examples for each tag/class), and have a total of 413 songs.
(b) Vocals-2	This dataset contains 3 tags/classes: Male, Female and Male-Female, for a total of 633 files. Here the task is to recognize either the vocal gender, or if some singers with different genders are singing.
(c) Pop-Appeal	The 983 songs of this dataset are classified with 5 repertories, which are numbered from 1 to 5. This “pop-appeal” value represents the “pop” characteristic of the song. For example, the music of “Amon Tobin” has no pop-appeal, but the one of the “Jackson 5” has more pop-appeal. It should be remarked that this task is actually a regression task since all the “tags/classes” are sorted according a given criterion. Nevertheless, it can be treated as a single-label classification task. On one hand it makes possible the use of advanced classification techniques, but on the other hand, it does not take into account the order of the classes.

(d) Intensity	This task concerns the “Intensity” property of a song. It appears as an integer number from 1 to 5. As the “Pop-Appeal” task, it should be treated as a regression problem, but for the same reason, it will be treated as a single-label classification problem. 812 songs are in the dataset.
(e) Instrumentation	This dataset contains 1983 files, which are stored in 13 instrument-like tags/classes: Live-Drums, Choir, Acapella, Synthetic-Drums, Orchestral, Strings, Acoustic-Guitar, Electric-Guitar, Piano, Percussions, Brass-Final, Speechiness, and Whistle. The tags/classes describe the presence of a characteristic instrument or sound, which can be dominant or just present if it is quite rare. For example, most of the songs of the class/tag “Piano” have the piano as dominant instrument, but many of the songs of “Choir” just have a part with a choir. Note that this dataset should be treated as a multi-label task, but unfortunately for most of the songs, only one tag is informed. One typical example is the song “Led Zeppelin - Your Time Is Gonna Come”, which contains a drum, guitars and a choir, but it is only annotated as “Choir”. Consequently, a standard single-label classification task is applied here, and the songs with more than one annotated tags are automatically excluded (148 files). Nevertheless, this does not fully solve the problem, and the performances will be affected by this lack of annotations, see below.
(f) Timbre	For this dataset, 786 files are classified into 6 tags/classes, which describe the timbre of the songs: Hard, Dark, Cold, Bright, Warm and Soft. As for the Instrumentation, this problem is a multi-label classification problem, but it will be treated as a single-label classification problem (by removing the duplicates). Note that except the “Hard” timbre (which is over-represented), the classes are relatively balanced.
(g) Genre	This task contains 1897 files classified using 10 relatively uniform tags/classes. These tags/classes indicate the musical genre of the songs. They are: World-music, Pop, Soul-Funk, Folk, Dance, Rock, Hip-Hop, Blues, Classical, and Jazz.
(h) Style	This dataset contains 9417 files stored in 61 tags/classes, which indicate the style of a song. They are for example: Balkan, Progressive-Rock, Disco. Again, this is a multi-label classification problem, but it will be treated as a single-label classification problem (by removing the 1304 duplicates). That's why only 8113 files are used. Note that even if the number of files is relatively high, the number of different tags/classes is also really high, which does not facilitate the task.

2.2.2 Description of the Algorithms Used

When applying ircamclassification to solve the 8 tag-family classification problems, we simultaneously tested some combinations of audio features and machine learning algorithms, including ARV Modelling (AutoRegressive Multi-Variate), Modulation Spectrum, GMM-UBM Super-Vectors with normalized MFCC or ARV-Filtering for the audio features, such as PCA (Principal Component Analysis), LDA (Linear Discriminant Analysis), or SVM (Support Vector Machine). Among all tested combinations, the best one will be chosen.

Additionally, during the first year of the ABC_DJ we have extended ircamclassification by adding two new modules: audio augmentation and audio segmentation. Both have the ability to increase the size of the training set and make the trained models more robust. They can be combined together.

2.2.3 Audio Augmentation

Audio augmentation, which is commonly named “data augmentation”, consists in creating new audio data by modifying the original audio data. Both the original and the modified audio data are then used for training the machine learning system.

We have developed a MATLAB toolbox, which can apply a lot of common sound modifications to an audio file, such as: equalizing, filtering, pitch shifting, noise addition, distortion, digital encoding, and dynamics compression. To derive a large number of possible modifications, this toolbox is able to apply the modifications in chain.

The advantages of this approach are:

- It increases the number of data for the training,
- It allows to train models which are more robust compared to common audio modifications,
- It may solve the problem of “uniform data”. For example, often, all the songs of a given dataset are encoded with the same MP3 codec. Then training a model with a unique codec, may provide unpredictable results when the classification is done on files encoded with a different codec (this problem has been observed). To solve this, the alteration toolbox makes possible to simulate different encoding algorithms, and the training set is therefore more representative of possible audio signals.

Note that for the tag/class prediction of a song, only the original signal is used.

2.2.4 Audio Segmentation

A common approach in audio classification is to derive a single feature vector representing the whole song duration. This can be achieved by taking the mean of each component of the instantaneous feature vector over the song duration, or by modelling the components distribution (with a single Gaussian model or a GMM for example). With this approach, for a training set of N songs, the classifier is trained with no more than N feature vectors which can be too low.

An alternative to this consists in operating the classification at the segment level (a segment is a 10 to 30 seconds extract of the song) rather than at the song level. Using M segments per song; the training set contains $N \times M$ feature vectors. For an unknown song, predicting its class is based on the computation of the segments class affinities. A unique class is derived for each song using a majority-voting rule.

The first advantage of this approach is that it significantly increases the size of the training set. The second advantage of this approach is that the classification is local in time. This allows for example time-varying classes such as occurring in radio streams for example.

2.2.5 Results of Classification

We present here the results of the classification for the 8 tag-families presented above.

First, the combination of feature types providing the best results has been chosen for each task separately. And concerning the classification algorithms, we finally obtained the best results using an LDA on each feature type for the dimension reductions, and a well-tuned SVM for the classification of songs, or segments.

The evaluation has been done using a 5 folds cross validation procedure (taking in turn 1/5 of the dataset for testing and the remaining 4/5 for training). Therefore, each song is used for testing once, and 4 times for training. Also, in order to avoid the "artist effect" (songs from the same artist tend to be very close) we applied an "artist-filter" (all the songs from a given artist are either in the train or the test set but never in both).

(a) Vocals-1	With 2 classes, a random decision would yield a result of 50%. The best configuration without audio augmentation and audio segmentation leads to a mean recall of 87%. Using audio augmentation and audio segmentation leads to **92.5%**, i.e. a 5% gain. The use of those has therefore a significant impact. Note that, as it is usually observed, the segmentation results in a higher benefit compared to the audio augmentation. It should be noted that the evaluation results presented are obtained using only the original audio for testing. While the benefit of audio augmentation is less clear in this case, it makes the trained system more robust to any degradation.
(b) Vocals-2	With 3 classes, a random decision would yield a result of 33%. The best configuration without audio augmentation and audio segmentation leads to a mean recall of 55%. Using audio augmentation and audio segmentation leads to **63.6%**. The results are however quite disappointing compared to the random 33%. A possible explanation for these low results comes from the way the training is performed. Actually, to train a "male" classifier the content of the whole track duration is used, whereas only segments of the track contain singing voice. The classifier therefore also used instrumental segments to train the "male" concept, which is nonsense. Actually, we do not have the information of the singing voice location inside the track. A possibility would be to use the "Vocals-1" classifier to obtain those. Another difficulty comes from the "mix" class (Male+Female) which is not clearly discriminated, even by human listeners. For example, most of the time a single singer performs the whole song, and only during a short amount of time (usually during the chorus) one or

	several singers of different genders appear, cf. e.g. the song: "Barry White".
(c) Pop-Appeal	With 5 classes, a random decision would yield a result of 20%. The best configuration without audio augmentation and audio segmentation leads to a mean recall of 38 %. Using audio augmentation and audio segmentation leads to **56.1%**. Of course, using a real regression task should allow increasing the recognition score since the current single-label classification paradigm does not allow taking into account the proximity of the classes.
(d) Intensity	With 5 classes, a random decision would yield a result of 20%. Again, applying a classification task, the best result is obtained for segmentation, with 68.8%. Adding the augmentation, the result is lower, **67.4%**, which is not significant because of the error margin. Without segmentation and augmentation the result is significantly lower, only 47%. Since the problem is originally a regression problem, the same remark as above can be done, cf. Pop-Appeal.
(e) Instrumentation	With 13 classes, a random decision would yield a result of 7.7%. Our prediction system provides a result of **53.2 %**, with both audio augmentation and audio segmentation. Again, the benefit of the two approaches is significant because it is 37 % without them. In spite of the difficulties mentioned earlier, the result of classification (53.45%) is not bad in comparison to the value of random prediction (7.7%).
(f) Timbre	With 6 classes, a random decision would yield a result of 16.6%. The best result obtained is **45.7%** with both audio augmentation and audio segmentation, whereas it is only 32% without. Note that the relative bad result of this task may be explained by three reasons: 1) this kind of task is not always well defined, and even for human listeners a choice may not be consensual; 2) the classes are usually based on cognitive properties, and most standard machine learning algorithms have some difficulties to treat them; 3) this task has been actually designed for a multi-labelling task, some songs are stored in several categories.
(g) Genre	With 10 classes, a random decision would yield a result of 10%. In spite of the significant benefit of the audio augmentation and audio segmentation, we remark that the results are relatively disappointing compared to the literature.

	Here the best results are **62.1%**, this is largely better than random (10%) but relatively lower than the results of the literature on a similar dataset, such as the GTZAN dataset.
(h) Style	With 61 classes, a random decision would yield a result of 1.64%. The best result obtained is **45.7%** with both audio augmentation and audio segmentation, whereas it is only 38% without. Compared to the result of a random decision (1,64%), the results obtained are good. Even if the dataset has a lot of songs, 8113 files without duplicates, the average number of song per class is only 133, increasing the size of the dataset then may lead to increasing results.

The figures below illustrate the confusion matrices in percentage for five tasks.

Concerning the tasks Pop-Appeal and Intensity, which are originally regression tasks, we noticed that in spite of the quite disappointing scores (56% and 67%), most of the wrong predictions are classified in neighbour levels. Consequently, the results of the prediction are coherent and meaningful.

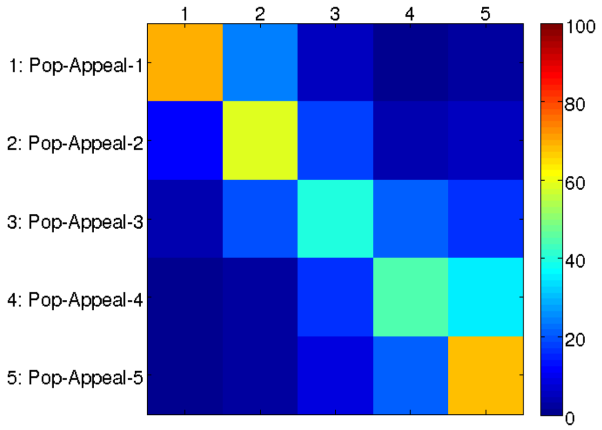


Figure: Confusion matrix for Pop-Appeal

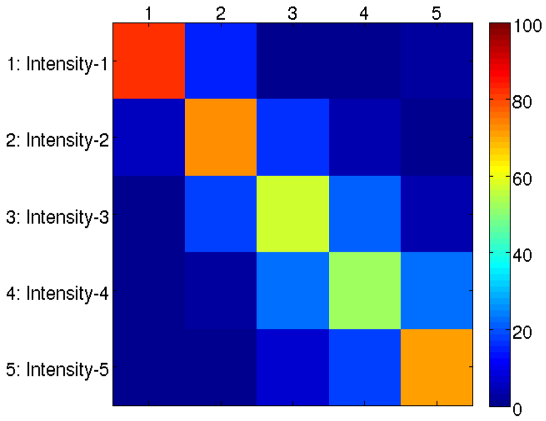


Figure: Confusion matrix for Intensity

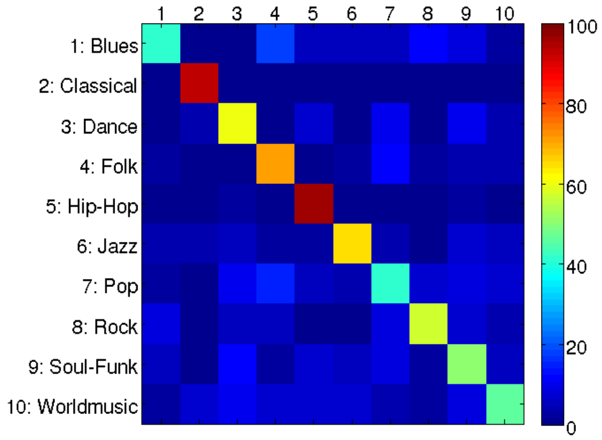


Figure: Confusion matrix for Genre

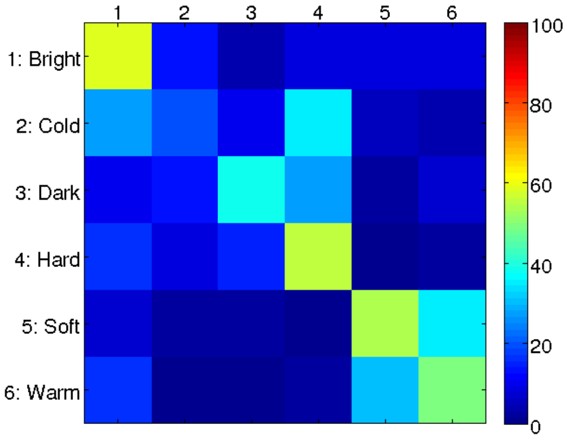


Figure: Confusion matrix for Timbre

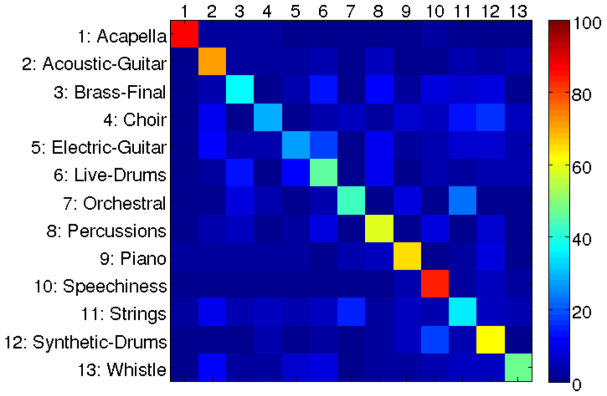


Figure: Confusion matrix for instrumentation

For the confusion matrices of the tasks Genre and Instrumentation, we can also see meaningful confusions. For example, there are obvious similarities of confusion between *Rock* and *Blues*, or between *Instrumental* and *Strings*.

Unfortunately, concerning the task Timbre we observed quite strong errors, with a mean recall of 45.7%. For example, the songs of the class *Cold* are most of the time recognized as *Hard* and rarely as *Cold*. Also, we can notice the high confusion between the classes *Warm* and *Soft*. As said above, these kinds of classification tasks are usually quite difficult.

Nevertheless, as with Genre and Instrumentation, the confusion matrix of the task Styles with 61 classes and a score of 45.7%, not shown here, reveals really meaningful errors, i.e. most of the confusions are with neighbour classes: for example *R&B* and *Soul*, or *Deep-House*, *House* and *Tech-House*.

2.2.6 Conclusion

In this part we presented the results obtained by applying ircamclassification and its two new extensions (audio augmentation and audio segmentation) to the prediction of the tags within the 8 tag-families/datasets.

First, we showed that the prediction scores are largely above a random decision. For example, we achieved 45.7% for the tags of the style-family. This is 28 times better than a random classifier (1.64%). This proves the ability of our algorithm to classify the songs.

Second, by examining the confusion matrices, we noticed that in case of errors, the wrongly predicted class is most of the time semantically similar to the ground-truth class. This reveals the strong coherence of the classification.

We also showed that for all tag-families/datasets, the two proposed extensions (audio augmentation and audio segmentation) provided a significant benefit.

3 Auto-Tagging System for “Hard” Features

Executive Summary: Within ABC_DJ, our system to estimate direct “hard” features is integrated into the MLM. The direct “hard” features are then used for the creation of playlists. The system to estimate the low-level audio features and the indirect “hard” features is used as input to TU Berlin model to predict the GMBI attributes which are also used for the creation of playlists.

3.1 Auto-Tagging System for HearDis! “Hard” Features

The list of “hard” features defined by HearDis! has been processed in a straightforward way as follows:

- BPM feature is provided by ircambeat [Peeters, 2011b] within imdABCDJ software.
- Key and Mode features are provided by ircamkeymode [Peeters, 2006] within imdABCDJ software.
- Loudness feature is provided by ircamdescriptor [Peeters, 2004] within imdABCDJ software.
- MonoCompatibility and LowQualitySource features are provided as part of the new Audio Quality features [Fourer, 2017] within imdABCDJ software (see part 4 for a description of the algorithms).

3.2 Auto-Tagging System for TU Berlin “Hard” Features

Among the list of features defined by TU Berlin, we can distinguish two main types of features:

1) Low-level audio features (such as statistics of MFCCs or features obtained from using the TimbreToolbox).
2) Indirect “Hard” features (i.e. features indirectly derived from a previously estimated Hard features). An example of this is the “number of II-V-I chord transitions” which requires the previous estimation the Hard features “chord succession over time”.

The whole list of TU Berlin features contains over 140 items. IRCAM has achieved the development to calculate over 80 of these features. We also now use both the audio excerpt and the full audio track to better estimate the number of segments, and ported the Timbre Toolbox Matlab version to the python script imdABCDJhardfeatures.py.

There were three main reasons that proved problematic for developing a feature, and that might limit its practical usefulness:

1) The meaning of the features is unclear when applied to a polyphonic time-varying signal (such as music). This is the case of FundamentalFrequency or TemporalCentroid features.
2) The previously estimated Hard features are not detailed enough to compute the Indirect “Hard” feature. This is the case of the “Ratio of sum of additional notes of all triads / total number of triads” which cannot be computed from ircamchord since this one only estimates major or minor chords.
3) The previously estimated Hard features are not reliable enough to compute. This is the case of the “Ratio #minor seconds melodic interval divided by #melody notes” which

require the computation of a very clean dominant melody estimation which is not currently possible.

In the following chapter we provide the details of the computed features for each feature family.

For each family, green cells indicate features already implemented, red ones indicate features that are not implemented, yellow indicates new features that have been implemented additionally to the requirements after M18 according to feedback from TUB.

3.2.1 Structure Features

The following set of features is computed using the output of ircamsummary [Kaiser, 2013] within imdABCDJ software (which computes the temporal structure of a music track).

Feature ID or Feature Group	Definition / Analytic Function
ICS_Part_Sequence	Music Parts contained in sample defined by 1) Label of music parts 2) Start-Time of music parts 3) End-Time of parts
ICS_Part_Sequence_Total	The total number of music parts contained in the sample
ICS_Part_Sequence_Unique	The total number of unique music parts contained in the sample

3.2.2 Rhythm Features

The following set of features corresponds directly to the output of ircambeat [Peeters, 2011b] within imdABCDJ software. AccentStruct is not currently computed.

Rhythm and Time Features	
IRC_Length	Length of the musical excerpt
ICB_Meter	Time signature as defined within ICB_Meter (2/2 or 3/2). This feature is essential for Feature ID AccentStruc.
ICB_BPM_Mean	Tempo / Tempo Fluctuation in Beats per Minute
ICB_BPM_SD	Tempo / Tempo Fluctuation in Beats per Minute
ICB_perc_norm	Percussivity of Rhythm
ICB_complex_norm	Complexity of Rhythm
ICB_speed_norm_A	Speed of Track (A?)
ICB_speed_norm_B	Speed of Track (B?)
ICB_periodicity	Periodicity of rhythmic structure ("Fractalness")
AccentStruc	Accent Structure: Accentuation on semiquater notes (16tel), based on loudness and duration of note, BPM, bar recognition

3.2.3 Key Features

The following set of features is computed using the output of ircamkeymode [Peeters, 2006] within imdABCDJ software. Since ircamkeymode only estimates a global key/mode (not a time-varying one), it was not possible to implement the Key_Changes feature.

Key Features	
ICK_Key_PC	Key of music excerpt consisting of dominant key pitch class, dominant key note and dominant key mode (major/minor).
ICK_Key_KN	Key of music excerpt consisting of dominant key pitch class, dominant key note and dominant key mode.
ICK_Key_KM	Key of music excerpt consisting of dominant key pitch class, dominant key note and dominant key mode.

Key_Changes	Total number of key changes. Value is based on succession of used Keys (e.g. C Maj --> A Min, --> C Maj) as specified in ICK_Key_NEW. Feature is essential for chord features.

3.2.4 Chord Features

The following set of features is computed using the output of ircamchord [Papadopoulos, 2010] within imdABCDJ software. Since ircamchord only estimates Major and minor chord, and since it does not estimate the bass note, it was not possible to implement all TU Berlin features.

Chord Features	
ICC_Chord Sequence	Sequence of chords specified as chord symbol, pitch class, mode, degree, bass note, added notes, onset (time), offset (time), onset (beat), offset (beat), loudness. This chord sequence is the basis for all subsequent Chord Features
Chords_Num_01	Total number of chords divided by track duration in seconds
Chords_Num_02	Number of unique chords divided by track duration in seconds
Chords_Mode_01	Ratio of minor chords / total number of chords
Chords_Mode_02	Ratio of major chords / total number of chords If denominator is zero, the whole output should be set to zero
Chords_Mode_03	Ratio of major chords / minor chords
Chords_Mode_04	Ratio of non major or minor-based chords / total number of chords If denominator is zero, the whole output should be set to zero
Chords_Add_01	Ratio of pure triads / triads with additional notes
Chords_Add_02	Ratio of sum of additional notes of all triads / total number of triads If denominator is zero, the whole output should be set to zero
Chords_Func	Ratio of functional chords / non-functional chords. Functional is defined as root note of the chord is on the degree I, ii, iii, IV, V or iv where song key mode is "major" or root note of chord is on degree i, III, iv, v, V, VI or VII where song key mode is "minor". If denominator is zero, whole output should be set to zero.
Chords_Bass_01	Ratio of chords with root note as bass note / chords with different note as bass note
Chords_Bass_02	Ratio of chords with root note as bass note / chords with 3 (Third) as bass note. If denominator is zero, the whole output should be set to zero.
Chords_Bass_03	Ratio of chords with root note as bass note / chords with 5 (Fifth) as bass note. If denominator is zero, the whole output should be set to zero.
Chords_Bass_04	Ratio of chords with root note as bass note / chords with 7 (Seventh) as bass note. If denominator is zero, the whole output should be set to zero.
Chords_Cad_01	Perfect cadence type 1: Ratio of total number of successions [5, 1] / total number of harmony changes (i.e. two subsequent chords having a different root note). If denominator is zero, the whole output should be set to zero.
Chords_Cad_02	Perfect cadence type 2: Ratio of total number of root note successions [4, 5, 1] / total number of harmony changes (i.e. two subsequent chords having a different root note). If denominator is zero, the whole output should be set to zero.

Chords_Cad_03	Perfect cadence type 3: Ratio of total number of root note successions [2, 5, 1] / total number of harmony changes (i.e. two subsequent chords having a different root note)
Chords_Turn_01	Turnaround "1625": Ratio of total number of root note successions [1,6,2,5] + [6,2,5,1] + [2,5,1,6] + [5,1,6,2] divided by total number of harmony changes (i.e. two subsequent chords having a different root note). If denominator is zero, the whole output should be set to zero.
Chords_Turn_02	Turnaround "Blues": Ratio of total number of root note successions [1,4,1,5,4,1] / total number of harmony changes (i.e. two subsequent chords having a different root note). If denominator is zero, the whole output should be set to zero.
Chords_Turn_03	Turnaround "Pop": Ratio of total number of root note successions [I, V, ii, IV] / total number of harmony changes (i.e. two subsequent chords having a different root note). If denominator is zero, the whole output should be set to zero.
Chords_TonicDist	Tonic distance: Mean Number of chord changes (i.e. changes of root note or mode) until the next tonic (degree 1). If only the tonic is played, the Tonic distance is zero. If the tonic is never played (highly unlikely), the distance to the most frequent chord in the song should be calculated.

