TOWARDS QUERY BY SINGING/HUMMING ON AUDIO DATABASES

Alexander Duda, Andreas Nürnberger and Sebastian Stober
Faculty of Computer Science
Otto-von-Guericke-University Magdeburg, Germany
{nuernb,stober}@iws.cs.uni-magdeburg.de

ABSTRACT

Current work on Query-by-Singing/Humming (QBSH) focusses mainly on databases that contain MIDI files. Here, we present an approach that works on real audio recordings that bring up additional challenges. To tackle the problem of extracting the melody of the lead vocals from recordings, we introduce a method inspired by the popular "karaoke effect" exploiting information about the spatial arrangement of voices and instruments in the stereo mix. The extracted signal time series are aggregated into symbolic strings preserving the local approximated values of a feature and revealing higher-level context patterns. This allows distance measures for string pattern matching to be applied in the matching process. A series of experiments are conducted to assess the discrimination and robustness of this representation. They show that the proposed approach provides a viable baseline for further development and point out several possibilities for improvement.

1 INTRODUCTION

Query-by-Singing/Humming (QBSH) as introduced in [6] is a popular content-based music retrieval method where the user enters a search query by singing, humming, or whistling it into a microphone. So far, work on QBSH systems has mainly focused on databases containing pieces of music in a symbolic description, usually MIDI. We describe an approach for QBSH on audio recordings instead. From these recordings, descriptive features are extracted and aggregated in symbolic strings which allow using distance measures for string pattern matching. However, music in general consists of several instruments or voices playing harmonically or in opposition to each other at the same time. In MIDI, each instrument usually has its own track, allowing straightforward separation of the individual voices. In real audio recordings, however, audio information of all instruments and voices is mixed and stored in all channels. Nevertheless, users of a QBSH system usually want to query the melody sung by the lead voice or played by a solo instrument. Consequently, the recordings need to be reduced to a somewhat more precise representation of components related to melody or lyrics.

Before we introduce our methodology in Section 3, we briefly discuss related work. Section 4 describes the experiments conducted to evaluate our approach. The results of the experiments are presented in Section 5.

2 RELATED WORK

In [17] a retrieval method for audio is proposed that restricts the frequency range to 22 semitones. Furthermore, the songs need to be manually segmented into semantically meaningful phrases. The authors argue that usually a query starts at the beginning of such a phrase and thus confine query comparison to the beginning of phrases without further local alignment. We make a similar assumption to improve performance but matching is not restricted to that case. In [14] an automated way for segmentation is presented using a normal CD recording as well as a karaoke track of the same song, which is usually not available and thus, we avoid using such additional information. Instead, we try to infer higher-level representations directly.

Current techniques of melody extraction from polyphonic recordings as [3, 5, 13] are still vulnerable to interferences from instruments. Here, source-separation approaches such as [4, 15] could help but still have many limitations: The method proposed in [15] works well on all single-note harmonic instruments including voice but has problems when drums are present or multiple instruments play the same note or an octave. The approach presented in [4] only works for vocals and up to three voices. In contrast to these approaches, we do not aim to achieve a perfect separation into individual voices. Instead, we are mainly interested in characteristics that are reproducable by a human singer. In this work we focus on the lead vocals or solo instruments.

3 METHODOLOGY

3.1 Voice Separation

We apply a two-step filtering to reduce the impact of backing instruments and voices in the audio recordings. First, a *band-path-filter* from 300 Hz to 3000 Hz is used to remove some instrument components, while keeping most of the lead voice. The second step exploits the spatial arrangement of instruments and voices in the stereo mix and could be described as *inverse karaoke* effect. It is inspired by the *center pan removal* technique used by most karaoke machines to remove the lead voice from a typical rock/pop

song: One stereo channel is inverted and mixed with the other one into a mono signal. The lead voice and solo instruments are usually centered in the stereo mix whereas most instruments and backing vocals are out of center. Applying the above mentioned transformation drastically reduces the power level of the centered signals and thus can be used to remove the lead voice.

