Oral Session 4

Retrieval
SPOTTING A QUERY PHRASE FROM POLYPHONIC MUSIC AUDIO SIGNALS BASED ON SEMI-SUPERVISED NONNEGATIVE MATRIX FACTORIZATION

Taro Masuda\textsuperscript{1}  Kazuyoshi Yoshii\textsuperscript{2}  Masataka Goto\textsuperscript{3}  Shigeo Morishima\textsuperscript{1}

\textsuperscript{1}Waseda University  \textsuperscript{2}Kyoto University  \textsuperscript{3}National Institute of Advanced Industrial Science and Technology (AIST)

masutaro@suou.waseda.jp  yoshii@i.kyoto-u.ac.jp  m.goto@aist.go.jp  shigeo@waseda.jp

ABSTRACT

This paper proposes a query-by-audio system that aims to detect temporal locations where a musical phrase given as a query is played in musical pieces. The “phrase” in this paper means a short audio excerpt that is not limited to a main melody (singing part) and is usually played by a single musical instrument. A main problem of this task is that the query is often buried in mixture signals consisting of various instruments. To solve this problem, we propose a method that can appropriately calculate the distance between a query and partial components of a musical piece. More specifically, gamma process nonnegative matrix factorization (GaP-NMF) is used for decomposing the spectrogram of the query into an appropriate number of basis spectra and their activation patterns. Semi-supervised GaP-NMF is then used for estimating activation patterns of the learned basis spectra in the musical piece by presuming the piece to partially consist of those spectra. This enables distance calculation based on activation patterns. The experimental results showed that our method outperformed conventional matching methods.

1. INTRODUCTION

Over a decade, a lot of effort has been devoted to developing music information retrieval (MIR) systems that aim to find musical pieces of interest by using audio signals as the query. For example, there are many similarity-based retrieval systems that can find musical pieces having similar acoustic features to those of the query\cite{5,13,21,22}. Audio fingerprinting systems, on the other hand, try to find a musical piece that exactly matches the query by using acoustic features robust to audio-format conversion and noise contamination\cite{6,12,27}. Query-by-humming (QBH) systems try to find a musical piece that includes the melody specified by users’ singing or humming\cite{19}. Note that in general information of musical scores\cite{9,16,23,31,39} (such as MIDI files) or some speech corpus\cite{36} should be prepared for a music database in advance of QBH. To overcome this limitation, some studies tried to automatically extract main melodies from music audio signals included in a database\cite{25,34,35}. Other studies employ chroma vectors to characterize a query and targeted pieces without the need of symbolic representation or transcription\cite{2}.

We propose a task that aims to detect temporal locations at which phrases similar to the query phrase appear in different polyphonic musical pieces. The term “phrase” means a several-second musical performance (audio clip) usually played by a single musical instrument. Unlike QBH, our method needs no musical scores beforehand. A key feature of our method is that we aim to find short segments within musical pieces, not musical pieces themselves. There are several possible application scenarios in which both non-experts and music professionals enjoy the benefits of our system. For example, ordinary users could intuitively find a musical piece by playing just a characteristic phrase used in the piece even if the title of the piece is unknown or forgotten. In addition, composers could learn what kinds of arrangements are used in existing musical pieces that include a phrase specified as a query.

The major problem of our task lies in distance calculation between a query and short segments of a musical piece. One approach would be to calculate the symbolic distance between musical scores. However, this approach is impractical because even the state-of-the-art methods of...
automatic music transcription [4,11,17,29,38] work poorly for standard popular music. Conventional distance calculation based on acoustic features [5] is also inappropriate because acoustic features of a phrase are drastically distorted if other sounds are superimposed in a musical piece. In addition, since it would be more useful to find locations in which the same phrase is played by different instruments, we cannot heavily rely on acoustic features.