3.2.5 Melody Features

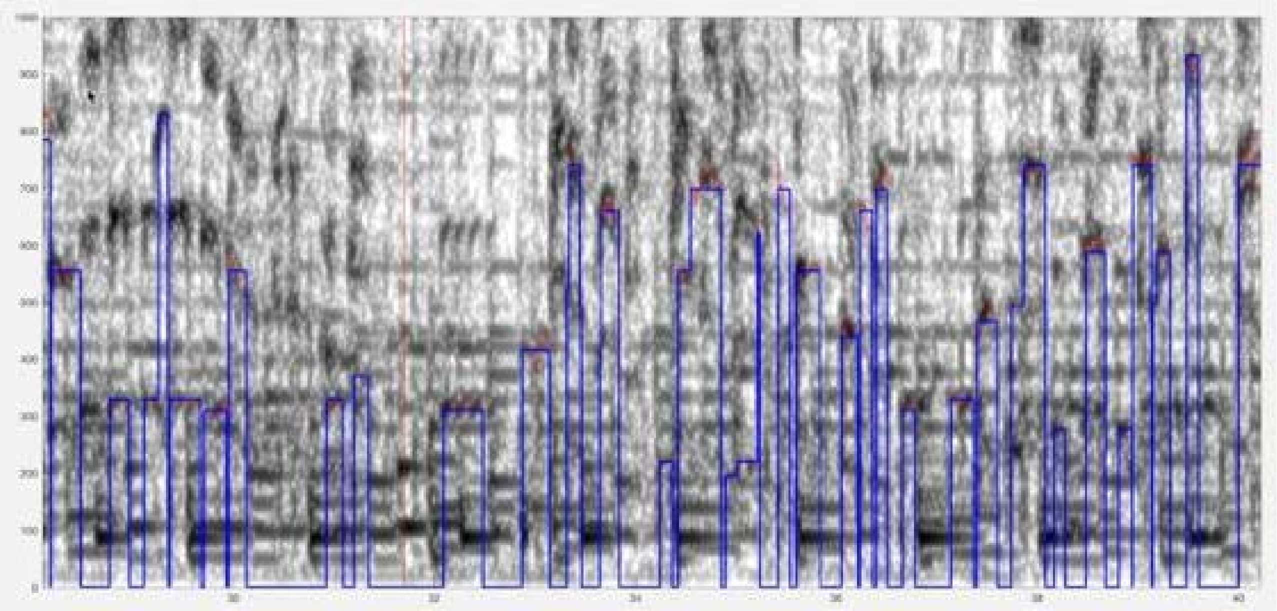
The initially developed melody features were based on

- Transcoding an audio signal to the continuous signal of the dominant fundamental frequency. This is done using the state of the art MTG essentia-extractors-v2.1_beta2/streaming_predominantmelody [Salamon, 2013].
- Converting this continuous signal of fundamental frequency to a sequence of discrete notes. This is done using a specifically developed hidden Markov model algorithm in which states are notes or notes with vibrato.
- Computing TU Berlin features from this sequence of notes.

The main difficulties encountered, which prevented the extraction to be performed, are:

- Difficulty to track correctly the dominant melody when the melody is not in the foreground,
- Difficulty to track only the dominant melody (and not to skip to other non-melodic parts),
- Difficulty to distinguish automatically music tracks with melody and without (Rap music)
- Difficulty to convert the fundamental frequency to pitch (especially when dealing with large vibrato and portamento)
- Necessity to take note duration into account in the definition of the features

Below is an example illustrating these difficulties. It shows the spectrogram of Track 03 of TU-Berlin corpus [Latin]. We can see that the melodies are interleaved, and thus difficult to estimate.



Note that in section 3.3 we describe research into a new algorithm for main melody extraction based on Convolutional and Recurrent Neural Networks which could in the future provide a more robust estimation of features derived from melody.

Melody Features	
Melody_Suc	Sequence/Matrix of melody notes as [pitch (e.g. "c4"), onset (time), offset (time), onset (beat), offset (beat), loudness, instrument]
Melody_Compl_01	#different pitches
Melody_Compl_02	#different pitch classes
Melody_Compl_03	Ratio #different pitches divided by #melody notes
Melody_Compl_04	Ratio #different pitch classes divided by #melody notes
Melody_Compl_05	Ratio #scale notes divided by #non-scale notes
Melody_Interval_01	Ratio #perfect unisons divided by #melody notes
Melody_Interval_02	Ratio #minor seconds divided by #melody notes
Melody_Interval_03	Ratio #major seconds divided by #melody notes
Melody_Interval_04	Ratio #minor thirds divided by #melody notes
Melody_Interval_05	Ratio #major thirds divided by #melody notes
Melody_Interval_06	Ratio #perfect fourths divided by #melody notes
Melody_Interval_07	Ratio #augmented fourths (trintoni) divided by #melody notes
Melody_Interval_08	Ratio #perfect fifths divided by #melody notes
Melody_Interval_09	Ratio #minor sixths by #melody notes
Melody_Interval_10	Ratio #major sixths divided by #melody notes
Melody_Interval_11	Ratio #of minor sevenths divided by #melody notes
Melody_Interval_12	Ratio #of major sevenths divided by #melody notes
Melody_Interval_13	Ratio #of perfect octaves divided by #melody notes

Melody_Interval_14	Ratio #small intervals (perfect unison up to major third) divided by #large intervals (perfect fourth up to perfect octave)
Melody_Direction	Ratio of #upwards intervals divided by #downwards intervals
Melody_NoteLength	Number of different types of note lengths within the melody (e.g. full, half, quarter, etc)

3.2.6 Timbre Set-A Features

The following set of features is computed using either

- the output of ircamdescriptor [Peeters, 2004] within imdABCDJ software (sharpness, distribution of loudness)
- or the output of a new algorithm based on the Harmonic, Percussive, Noise separation [Driedger , 2014] of the HPS of [Fitzgerald , 2010] (DecSinus/DecNoise/DecTrans)

Timbre and Dynamic Features	
DecSinus	The proportion of sinusoidal components (Decomposition of Recording)
DecNoise	The proportion of noise components (Decomposition of Recording)
DecTrans	The proportion of transient components (Decomposition of Recording)
Timbre_Dissonance	Harmonic distance of fundamentals and partials of sinusoidal signals to multiples of root note frequencies (based on sinusoidal component of example, not noise or transient components - as being defined in feature ID **DecSinus**). The idea is: 1. Detect root note frequency of song (key) 2. Determine root note frequencies of the remaining 11 pitch classes 3. Measure the distance of partials to pitch class frequencies (or multiples) per new harmony (e.g. based on new chord symbol) 4. Sum up distances of the partials to the respectively determined pitch class frequency (or multiple of it)
Roughness_Mean	Roughness of sample based IRCAM model
Roughness_SD	Roughness of sample based IRCAM model
Sharpness_Mean	Sharpness of sample based IRCAM model
Sharpness_SD	Sharpness of sample based IRCAM model
Dist_Loud_Mean	Estimation of dynamic range compressor parameters from sound recordings. Based on "Audio Quality Algorithm"
Dist_Loud_SD	Estimation of dynamic range compressor parameters from sound recordings. Based on "Audio Quality Algorithm"

3.2.7 Timbre Set-B Features

The following set of features was extracted using the TimbreToolbox [Peeters, 2011]. A specific python version has been developed within the project.

Timbre Toolbox Features	
TTB_RMS_Energy_critical_bands_by ERBfft_Mean	Frequency Spectrum and Frequency Spectrum Fluctuation. (Defined by TTB_RMS Energy Envelope (critical bands by ERBfft)
TTB_RMS_Energy_critical_bands_by ERBfft_SD	Frequency Spectrum and Frequency Spectrum Fluctuation. (Defined by TTB_RMS Energy Envelope (critical bands by ERBfft)

TTB_Fundamental Frequency_Mean	Fundamental Frequency (Dominant Pitch)
TTB_Fundamental Frequency_SD	Fundamental Frequency (Dominant Pitch)
TTB_RMS Energy_Envelope (technical, by STFT)_Mean	Sound Level / Level Fluctuation / Level Compression (RMS-Energy Envelope, Peeters et al (2011), p. 2905
TTB_RMS Energy_Envelope (technical, by STFT)_SD	Sound Level / Level Fluctuation / Level Compression (RMS-Energy Envelope, Peeters et al (2011), p. 2905
TTB_RMS Energy_Envelope (technical, by STFT)_Crest	Sound Level / Level Fluctuation / Level Compression (RMS-Energy Envelope, Peeters et al (2011), p. 2905
TTB_Frame_Energy_Mean	Overall Sound Energy of Sample
TTB_Frame_Energy_SD	Overall Sound Energy of Sample
TTB_Temporal_Centroid_Mean	Center of Gravity in Terms of Sound Energy
TTB_Temporal_Centroid_SD	Center of Gravity in Terms of Sound Energy
TTB_Spectral_Centroid_Mean	Center of Gravity in Terms of Sound Spectrum
TTB_Spectral_Centroid_SD	Center of Gravity in Terms of Sound Spectrum
TTB_Spectral_Spread_Mean	Breadth of Sound Spectrum --> all spectral TTB features should use the ERBfft Auditory Model (Window Size 40ms)
TTB_Spectral_Spread_SD	Breadth of Sound Spectrum --> all spectral TTB features should use the ERBfft Auditory Model (Window Size 40ms)
TTB_Spectral_Skewness_Mean	Asymetry of Sound Spectrum (LowFreq vs. HighFreq Domininance)
TTB_Spectral_Skewness_SD	Asymetry of Sound Spectrum (LowFreq vs. HighFreq Domininance)
TTB_Spectral_Kurtosis_Mean	Flatness of Sound Spectrum
TTB_Spectral_Kurtosis_SD	Flatness of Sound Spectrum
TTB_Spectral_Slope_Mean	Slope of Sound Spectrum
TTB_Spectral_Slope_SD	Slope of Sound Spectrum
TTB_Spectral_Decrease_Mean	Slope of Sound Spectrum
TTB_Spectral_Decrease_SD	Slope of Sound Spectrum
TTB_Spectral_Rolloff_Mean	Slope of Sound Spectrum
TTB_Spectral_Rolloff_SD	Slope of Sound Spectrum
TTB_Spectro-temporal_variation_Mean	Change in Overall Sound Color across Time
TTB_Spectro-temporal_variation_SD	Change in Overall Sound Color across Time
TTB_Spectral_Flatness_Mean	Noisiness vs. Harmonicness
TTB_Spectral_Flatness_SD	Noisiness vs. Harmonicness
TTB_Spectral_Crest_Mean	Noisiness vs. Harmonicness
TTB_Spectral_Crest_SD	Noisiness vs. Harmonicness
TTT_Enmi	temporary energy envelope rms-envelope minimum in a
TTT_Enme	temporary energy envelope rms-envelope mean in a
TTT_Enst	temporary energy envelope rms-envelope standard deviation in a
TTA_01mi	audio signal autocorrelation band 1 minimum

TTA_01ma	audio signal autocorrelation band 1 maximum
TTA_02ma	audio signal autocorrelation band 2 maximum
TTA_02me	audio signal autocorrelation band 2 mean
TTA_08mi	audio signal autocorrelation band 8 minimum
TTA_08me	audio signal autocorrelation band 8 mean
TTA_12mi	audio signal autocorrelation band 12 minimum
TTA_12ma	audio signal autocorrelation band 12 maximum
TTF_Sdme	ERBFFT spectral decrease mean
TTF_Svma	ERBFFT spectro-temporal variation maximum
TTF_Svme	ERBFFT spectro-temporal variation mean
TTF_Femi	ERBFFT frame energy minimum in i
TTF_Fema	ERBFFT frame energy maximum in i
TTF_Feme	ERBFFT frame energy mean in i
TTF_Fest	ERBFFT frame energy standard deviation in i
TTF_Sfme	ERBFFT spectral flatness mean
TTG_Fema	Timbre Toolbox ERBgammatone Frame Energy Maximum in I
TTG_Feme	Timbre Toolbox ERBgammatone Frame Energy Mean in I
TTG_Fest	Timbre Toolbox ERBgammatone Frame Energy Standard Deviation in I
TTG_Skst	Timbre Toolbox ERBgammatone Spectral Kurtosis Standard Deviation
TTG_Ssme	Timbre Toolbox ERBgammatone Spectral Spread Mean in F
TTG_Svmi	Timbre Toolbox ERBgammatone Spectro-temporal Variation Minimum
TTG_Swme	Timbre Toolbox ERBgammatone Spectral Skewness Mean
TTG_Swst	Timbre Toolbox ERBgammatone Spectral Skewness Standard Deviation
TTH_Heme	harmonic energy mean in a^2
TTH_Noma	noisiness maximum
TTH_Nome	noisiness mean
TTH_Nost	noisiness standard deviation
TTH_F0st	fundamental frequency standard deviation in hz
TTH_Ihma	inharmonicity maximum
TTH_Ihst	inharmonicity standard deviation
TTH_Scmi	Timbre Toolbox Harmonic Spectral Centroid Minimum in F
TTH_Scst	Timbre Toolbox Harmonic Spectral Centroid Standard Deviation in F
TTH_Slma	Timbre Toolbox Harmonic Spectral Slope Maximum in 1/F
TTH_Slme	Timbre Toolbox Harmonic Spectral Slope Mean in 1/F
TTH_Slmi	Timbre Toolbox Harmonic Spectral Slope Minimum in 1/F

TTH_Slst	Timbre Toolbox Harmonic Spectral Slope Standard Deviation in 1/F
TTH_Svme	Timbre Toolbox Harmonic Spectro-temporal Variation Mean
TTH_Svst	Timbre Toolbox Harmonic Spectro-temporal Variation Standard Deviation
TTH_Swmi	Timbre Toolbox Harmonic Spectral Skewness Minimum
TTH_T3me	tristimulus 3 mean
TTH_T3st	tristimulus 3 standard deviation
TTH_Hdma	harmonic spectral deviation maximum in a
TTH_Hdme	harmonic spectral deviation mean in a
TTM_Fema	Timbre Toolbox STFTmagnitude Frame Energy Maximum in I
TTM_Feme	Timbre Toolbox STFTmagnitude Frame Energy Mean in I
TTM_Fest	Timbre Toolbox STFTmagnitude Frame Energy Standard Deviation in I
TTM_Scme	Timbre Toolbox STFTmagnitude Spectral Centroid Mean in F
TTM_Sdmi	Timbre Toolbox STFTmagnitude Spectral Decrease Minimum
TTM_Sfma	Timbre Toolbox STFTmagnitude Spectral Flatness Maximum
TTM_Skme	Timbre Toolbox STFTmagnitude Spectral Kurtosis Mean
TTM_Slme	Timbre Toolbox STFTmagnitude Spectral Slope Mean in 1/F
TTM_Srme	Timbre Toolbox STFTmagnitude Spectral Rolloff Mean in F
TTM_Srst	Timbre Toolbox STFTmagnitude Spectral Rolloff Standard Deviation in F
TTM_Ssme	Timbre Toolbox STFTmagnitude Spectral Spread Mean in F
TTM_Ssmi	Timbre Toolbox STFTmagnitude Spectral Spread Minimum in F
TTM_Stma	Timbre Toolbox STFTmagnitude Spectral Crest Maximum
TTM_Stmi	Timbre Toolbox STFTmagnitude Spectral Crest Minimum
TTP_Fema	Timbre Toolbox STFTpower Frame Energy Maximum in I
TTP_Femi	Timbre Toolbox STFTpower Frame Energy Minimum in I
TTP_Fest	Timbre Toolbox STFTpower Frame Energy Standard Deviation in I
TTP_Sdst	Timbre Toolbox STFTpower Spectral Decrease Standard Deviation
TTP_Sfme	Timbre Toolbox STFTpower Spectral Flatness Mean
TTP_Slme	Timbre Toolbox STFTpower Spectral Slope Mean in 1/F
TTP_Srmi	Timbre Toolbox STFTpower Spectral Rolloff Minimum in F
TTP_Ssmi	Timbre Toolbox STFTpower Spectral Spread Minimum in F
TTP_Stmi	Timbre Toolbox STFTpower Spectral Crest Minimum
TTP_Stst	Timbre Toolbox STFTpower Spectral Crest Standard Deviation
TTP_Svmi	Timbre Toolbox STFTpower Spectro-temporal Variation Minimum
TTP_Swst	Timbre Toolbox STFTpower Spectral Skewness Standard Deviation

3.2.8 Timbre Set-C Features

The following set of features has been extracted using either

- the output of ircamdescriptor [Peeters, 2004] within imdABCDJ software (sharpness, distribution of loudness)
- or the output of the new Audio Quality features [Fourer, 2017] within imdABCDJ software (see part 4 for a description of the algorithms).

Other Features	
MFCC	Psychoacoustic spectrum which approximates the human auditory system's response more closely than the linearly-spaced frequency bands
MFCC_Band_01_MEAN	MFCC dimension / band 01
MFCC_Band_01_SD	MFCC dimension / band 01
MFCC_Band_02_MEAN	MFCC dimension / band 02
MFCC_Band_02_SD	MFCC dimension / band 02
MFCC_Band_03_MEAN	MFCC dimension / band 03
MFCC_Band_03_SD	MFCC dimension / band 03
MFCC_Band_04_MEAN	MFCC dimension / band 04
MFCC_Band_04_SD	MFCC dimension / band 04
MFCC_Band_05_MEAN	MFCC dimension / band 05
MFCC_Band_05_SD	MFCC dimension / band 05
MFCC_Band_06_MEAN	MFCC dimension / band 06
MFCC_Band_06_SD	MFCC dimension / band 06
MFCC_Band_07_MEAN	MFCC dimension / band 07
MFCC_Band_07_SD	MFCC dimension / band 07
MFCC_Band_08_MEAN	MFCC dimension / band 08
MFCC_Band_08_SD	MFCC dimension / band 08
MFCC_Band_09_MEAN	MFCC dimension / band 09
MFCC_Band_09_SD	MFCC dimension / band 09
MFCC_Band_10_MEAN	MFCC dimension / band 10
MFCC_Band_10_SD	MFCC dimension / band 10
MFCC_Band_11_MEAN	MFCC dimension / band 11
MFCC_Band_11_SD	MFCC dimension / band 11
MFCC_Band_12_MEAN	MFCC dimension / band 12
MFCC_Band_12_SD	MFCC dimension / band 12
MFCC_Band_13_MEAN	MFCC dimension / band 13
MFCC_Band_13_SD	MFCC dimension / band 13
StereoSpread_Mean	Stereo Spread of sample (Channel_Level_ Difference) Cross channel correlation, level differences, time delays

StereoSpread_SD	Stereo Spread of sample (Channel_Level_ Difference) Cross channel correlation, level differences, time delays
MonoCompatibility_Mean	Mono compatibility (Cross-Channel Correlation).
SamplingRate	Sampling rate rate of sample

3.3 Main Melody Estimation with Recurrent Neural Networks

In this task, we considered the automatic extraction of the main melody from audio files. We investigated neural network methods to improve on the state-of-the-art.
Note that this task started at M24 and was scheduled to last until M36 as further prospective research, and was not to be integrated in the development output of the project.

3.3.1 State of the Art

The model introduced in [Bittner+2017] is a convolutional neural network architecture to analyze a harmonic constant-Q transform (CQT) representation of the audio signal in order to extract a salience representation. Using a peak-picking method on this representation, it achieves state-of-the-art performances.

3.3.2 Method: Harmonic CQT

The Harmonic CQT is a stack of k CQT representations of the same audio signal performed with k different minimum frequencies f_{ok}, each f_{ok} being twice the previous one:

$$f_{0k} = 2 * f_{0(k-1)}$$

This way, each minimum frequency of each CQT representation share a harmonic relation to the minimum frequencies of all other CQT representations. As a result, the different CQTs provide a harmonic view of the frequency domain representation of the original signal. These are then stacked together so as to obtain a 3D tensor displaying a third axis where harmonically-related frequencies appear. This dimension can then provide some timbre information, carrying the relationship between different harmonics within the signal at each frequency f (see figure below).

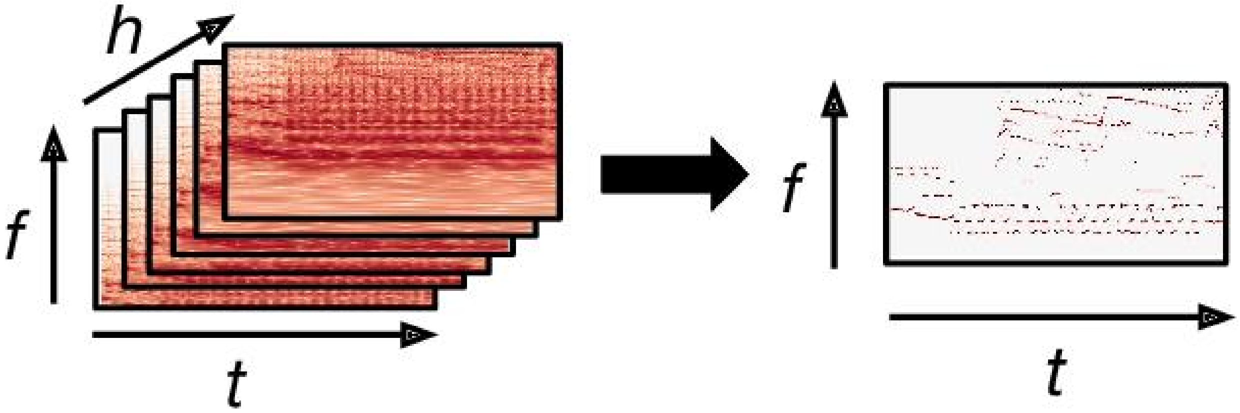


Figure 1: The harmonic CQT representation

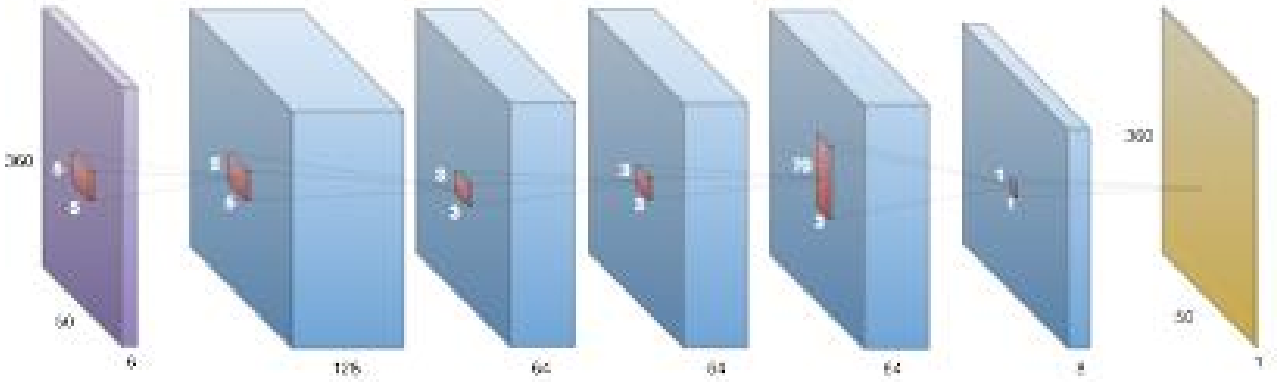


Figure 2: State-of-the-art CNN network

3.3.3 Deep Salience Representation with Convolutional Networks

The model introduced in [Bittner+2017] uses convolutional layers to analyze the Harmonic CQT and extract a salience representation, as shown in figure 2. It analyzes chunks of 590 ms of audio signal and outputs a 2D salience representation of it. The filters in the convolutional layers attempt to identify some patterns within the 590 ms signal, along the time axis as well as the frequency axis. The size of the filters in the first layers are chosen so as to be able to identify patterns over 5 ms and 1 semitone. Filters in a subsequent layer are larger along the frequency axis so as to accommodate for octave errors (notes detected at the wrong octave).

3.3.4 Temporal Modelling with Recurrent Neural Networks

The representation described above is used as input to train a recurrent neural network (RNN) classifier with 72 softmax outputs, one for each note across 6 octaves (C1 to C6). We use recurrent connections, as shown in figure 3, for each neuron, in order to build a temporal model over the sequence of inputs, as shown in figure 4. For memory reasons we train the network on sequences of 500 frames, which correspond to 5.8 s of audio. The temporal dependencies that the network learns are therefore limited to 6 s. This can of course be modified by training the network over longer sequences.