Now the idea is to invert this effect, so that the pre-processed version of a song yields a high portion of the lead voice, while most other instruments are filtered out. Unfortunately, there is no simple way to keep the center pan. Inverting the karaoke result and mixing it with a mono version of the original will not work. The Audacity audio editor [1] that we used for the pre-processing provides a function called *noise profile*. It derives a power spectrum of frequencies from a noise track defining a noise signal which can then be removed from any other track. This filter works well for removing monotonic noise, e.g. white noise or growling. However, defining all the background as noise, the noise profile becomes rather imprecise resulting in removal of foreground parts which yields warbling artefacts. To reduce this effect, a *local noise profile* is defined as the noise profile of a narrow 2 s time window, which is moved along the track with 1 s overlap.

3.2 Feature Selection and Extraction

In this work, we concentrate on the following small set of manually selected features (as well as their 1^{st} and 2^{nd} derivatives) that seem to be promising in terms of discrimination and robustness: *audio power (AP)* [8], *audio fundamental frequency (AFF)* [8], *chroma* [12], *mel frequency cepstral coefficients (MFCC)* [11], and *formant frequencies (FF)* [2]. For these features we empirically determined a frame size and hopsize (i.e the span between the starting times of two succeeding frames) of 30 ms in preliminary experiments as optimal for the extraction. This time span roughly corresponds to the length of a $\frac{1}{32}$ note assuming a common tempo of 70–120 beats-per-minute. It should be the finest resolution for a melody track and is a natural factor of the more likely note lengths, $\frac{1}{4}$, $\frac{1}{8}$ and $\frac{1}{16}$, allowing a natural aggregation factor in the following aggregation step and thus avoiding interpolation.

3.3 Aggregation and Discretization

The extracted features are converted into a symbolic representation by reducing the number of possible values to reveal identifiable and repeating patterns. We extend the *symbolic aggregate approximation (SAX)* approach [10] that uses *piecewise aggregate approximation (PAA)* [16] for decomposing the time series into fixed length intervals. [2] The aggregated vector $\overline{C}$ of length w for a time series C of length n is computed as [10]

$$\overline{c}_i = \frac{w}{n} \sum_{j=\frac{n}{w}(i-1)+1}^{\frac{n}{w}i} c_j \quad , 0 < i \leq n \tag{1}$$

and normalized with respect to mean and standard deviation. SAX uses a lookup table for discretization that contains a symbol for each quantile of the value distribution of the feature to guaranty an equally distributed alphabet. This lookup table is highly dependent on the distributions type of the feature to discretize. The values of audio power, MFCC, fundamental and formant frequencies are exponentially distributed whereas the values of chroma bins are lognormal distributed. As the original SAX only allowed Gaussian distributions, we added further lookup tables and the distribution type as additional parameter for the discretization process. Further, in SAX the (edit) distance between symbols is defined as the absolute difference between the centroids of the area they represent, approximating the *absolute* distance between the values they have aggregated. However, as a *relative*, quality based distance was desired in the context of this work, we defined consecutive symbols to be equidistant instead.

Following parameters have an impact in this step: the aggregation factor, the alphabet size and the distribution type. As mentioned in Section 3.2, the former is a natural number which results in a time span per symbol that corresponds roughly to the common note lengths. Though the PAA is capable of aggregating time series by rational factor numbers, doing so would imply some interpolation. The feature values are mapped to an alphabet, assuming there is an optimal alphabet size for each feature which has to be determined experimentally. The alphabet size affects the resolution of the representation. Increasing the alphabet size may veil patterns, whereas decreasing it reduces its discriminatory power. Finally, from the distribution type the respective quantiles can be derived to be used in for the symbol lookup table. Here, the global distribution of the feature values on all songs of the database or a local one referring to an indivual song can be taken into account. In the latter case, the symbols of the alphabet for a specific feature would correspond to (slightly) different value intervals depending on the particular song. This could be a disadvantage but on the other hand it ensures that the alphabet is optimally utilized for each song. Both options were tested in our experiments.