In this paper we propose a novel method that can perform phrase spotting by calculating the distance between a query and partial components of a musical piece. Our conjecture is that we could judge whether a phrase is included or not in a musical piece without perfect transcription, like the human ear can. More specifically, gamma process nonnegative matrix factorization (GaP-NMF) [14] is used for decomposing the spectrogram of a query into an appropriate number of basis spectra and their activation patterns. Semi-supervised GaP-NMF is then used for estimating activation patterns of the fixed basis spectra in a target musical piece by presuming the piece to partially consist of those spectra. This enables appropriate matching based on activation patterns of the basis spectra forming the query.

2. PHRASE SPOTTING METHOD

This section describes the proposed phrase-spotting method based on nonparametric Bayesian NMF.

2.1 Overview

Our goal is to detect the start times of a phrase in the polyphonic audio signal of a musical piece. An overview of the proposed method is shown in Figure 1. Let $X \in \mathbb{R}^{M \times N_x}$ and $Y \in \mathbb{R}^{M \times N_y}$ be the nonnegative power spectrogram of a query and that of a target musical piece, respectively. Our method consists of three steps. First, we perform NMF for decomposing the query $X$ into a set of basis spectra $W^{(x)}$ and a set of their corresponding activations $H^{(x)}$. Second, in order to obtain temporal activations of $W^{(x)}$ in the musical piece $Y$, we perform another NMF whose basis spectra consist of a set of fixed basis spectra $W^{(x)}$ and a set of unconstrained basis spectra $W^{(f)}$ that are required for representing musical instrument sounds except for the phrase. Let $H^{(y)}$ and $H^{(f)}$ be sets of activations of $Y$ corresponding to $W^{(x)}$ and $W^{(f)}$, respectively. Third, the similarity between the activation patterns $H^{(x)}$ in the query and the activation patterns $H^{(y)}$ in the musical piece is calculated. Finally, we detect locations of a phrase where the similarity takes large values.

There are two important reasons that “nonparametric” “Bayesian” NMF is needed. 1) It is better to automatically determine the optimal number of basis spectra according to the complexity of the query $X$ and that of the musical piece $Y$. 2) We need to put different prior distributions on $H^{(y)}$ and $H^{(f)}$ to put more emphasis on fixed basis spectra $W^{(x)}$ than unconstrained basis spectra $W^{(f)}$. If no priors are placed, the musical piece $Y$ is often represented by using only unconstrained basis spectra $W^{(f)}$. A key feature of our method is that we presume that the phrase is included in the musical piece when decomposing $Y$. This means that we need to make use of $W^{(x)}$ as much as possible for representing $Y$. The Bayesian framework is a natural choice for reflecting such a prior belief.

2.2 NMF for Decomposing a Query

We use the gamma process NMF (GaP-NMF) [14] for approximating $X$ as the product of a nonnegative vector $\theta \in \mathbb{R}^{K_x}$ and two nonnegative matrices $W^{(x)} \in \mathbb{R}^{M \times K_x}$ and $H^{(x)} \in \mathbb{R}^{K_x \times N_x}$. More specifically, the original matrix $X$ is factorized as follows:

$$X_{mn} \approx \sum_{k=1}^{K_x} \theta_k W_{mk}^{(x)} H_{kn}^{(x)}.$$

where $\theta_k$ is the overall gain of basis $k$, $W_{mk}^{(x)}$ is the power of basis $k$ at frequency $m$, and $H_{kn}^{(x)}$ is the activation of basis $k$ at time $n$. Each column of $W^{(x)}$ represents a basis spectrum and each row of $H^{(x)}$ represents an activation pattern of the basis over time.

2.3 Semi-supervised NMF for Decomposing a Musical Piece

We then perform semi-supervised NMF for decomposing the spectrogram of the musical piece $Y$ by fixing a part of basis spectra with $W^{(x)}$. The idea of giving $W$ as a dictionary during inference has been widely adopted [3, 7, 15, 18, 24, 26, 28, 30, 33, 38].

We formulate Bayesian NMF for representing the spectrogram of the musical piece $Y$ by extensively using the fixed bases $W^{(x)}$. To do this, we put different gamma priors on $H^{(y)}$ and $H^{(f)}$. The shape parameter of the gamma prior on $H^{(y)}$ is much larger than that of the gamma prior on $H^{(f)}$. Note that the expectation of the gamma distribution is proportional to its shape parameter.