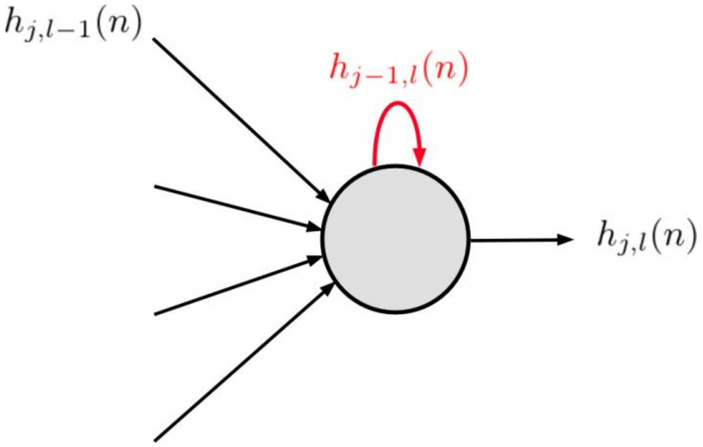


Figure 3: Graph of a recurrent neuron

3.3.5 Modelling Long-Dependencies with Stateful RNN

In order to account for long-term dependencies in the music, which in the case of dominant melody transcription equate to melodic patterns, it is possible to let the RNN's memory run through entire songs during training. This way the RNN keeps track of recurring patterns across entire songs and can therefore detect melody patterns.

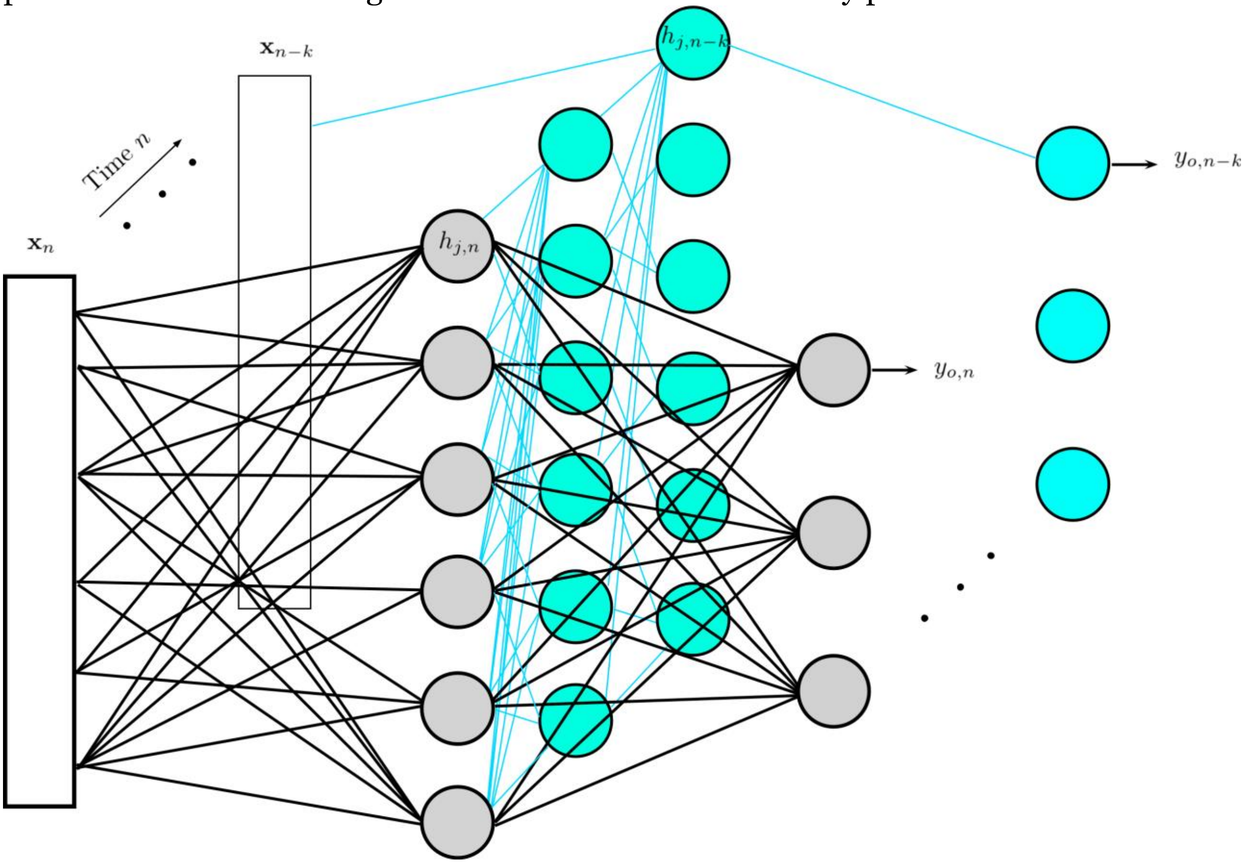


Figure 4: RNN 'unfolding' in time to illustrate the temporal dimension

3.3.6 RNN Results

The RNN model was run on Deep Salience representations of the *MedleyDB* dataset, with GRU units. Figure 5 gives an example illustration of the effect of the RNN on the input from the Deep Salience representation, which seems to be positive, as the melody line is clearly identifiable at the output of the RNN. However, this impact seems to be negative in terms of metrics. The final results on the test dataset, computed with the 'mireval' python library [Raffel+2014] are slightly lower for our model (7) than the state-of-the-art results, shown in figure 6. To understand why, we provide our model's confusion matrix in figure 9 and compare it with state-of-the-art's figure 8.
It appears that the confusion matrix for our model displays many more errors in the immediate vicinity of the diagonal (semi-tone errors), even though there seem to be less errors further away from the diagonal.
The exact count of errors is as follows:

- semi-tone errors: 60610 (vs 9451 for sota)
- tone errors: 4114 (vs 4423 for sota)
- within octave errors (besides tone and semi-tone): 19826 (vs 20063 for sota)
- rest of errors: 8724 (vs 13133 for sota)

It seems that our model suffers from making many semi-tone errors, despite working well as a “smoother” since less mistakes are made outside the octave.

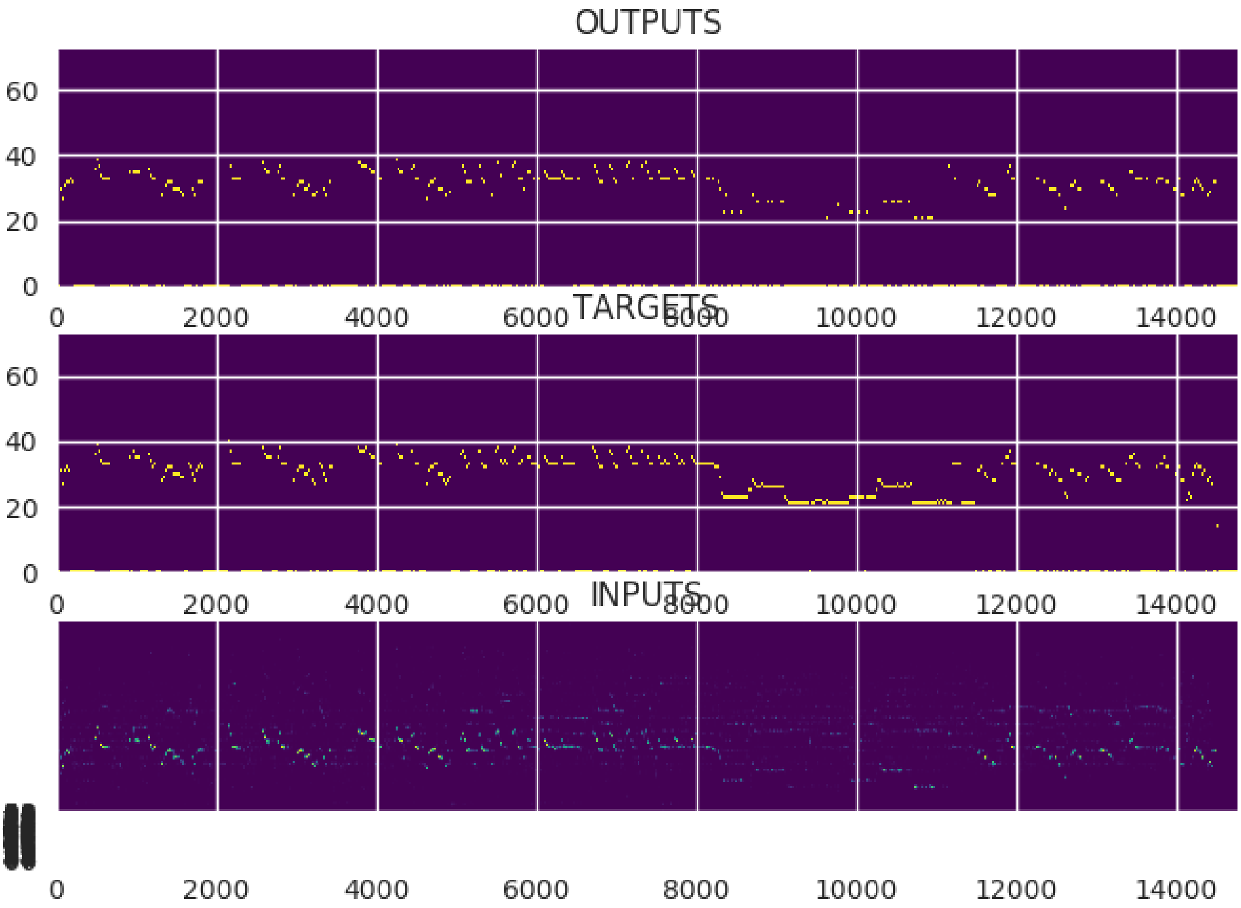


Figure 5: Comparison between input from deep salience representation, targets and output of RNN, on song "Night Owl" by "A Classic Education"

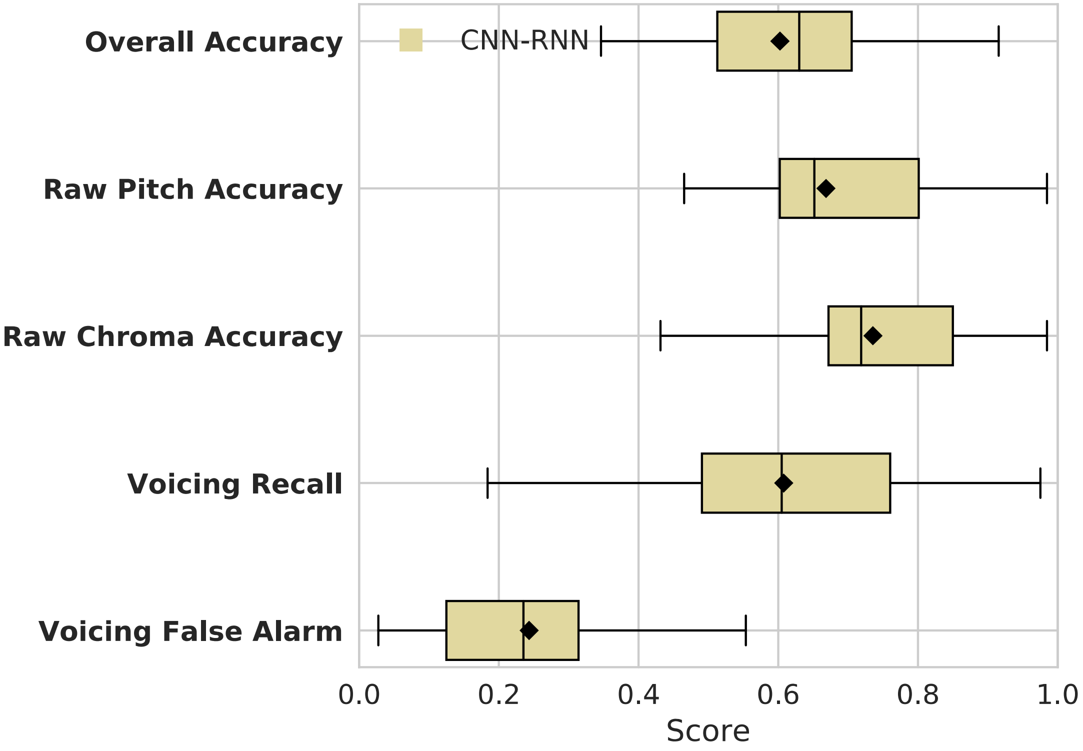


Figure 6: Results for State-of-the-art's predictions

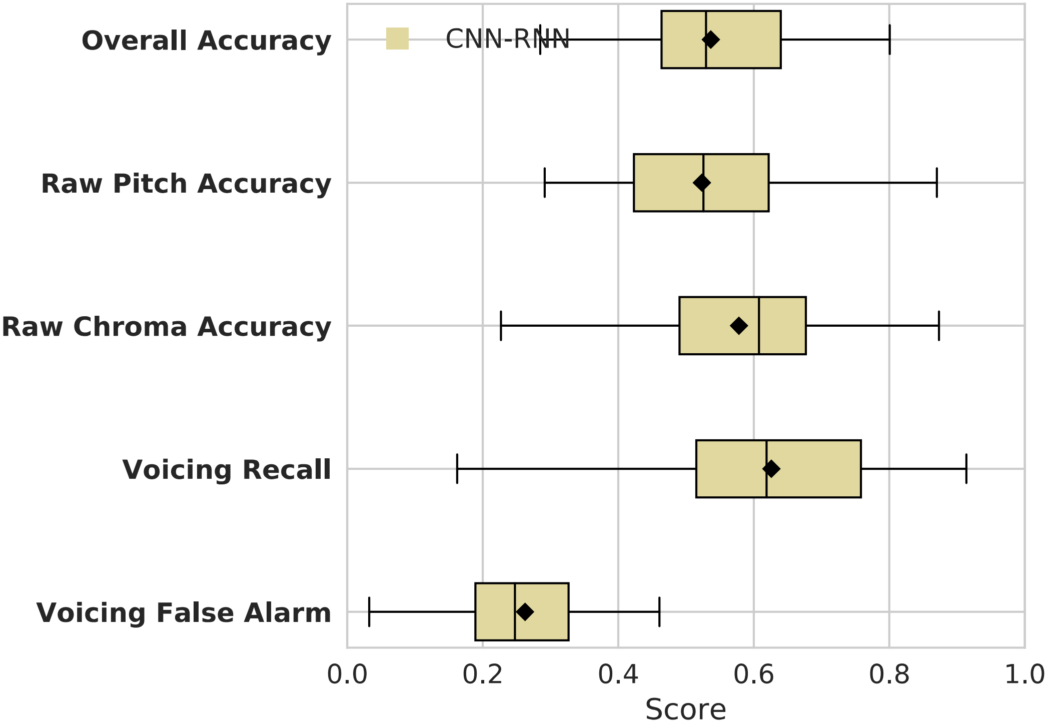


Figure 7: Results for RNN's predictions

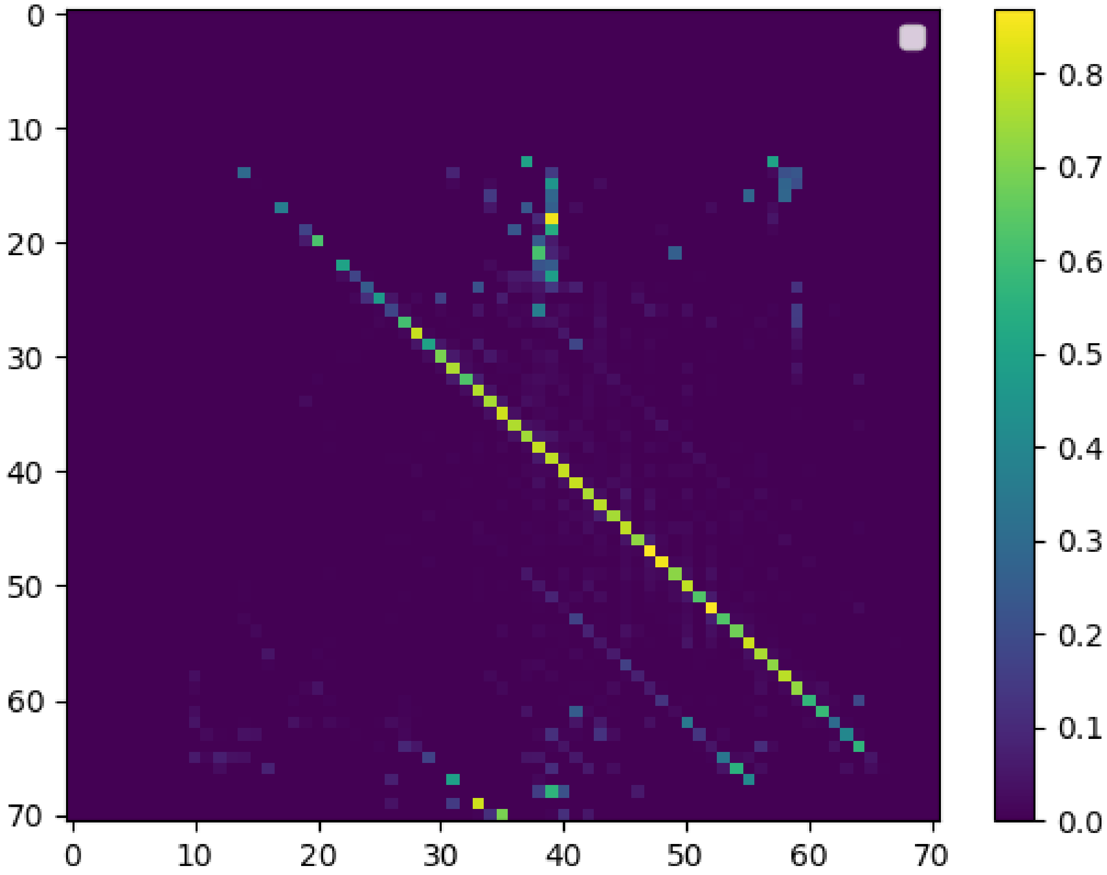


Figure 8: Confusion matrix for state-of-the-art over entire test set

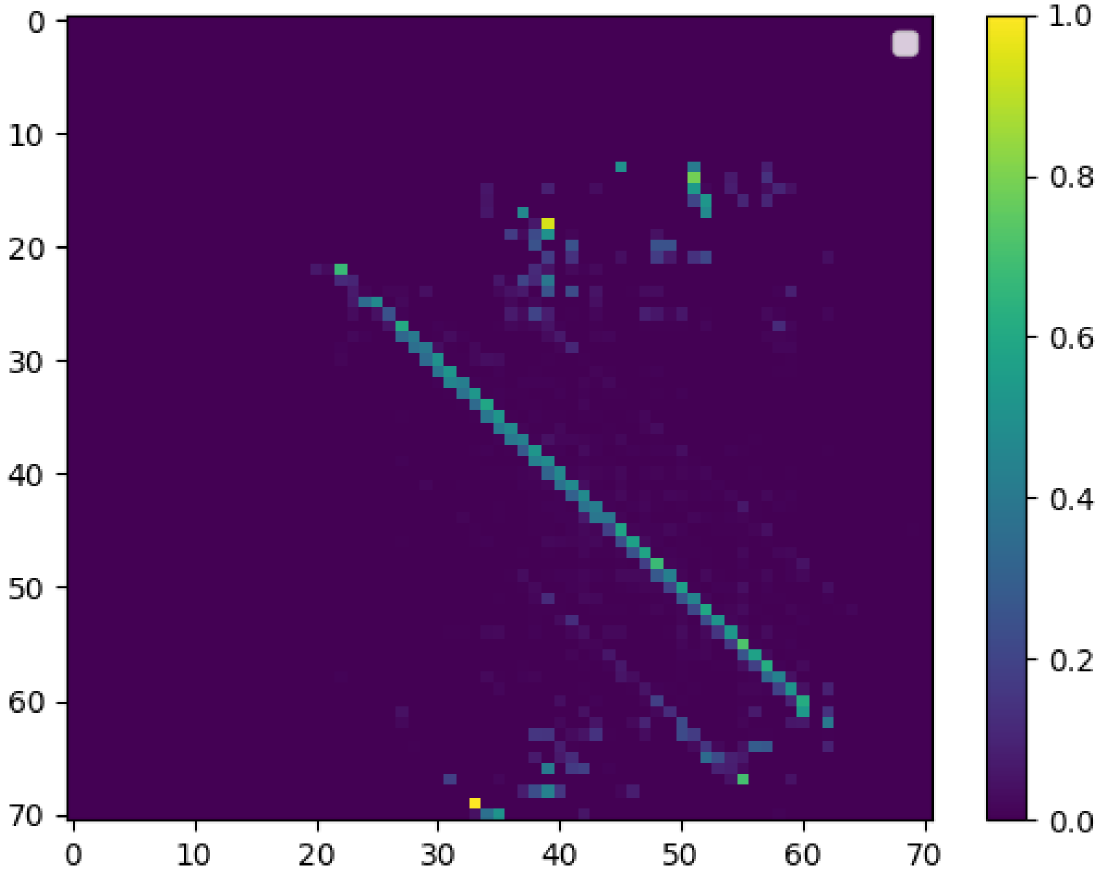


Figure 9: Confusion matrix for RNN over entire test set

3.3.7 Stateful RNN

Trying to identify long-term dependencies requires many songs in order to learn the evolution of melodies over each song. However, the *Medleydb* dataset only contains 128 songs, of which 70 only are used for training. This comes down to showing only 70 samples of a category of data to a model and expecting it to learn some common structures inside these samples. For those reasons the results using this model were very poor, as shown in figure 10.

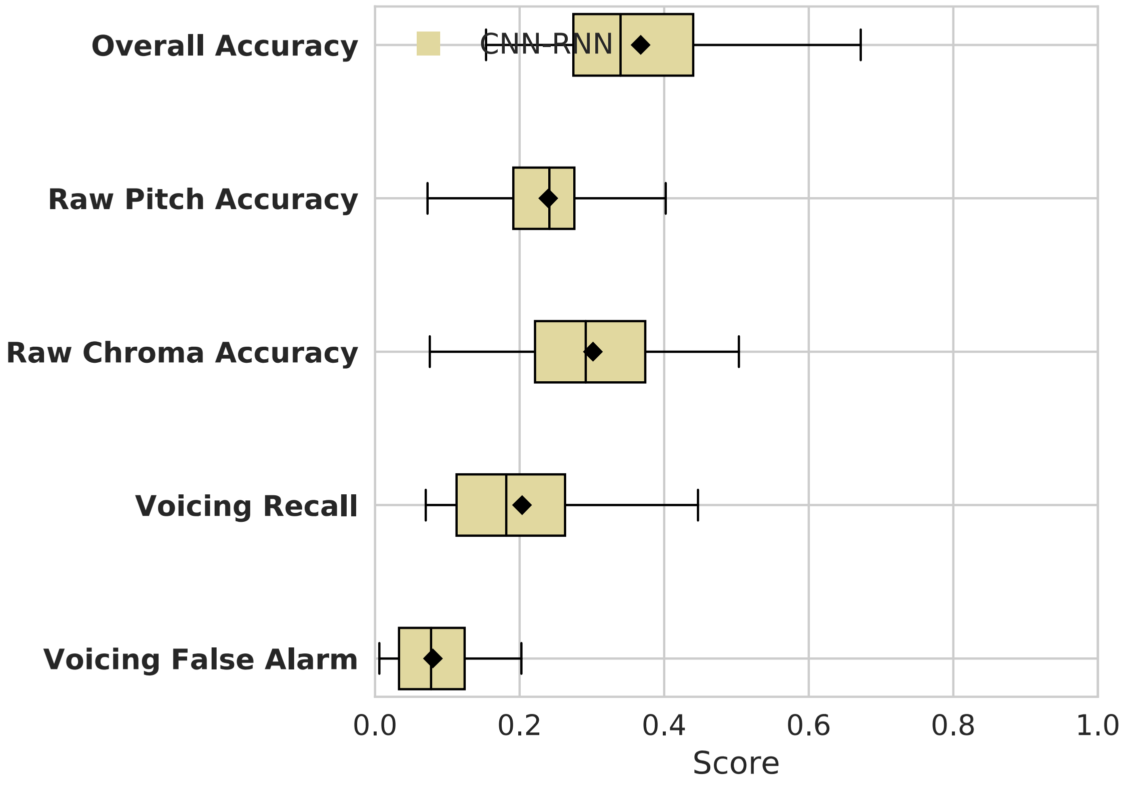


Figure 10: Results for Stateful RNN

3.4 References

[Bittner+2017] Rachel M. Bittner, Brian McFee, Justin Salamon, Peter Li, and Juan Pablo Bello. Deep salience representations for f0 estimation in polyphonic music. In ISMIR, 2017.