3.4 High-Level Patterns

For demonstration purposes, we also tried to derive high-level patterns, describing generic shapes of a time series. As there is no automatic way of identifying such patterns, we restrict ourselves to the audio power since some patterns can be defined intuitively. A Gaussian de-noising filter with length twelve is applied, while each value describes a frame of 30 ms. From observation, four types of patterns can be defined: *Flat patterns* refer to sections of silence or quiet background that have a low mean value and a low variance, or lie between two elevations. *Smooth elevations* have a mean signal above a certain threshold

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

[2] We restrict ourselves to fixed interval lengths as we do not expect an improvement by allowing a varying and adaptive interval length.

with only one peak, probably describing single syllables or primary features that last for some frames. *Toothy structures*, are elevations with a mean signal above a certain threshold and more than one peak, that probably occur, when two or more smooth elevations are very close to each other, so there is no flat section between them. All other sections are classified as *undefined or noisy regions* that can result from quiet singing or filtered out instruments. Each pattern is stored along with its length. The distance of two patterns is again determined by a lookup table, that has been manually defined, but could also be optimized by some machine learning technique for further improvement.

3.5 Distance Computation

To compute the distance of the query and a song, well established distance measures for string pattern matching were used: *edit distance (Levenshtein Distance)* [9], *continuous edit distance* [7] and *n-grams* [1]. However, some considerations had to be made in order to assure pitch invariance, which is necessary because users will most often not sing or hum with the right pitch. Here, the only affected features are the fundamental frequency and the chroma. For the former this is not a problem because the symbols of the transformed signal do not refer to specific pitches anymore but to frequency bands derived from a pitch distribution. For the latter the problem can be overcome by rotating the chroma bins during the distance computation which is analogous to a transposition.

4 EXPERIMENTS

From a private collection, stereo recordings of 200 well-known songs of pop, rock, beat and soul were selected to form the test database. [3] Six experiments have been conducted on that database to assess the proposed approach:

(1) Querying with 150 ***MIDI files***, each containing solely the melody of a single song from the test database.

(2) Querying with 150 ***humanized MIDI files***, obtained by altering tempo, pitch and pauses of the files from (1).

(3) Querying with 150 ***hummed queries*** – each for a different song – from a non-professional singer.

(4) Querying with 130 ***sung queries*** from 7 non-professional singers (multiple queries for several songs).

(5) Querying 10 songs with ***modified recording snippets*** as in (2) and an extra snippet from a ***live recording***.

(6) Querying with the ***sung queries and additinal information*** on whether the query refers to the beginning, the chorus or "anything else" of a song for boosting.

The queries were 10s long and transcribed as described in Section 3. For each test scenario and parameter combination two performance measures on all queries were computed. We define the *Mean of Accuracy (MoA)* as:

$$MoA = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{n - rank(t_i)}{n-1}\right) \quad (2)$$

[3] The extracted features can be made available. For a song list, see `http://irgroup.cs.uni-magdeburg.de/mir`

MIDI Queries	ground truth		humanized	
	MoA	MRR	MoA	MRR
simple features				
1st MFCC (MFCC1)	0.5965	0.0338	0.5678	0.0289
2nd-5th MFCC	0.6096	0.0735	0.5677	0.0252
Audio Power (AP)	**0.6262**	**0.0574**	0.5522	0.0495
Fundamental Freq.	0.6098	0.0494	0.5678	0.0289
1st Formant (FF1)	0.5398	0.0344	--	--
Chroma	**0.6328**	**0.1242**	**0.6380**	**0.0618**
1st derivatives				
dAP	**0.6237**	**0.0924**	**0.6189**	**0.0872**
dChroma	0.5490	0.0320	--	--
high-level patterns				
HLP(AP)	0.5390	0.0350	--	--
feature combinations				
(AP, dAP)	0.6708	0.0764	0.6085	**0.0826**
(AP, dAP, Chroma)	**0.6979**	**0.1426**	**0.6439**	0.0820