2.4 Correlation Calculation between Activation Patterns

After the semi-supervised NMF is performed, we calculate the similarity between the activation patterns $H^{(x)}$ in the query and the activation patterns $H^{(y)}$ in a musical piece to find locations of the phrase. We expect that similar patterns appear in $H^{(y)}$ when almost the same phrases are played in the musical piece even if those phrases are played by different instruments. More specifically, we calculate the sum of the correlation coefficients $r$ at time $n$ between $H^{(x)}$ and $H^{(y)}$ as follows:

$$r(n) = \frac{1}{K_x N_x} \sum_{k=1}^{K_x} \frac{\sum_{k=1}^{K_x} \langle h_{k1}^{(y)} - \overline{h_{k1}^{(y)}} \rangle^T \langle h_{kn}^{(y)} - \overline{h_{kn}^{(y)}} \rangle}{\| h_{k1}^{(x)} - \overline{h_{k1}^{(x)}} \|_2 \| h_{kn}^{(y)} - \overline{h_{kn}^{(y)}} \|_2},$$

where

$$K_{ki}^{(x)} = \left[ H_{ki}^{(x)}, \ldots, H_{k(i+N_x-1)}^{(x)} \right]^T,$$

$$K_{kn}^{(x)} = \frac{1}{N_x} \sum_{j=1}^{N_x} H_{k(n+j-1)}^{(x)} \times [1 \cdots 1]^T.$$
Finally, we detect a start frame $n$ of the phrase by finding peaks of the correlation coefficients over time. This peak picking is performed based on the following thresholding process:

$$r(n) > \mu + 4\sigma,$$  (5)

where $\mu$ and $\sigma$ denote the overall mean and standard deviation of $r(n)$, respectively, which were derived from all the musical pieces.

2.5 Variational Inference of GaP-NMF

This section briefly explains how to infer nonparametric Bayesian NMF [14], given a spectrogram $V \in \mathbb{R}^{M \times N}$. We assume that $\theta \in \mathbb{R}^K$, $W \in \mathbb{R}^{M \times K}$, and $H \in \mathbb{R}^{K \times N}$ are stochastically sampled according to a generative process. We choose a gamma distribution as a prior distribution as follows:

$$
P(W_{mk}) = \text{Gamma}\left(a^{(W)}_{mk}, b^{(W)}_{mk}\right),$$

$$
P(H_{kn}) = \text{Gamma}\left(a^{(H)}_{kn}, b^{(H)}_{kn}\right),$$

$$
P(\theta_k) = \text{Gamma}\left(\frac{\alpha}{K}, \alpha/c\right),$$

where $\alpha$ is a concentration parameter, $K$ is a sufficiently large integer (ideally an infinite number) compared with the number of components in the mixed sound, and $c$ is the inverse of the mean value of $V$:

$$c = \left(\frac{1}{MN} \sum_{m} \sum_{n} V_{mn}\right)^{-1}.$$  (7)

We then use the generalized inverse-Gaussian (GIG) distribution as a posterior distribution as follows:

$$
q(W_{mk}) = \text{GIG}\left(\gamma^{(W)}_{mk}, \rho^{(W)}_{mk}, \tau^{(W)}_{mk}\right),
$$

$$
q(H_{kn}) = \text{GIG}\left(\gamma^{(H)}_{kn}, \rho^{(H)}_{kn}, \tau^{(H)}_{kn}\right),$$

$$
q(\theta_k) = \text{GIG}\left(\gamma^{(\theta)}_{k}, \rho^{(\theta)}_{k}, \tau^{(\theta)}_{k}\right).$$

To estimate the parameters of these distributions, we first update other parameters, $\phi_{kmn}, \omega_{mn}$, using the following equations.

$$
\phi_{kmn} = \mathbb{E}_q\left[\frac{1}{\theta_k W_{mk} H_{kn}}\right]^{-1},$$  (9)

$$
\omega_{mn} = \sum_k \mathbb{E}_q [\theta_k W_{mk} H_{kn}].$$  (10)