[Driedger , 2014] J. Driedger, M. Mueller, and S. Disch. Extending harmonic-percussive separation of audio signals. In Proc. of ISMIR (International Society for Music Information Retrieval), Taipei, Taiwan, 2014.

[Fitzgerald , 2010] D. Fitzgerald. Harmonic/percussive separation using median filtering. In Proc. of DAFx (International Conference on Digital Audio Effects), Graz, Austria, 2010.

[Fourer, 2017] D. Fourer and G. Peeters. Objective characterization of audio signal quality: Applications to music collection description. In Proc. of IEEE ICASSP (International Conference on Acoustics, Speech, and Signal Processing), New Orleans, USA, March, 5–9 2017.

[Kaiser, 2013] F. Kaiser and G. Peeters. Multiple hypotheses at multiple scales for audio novelty computation within music. In Proc. of IEEE ICASSP (International Conference on Acoustics, Speech, and Signal Processing), Vancouver, British Columbia, Canada, May 2013.

[Papadopoulos, 2010] H. Papadopoulos and G. Peeters. Joint estimation of chords and downbeats from an audio signal. Audio, Speech and Language Processing, IEEE Transactions on, 19(1):138 – 152, January 2010.

[Peeters, 2004] G. Peeters. A large set of audio features for sound description (similarity and classification) in the cuidado project. Cuidado project report, Ircam, 2004.

[Peeters, 2006] G. Peeters. Chroma-based estimation of musical key from audio-signal analysis. In Proc. of ISMIR (International Society for Music Information Retrieval), pages 115–120, Victoria, BC, Canada, 2006.

[Peeters, 2011] G. Peeters, B. Giordano, P. Susini, N. Misdariis, and S. McAdams. The timbre toolbox: Extracting audio descriptors from musical signals. JASA (Journal of the Acoustical Society of America), 130(5), November 2011.

[Peeters, 2011b] G. Peeters and H. Papadopoulos. Simultaneous beat and downbeat-tracking using a probabilistic frame- work: theory and large-scale evaluation. Audio, Speech and Language Processing, IEEE Transactions on, 19(6):1754–1769, August 2011.

[Raffel+2014] Colin Raffel, Brian McFee, Eric J. Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel P. W. Ellis. Mireval: A transparent implementation of common MIR metrics. In Hsin−MinWang, Yi−Hsuan Yang, and Jin Ha Lee, editors, ISMIR, pages 367–372, 2014.

[Salamon, 2013] J. Salamon. Melody Extraction from Polyphonic Music Signals. PhD thesis, Music Technology Group (MTG), Universitat Pompeu Fabra, Barcelona, 2013.

4 Audio Quality Characterization

Executive Summary: An intensive bibliographical study was completed and allowed us to build a new collection of about sixty audio quality features. These features have been evaluated on two classification tasks: audio alteration effect detection and music decade prediction. Our results are promising both in the supervised and in the unsupervised cases (the unsupervised approach is not detailed in this report and only concerns the detection of some audio degradation effects). For the decade prediction task (which consists in predicting the decade when a musical track was recorded), we obtain results comparable to a firstly proposed state-of-the-art method [Tard11]. A paper related to this work was accepted [Fourer17b] and an audio tagging software based on these results has been implemented in MATLAB.

Within ABC_DJ, the audio quality characterization is integrated into the MLM. It can be used to create playlists with homogeneous quality or, when several occurrences of a track exist, select the one with the highest quality.

4.1 Introduction

Audio signal quality can be related to subjective and objective audio signal attributes resulting from a sophisticated digital signal processing chain. Despite a consistent definition of audio quality has not been offered yet, researchers agree that it depends on a combination of transformations applied to the audio signal from its studio recording (or pure synthesis) to the resulting final mix obtained after mastering [Duar13].

Knowing the audio quality of a music track is full of interest for applications such as music streaming or playlist generation since it allows to decide which sound file (when several occurrences of the same music track exist in a database) has better quality or should be discarded. Among the first works related to the objective description of audio quality, the standard ISO/IEC 15938-4 (MPEG-7 Audio) [Bitz02] proposes a set of informative features to describe the audio content and the signal quality. More recently, audio quality has re-gained interest. In 2011, [Tard11] and [Duar13] propose to use a set of audio quality features to estimate the decade during which a musical piece was recorded and help the navigation in large music collection. In 2015, [Wils15] performs a set of perceptual experiments in which users judge the audio quality through listening tests. This leads to an audio quality lexicon. [Kend15, Fazen15, Wils16] propose a set of audio quality features to predict the results of the perceptual experiments using a machine learning approach.

In this document, we propose an extension of this approach, i.e. we propose an objective description of the audio quality. We aim at describing the audio signal content related to the mixing process and the signal quality. Hence, this approach is directly related to the audio signal reverse engineering problem [Reiss10, Gorl13], which finds applications in music description, audio branding [Bron09], automatic playlist generation or automatic music mixing [Barch09].

4.2 Overview of the Proposed Method

A music audio signal is the result of the mixing of a set of effects and transformations applied on separated tracks (elementary signals) in order to obtain an artistic mixture [Barch09]. Furthermore, after studio mixing, audio signals can also be degraded by signal transformations resulting from user manipulation (e.g. remixing, resampling, lossy compression, etc.). Hence, the audio quality characterization problem addressed here consists in either obtaining cues about the signal mixing process or (ideal case) recovering the exact signal properties related to the transformations, which have been applied to the signal. The solution proposed here is based on a machine-learning framework, which aims to predict the exact kind of transformation, which have been applied on a signal.

4.3 Considered Audio Signal Alteration Effects

In the present work we only consider a restricted set of signal alterations. Those however cover a wide range of commonly used audio transformations, as often addressed in the music processing literature.

Alteration name	Profiles	# of classes
Dynamic range control	- Reference artistic studio mix - No compression (instantaneous mix) - Dynamic range compression (sox)	1 1 5
Spatialization	- Reference artistic studio stereo mix - Mono mix - Amplitude panning - Phase panning - HRTF	1 1 4 4 4
Lossy compression	- Original uncompressed WAV file - MP3 compression (LAME encoder)	1 4
Content alteration	- Resampling - Addition of a white Gaussian noise	5 5

4.4 Audio Quality Features

For the purpose of describing the audio quality, we collect previously proposed audio quality features from the literature. These features correspond to functions, which are directly applied to audio signal in order to obtain cues related to the audio quality. Some of these functions (DH, AS, CD, SE, BW) are summarized by a time series obtained by computing statistics (mean, standard deviation, inter-quartile range, median, skewness, entropy, etc.) on the output provided by the initially proposed function.

Designation	Feature name	Label	# of feat.
Mixture dynamic range	- Dynamic histogram - Average spectrum	DH AS	12 12
Stereo quality	- Cochleagram difference - Spectral Stereo Phase Spread - Monophony detector - Cross-channel correlation - Relative delay - Balance	CD SSPS isMono CCCor Rdelay Bal	5 1 1 1 1 1

Signal content	- DC-offset - Root Mean Squared amplitude - Spectral Entropy - Frequency Bandwidth - Background noise level	DCOff aRMS SE BW BNL	1 1 10 10 1
Total number of features			57

4.5 Numerical Results

To validate our proposed audio quality features in a machine-learning framework, two prediction tasks have been considered and evaluated on their respective annotated dataset.

For the supervised classification tasks, we systematically compare the results respectively provided by the KNN (*k*-nearest neighbors), LDA (Linear Discriminant Analysis) and SVM (Support Vector Machines) with a radial basis kernel. The classification results for each class are expressed in term of recall and of accuracy, taking values in [0,1] range where the highest is the best.

4.5.1 Audio Signal Alteration Prediction

For this experiment, the MedleyDB database [Bittner14] has been used. A set of alteration effects has been applied on each musical piece (122) for which separated track and an artistic studio mixture is available. The supervised classification is evaluated using a 3-fold cross validation scheme separately on each class of alteration effect (dynamic range, spatialization, lossy compression and content alteration).

Method	Dynamic range control class name							Accuracy
	No comp.	Stud.	speech	stream	Spe./mus.	Mus1.	Mus2.	
KNN	0.36	0.80	0.23	0.08	0.26	0.44	0.06	0.32
LDA	0.72	0.98	**0.65**	**0.48**	**0.89**	**0.96**	**0.27**	**0.71**
SVM	**0.90**	**0.99**	0.48	0.37	0.23	0.95	0.09	0.57

Method	Spatialization class name					Accuracy
	Stud. mix	mono	Amp. pan.	Phs. Pan.	HRTF	
KNN	0.31	0.34	0.90	0.85	0.98	0.83
LDA	0.94	**1**	0.97	0.57	**1**	0.86
SVM	**0.96**	0.89	**1**	**0.97**	0.99	**0.98**

Method	Lossy compression class name					Accuracy
	Orig. wav	Mp3 320kbs	Mp3 128kbs	Mp3 64kbs	Mp3 16kbs	
KNN	0.34	0.20	0.20	0.99	1	0.55
LDA	0.73	**0.80**	0.85	1	1	0.88
SVM	**0.75**	0.59	**0.43**	**1**	**0.99**	**0.98**

Method	Content alteration class name										Acc.
	8khz	16kHz	32kHz	44kHz	96kHz	-15dB	-5dB	10dB	20dB	45dB	
KNN	0.83	0.72	0.51	0.25	0.32	**1**	**1**	0.90	0.61	0.24	0.64
LDA	0.87	**0.89**	**0.81**	0.55	**0.68**	**1**	**1**	**0.98**	**0.94**	**0.77**	**0.85**
SVM	**0.90**	0.80	0.70	**0.57**	0.65	0.99	**1**	0.89	0.66	0.46	0.76

4.5.2 Music Decade Prediction

For this task, which aims to predict the decade when a track was recorded, we consider a previously annotated dataset used in [Tard11]. For the experiment, we use a supervised random 3-folds cross validation scheme, after applying an artist filter on each fold. Our results show an improvement of the overall classification accuracy compared to [Tard11] without the usage of MFCC (Mel-Frequency Cepstrum Coefficients) as a classification feature.

Method	Class name					Accuracy
	60s	70s	80s	90s	2000s	
KNN	0.77	0.38	0.63	0.49	0.71	0.60
LDA	0.69	**0.43**	0.62	0.52	0.77	0.60
SVM	**0.83**	0.31	**0.69**	**0.55**	**0.79**	**0.63**

4.6 Conclusions

A set of audio features for the automatic characterization of audio signal quality has been proposed and evaluated on research databases. These results have been used to implement a new music auto-tagging software based on alteration effects detection and music decade prediction. These results pave the way for more sophisticated systems designed for automatic mixing, playlist generation or database indexing. Future work will consist in further investigating a larger set of “realistic” signal alterations and using audio quality annotated dataset as in [Wils16a, Wils16b].

4.7 References

[Barch09] Daniele Barchiesi and Josh Reiss, “Automatic target mixing using least-squares optimization of gains and equalization settings,” in Proc. Digital Audio Effects Conf. (DAFx’09) , 2009, pp. 7–14.

[Bittner14] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, “MedleyDB: A multitrack dataset for annotation-intensive MIR research,” in *Proc. IS- MIR’14*, Taipei, Taiwan, Oct. 2014.

[Bitz02] Joerg Bitzer and Juergen Herre, “Coding of moving pictures and audio ISO/IEC JTC 1/SC 29/WG 11,” Tech. Rep., International Organization for Standardization Organization Internationale Normalization, Shanghai, China, Oct. 2002.

[Bron09] K. Bronner and H. Rainer, Audio Branding. Brands, Sound and Communication, Nomos, 2009.

[Duar13] Pedro Duarte Pestana, Zheng Ma, Joshua D. Reiss, Alvaro Barbosa, and Dawn A. A. Black, “Spectral characteristics of popular commercial recordings 1950-2010,” in 135th AES Convention, NY, USA, Oct. 2013.

[Fazen15] Bruno Fazenda, Paul Kendrick, Trevor Cox, Francis Li, and Iain Jackson, “Perception and automated assessment of audio quality in user generated content,” in 139th AES Convention, Manchester, UK, Oct. 2015.

[Fourer17b] D. Fourer and G. Peeters, “Objective characterization of audio signal quality: applications to music collection description”, in Proc. IEEE International Conference on Acoust., Speech and Signal Process. (ICASSP’17), Aug. 2017.

[Gorl13] S. Gorlow and S. Marchand, “Reverse engineering stereo music recordings pursuing an informed two-stage approach,” in Proc. Digital Audio Effects Conf. (DAFx’13) , 2013.

[Kend15] Paul Kendrick, Francis Li, Bruno Fazenda, Iain Jackson, and Trevor Cox, “Perceived audio quality of sounds degraded by nonlinear distortions and single-ended assessment using hasqi,” Journal of the Audio Engineering Society, vol. 63, no. 9, pp. 698–712, 2015.

[Reiss10] J. Reiss D. Barchiesi, “Reverse engineering of a mix”, Journal of the Audio Engineering Society, vol. 58, pp. 563–576, 2010.

[Tard11] D. Tardieu, E. Detruty, and G. Peeters, “Production effect: Audio features for recordings techniques description and decade prediction,” in Proc. Digital Audio Effects Conf. (DAFx’11), Sept. 2011, pp. 441–446.

[Wils15] Alex Wilson and Bruno M Fazenda, “A lexicon of audio quality,” in Proceedings of the 9th Triennial conference of the European Society for the Cognitive Sciences of Music (ESCOM 2015), Manchester, UK, 2015.

[Wils16a] Alex Wilson and Bruno M Fazenda, “Perception of audio quality in productions of popular music,” Journal of the Audio Engineering Society, vol. 64, no. 1/2, pp. 23–34, 2016.

5 Specific Audio Content Based Features in Order to Improve Music Mixes

5.1 Blind Audio Source Separation (BASS) for Singing Voice Characterization

Executive Summary: We address here the audio source separation problem in the blind case (i.e. without trained model) which aims at estimating an arbitrary number of music sources from a single- or two-channel mixture. This configuration which is the most challenging one, requires to investigate new paradigms or to enhance existing ones. Thus, solving this configuration can theoretically enhance the foreground instrument signal (e.g. singing voice) and may improve the extraction of high-level features.

For this purpose, we investigated three approaches:

The first approach, inspired by the Computational Auditory Scene Analysis (CASA) literature [Breg90], provides promising results for non-stationary sinusoidal modeling. The local estimation of amplitude modulation (AM) and frequency modulation (FM) are then used for blind source separation through time-frequency clustering. This work is based on very recent theoretical findings related to [Fourer17a] which were extended to audio signal and source separation in a recent published paper [Fourer17c, Fourer18b].

The second approach directly operates in a time-frequency representation and uses image processing methods to apply source-specific filtering to recover the isolated sources, which compose a mixture. In our works, we focused on 3 promising methods:

1) Kernel Additive Modeling (KAM) proposed in [Liutkus14],
2) the Robust Principal Component Analysis (RPCA) [Candes11] and
3) Jeong-Lee's method based on global optimization [Jeong14].

A complete comparative evaluation of all three approaches was completed in 2018 and published as a preprint [Fourer18a].

The third approach aims at improving the source separation quality by investigating the role of the used time-frequency representation of the mixture. To this end, we combine several new synchrosqueezing techniques to enhance the results of a state-of-the-art BASS method called DUET [Jourjine 00] which can separate an arbitrary number of source from a two-channel mixture.

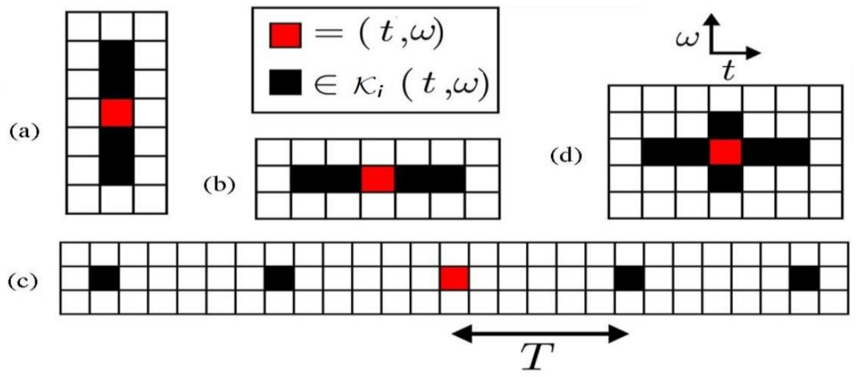
Within ABC_DJ, source separation can be used - either to increase the possibility of advanced mixing of the tracks in a playlist (by isolating completely the voice or removing it) – or to enhance a specific instrument in the mix to facilitate its detection and therefore improves the prediction of “soft” features. So far, only the second seems achievable in a short term.

5.1.1 Introduction

Since many years, the audio source separation problem has been addressed and remains intensively studied. While it aims to recover the isolated signal of each source (e.g. the instruments), which composes an observed mixture, it finds many applications like music remixing (karaoke, re-spatialization, isolated effects applied on a source), signal denoising (or signal enhancement) or polyphonic music processing. Here, we address the blind case where several sources (2 or more) are present in a single-channel instantaneous mixture expressed as the sum of the composing sources signals.

In a first approach, we propose to solve this problem thanks to local modulation estimation, which is used as a separation cue. This idea was developed for many years in the Computational Auditory Scene Analysis (CASA) methods [Breg90] assuming that a human ear can segregate sound sources signals because they have different evolution respect to time (e.g. common onset, common AM/FM parameters, etc.). This idea has very recently regained interest and has been validated in new experiments proposed in [Stoter16, Creag15, Creag16]. Here, we propose then a further investigation thanks to new advances in time-frequency analysis theory [Fourer17a].

In a second approach, this problem is solved through morphological filtering in the time-frequency domain. This approach is based on the observation that each source type has a specific time-frequency structure, which can be used for separation. The KAM approach [Liutkus14] uses median filtering with a source-specific kernel as illustrated below (cf. Figure taken from [Liutkus14]). For example, a percussive source can be described by the kernel (a) and a harmonic source can either be described by (b) or (d) if it has some modulations. The kernel (c) will fit for repetitive sources such as the musical background accompaniment. This idea is also used by Jeong-Lee approach but with a different mathematical formulation [Jeong14]. In [Candes11], RPCA is used to isolate the musical background accompaniment from the singing voice since they correspond to distinct matrices: the musical background spectrogram is a low-rank matrix and the singing voice spectrogram corresponds to a sparse full-rank matrix.



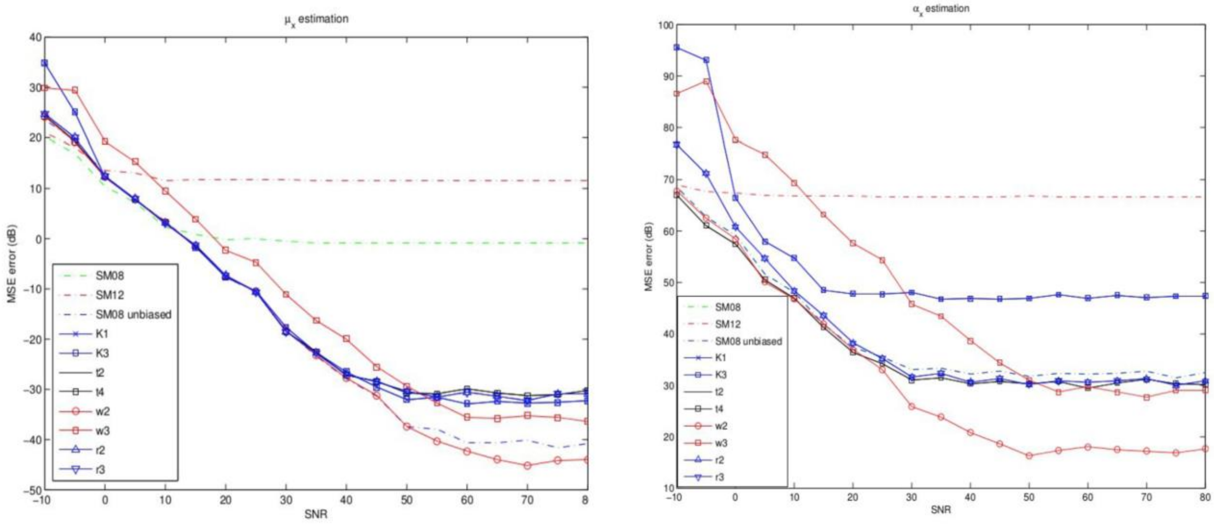
5.1.2 Approach 1: Source Separation by AM-FM Time-Frequency Clustering

We propose now an extension of the method proposed in [Fourer17a][Fourer18b] which allows a more accurate estimation of AM (amplitude modulation) and FM (frequency modulation) parameters at each time-frequency coordinate. This novel approach is based on the partial derivatives respect to time and to frequency, of the considered time-frequency representation: here we consider the STFT (Short-Time Fourier Transform of the input signal) used to compute a spectrogram.

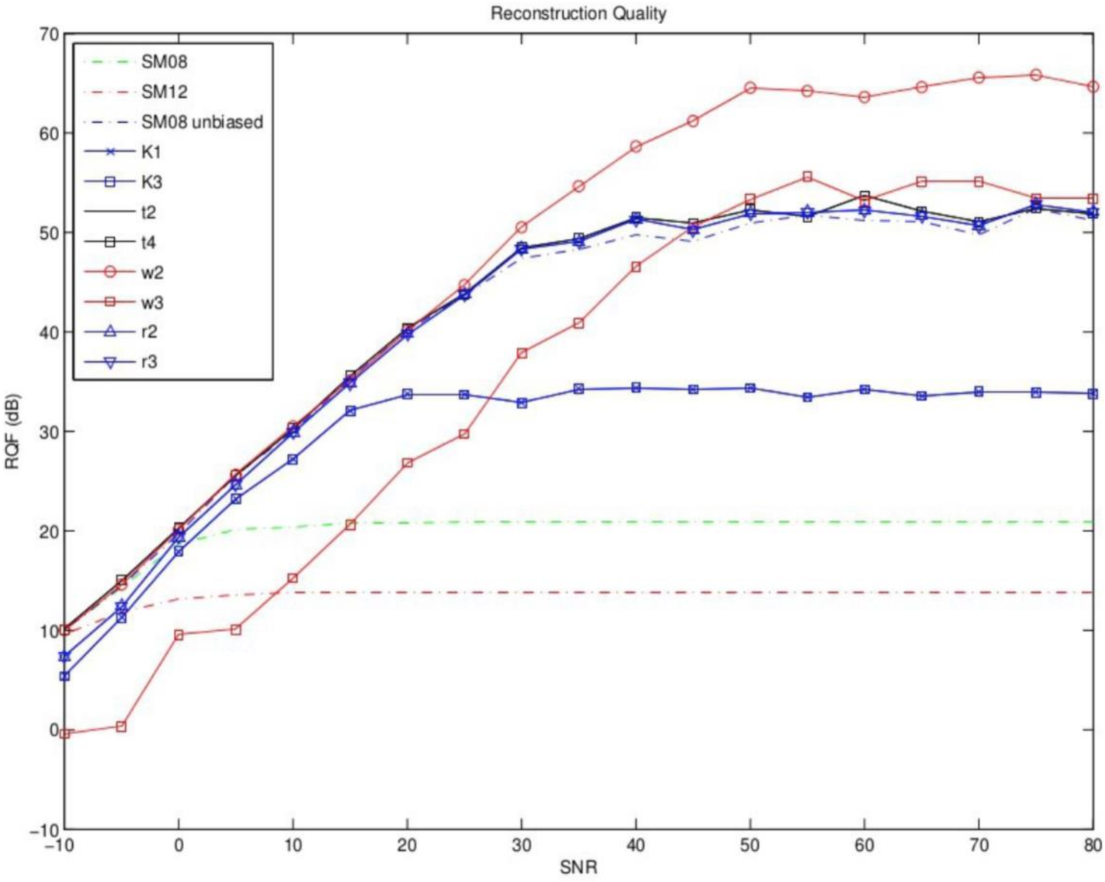
We also propose to apply these results in order to obtain a more accurate audio analysis-synthesis framework based on sinusoidal modeling. The overall proposed blind source separation method can be illustrated as follows where a time-frequency representation is computed from the input signal *x(t)* and is then clustered using the local modulations AM and FM estimated at each coordinate of the time-frequency plane.