Human Queries	hummed		sung	
	MoA	MRR	MoA	MRR
simple features				
1st MFCC (MFCC1)	0.5867	0.0393	**0.7325**	**0.1950**
2nd-5th MFCC	0.5361	0.0376	0.6325	0.1307
Audio Power (AP)	**0.6127**	**0.0549**	0.6794	0.1558
Fundamental Freq.	0.5113	0.0362	0.5586	0.0585
1st Formant (FF1)	0.5696	0.0251	0.6351	0.0821
Chroma	0.6095	0.0312	0.5879	0.1288
1st derivatives				
dMFCC1	0.5574	0.0641	0.6062	0.1032
high-level patterns				
HLP(AP)	**0.5970**	**0.0588**	**0.6925**	**0.1454**
feature combinations				
(MFCC1, dMFCC1)	0.5796	0.0456	0.7552	0.2164
(MFCC1, FF1)	--	--	0.7447	**0.2457**
(MFCC1, dMFCC1, FF1)	--	--	**0.7635**	0.2329

Table 1. Selected results for experiments (1)–(4). Remarkable values are highlighted. In case of missing values, the feature or combination was not further examined because no improvement was expected.

It depicts the average rank at which the target was found for each query with a value of 0.5 describing random, 0.57 to 0.67 mediocre, and above 0.67 good accuracy. The *Mean Reciprocal Rank (MRR)* as used in the MIREX 2006 QBSH known item retrieval task [4] is defined as:

$$MRR = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{rank(t_i)}\right) \quad (3)$$

and gives a hint of how often the target reaches one of the first ranks. It is highly depending on the size of the database. A good MRR on 200 songs would be above 0.2.

5 RESULTS

Generally, the continuous edit distance, an aggregation level of 4 and an alphabet size of 12 showed the best results, except for chroma, with only 3 symbols per bin. Using a local feature distribution for each song to compute the symbol lookup table yielded better results than using a global one. For the best parameter combination, Table 1 shows the average performance values for tests (1)–(4).

MIDI queries: Audio power and chroma were adequate working features for these tests. Especially, chroma compensated the difficulties introduced in test (2). A combination with the 1^{st} derivative of the audio power led to a remarkable performance gain, probably because of added

[4] `http://www.music-ir.org/mirex2006/`

context information. However, the higher-level patterns adding even more context failed but since they are designed for real recordings, this result is no setback. Similarly, the formant frequencies performed as poorly as expected, for they are bound to vowels and consonants.

Hummed and sung queries: The 1^{st} MFCC proved to be the best standalone feature and performed for sung queries even better than the audio power. This is not surprising since singers try to reproduce what they have heard. Combining it with its derivate and the 1^{st} formant yielded the best results. The remaining MFCCs are not so discriminative as they describe the shape of the spectrum which contains more artefacts from instruments. Likewise, only the 1^{st} formant frequency yields acceptable results. Especially remarkable is the performance of the higher-level patterns that is equal or slightly better than the simple audio power feature and can still be improved by refined patterns (see also 3.4). On the other hand, the fundamental frequency was only slightly better than random, possibly because of extraction errors.

Control Condition: Transposition and levelling (raising the audio power for quieter while keeping the same level for louder sections) had no impact. For these and unmodified snippets the target song was always ranked highest. Tempo changes by up to 10% had only a negligible impact of less than 1%. Only shortening sections of silence to 200 ms and 100 ms caused a drop of the MoA to 0.89 and 0.76 respectively. For the live version the MoA was 0.79. Thus, the proposed approach shows robustness in case of reasonable distortions of the query.