After obtaining $\phi_{kmn}$ and $\omega_{mn}$, we update the parameters of the GIG distributions as follows:

$$
\gamma^{(W)}_{mk} = a^{(W)}_{mk}, \quad \rho^{(W)}_{mk} = b^{(W)} + \mathbb{E}_q[\theta_k] \sum_n \frac{E_q[H_{kn}]}{\omega_{mn}},$$

$$
\tau^{(W)}_{mk} = \mathbb{E}_q \left[\frac{1}{\theta_k}\sum_n V_{mn} \phi^2_{kmn} \mathbb{E}_q \left[\frac{1}{H_{kn}}\right]\right],$$  (11)

$$
\gamma^{(H)}_{kn} = a^{(H)}_{kn}, \quad \rho^{(H)}_{kn} = b^{(H)} + \mathbb{E}_q[\theta_k] \sum_m \frac{E_q[W_{mk}]}{\omega_{mn}},$$

$$
\tau^{(H)}_{kn} = \mathbb{E}_q \left[\frac{1}{\theta_k}\sum_n V_{mn} \phi^2_{kmn} \mathbb{E}_q \left[\frac{1}{W_{mk}}\right]\right],$$  (12)

$$
\gamma^{(\theta)}_{k} = \frac{\alpha}{K}, \quad \rho^{(\theta)}_{k} = \alpha/c + \sum_m \sum_n \frac{E_q[W_{mk} H_{kn}]}{\omega_{mn}},$$

$$
\tau^{(\theta)}_{k} = \sum_{m} \sum_{n} V_{mn} \phi^2_{kmn} \mathbb{E}_q \left[\frac{1}{W_{mk} H_{kn}}\right].$$  (13)

The expectations of $W$, $H$, and $\theta$ are required in Eqs. (9) and (10). We randomly initialize the expectations of $W$, $H$, and $\theta$ and iteratively update each parameter by using those formulas. As the number of iterations increases, the value of $E_q[\theta_k]$ over a certain level $K^+$ decreases. Therefore, if the value is 60 dB lower than $\sum_k E_q[\theta_k]$, we remove the related parameters from consideration, which makes the calculation faster. Eventually, the number of effective bases, $K^+$, gradually reduces during iterations, suggesting that the appropriate number is automatically determined.

3. CONVENTIONAL MATCHING METHODS

We describe three kinds of conventional matching methods used for evaluation. The first and the second methods calculate the Euclidean distance between acoustic features (Section 3.1) and that between chroma vectors (Section 3.2), respectively. The third method calculates the Itakura-Saito (IS) divergence between spectrograms (Section 3.3).

3.1 MFCC Matching Based on Euclidean Distance

Temporal locations in which a phrase appears are detected by focusing on the acoustic distance between the query and a short segment extracted from a musical piece. In this study we use Mel-frequency cepstrum coefficients (MFCCs) as an acoustic feature, which have commonly been used in various research fields [1, 5]. More specifically, we calculate a 12-dimensional feature vector from each frame by using the Auditory Toolbox Version 2 [32]. The distance between two sequences of the feature vector extracted from the query and the short segment is obtained by accumulating the frame-wise Euclidean distance over the length of the query.

The above-mentioned distance is iteratively calculated by shifting the query frame by frame. Using a simple peak-picking method, we detect locations of the phrase in which the obtained distance is lower than $m - s$, where $m$ and $s$ denote the mean and standard deviation of the distance over all frames, respectively.
3.2 Chromagram Matching Based on Euclidean Distance

In this section, temporal locations in which a phrase appears are detected by directly calculating the Itakura-Saito (IS) divergence [8,37] between the query X and the musical piece Y. The use of the IS divergence is theoretically justified because the IS divergence poses a smaller penalty than standard distance measures such as the Euclidean distance and the Kullback-Leibler (KL) divergence when the power spectrogram of the query is included in that of the musical piece.