The analyzed mixture is first expanded into a sum of sinusoidal components, which are then grouped according to the Coherent Amplitude Modulation (CAM) and the Coherent Frequency Modulation (CFM) features. In order to preserve the time continuity of each source, the spectral centroid and the CFM computed at each frame are connected to the closest candidate of the previous adjacent frame.

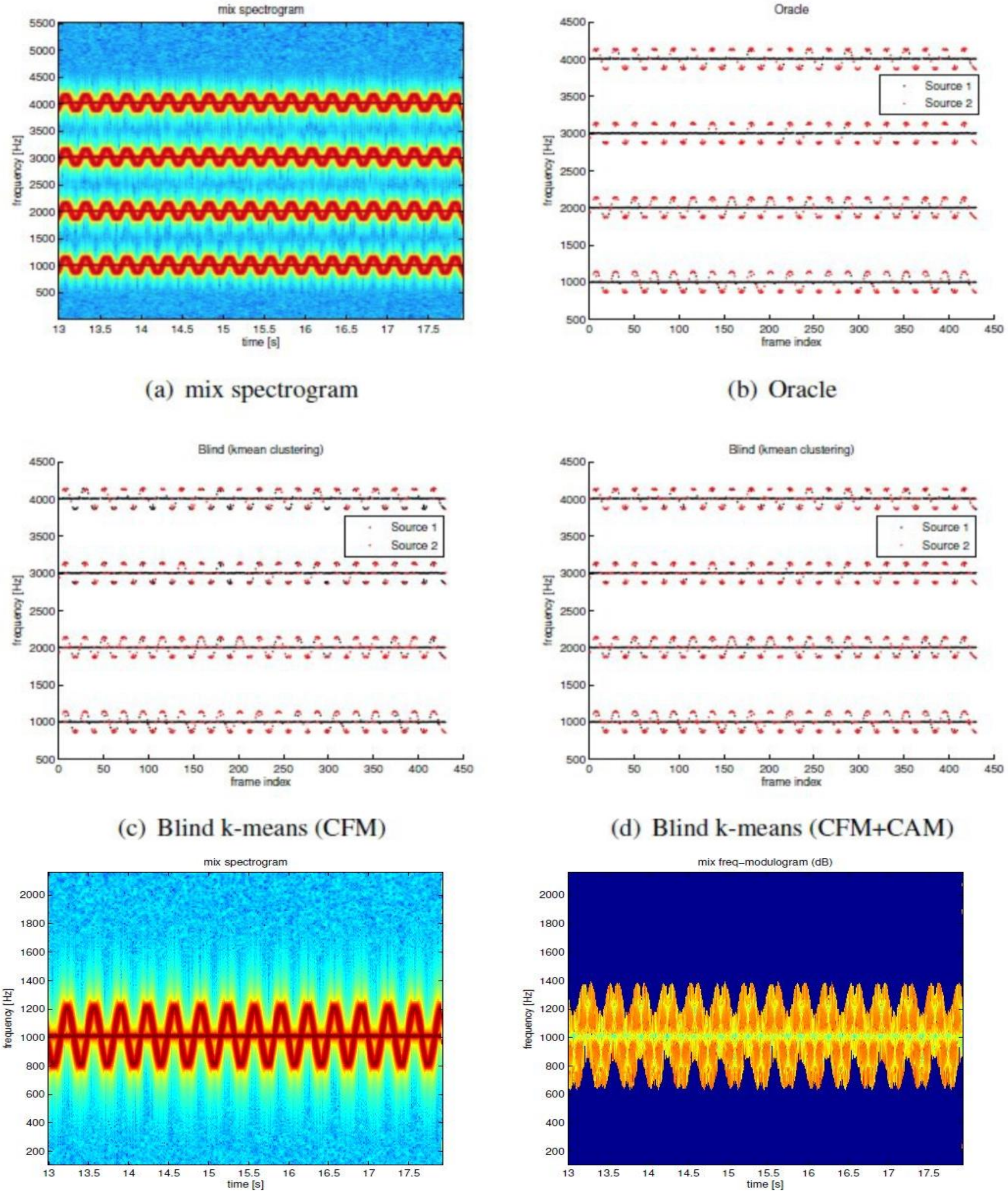
5.1.2.1 Audio signal modelling and local modulation estimation



The new proposed AM and FM estimators denoted k1, k3, t2, t4, w2, w3, r2 and r3 have been compared to the state-of-the-art ones [Mar08, Mar12]. The figure above shows a major improvement of these estimators, measured in term of the MSE (Mean Squared Error) as a function of the SNR (signal-to-Noise Ratio) when the input signal is merged with a white Gaussian noise. This improvement allows a better signal analysis based on sinusoidal modeling as illustrated in the figure below which compares the signal Reconstruction Quality Factor of a sinusoid merged with a white Gaussian noise with a SNR varying from -10 dB to 80 dB.



5.1.2.2 Blind source separation of synthetic signals



(a) mix spectrogram

(b) Oracle

(c) Blind k-means (CFM)

(d) Blind k-means (CFM+CAM)

The figures above show a separation example of two 5-seconds long sounds with different frequency modulation as illustrated in the first figure. The estimated sources can then be compared to the oracle (the ground truth solution). This result was obtained after a frequency tracking of each component and a local unsupervised clustering (k-mean) applied on the estimated sinusoidal components using the coherent local modulation features (CAM and CFM) estimated from the signal. This interesting result can be of interest for the separation of unison sounds (when 2 instruments play the same pitch).

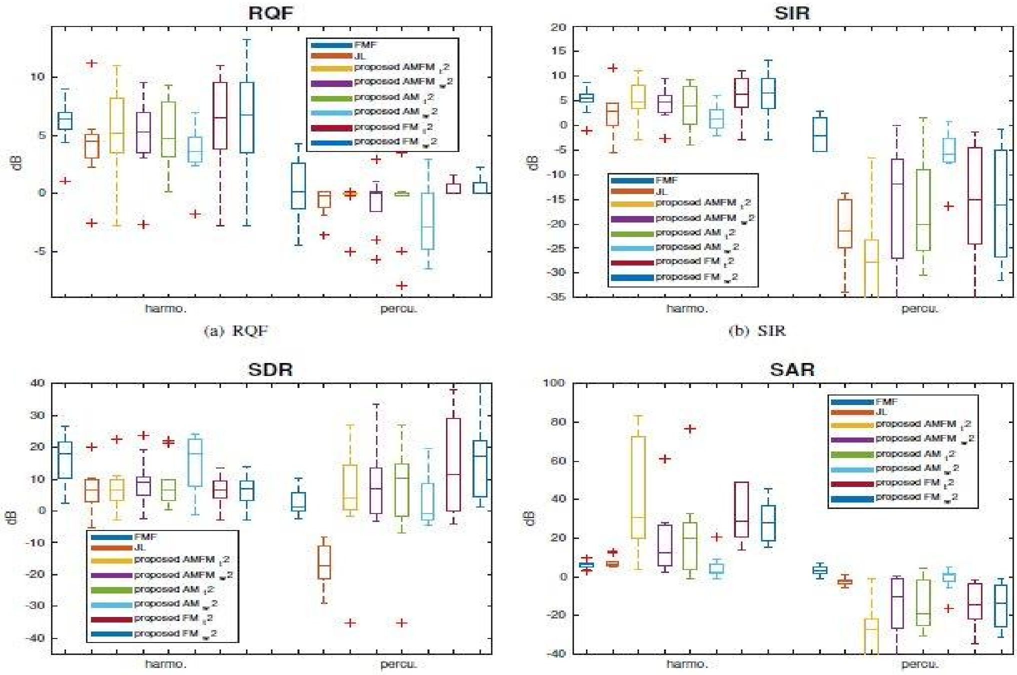
5.1.2.3 Blind source separation of real-world music mixture

The following table shows the separation results on a real-world mixture made of two audio sources (guitar + singing voice). The separation results are measured in terms of Reconstruction Quality Factor (RQF) as defined in [Fourer17c], Signal-to-Interference-Ratio (SIR), Signal-to-Artifact-Ratio (SAR) and Signal-to-Distortion-Ratio (SDR) as proposed by Vincent et al. in the BSS_EVAl [Vincent06]. Our results show different separation quality depending on the used features through k-means algorithm (used for unsupervised clustering).

Method	RQF (dB)	SIR (dB)	SDR (dB)	SAR (dB)
Oracle	11.67/10.33	23.84/26.86	11.41/9.91	11.69/10.01
MFC-kmeans	**5.42/7.42**	**9.31/12.65**	**3.97/6.60**	5.96/**8.06**
MAC-kmeans	0.67/2.64	0.60/**12.26**	-0.47/0.23	**8.80**/0.76
MFC/MAC-kmeans	0.78/2.73	0.67/**13.01**	-0.28/0.78	**9.40**/1.26

5.1.2.4 Application to harmonic/percussive source separation (HPSS)

In [Fourer 19], we propose a novel HPSS method based on the recently introduced local AM-FM estimators. We show that these estimators can also be used to discriminate the harmonic part from the percussive part of a music audio mixture. This method blindly operates in the time-frequency plane and assigns each point to a source according to its local modulation rate that is expected to be higher for percussive sounds than for harmonic components. This technique offers a simple and elegant mathematical formulation of the HPSS problem and can provide competitive results outperforming state-of-the-art methods when comparatively evaluated on a music dataset. Our results (presented in the figure below) compare our proposed algorithm (using estimators t2, w2) with the Fitzgerald median Filtering (FMF) HPSS [Fitzgerald10] and the Jeong Lee (JL) HPSS methods [Jeong14]. Our proposal obtains comparable and sometimes better results, especially for estimating the harmonic part as measured in terms of BSSEval [Vincent06].



(a) RQF (b) SIR

(c) SDR (d) SAR

Fig. 2. HPSS comparative results measured in terms of BssEval[20] for the Cano10 dataset.

5.1.2.5 Conclusion and Future Work

The main contribution of this research can be summarized as follows.

1) new enhanced estimators for local modulation (AM and FM) have been proposed and obtain a better accuracy than the state-of-the art methods.
2) a first application of these new estimators has been applied to audio sinusoidal modeling. The new methods obtain informal promising perceptual results and better objective results (measured in term of RQF) for audio signal analysis-synthesis (audio results here: http://www.fourer.fr/publi/spl18).
3) we obtain promising results for single-channel blind audio music source separation method based on local modulation. Two methods were developed: one allowing the separation of an arbitrary number of harmonic sources [Fourer18b]. The second allowing the separation of the harmonic part from the percussive part [Fourer19].

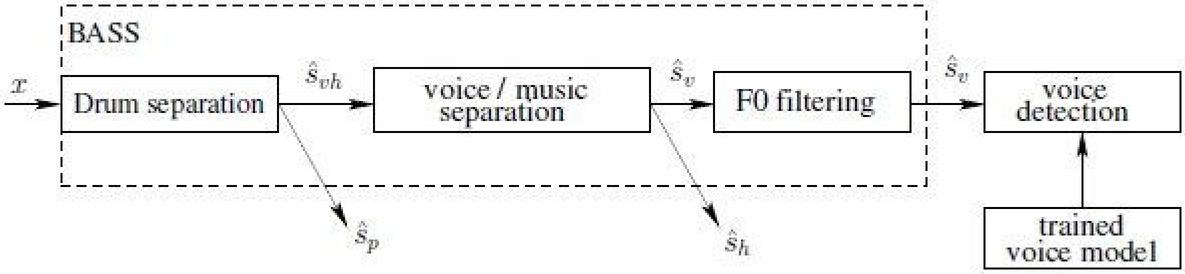
5.1.3 Approach 2: Source Separation through Time-Frequency Representation Morphological Filtering

This method assumes that the foreground voice and the instrumental music background have significantly different time-frequency regularities, which can be exploited to affect each time-frequency point to a source. To illustrate this idea, vertical lines are visible in a drum spectrogram due the spectral regularities at each instant, contrarily to a harmonic source which has horizontal lines due to the regularities over time at each active frequency (i.e. the partials). A recent comparative study [Lehner15] leads us to three very promising approaches, which can be summarized as follows.

1) Jeong and Lee [Jeong14] propose a total variation approach and minimize a convex auxiliary function, related to the temporal continuity (for harmonic sources), the spectral continuity (for percussive sounds) and the sparsity for the leading singing voice. The solutions provide estimates of the spectrogram of each source.

2) In [Candes11], the authors propose a voice/music separation using Robust Principal Component Analysis (RPCA) to decompose the mixture spectrogram into two matrices: a low rank matrix associated to the spectrogram of the repetitive musical background, and a sparse matrix associated to the lead instrument which plays the melody.
3) In [Liutkus15], the authors introduce the Kernel Additive Modelling (KAM) framework which unifies several approaches: - REPET [Rafii13], - Harmonic Percussive Source Separation (HPSS) through median filtering [Fitzgerald10]) using the specific regularities of the TFR associated to a source. In this framework, each source is characterized by a kernel, which models the vicinity of each time-frequency point in a spectrogram. This allows estimating each source through median filtering based on its specific kernel. This idea was extended through other source-specific kernels in [Liutkus14], [Liutkus15], [Kim16].

Hence, each method can be used in the following scheme as a Blind Audio Source Separation (BASS) step to obtain a singing voice detection system. This system can then work with or without a trained voice model as an unsupervised or a supervised singing voice method:

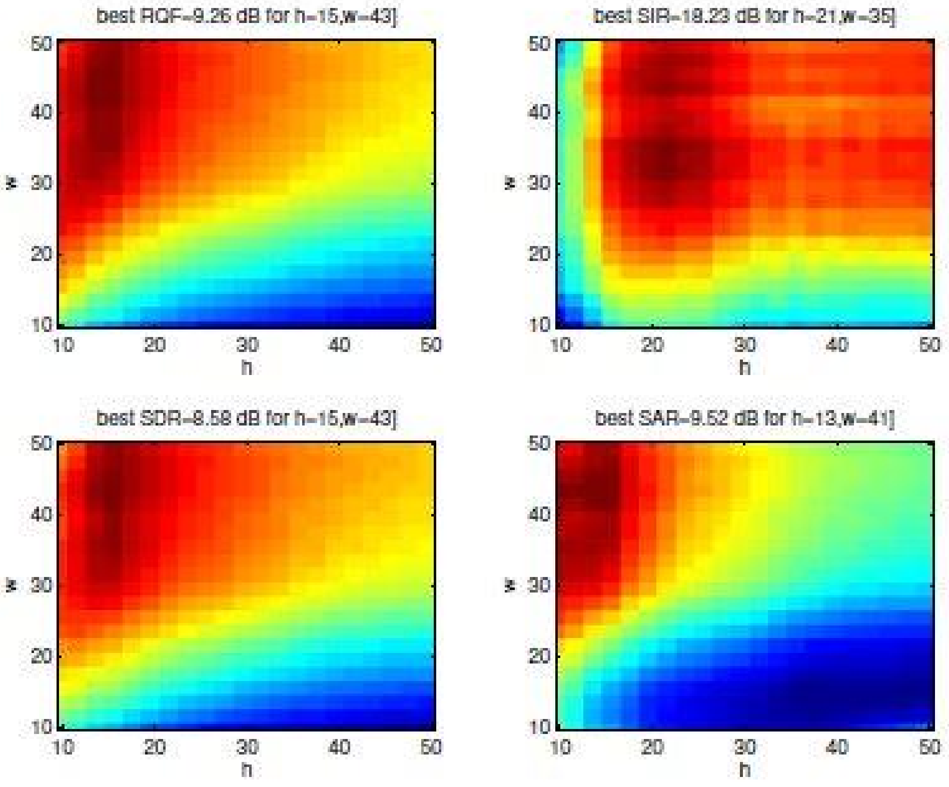


The first proposed implementation uses KAM with a REPET model combined with a threshold applied on the energy ratio between the signal associated to the voice signal and the signal associated to the music accompaniment (also called Voice-to-Music Ratio or VTMR).

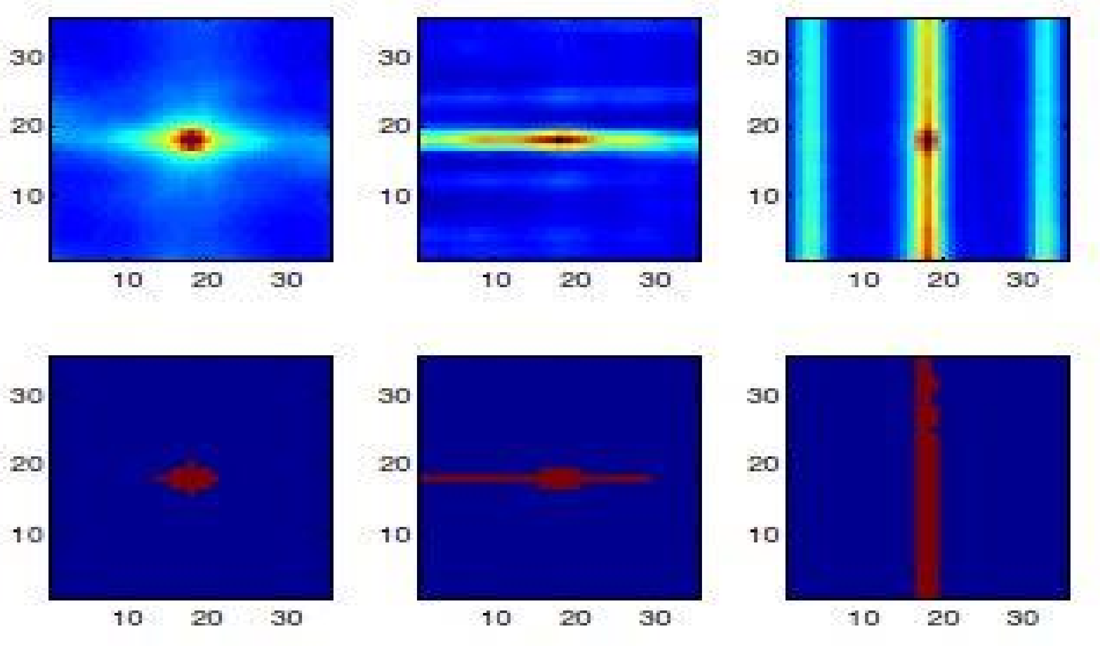
5.1.3.1 Towards a training method for supervised KAM-based source separation

Despite the KAM framework providing promising source separation results, this approach depends on the choice of the kernel and its parameters. However, to the best of our knowledge, no method exists in order to choose the best source-specific kernel to use.

First, we investigated through a grid search the parameters (dimensions w and h of the kernels), which provide the best separation results according to BSS_EVAL, when they are applied on a typical source separation problem (here the separation of a singing voice, from a drum set for which the kernels are known).
In the figure below, the red areas correspond to the best score (RQF, SIR, SAR and SDR) obtained as a function of the width w and the height h of the used source-specific kernels.



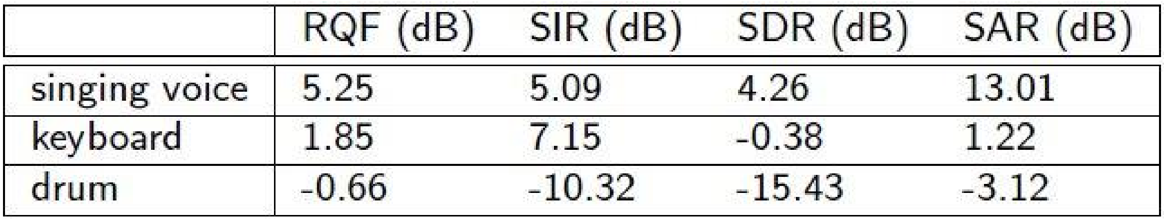
Furthermore, we also proposed a new algorithm which can provide a source-specific kernel from a previously computed time-frequency representation (i.e. a spectrogram). The main idea consists in computing the vicinity map corresponding to an averaged and reduced spectrogram after visiting each time-frequency point of a time-frequency representation. As a result, we can obtain kernels as the ones illustrated below respectively (from the left to the right) for a singing voice, a keyboard synthesizer and a drum set.



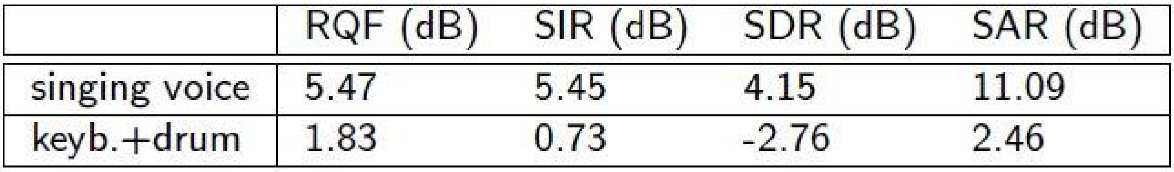
5.1.3.2 Source separation results through morphological clustering

For this evaluation, we compared the separation results on a mixture made of 3 sources (singing voice, piano and drum) sampled at 16kHz using the best parameters for each method (according to their respective paper).

1) Jeong-Lee's method (with and without drum separation)

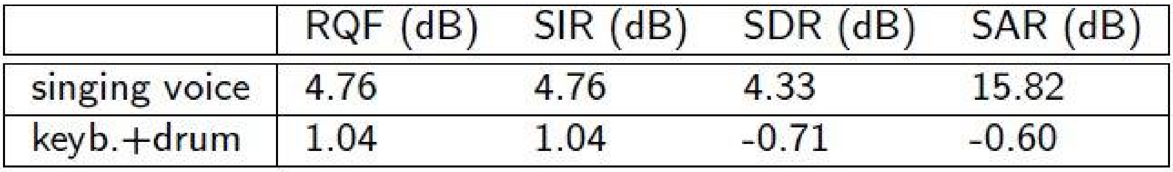


	RQF (dB)	SIR (dB)	SDR (dB)	SAR (dB)
singing voice	5.25	5.09	4.26	13.01
keyboard	1.85	7.15	-0.38	1.22
drum	-0.66	-10.32	-15.43	-3.12



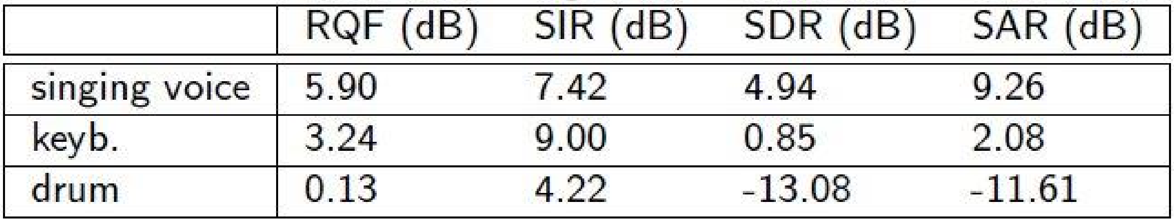
	RQF (dB)	SIR (dB)	SDR (dB)	SAR (dB)
singing voice	5.47	5.45	4.15	11.09
keyb.+drum	1.83	0.73	-2.76	2.46

2) RPCA

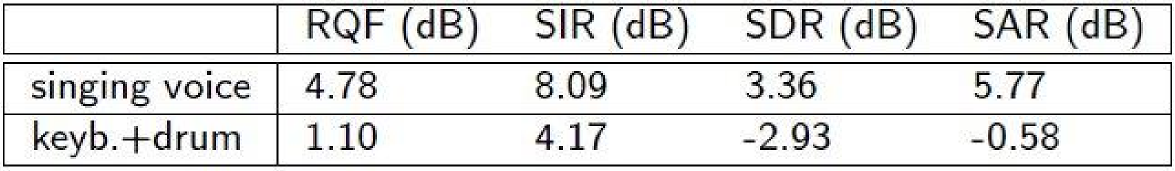


	RQF (dB)	SIR (dB)	SDR (dB)	SAR (dB)
singing voice	4.76	4.76	4.33	15.82
keyb.+drum	1.04	1.04	-0.71	-0.60

3) KAM (proposed trained and manually corrected set of kernels [Fourer17d], and REPET kernels voice/music [Liutkus15])



	RQF (dB)	SIR (dB)	SDR (dB)	SAR (dB)
singing voice	5.90	7.42	4.94	9.26
keyb.	3.24	9.00	0.85	2.08
drum	0.13	4.22	-13.08	-11.61



	RQF (dB)	SIR (dB)	SDR (dB)	SAR (dB)
singing voice	4.78	8.09	3.36	5.77
keyb.+drum	1.10	4.17	-2.93	-0.58

5.1.3.3 Conclusion and Future Work

In this work, we provided three main contributions:

1) A comparative study of 3 robust sources separation methods based on morphological filtering

2) A new algorithm for kernel training designed for the KAM approaches

3) A new singing voice detection method, which allows unsupervised classification.