Using Additional Information: Boosting beginnings by 10% and each possible chorus by 5% increased the MoA to 0.79 and the MRR to 0.3. In 23.3% of all queries the target song was ranked first, for 30% it was amongst the top 3 and for 41.1% in the top 10. This could be significantly improved by using more sophisticated methods, e.g. for segmentation or chorus detection.

6 CONCLUSION

In this paper we presented an approach for QBSH on real audio recordings that showed some promising results in first experiments and still leaves much room for improvements such as filtering out drum beats from the center pan – as tracks with a high beat portion tended to be harder to find – or using more robust and elaborate melody extraction algorithms [3]. We demonstrated that higher-level patterns perform at least equally well in comparison to simple feature values. A method for automatically learning the characteristic higher-level patterns from a set of time series for a feature would allow to further apply and investigate higher-level patterns that might not only improve retrieval performance but also significantly speed up processing as the signal information is even further compressed. This would not only be interesting for QBSH, but for the whole pattern matching domain. This way, QBSH on real audio data has a high chance to become a standard application in many multimedia retrieval scenarios.

References

[1] J.-M. Batke, G. Eisenberg, P. Weishaupt, and T. Sikora. Evaluation of distance measures for MPEG-7 melody contours. In *Int. Workshop on Multimedia Signal Processing*, 2004.

[2] P. R. Cook. *Real Sound Synthesis for Interactive Applications*. A. K. Peters, Ltd., 2002.

[3] K. Dressler. Sinusoidal extraction using an efficient implementation of a multi-resolution FFT. In *Proc. of the Int. Conf. on Digital Audio Effects (DAFx'06)*, 2006.

[4] S. Dubnov, J. Tabrikian, and M. Arnon-Targan. Speech source separation in convolutive environments using space-time-frequency analysis. *EURASIP J. on Applied Signal Processing*, 2006.

[5] D. Gerhard. Pitch track target deviation in natural singing. In *Proc. of ISMIR'05*, 2005.

[6] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith. Query by humming: Musical information retrieval in an audio database. In *ACM Multimedia*, 1995.

[7] N. Jacobs, F. V. den Borre, L. Smeets, E. Schoofs, and H. Blockeel. A symbolic approach to music recognition, 2003.

[8] H.-G. Kim, N. Moreau, and T. Sikora. *MPEG-7 Audio and Beyond: Audio Content Indexing and Retrieval*. John Wiley & Sons, 2005.

[9] V. I. Levenshtein. Binary codes capable of correcting spurious insertions and deletions of ones. *Problems of Information Transmission*, 1:8–17, 1965.

[10] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proc. of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery (DMKD'03)*, 2003.

[11] B. Logan. Mel frequency cepstral coefficients for music modeling. In *Proc. of ISMIR'00*, 2000.

[12] M. Müller, F. Kurth, and M. Clausen. Audio matching via chroma-based statistical features. In *Proc. of ISMIR'05*, 2005.

[13] R. P. Paiva. On the detection of melody notes in polyphonic audio. In *Proc. of ISMIR'05*, 2005.

[14] W.-H. Tsai, H.-M. Yu, and H.-M. Wang. Query-by-example technique for retrieving cover versions of popular songs with similar melodies. In *Proc. of ISMIR'05*, 2005.

[15] J. Woodruff and B. Pardo. Using pitch, amplitude modulation, and spatial cues for separation of harmonic instruments from stereo music recordings. *EURASIP J. on Adv. in Signal Processing*, 2007.

[16] B.-K. Yi and C. Faloutsos. Fast time sequence indexing for arbitrary lp norms. In *Proc. of the 26th Int. Conf. on Very Large Data Bases (VLDB'00)*, 2000.

[17] H.-M. Yu, W.-H. Tsai, and H.-M. Wang. A query-by-singing technique for retrieving polyphonic objects of popular music. In *Proc. of the 2nd Asia Information Retrieval Symposium (AIRS'05)*, 2005.