To efficiently find phrase locations, we use a dynamic programming (DP) matching method based on the IS divergence. First, we make a distance matrix D ∈ R_{N_x×N_y} in which each cell D(i, j) is the IS divergence between the i-th frame of X and the j-th frame of Y (1 ≤ i ≤ N_x and 1 ≤ j ≤ N_y). D(i, j) is given by

\[ D(i, j) = D_{IS}(X_i|Y_j) = \sum_m \left( -\log \frac{X_{mi}}{Y_{mj}} + \frac{X_{mi}}{Y_{mj}} - 1 \right), \]

where m indicates a frequency-bin index. We then let E ∈ R_{N_x×N_y} be a cumulative distance matrix. First, E is initialized as E(1, j) = 0 for any j and E(i, 1) = ∞ for any i. E(i, j) can be sequentially calculated as follows:

\[ E(i, j) = \min \left\{ \begin{array}{l} 1) E(i-1, j-2) + 2D(i, j-1) \\
2) E(i-1, j-1) + D(i, j) \\
3) E(i-2, j-1) + 2D(i-1, j) \end{array} \right\} + D(i, j). \]

Finally, we can obtain E(N_x, j) that represents the distance between the query and a phrase ending at the j-th frame in the musical piece. We let C ∈ R_{N_x×N_y} be a cumulative cost matrix. According to the three cases 1), 2), and 3), C(i, j) is obtained as follows:

\[ C(i, j) = \left\{ \begin{array}{l} 1) C(i-1, j-2) + 3 \\
2) C(i-1, j-1) + 2 \\
3) C(i-2, j-1) + 3. \end{array} \right\} \]

This means that the length of a phrase is allowed to range from one half to two times of the query length.

Phrase locations are determined by finding the local minima of the regularized distance given by E_{IS}(N_x, j). More specifically, we detect locations in which values of the obtained distance are lower than M − S/10, where M and S denote the median and standard deviation of the distance over all frames, respectively. A reason that we use the median for thresholding is that the distance sometimes takes an extremely large value (outlier). The mean of the distance tends to be excessively biased by such an outlier. In addition, we ignore values of the distance which are more than 10^8 when calculating S for practical reasons (almost all values of E_{IS}(N_x, j) range from 10^4 to 10^8). Once the end point is detected, we can also obtain the start point of the phrase by simply tracing back along the path from the end point.

3.3 DP Matching Based on Itakura-Saito Divergence

In this section, temporal locations in which a phrase appears are detected by directly calculating the Itakura-Saito divergence IS between the query and the musical piece. The use of the IS divergence is theoretically justified because the IS divergence poses a smaller penalty than standard distance measures such as the Euclidean distance and the Kullback-Leibler (KL) divergence when the power spectrogram of the query is included in that of the musical piece.

3.4 EXPERIMENTS

This section reports comparative experiments that were conducted for evaluating the phrase-spotting performances of the proposed method described in Section 2 and the three conventional methods described in Section 3.

4. EXPERIMENTS

This section reports comparative experiments that were conducted for evaluating the phrase-spotting performances of the proposed method described in Section 2 and the three conventional methods described in Section 3.

4.1 Experimental Conditions

The proposed method and the three conventional methods were tested under three different conditions: 1) Exactly the same phrase specified as a query was included in a musical piece (exact match). 2) A query was played by a different kind of musical instruments (timbre change). 3) A query was played in a faster tempo (tempo change).

We chose four musical pieces (RWC-MDB-P-2001 No.1, 19, 42, and 77) from the RWC Music Database: Popular Music [10]. We then prepared 50 queries: 1) 10 were short segments excerpted from original multi-track recordings of the four pieces. 2) 30 queries were played by three kinds of musical instruments (nylon guitar, classic piano, and strings) that were different from those originally used in the four pieces. 3) The remaining 10 queries were played by the same instruments as original ones, but their tempi were 20% faster. Each query was a short performance played by a single instrument and had a duration ranging from 4 s to 9 s. Note that those phrases were not necessarily salient (not limited to main melodies) in musical pieces. We dealt with monaural audio signals sampled at 16 kHz and applied the wavelet transform by shifting short-time frames with an interval of 10 ms. The reason that we did not use short-time Fourier transform (STFT) was to attain a high resolution in a low frequency band. We determined the standard deviation of a Gabor wavelet function to 3.75 ms (60 samples). The frequency interval was 10 cents and the frequency ranged from 27.5 (A1) to 8000 (much higher than C8) Hz.