A journal paper is in preparation including a further study [Fourer18a].

5.1.3.4 References

[Breg90] A. S. Bregman, Auditory scene analysis, MIT Press: Cambridge, MA, 1990.

[Candes 11] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" Journal of the ACM (JACM), vol. 58, no. 3, p. 11, 2011.

[Creag15] Elliot Creager, "Musical source separation by coherent frequency modulation cues," M.S. thesis, Department of Music Research, Schulich School of Music, McGill University, Dec. 2015.

[Creag16] Elliot Creager, Noah D. Stein, Roland Badeau, and Philippe Depalle, "Nonnegative tensor factorization with frequency modulation cues for blind audio source separation," in 17th International Society for Music Information Retrieval (ISMIR) Conference, New York, NY, United States, Aug. 2016.

[Fitzgerald10] D. Fitzgerald, "Harmonic/percussive separation using median filtering,"in Proc. Digital Audio Effects Conference (DAFx-10). Dublin Institute of Technology, 2010.

[Fourer17a] D. Fourer, F. Auger, K. Czarnecki, Meignen, and Flandrin, "Chirp rate and instantaneous frequency estimation", IEEE Signal Processing Letters. Vol. PP. Issue 99. DOI: 10.1109/LSP.2017.2714578. June 2017.

[Fourer17c] D. Fourer, G. Peeters and F. Auger, "Estimation of local AM/FM modulation: applications to sinusoidal modeling and to blind source separation", in Proc. GRETSI'17, Sept. 2017 (in french).

[Fourer18a] D. Fourer and G. Peeters, "Single-Channel Blind Source Separation for singing voice detection: A comparative study", *arXiv:1805.01201*, Mar. 2018.

[Fourer18c] Dominique Fourer and Geoffroy Peeters, "Fast and adaptive blind audio source separation using recursive levenberg-marquardt synchrosqueezing," in *Proc. IEEE ICASSP*, Calgary, Canada, Apr. 2018.

[Fourer18b] D. Fourer, F. Auger, and G. Peeters, "Local AM/FM parameters estimation: application to sinusoidal modeling and blind audio source separation," *IEEE Signal Processing Letters*, Aug. 2018.

[Fourer19] D.Fourer and G.Peeters, "Blind Harmonic/Percussive source separation using time-frequency reassignment-based AM-FM estimation," submitted to ICASSP 2019.

[Jeong 14] I.-Y. Jeong and K. Lee, "Vocal separation from monaural music using temporal/spectral continuity and sparsity constraints," IEEE Signal Process. Lett., vol. 21, no. 10, pp. 1197–1200, 2014.

[Kim16] H.-G. Kim and J. Y. Kim, "Music/voice separation based on kernel back-fitting using weighted beta-order MMSE estimation," ETRI Journal, vol. 38, no. 3, pp. 510–517, Jun. 2016.

[Lehner 15] B. Lehner and G. Widmer, "Monaural blind source separation in the context of vocal detection," in Proc. of the International Society for Music Information Retrieval Conference (ISMIR), 2015, pp. 309–315.

[Liutkus 14] A. Liutkus, D. Fitzgerald, Z. Rafii, B. Pardo, and L. Daudet, "Kernel additive models for source separation," IEEE Trans. Signal Process., vol. 62, no. 16, pp. 4298–4310, Aug. 2014.

[Liutkus15] A. Liutkus, D. Fitzgerald, and Z. Rafii, "Scalable audio separation with light kernel additive modelling," in Proc. IEEE International Conference on Acoust., Speech and Signal Process. (ICASSP), Brisbane, Australia, Apr. 2015, pp. 76–80.

[Mar08] S. Marchand and P. Depalle, "Generalization of the derivative analysis method to non-stationary sinusoidal modeling," in Proc. Digital Audio Effects Conference (DAFx'08), Sept. 2008, pp. 281–288.

[Mar12] Sylvain Marchand, "The simplest analysis method for non-stationary sinusoidal modeling," in Proc. Digital Audio Effects Conference (DAFx'12) , York, UK, Sept. 2012, pp. 23–26.

[Rafii13] Z. Rafii and B. Pardo, "Repeating pattern extraction technique (repet): A simple method for music/voice separation," IEEE/ACM Trans. on Audio, Speech, and Language Processing, vol. 21, no. 1, pp. 73–84, 2013.

[Stoter16] F. R. Stöter, A. Liutkus, R. Badeau, B. Edler, and P. Magron, "Common fate model for unison source separation," in 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) , Mar. 2016, pp. 126–130.

[Vincent 06] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," IEEE Transactions on Audio, Speech, and Language Processing (TASLP), vol. 14, no. 4, pp. 1462–1469, Jul. 2006.

5.1.4 Approach 3: Blind Source Separation of Stereo Mixtures Using Synchrosqueezing

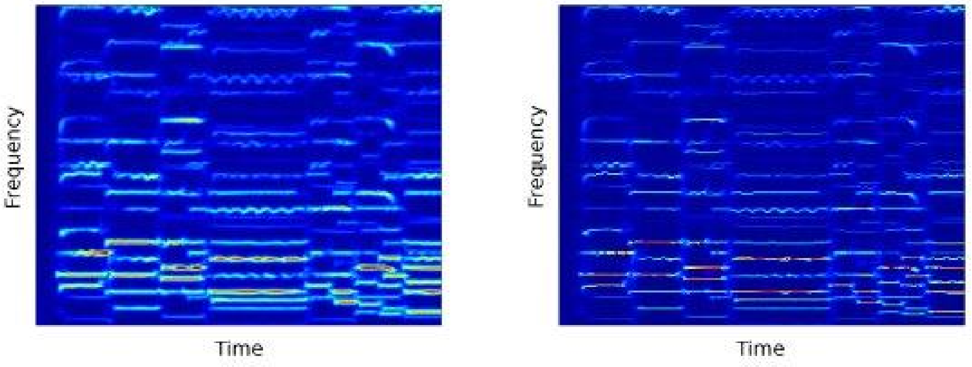
This approach revisits the Degenerate Unmixing Estimation Technique (DUET) for blind audio separation of an arbitrary number of sources given two mixtures through a recursively computed and adaptive time-frequency representation. Recently, synchrosqueezing was introduced [Daubechies 11] [Fourer 17] as a promising signal disentangling method which provides reversible and sharpen time-frequency representations. Thus, it can reduce overlaps between the sources in the time-frequency plane and can improve the sources' sparsity that is exploited by source separation techniques.

Another innovative part of this contribution consists in extending the synchrosqueezing method using the Levenberg-Marquardt algorithm to allow the computation of adaptive and adjustable time-frequency representations. As a result, our method improves the quality of the source separation process while remaining suitable for real-time applications.

5.1.4.1 Synchrosqueezing transform

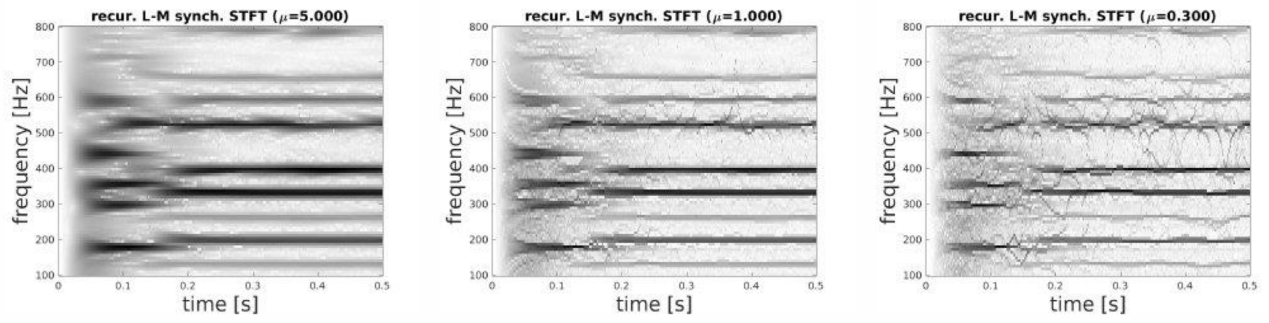
Synchrosqueezing [Daubechies 11] is a sharpening technique which improves the readability of a time-frequency representation while admitting a signal reconstruction formula. It is a post-processing technique which uses the frequency reassignment operators [Auger 95] to map values of a transform (here the short-time Fourier transform is considered) to new coordinates closer to the real support of the original signal in the time-frequency plane.

It results an increase of the readability of the time-frequency representation as illustrated below where the classical spectrogram of a music signal is compared to the squared modulus of the corresponding synchrosqueezed short-time Fourier transform.



(a) spectrogram $|X^h(t,\omega)|^2$ (b) synchrosqueezing $|\mathcal{S}X^h(t,\omega)|^2$

A second improvement first proposed in [Auger12] [Fourer 16] allows to make the synchrosqueezing transform adjustable through a damping parameter μ by applying the Levenberg-Marquardt algorithm of the reassignment operators. The result can be illustrated below where lower values for μ leads to a better energy concentration but with an increase of the sensibility to noise.

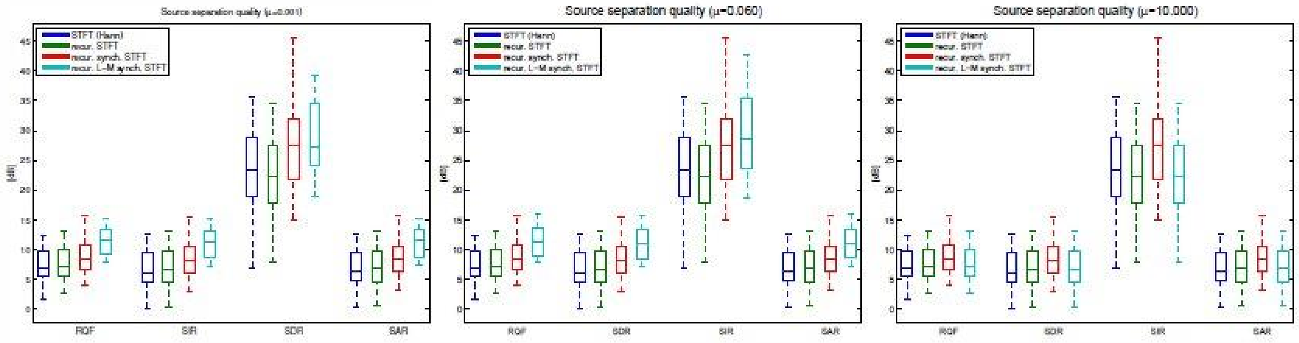


(c) $\mu = 5$ (d) $\mu = 1$ (e) $\mu = 0.3$

5.1.4.2 Numerical results

We comparatively evaluated the separation results on the freely available Bach10 music dataset (http://music.cs.northwestern.edu/data/Bach10.html) which contains 10 mixtures made of 4 instrumental sources. After generating random stereo mixtures from the isolated stems, we compare the results provided by our modified DUET algorithm [Jourjine 00] when combined with the proposed time-frequency representation computation methods. This experiment compares the results provided by recursive synchrosqueezing and recursive LM-synchrosqueezing with classical and recursively computed STFT.

The results show that the best separation results measured in terms of BssEval [Vincent 06] are provided by the LM-synchrosqueezing with μ=0.06. This optimal value for μ was empirically tuned in order to obtain the best disjoint-orthogonality of the 4 composing sources in the overall dataset.



(f) $\mu = 0.001$ (g) OPTIMAL value $\mu = 0.06$ (h) $\mu = 10$

Comparative source separation results measured in terms of Bss Eval [5], provided by the proposed methods and classical STFT applied on the Bach10 dataset. The results are obtained with different values of the damping parameter μ used by the recursive Levenberg-Marquardt synchrosqueezed STFT (other TFRs are not affected by μ).

5.1.4.3 Conclusion and future works

We have proposed new extensions of the DUET source separation algorithm using a recursive implementation of the Levenberg-Marquardt synchrosqueezed STFT . We have shown that synchrosqueezing can provide a sharpen and reversible time-frequency representations which improve the disjoint orthogonality between the sources. It results in a significant improvement of the source separation results in comparison with classical TFRs (an increase of the SIR of about +5dB in average). Future works will investigate new source separation methods based on time-frequency masking when they are combined with synchrosqueezing technique (e.g. CNN, NMF).

5.1.4.4 References

[Jourjine 00] A. Jourjine, S. Rickard, and O. Yilmaz, “Blind separation of disjoint orthogonal signals : Demixing N sources from 2 mixtures,” in Proc. IEEE ICASSP,Istanbul, Turkey, June 2000, vol. 5, pp. 2985–2988.

[Auger 95] F. Auger and P. Flandrin, “Improving the readability of time-frequency and time-scale representations by the reassignment method’ IEEE Trans. SignalProcess., vol.43, no. 5, pp. 1068–1089, May 1995.

[Daubechies 11] Daubechies, Ingrid, Jianfeng Lu, and Hau-Tieng Wu. "Synchrosqueezed wavelet transforms: An empirical mode decomposition-like tool." *Applied and computational harmonic analysis* 30.2 (2011): 243-261.

[Auger 12] Auger, François, E. Chassande-Mottin, and Patrick Flandrin. "Making reassignment adjustable: The levenberg-marquardt approach." *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012.

[Fourer 16] D. Fourer, F. Auger, and P. Flandrin, “Recursive versions of the Levenberg-Marquardt reassigned spectrogram and of the synchrosqueezed STFT,” in Proc. IEEE ICASSP, Shanghai, China, May 2016, pp. 4880–4884.

[Fourer 17] D. Fourer, J. Harmouche, J. Schmitt, T. Oberlin, S. Meignen, F. Auger, and P. Flandrin, “The ASTRES toolbox for mode extraction of non-stationary multicomponent signals”, in Proc. EUSIPCO, Kos island, Greece, Aug. 2017, pp. 1170–1174. https://github.com/dfourer/ASTRES_toolbox

[Vincent 06] E. Vincent, R. Gribonval, and C. Févotte, “Performance measurement in blind audio source separation,” IEEE Transactions on Audio, Speech, and LanguageProcessing (TASLP), vol. 14, no. 4, pp. 1462–1469, Jul. 2006.

5.2 Convolutional Neural Network (ConvNet) for Singing Voice

Executive Summary: We address here the audio singing voice segmentation problem from a polyphonic mixture. The goal is to propose a method that can label each time segment of an audio signal as containing or not containing a singing voice.

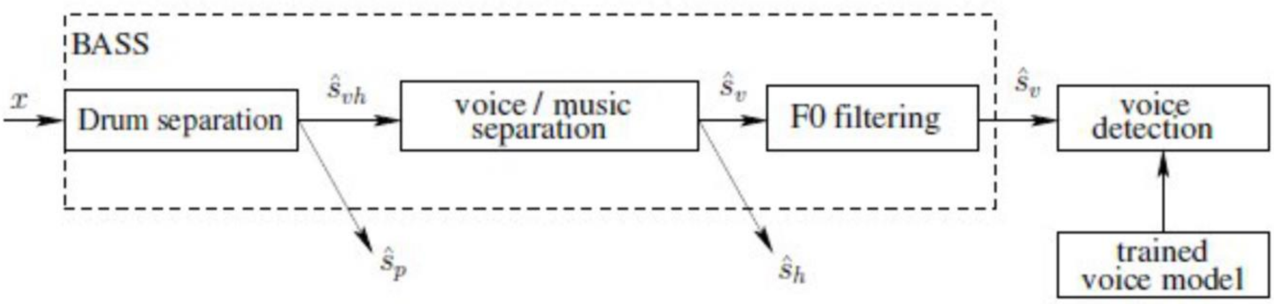
For this purpose, two approaches were investigated:

1) A new unsupervised method based on Blind Audio Source Separation (BASS), which uses the estimated signals associated to the singing voice and the one associated to the music accompaniment.

2) A recent state-of-the-art supervised method based on Convolutional Neural Networks (ConvNet) first proposed by [Schluter15], improved in [Schluter16] and currently investigated in the PhD-work of Alice Cohen-Hadria at IRCAM.

Within ABC_DJ, singing voice detection is used for “soft” auto-tagging and therefore playlist generation. Singing voice segmentation is used as part of the temporal description of the track which is to be used to facilitate automatic mixing between two tracks.

5.2.1 Unsupervised Method based on Blind Audio Source Separation



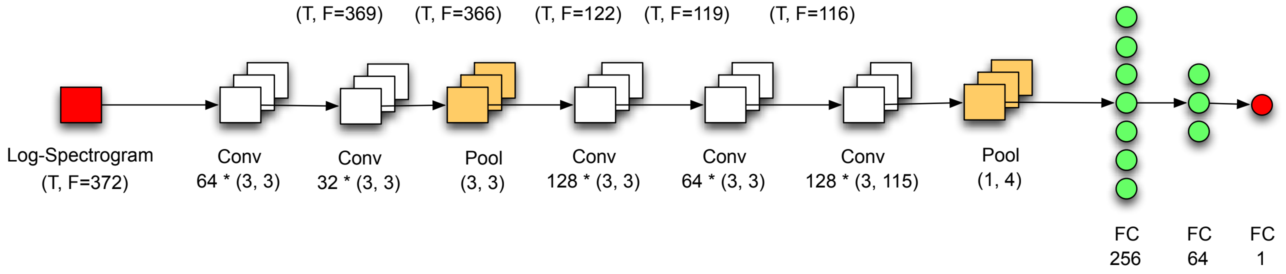
This method uses a BASS method as a preliminary step to recover a singing voice signal through different source assumption (non-repetitive, full-rank sparse matrix, etc.).

The estimated signal for the singing voice is then reinforced through a band-pass filtering in the range [120Hz, 3000Hz] and by applying a harmonic comb mask (obtained after estimating the fundamental frequency using the YIN algorithm [Cheveigne02]).

Finally, a decision is made based on a user-defined threshold, which is applied on the Voice-to-Music Ratio (VTMR). The VMR is defined as the energy ratio between the two signals through a sliding window.

5.2.2 Supervised Method based on ConvNet

This approach uses a classical supervised machine-learning framework where a ConvNet is trained on annotated samples before applying the classification on the testing samples using the trained model. We re-implement here the architecture of the network proposed in [Schluter16] using Log-Spectrogram as input. The architecture of the network is indicated in the Figure below.



5.2.3 Results

For this evaluation, we used three annotated datasets the audio tracks of which have a Creative Commons license. Each dataset has been split into a training-set and a test-set such as:

- JA: Jamedo [Ramona08]: 93 tracks (train:61, test:16)
- MI: MIR1K [Hsu10]: 1000 tracks (train: 828, test: 172)
- ME: MedleyDB [Bittner14 : 122 tracks (only 60 with voice) (train: 62, test: 34)

We considered a frame duration of 46 ms with a hop size of 16 ms.

The results are given in terms of mean recall (defined as the average between the voice and the non-voice recall).

The two tables below correspond respectively to two different experiments.

The first experiment considers each dataset independently. For each dataset, we indicate the results obtained by training the system on its training-set (except for the unsupervised approach which doesn't use training) and testing it on its testing-set.

In this experiment, the unsupervised approach corresponds to the KAM BASS method using a REPET kernel [Liutkus14 ,Liutkus15] combined with the proposed VTMR criterion (the drum was not previously separated and a F0-filter was not applied).

Dataset	**Unsupervised**	**ConvNet**
JA-train	0.56 (JA-test)	0.89 (JA-test)
MI-train	0.59 (MI-test)	0.90 (MI-test)
ME-train	0.73 (ME-test)	0.86 (ME-test)

The second experiment corresponds to a cross-dataset experiment. We only test here the ConvNet approach and test if this supervised approach can lead to over-fitting. For this we compare the results obtained by

- **Self-dataset**: training on the train parts of A + B, testing on the test parts of A+B
- **Cross-dataset**: training in the train parts of A+B, testing on the test part of C

Datasets	**Self-dataset**	**Cross-dataset**
JA-train + MI-train	0.89 (JA-test + MI-test)	0.75 (ME-test)
JA-train + ME-train	0.86 (JA-test + ME-test)	0.65 (MI-test)
ME-train + MI-train	0.84 (ME-test + MI-test)	0.77 (JA-test)

As one can see, when the dataset of the test part differ from the dataset of training part (cross-dataset) the mean-recall significantly decreases.

The results of the ConvNet from Table 1 ME-train/ME-test is 0.86; from Table 2 JA-train+MI-train/ME-test is only 0.75.

5.2.4 Conclusions and Future Work

The supervised singing voice method based on ConvNet obtains very promising results, which correspond to the state-of-the-art (in 2017). The unsupervised method shows its capability to obtain good results, as illustrated for the ME-test (0.73), which is comparable to the cross-dataset results with ConvNet (0.75).

Since the unsupervised approach has the advantage to not rely on any training (which is often computationally expensive and can lead to models that overfit the data), we still consider it as an interesting option. Future works will therefore consist in improving it. Especially we will optimize the choice of the kernels through a further comparative study in a both BASS/VD framework (cf. Paper [Fourer18a] published as preprint).

5.2.5 References

[Bittner14] R. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, "Medleydb: A multitrack dataset for annotation-intensive MIR research," in Proc. of the International Society for Music Information Retrieval Conference (ISMIR), Taipei, Taiwan, Oct. 2014

[Cheveigne02] A. de CHEVEIGNÉ et H. KAWAHARA: YIN, a fundamental frequency estimator for speech and music. Journal of the Acoustical Society of America, 111(4):1917–1930, 2002.

[Fourer18a] D. Fourer and G. Peeters, "Single-Channel Blind Source Separation for singing voice detection: A comparative study", *arXiv:1805.01201*, Mar. 2018.

[Ramona08] M. Ramona, G. Richard, and B. David, "Vocal detection in music with support vector machines," in Proc. ICASSP '08, Mar. 2008, pp. 1885–1888.

[Hsu10] C.-L. Hsu and J.-S. R. Jang, "On the improvement of singing voice separation for monaural recordings using the mir-1k dataset," IEEE/ACM Trans. on Audio, Speech, and Language Processing, vol. 18, no. 2, pp. 310–319, 2010.

[Liutkus 14] A. Liutkus, D. Fitzgerald, Z. Rafii, B. Pardo, and L. Daudet, "Kernel additive models for source separation," IEEE Trans. Signal Process, vol. 62, no. 16, pp. 4298–4310, Aug. 2014.

[Liutkus15] A. Liutkus, D. Fitzgerald, and Z. Rafii, "Scalable audio separation with light kernel additive modelling," in Proc. IEEE International Conference on Acoust., Speech and Signal Process. (ICASSP), Brisbane, Australia, Apr. 2015, pp. 76–80.

[Schluter16] J. Schluter, "Learning to pinpoint singing voice from weakly labelled examples," in Proc. of the International Society for Music Information Retrieval Conference (ISMIR), 2016, pp. 44–50.

5.3 Convolutional Neural Network (ConvNet) for Music Structure Boundaries

Executive Summary: We address here the problem of estimating the temporal boundaries of music structure. We extend a previously proposed method based on Convolutional Neural Network. More precisely, we propose a new input representation (the time-time Self Similarity Matrix) and we propose to use the depth of the input layer of a ConvNet to represent various viewpoints of the audio signal content (MFCC and Chroma). Our proposals allow to estimate the music structure boundaries more precisely than what can be achieved using current state-of-the-art methods.

Within ABC_DJ, music structure boundaries estimation is used as part of the temporal description of the track which is to be used to facilitate automatic mixing between two tracks.

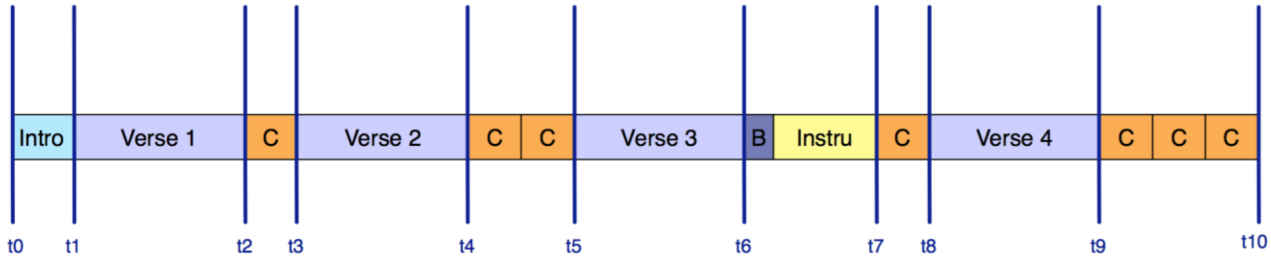
5.3.1 Introduction

Music structure discovery (MSD) is a recent research field, which aims at estimating automatically the temporal structure of a music track by analysing the characteristics of its audio signal over time. Such structure can be used for interactive browsing within a track or automatic summary generation, automatic DJ or computational musicology.

In MSD, the temporal structure of a music track is represented as a succession of segments. Such segments can correspond to

- Homogeneous audio content (also named “states”),
- Repeated audio content (also named “sequences” when they are non-homogeneous) or
- Non-homogeneous non-repeated content (in this case they are only defined as the temporal segment between two novelty boundaries).

In pop music, such segments can correspond to the verse, chorus or bridge parts of a song. In an MSD representation, each segment is characterized by its temporal boundaries and a label indicating its similarity with the other segments. In the present work, we only consider the problem of estimating the boundaries, i.e. the positions of the t0, t1, t2… in the following figure.



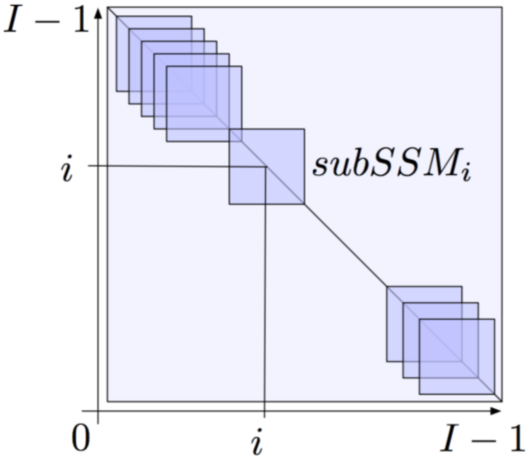
Research related to the automatic estimation of music structure started in 1999 with the work of Foote [Foote, 2000]. Until the accessibility of large annotated datasets, the methods used to estimate the music structure were mostly based on unsupervised learning algorithms: clustering or hidden Markov model applied to audio signal features, dynamic time warping, non-negative matrix factorization or singular value decomposition applied to a self-similarity matrix. Recently, large datasets of music annotated in structure have appeared (such as RWC, INRIA, SALAMI) allowing the use of supervised learning algorithms.

Following what happened in other domains (such as text or image recognition), neural networks with many hidden layers, a.k.a. deep learning, have allowed to largely increase the recognition results in various music audio recognition tasks (onset, beat, downbeat or music structure boundaries estimation). Various types of units were proposed to apply a network to an audio signal representation. [Boeck, 2011] or [Boeck, 2012] first proposed to use Bi-directional Long-Short-Term-Memory (BLSTM) units [Ullrich, 2014] or [Grill, 2015] later proposed to use the more tractable Convolutional units with even better recognition results. In this work, we will rely on Convolutional units. In this case, the network is named a Convolutional Neural Network (CNN or ConvNet).

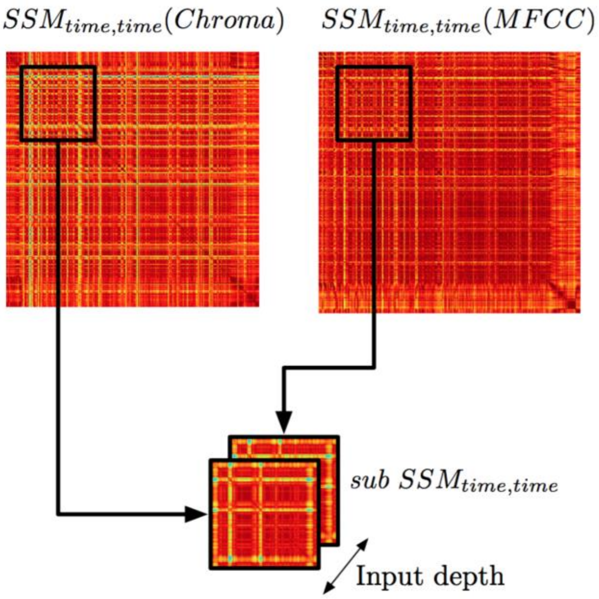
5.3.2 Proposed Method

In this work, we extend the work of [Grill, 2015] which propose to use ConvNet to estimate music boundaries. The full details of our method are described in our paper [Cohen-Hadria, 2017]. We only summarize its main points here and the differences with [Grill, 2015].

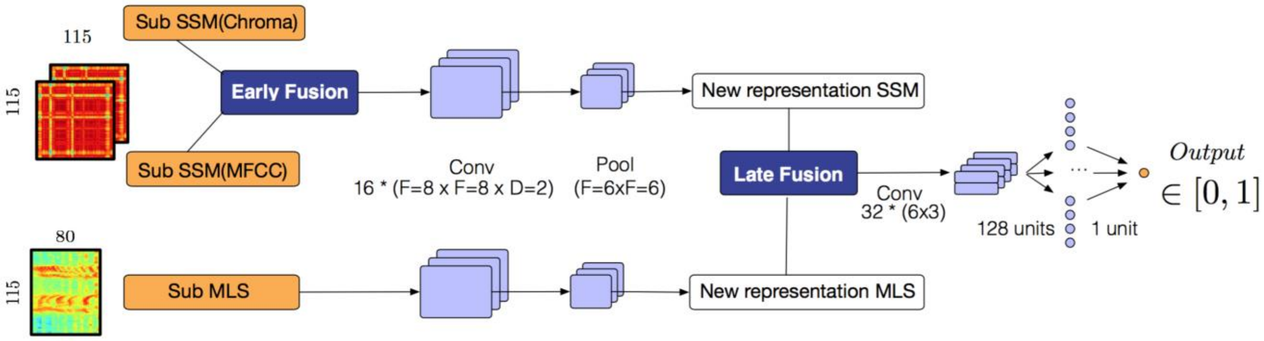
For the input layer of the network, we propose to use the succession over time of the square-sub-matrices centred on the main diagonal of a (time-time) Self-Similarity-Matrix (SSM). [Grill, 2015] proposed to use a (time-lag) SSM. In the case of homogeneous segments, the (time, time) SSM provides much sharper edges at the beginning and ending of segments than the lag-matrix. This representation was already used by Foote [Foote, 2000] to estimate music structure boundaries (by convolving it with a single predefined checker-board kernel) or by Kaiser and Peeters [Kaiser, 2013] (using several predefined kernels). We believe that using ConvNet on this representation will allow us to find even better kernels. This is illustrated in the following figure.



For the input layer of the network, we also propose to use its depth to represent various view-points on the audio content. Indeed, when computing a SSM, the choice of the signal representation plays a crucial role. Using Mel-Frequency-Cepstral-Coefficient (MFCC) or chroma as audio signal representation will lead to two different SSMs highlighting possibly two different temporal structures. We therefore propose to combine several SSMs using the depth of the input layer. Such depth is usually used in computer vision to represent the R, G, B colours of an image. This is illustrated in the following figure.



As in [Grill] we also use Mel-Log-Spectrogram as a third representation. A dedicated ConvNet is trained on it which is then merged (late-fusion) with the joint SSM(Chroma) and SSM(MFCC) network. This is illustrated in the following figure.



The output of the last neuron of the network gives a value between 0 and 1 indicating the likelihood of being a structure boundary. Based on this value, we tested two methods to decide on a boundary:

1) applying directly a threshold on the neuron value,

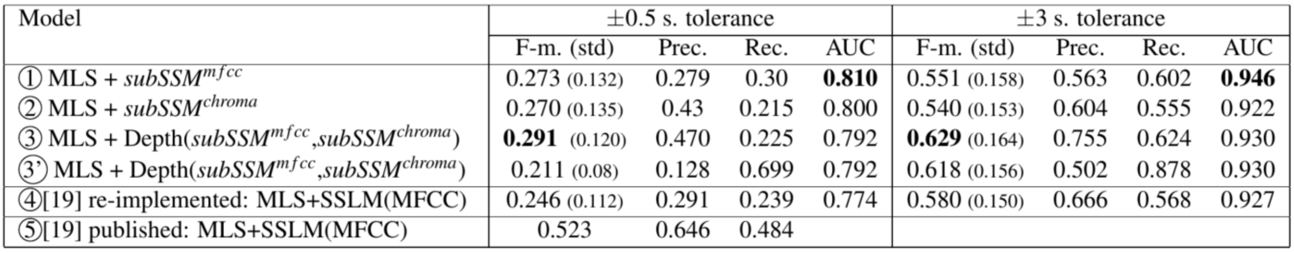
2) applying a peak-picking algorithm on the temporal sequence of neuron values.

5.3.3 Evaluation of the Proposed Method

We evaluate our structure boundary estimation method on the SALAMI dataset. For measuring the performances we used the Precision/Recall and F-Measure with two temporal precision windows (of 0.5s and 3s). We also used the AUC (Area Under the ROC Curve). This curve represents the True Positive rate versus the False Positive rate for all possible choices of threshold. The area under this curve represents the discriminative power of the method independently of the choice of a specific threshold.

It should be noted that we only had access to a part of the SALAMI dataset. Also, it should be noted that [Grill, 2015] used an additional private dataset for training their system in their [19] publication. In order to be able to compare our proposals to the method of [Grill, 2015] we therefore re-implemented [19] and tested it on our dataset. As one can see our re-implementation (0,246, row④ in the Table below) did not reach the published results of [19] (0,523, row⑤).

We first compare the use of a (time, lag) SSM (0.246 at row ④) to our proposal of (time, time) SSM (0.273 at row ①). The use of a (time, time) SSM seems beneficial at least for a precision window of 0.5s.
We then compare the early-fusion (using the depth of input layer) of a MFCC and Chroma (time, time) SSM (0.291 at row③) to the use of them individually (0.273 at row ①and (0.270 at row ②). Again, the early-fusion (using the depth of input layer) seems beneficial for both precision windows of 0.5s and 3s.
We finally compare the performances obtained using a peak-picking algorithm on the output neuron value (0.211 at row③) to ones obtained applying a peak-picking algorithm (0.291 at row ③'). The peak-picking algorithm improves a lot the results obtained for a precision window of 0.5s but not that much at 3s.



Model	±0.5 s. tolerance				±3 s. tolerance			
	F-m. (std)	Prec.	Rec.	AUC	F-m. (std)	Prec.	Rec.	AUC
① MLS + $subSSM^{mfcc}$	0.273 (0.132)	0.279	0.30	**0.810**	0.551 (0.158)	0.563	0.602	**0.946**
② MLS + $subSSM^{chroma}$	0.270 (0.135)	0.43	0.215	0.800	0.540 (0.153)	0.604	0.555	0.922
③ MLS + Depth($subSSM^{mfcc}$,$subSSM^{chroma}$)	**0.291** (0.120)	0.470	0.225	0.792	**0.629** (0.164)	0.755	0.624	0.930
③' MLS + Depth($subSSM^{mfcc}$,$subSSM^{chroma}$)	0.211 (0.08)	0.128	0.699	0.792	0.618 (0.156)	0.502	0.878	0.930
④[19] re-implemented: MLS+SSLM(MFCC)	0.246 (0.112)	0.291	0.239	0.774	0.580 (0.150)	0.666	0.568	0.927
⑤[19] published: MLS+SSLM(MFCC)	0.523	0.646	0.484					

5.3.4 References

[Boeck, 2011] Boeck, S. and Schedl, M. , "Enhanced beat tracking with context-aware neural networks", in Proc. of DAFx, 2011.

[Boeck, 2012] Boeck, S., Arzt, A., Krebs, F. and Schedl, M., “Online real-time onset detection with recurrent neural networks,” in Proc. of DAFx, 2012.

[Cohen-Hadria, 2017] A. Cohen-Hadria and G. Peeters. Music structure boundaries estimation using multiple self-similarity matrices as input depth of convolutional neural networks. In AES Conference on Semantic Audio, Erlangen, Germany, June 22–24 2017.

[Foote, 2000] Foote, J. (2000). Automatic audio segmentation using a measure of audio novelty. In Proc. of ICME.

[Grill, 2015] Grill, T. and Schlueter, J. (2015). Music Boundary Detection Using Neural Networks on Combined Features and Two-Level Annotations. In Proc. of ISMIR.

[Kaiser, 2013] Kaiser, F. and Peeters, G. (2013). Multiple hypotheses at multiple scales for audio novelty computation within music. In Proc. of ICASSP.

[Ullrich, 2014] Ullrich, K., Schlueter, J., and Grill, T. (2014). Boundary detection in music structure analysis using convolutional neural networks. In Proc. of ISMIR.

5.4 Cue Point Estimation

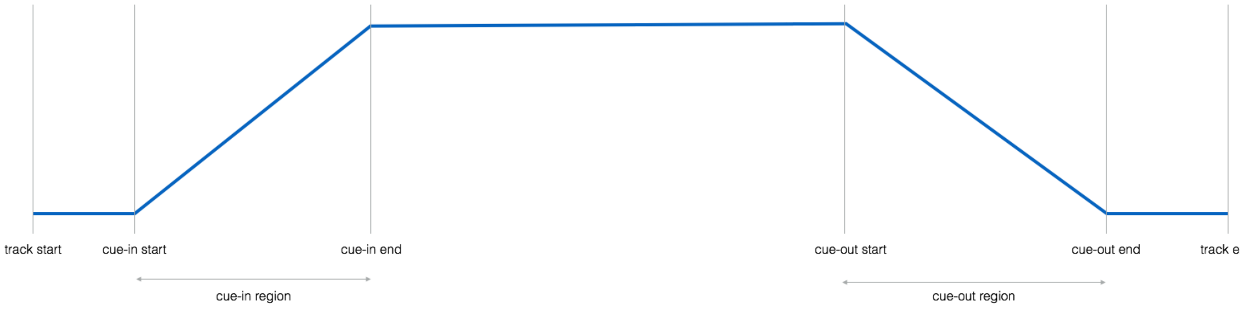
This task concerns the proposition of the *cue-regions* in the tracks where the fade-in and fade-out has to happen for the cross-fade to the next track in the automatic mix in the ISP. The automatically estimated proposition saves time for the human annotators of new tracks. It is thus related to Task T4.3 *Algorithms for sound design and feature developments for audio player*, because it is important for automatic mixing.
The proposed algorithm detailed below is based on the song structure estimation by the IrcamSummary module from T4.2, which determines significant parts of song, including the intro and outro sections which are used to place the cue-regions. This research has been presented at the Sound and Music Computing conference 2018 [Schwarz+2018].

The work is based on input provided by project partner *HearDis!*. Their music experts provided heuristic rules and an example database of tracks with cue points. The rules were then verified with the examples and those rules that could be realised computationally were implemented in a prototype automatic annotation software. The software was run on the example tracks and the results evaluated computationally and by human experts. Their feedback gave rise to adaptations in the algorithm, which improved the quality of the annotations. By following this structure, the chapter will also give an account of the informal iterative design process that allowed to keep the end users closely in the loop.

Also note that the project context is not DJ mixing for clubs or performance, but point-of-sale (PoS) automatic mixing in shops, based on semi-automatic music annotation and generated playlists. However, the automatically produced mixes should retain club-DJ quality (beat synchronicity, cross-fades).

Cue points define the regions where tracks in a DJ mix fade in or out to blend with other tracks (see [fig:cuepoints]).

DJs will usually choose them by hand according to the context of the current DJ set, based on their experience and familiarity with the specific track. However, when computer support or automation of DJ mixing is called for, we have to devise heuristics and an algorithm that can analyse the music content of a track in order to come close to the human decision.



[fig:cuepoints] Cue points and cue-regions with the resulting volume fade curve for one track in a DJ mix.

In our context of PoS automatic mixing, the automatically estimated cue-region proposition saves time for the human annotators of new tracks to be included in new automatically generated playlists, who only have to verify the automatic annotation and correct it if necessary.

5.4.1 Human Expert Rules

In order to get a high-level framing of the problem of cue point estimation from the point of view of the users, project partner *HearDis!* provided the content- and context-based criteria below for the choice of cue-regions for the aim of PoS automatic mixing. The concern in that case is to decide if the song can start immediately, or if the intro has to be shortened, possibly because it is an extended club-DJ-friendly version, and if the end has to be shortened because in-store music needs to change more often than club music in order to achieve a higher level of variety.

- Track is too long in general (more than 6 to 7 minutes)
- Intro is too repetitive (especially DJ-friendly versions)
- Intro is too quiet for too long (more than 4 to 8 bars) until track is of discernible loudness (at the PoS) EXCEPTION: Artist is already singing
- Intro is too noisy/non-musical
- Outro is too repetitive
- Loudness drops significantly but outro lasts longer than 4 to 8 bars EXCEPTION: Artist is still singing
- Outro is too noisy/non-musical
- Generally silence at the beginning and end of a track should be shortened to a minimum

We can already see that many of these points are dependent on audio and musical content (repetition, presence of voice) and even cultural context (what is *too noisy* or *non-musical*?).

The key point of the ensuing work was to find out which of these criteria were computationally feasible with the current tools, and whether the rate of errors with regard to the unfeasible criteria not modeled in our algorithm was acceptable.

5.4.2 Ground Truth Database of Cue Points

HearDis! provided a set of 30 example tracks in MP3 format, each in two versions:

1. the full-length track
2. the track shortened according to human-decided cue-in and cue-out regions with fades applied to them

We then annotated the start and end points of the cut regions, and the durations and kinds of fades or cuts by hand in text label format as produced by *Audacity*[1]. This does provide example cases of shortening and fade times. The results of a statistical analysis of the annotations are given in [tab:mean]–[tab:duration]. Statistics of cue-region durations are always given with the zero-duration regions removed, since they correspond to “cut” transitions (no cross-fade).

[1] http://audacity.sourceforge.net

segment	mean / median start time	mean / median / number of non-zero durations
cue in	13.6 / 8.9	1.3 / 0.9 / 23
cue out	290.0 / 316.7	8.4 / 9.2 / 4
track end	369.1 / 363.0	n/a

[tab:mean] Statistics of start time and duration of ground truth cue-regions for the 30 example tracks in seconds.

segment	min / max start time	min / max end time	min / max non-zero duration
cue in	0.0 / 57.4	0.0 / 57.4	0.2 / 4.3
cue out	163.9 / 418.0	166.6 / 428.8	2.0 / 14.6
track end	182.0 / 741.1	182.0 / 741.1	n/a

[tab:duration] Statistics of minimum/maximum start and duration of ground truth cue-regions for the 30 example tracks in seconds.

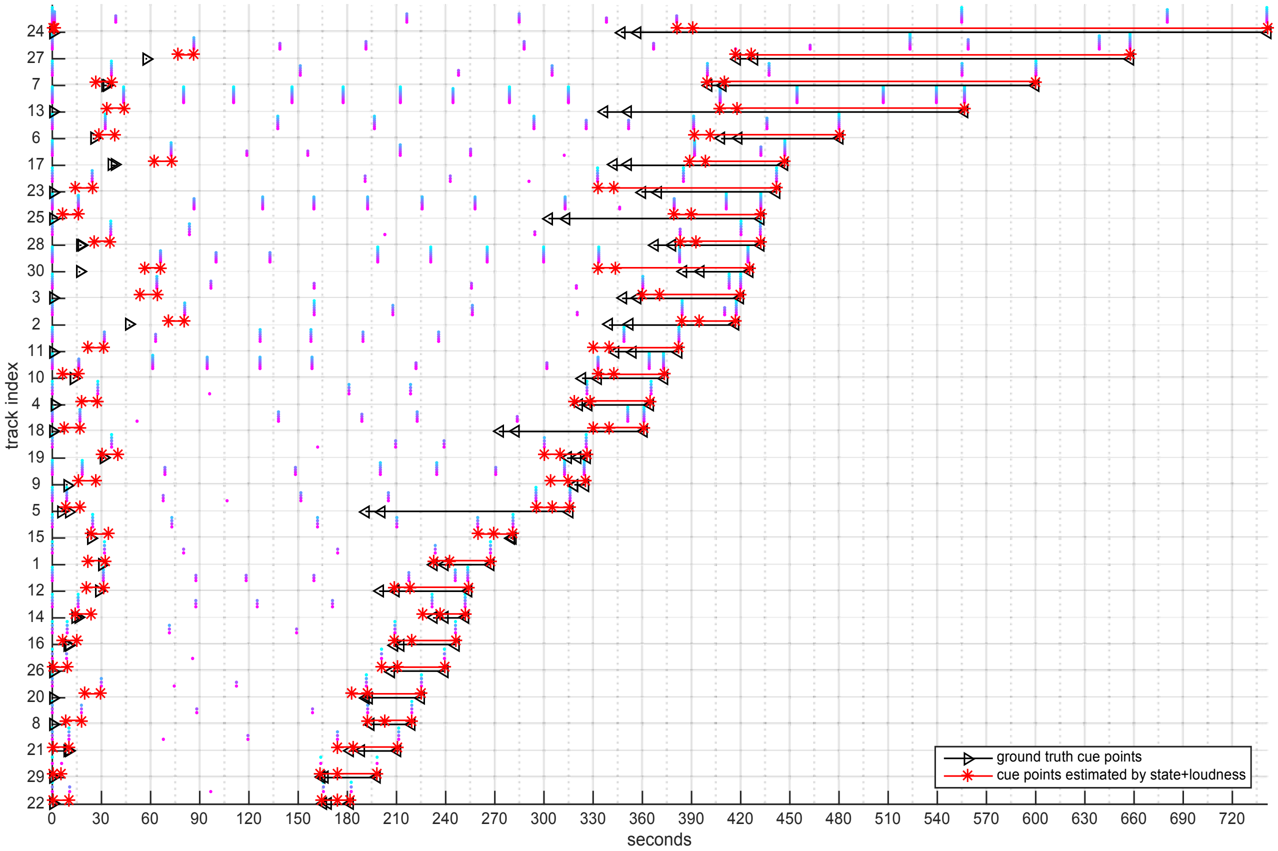
Furthermore, the examples revealed other content-based decisions, such as, in one track, removing one repetition of the exposition of a synth line by cutting the intro in half, or removing redundancy in long end parts of songs.

5.4.3 Comparison of Ground Truth Cue Points with Music Structure Analysis

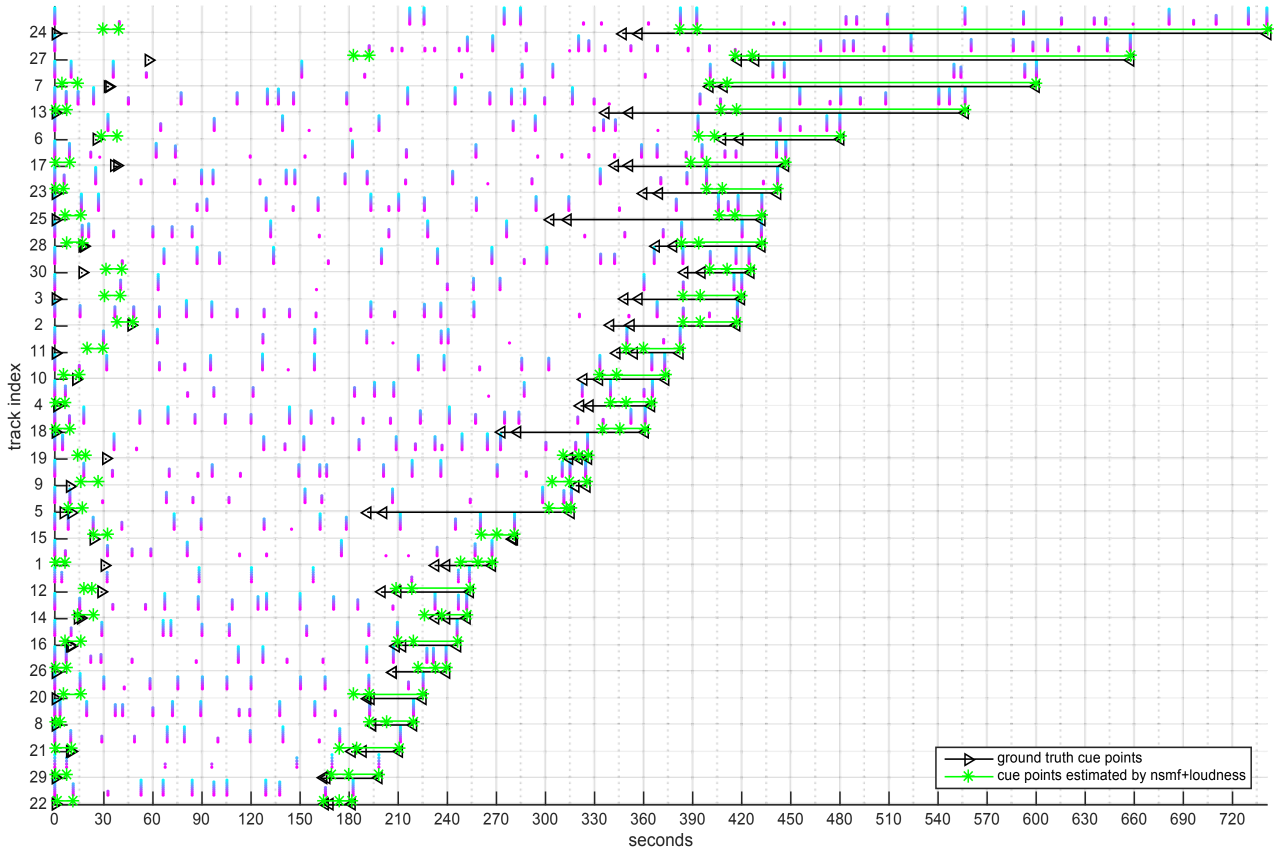
The audio-based music structure analysis algorithm [Kaiser2013] implemented in IrcamSummary divides a piece of music into its significant parts, and organises them into classes (e.g. corresponding to intro, outro, chorus, verse). This is done on multiple cue-points levels, where, from the lowest to the highest level, the structural segments are fused into larger classes.
Our hypothesis was that the intro and outro segments would stand out and could be a good basis for cue-regions. We verified this by calculating two versions of automatic structural analysis, using the tool *IrcamSummary*, on the full-length example tracks.

The first version, *state* mode, is based on a fusion of homogeneous state and sequence repetition segmentations. The second version, *NSMF* mode [Kaiser2012], uses non-negative matrix factorisation (NMF) of similarity matrices as a mid-level representation to classify the structure. This mode is called *NSMF* for *Non-negative Similarity Matrix Factorization*.

We then compare the structural segments with the human suggestions of cue-regions. The plots in [fig:structure-state] and [fig:structure-nsmf] show the ground truth cue-regions of the example tracks overlaid with multi-level structure analysis regions. Each level's segment boundaries are shown on one horizontal line as coloured dots, from lower levels in violet to higher levels in cyan, which proceed by fusing lower-level segments. Right-pointing triangles show the cue-in fade region, or cut, when only one triangle is visible (length of zero). Left-pointing triangles show the cue-out region and end point of the full-length example. Tracks are sorted by length, for easier observation of maximum final track length.



[fig:structure-state] Ground truth cue-regions of the example tracks (black) overlaid with IrcamSummary multi-level structure analysis regions in state mode (violet to cyan dots), and cue points estimated by final algorithm (red).



[fig:structure-nsmf] Ground truth cue-regions of the example tracks (black) overlaid with IrcamSummary multi-level structure analysis regions in NSMF mode (violet to cyan dots), and cue points estimated by final algorithm (green).

These plots reveal that, first, the lowest (most detailed) level *state* mode summary in [fig:structure-state] is more pertinent, since it has a segment structure better coinciding with the annotations (it is also beat-synchronous, unlike the *NSMF* mode summary in [fig:structure-nsmf]), and, second, that almost half of the songs were not shortened at the beginning:

- 14 cue-in start points are cuts at song start
- 16 cue-in start points are within the first structure segment
- only 3 cue-in segments are longer than 1 second[2]
- the cut-off point for long tracks is mostly between 5:30 and 6:00, with 3 tracks going until 7 minutes

5.4.4 Cue Point Estimation Heuristic Algorithm

The first proposed algorithm detailed below is solely based on the song structure estimation by the *IrcamSummary* module, which determines significant parts of the track, including the intro and outro sections which are used to place the cue-regions. For this first iteration, we wanted to see how far we would come only with song structure information, without taking the audio content into account.

The above observations from the example tracks suggest the following heuristic algorithm for the cue-region estimation (always of 10 s length):

```
function CUE-IN
    return cue region such that its end coincides with
           the end of the first long enough (10 s) structural region at the lowest level
end function

function CUE-OUT
    if the song is not too long ($<= 6$ min) then
        return start of the last structural region that is
               long enough (10 s) as cue-out start
    else
        ▷ for long songs, we apply the explicit rule to
          shorten songs that are too long (see 3.1)
        if there is a structural segment longer than 10 s in
           the time span between 5:30 and 7 minutes [6]
        then
            return the cue-out region placed at its start
        else
            return cue-out start at 5:30
        end if
    end if
end function
```

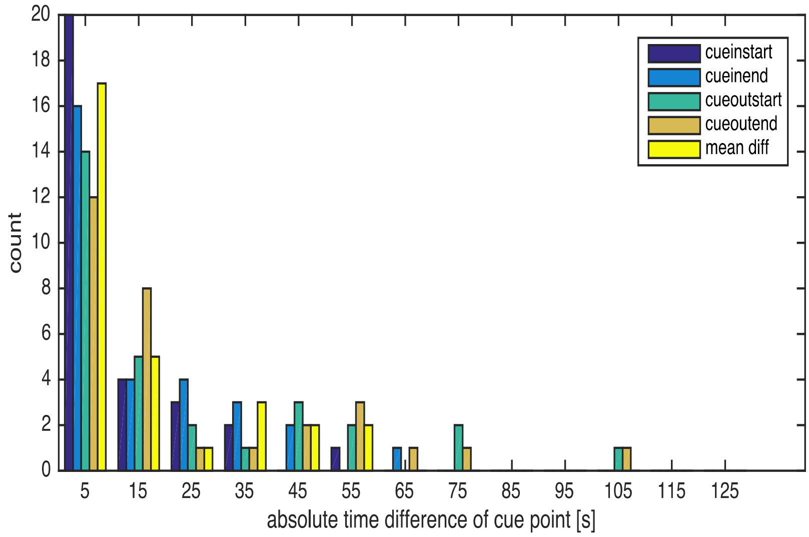
[fig:algorithm] Cue point estimation heuristic algorithm in pseudo-code.

[2] The frequent presence of cuts instead of fade regions are due to the fact that the existing PoS playout system at *HearDis*! does not yet do cross-fade mixing, so that the experts are trained to find cue points where tracks can cut from one to the next.

5.4.5 First Evaluation of Cue Point Detection

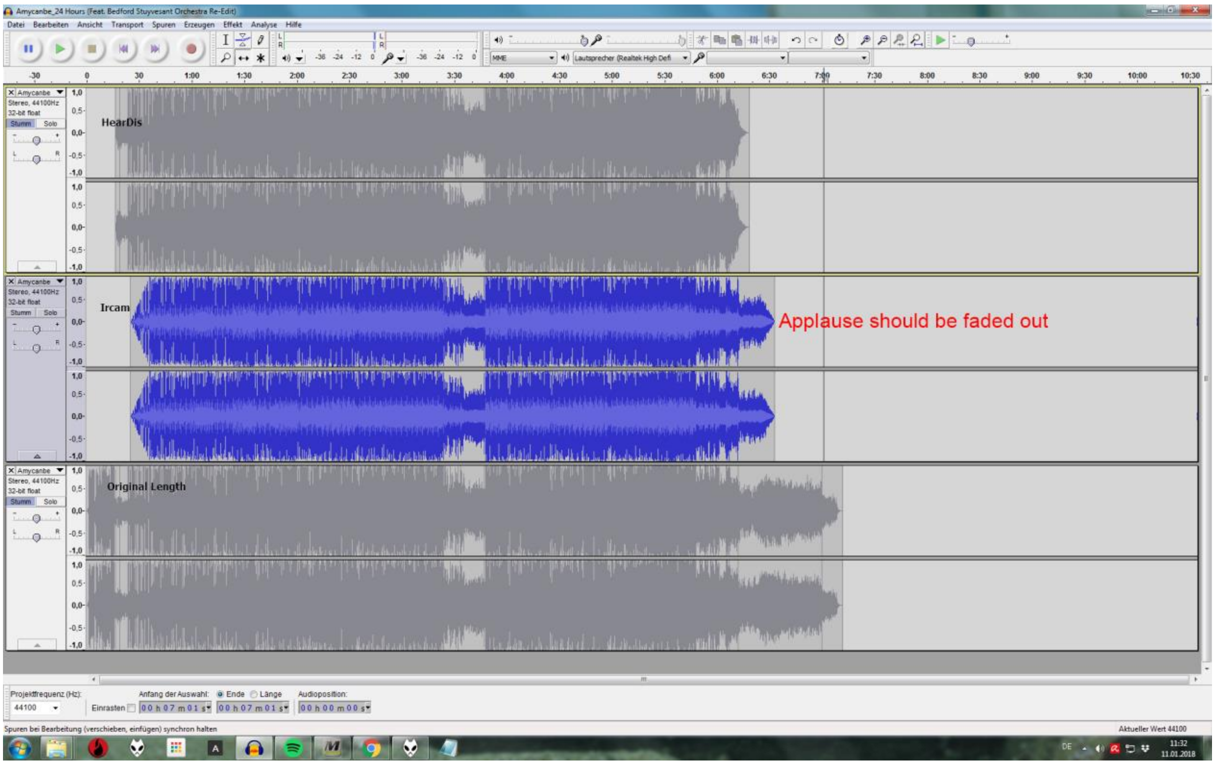
We created cue-region estimations for the tracks in the cue point example database. To facilitate evaluation, we also exported the example tracks faded at the estimated cue points. These examples were evaluated by music annotators at *HearDis!* in order to validate the heuristics. (But keep in mind that the cue point estimation is always only a starting point for a human annotator who solely is in the position to correctly judge the content and context of the tracks.)

Nevertheless, the numerical comparison between the estimated cue points and the hand-annotated ground-truth cue points from the test database in [fig:cuehist] shows that over 50% of estimated cue points are within 10 seconds from the manual choice and for two thirds of the examples the absolute time differences are smaller than 20 seconds. The two outliers with cue-end estimation differences over 60 s are due to a more flexible interpretation of the shortening rule by the human annotators (some tracks were left at much longer than the cutoff rule of 6 minutes).



[fig:cuehist] Histogram of the time difference between hand-annotated ground-truth cue points and cue points estimated based on state mode

A subjective but systematic evaluation of the automatically faded example tracks was carried out by the human music experts at *HearDis!*. They provided precise feedback in the form of screenshots with the 3 versions of each track aligned (original, human-cut, automatically cut) and remarks for the problematic cases (see [fig:evalex]).


Amycanbe_24 Hours (Feat. Bedford Stuyvesant Orchestra Re-Edit)
Datei Bearbeiten Ansicht Transport Spuren Erzeugen Effekt Analyse Hilfe
HearDis
Ircam
Applause should be faded out
Original Length
Projektfrequenz (Hz):
44100
Anfang der Auswahl:
Ende
Länge
Audioposition:
Einrasten
Spuren bei Bearbeitung (verschieben, einfügen) synchron halten
Aktueller Wert 44100

[fig:evalex] Example of subjective feedback: The human expert hand-aligned the original track (bottom), the manually cut and faded track (top), and the automatically cut and faded track (middle) to evaluate and comment on important differences.

The feedback was positive about the algorithmic choice of cue points, with the remark that, for PoS applications, it is always OK to cut more than a human annotator at beginning and end. There were only 5 problematic cases, listed in [tab:feedback].

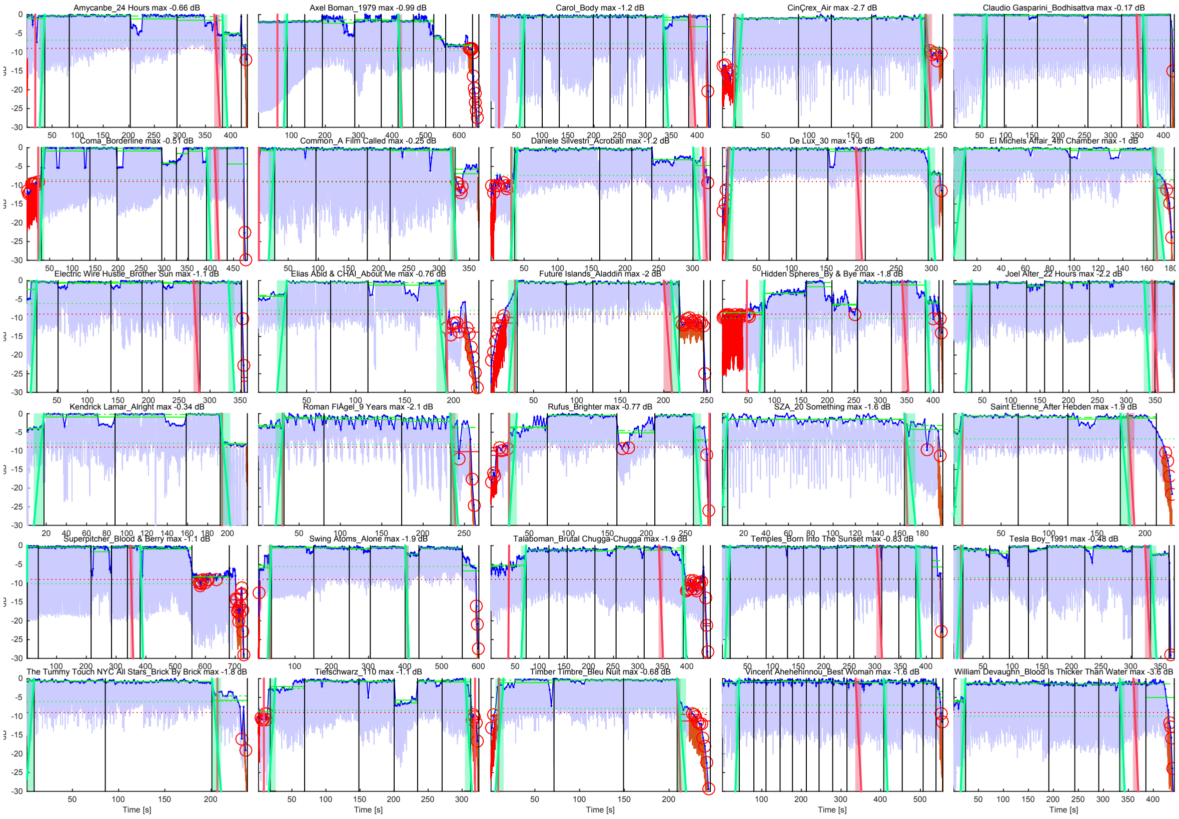
The remarks show the limits of our simple algorithm, where the human decision mobilises deep content- and context-dependent knowledge up to the cultural level (e.g. that applause is a special noise that marks the end of a performance).

position	evaluator remark	computable	observations
end	end applause should be faded out	no	music continues during applause
end	just noises?	no	free guitar + voice
end	too noisy and weird	no	fade in of 2sec of Morse code
start	intro too long?	yes	intro very low volume -24dB
end	outro too long	yes	outro -15 dB (last struct segment silence)

[tab:feedback] Feedback on individual tracks in the first version of cue point detection output by human expert, and assessment of computability.

5.4.6 Second Iteration

Based on the above feedback, we chose to implement the only computationally feasible content-based criterion for cue-in/cue-out estimation of loudness. From an analysis of the example tracks shown in [fig:rms], we could determine that a threshold of -9 dB relative to the max loudness of the track catches all the cases where an intro or outro had been considered as too quiet, without introducing false positives.[3]



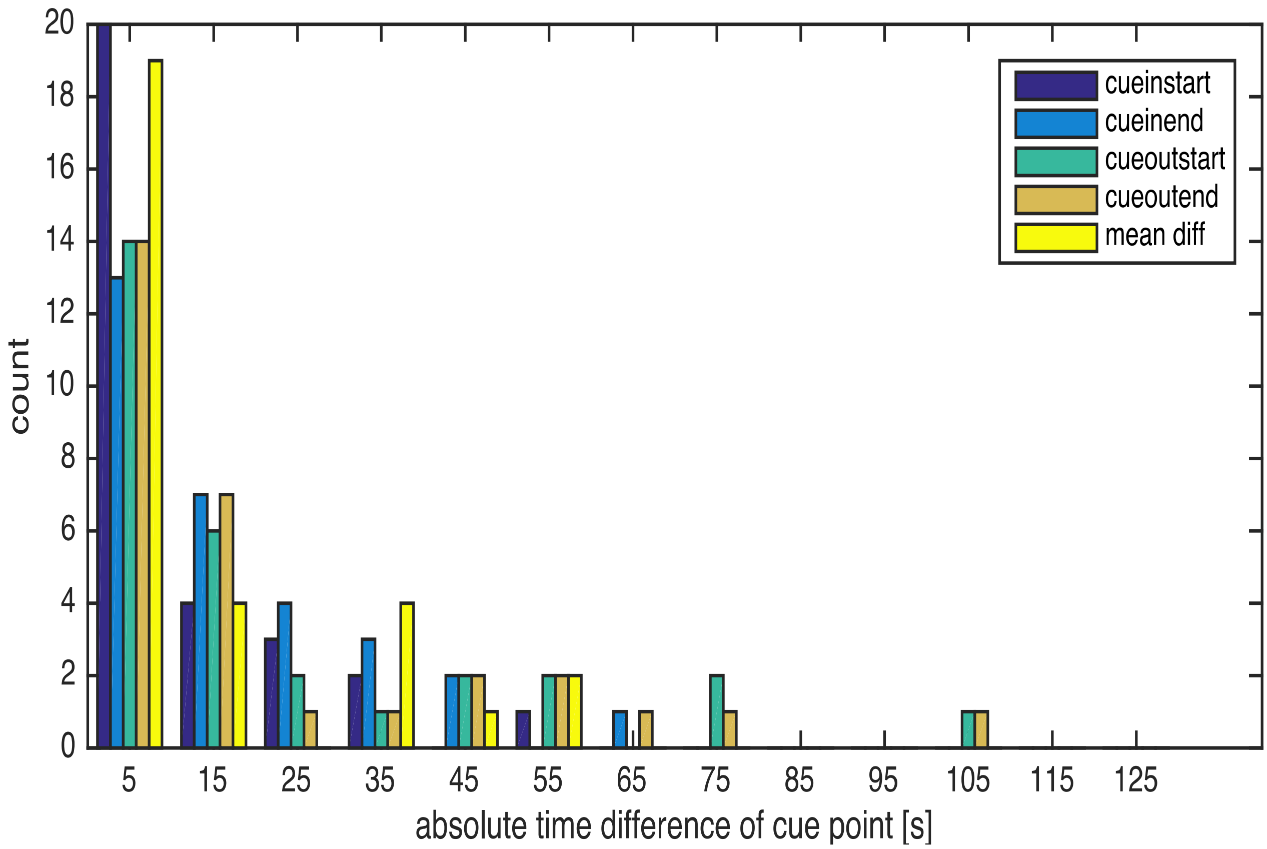
[fig:rms] Examination of the RMS values of the ground truth example tracks.

We then shift the cue-region until its minimum loudness is larger than -9 dB relative to the max loudness of the track. Loudness of a segment is calculated as the max peak RMS energy in 2s windows.

The second evaluation results, shown in [fig:cuehist2], slightly improve the difference between estimated cue-regions and manual choice, and has been approved by the human experts. The numeric evaluation also confirms our initial decision to choose *state* mode over *NSMF* mode: The found cue points are also shown in [fig:structure-state] and [fig:structure-nsmf].

Musically, the decision to place the cue-in region at the end of a song-structure region work very well, since at the end of the fade in of the new track, just when it has reached full volume, and the previous song has just vanished, there is a clear change in content (as predicted by the song structure), that catches the ear and clearly signals the start of the track.

[3] Note that our algorithm will only ever encounter professionally produced music that is optimised for being loud and punchy to stand out in radio or streaming listening conditions, so we're fairly confident that that threshold will be generalisable.



[fig:cuehist2] Histogram of the time difference between hand-annotated ground-truth cue points and estimated cue points with loudness criterion.

5.4.7 Implementation and Data Format for Cue Points

After a final validation by the music experts at *HearDis!*, the heuristic algorithm has been ported to C++ and released as part of imdABCDJ v2.1.0.
We defined an XML representation for the estimated cue-in/-out markers, to be integrated into the next version of the auto-tagging modules, with XML output as segment type (Task 4.4). Here is an example extract of a *musicdescription* XML file, defining cue-regions as segment descriptor number 15 (these definitions have been formalised in the *musicdescription* XML schema version 1.4.1):

```
<descriptiondefinition id="15">
    <type>cuetype</type>
    <generator name="imdABCDJ" version="1.2.1" date="2017-11-16" />
    <dictionary>
        <label name="cue-in" />
        <label name="cue-out" />
    </dictionary>
</descriptiondefinition>

<segment time="10.0000000000" length="7.2500000000" sourcetrack="0">
    <cuetype id="15" value="cue-in" type=“intro”/>
</segment>
<segment time="180.0000000000" length="10.25" sourcetrack="0">
    <cuetype id="15" value="cue-out"  type=“shorten”/>
</segment>
```

5.4.8 Conclusions of Cue Point Estimation

We presented a heuristic algorithm to estimate cue points for generating DJ-like mixes based on automatic annotations by state of the art MIR methods of music structure segmentation, coupled with domain knowledge of human experts, and backed by a database of example tracks. The iterative design process created a close feedback loop between researchers, developers, and expert users, quickly reaching a satisfactory solution for their specific needs of in-store music playout and audio branding.

In future work, we could examine which of the criteria in the human expert rules are possibly detectable by content descriptors and classifiers available in MIR research, e.g. voice detection, or develop specific descriptors and classifiers for the "music-ness" of audio. However, this would need many more annotated tracks to train the method. Before this effort is made, feedback should be gathered about the number of problematic cases in real-world usage of the existing system.

5.4.9 References

[Kaiser2012] F. Kaiser, "Music structure segmentation," Ph.D. dissertation, TU Berlin, 2012.

[Kaiser2013] F. Kaiser and G. Peeters, "A simple fusion method of state and sequence segmentation for music structure discovery," in ISMIR (International Society for Music Information Retrieval), Curitiba, Brazil, 2013, [Online]. Available: https://hal.archives-ouvertes.fr/hal-01106873.

[Schwarz+2018] Diemo Schwarz, Daniel Artur Schindler, Severino Spadavecchia. A Heuristic Algorithm for DJ Cue Point Estimation. Sound and Music Computing Conference, July 2018, Limassol, Cyprus.

6 Summary and Conclusion

Within Task 4.2, IRCAM is responsible for the estimation of a wide extent of features: from low-level audio features (such as derived from the TimbreToolbox, ircamdescriptor or newly developed ones based on HPSS), musical attributes (such as bpm), musicological attributes (such as the number of II-V-I within the estimated chords) to machine-learning based attributes (the "soft" features).

The goal of IRCAM has been two-fold:

- As technology provider for the other partners of the project, we need to guarantee that features are computed and this whatever the way they are estimated. This is done by first developing base-line technologies for all features.
- As research institution, we compare existing strategies and develop new ones to estimate these features with new innovative methods. For example, within ABC_DJ we tested for the same features (singing voice detection) several strategies: classical machine-learning method (improved by developing new audio features, data augmentation and data segmentation), deep-learning method (using Convolutional Neural Network) and recent source separation algorithms (using Blind Audio Source Separation algorithms such as Common Fate, KAM or RPCA method).