When a query was decomposed by NMF, the hyperparameters were set as α = 1, K = 100, a(W) = b(W) = a(H) = 0.1, and b(H) = c. When a musical piece was decomposed by semi-supervised NMF, the hyperparameters were set as a(W) = b(W) = 0.1, a(H) = 10, a(H) = 0.01, and b(H) = c. The inverse-scale parameter b(H) was adjusted to the empirical scale of the spectrogram of a target audio signal. Also note that using smaller values of α makes parameters sparser in an infinite space.

To evaluate the performance of each method, we calculated the average F-measure, which has widely been used in the field of information retrieval. The precision rate was defined as a proportion of the number of correctly-found phrases among all detected phrases.
<table>
<thead>
<tr>
<th>Method</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>F-measure (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFCC</td>
<td>24.8</td>
<td>35.0</td>
<td>29.0</td>
</tr>
<tr>
<td>Chroma</td>
<td>33.4</td>
<td>61.0</td>
<td>43.1</td>
</tr>
<tr>
<td>DP</td>
<td>1.9</td>
<td>55.0</td>
<td>3.6</td>
</tr>
<tr>
<td>Proposed</td>
<td>53.6</td>
<td>63.0</td>
<td>57.9</td>
</tr>
</tbody>
</table>

Table 1. Experimental results in a case that exactly the same phrase specified as a query was included in a musical piece.

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>F-measure (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFCC</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Chroma</td>
<td>18.1</td>
<td>31.7</td>
<td>23.0</td>
</tr>
<tr>
<td>DP</td>
<td>1.1</td>
<td>66.3</td>
<td>6.2</td>
</tr>
<tr>
<td>Proposed</td>
<td>26.9</td>
<td>56.7</td>
<td>36.5</td>
</tr>
</tbody>
</table>

Table 2. Experimental results in a case that a query was played by a different kind of instruments.

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>F-measure (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MFCC</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Chroma</td>
<td>12.0</td>
<td>19.0</td>
<td>14.7</td>
</tr>
<tr>
<td>DP</td>
<td>0.5</td>
<td>20.0</td>
<td>2.7</td>
</tr>
<tr>
<td>Proposed</td>
<td>15.8</td>
<td>45.0</td>
<td>23.4</td>
</tr>
</tbody>
</table>

Table 3. Experimental results in a case that the query phrases was played in a faster tempo.

4.2 Experimental Results

Tables 1–3 show the accuracies obtained by the four methods under each condition. We confirmed that our method performed much better than the conventional methods in terms of accuracy. Figure 2 shows the value of $r(n)$ obtained from a musical piece in which a query phrase (originally played by the saxophone) is included. We found that the points at which the query phrase starts were correctly spotted by using our method. Although the MFCC-based method could retrieve some of the query phrases in the exact-match condition, it was not robust to timbre change and tempo change. The DP matching method, on the other hand, could retrieve very few correct points because the IS divergence was more sensitive to volume change than the similarity based on spectrograms. Although local minima of the cost function often existed at correct points, those minima were not sufficiently clear because it was difficult to detect the end point of the query from the spectrogram of a mixture audio signal. The chroma-based method worked better than the other conventional methods. However, it did not outperform the proposed method since the chroma-based method often detected false locations including a similar chord progression.

Although our method worked best of the four, the accuracy of the proposed method should be improved for practical use. A major problem is that the precision rate was relatively lower than the recall rate. Wrong locations were detected when queries were played in staccato manner because many false peaks appeared at the onset of staccato notes.

As for computational cost, it took 29746 seconds to complete the retrieval of a single query by using our method. This was implemented in C++ on a 2.93 GHz Intel Xeon Windows 7 with 12 GB RAM.

5. CONCLUSION AND FUTURE WORK

This paper presented a novel query-by-audio method that can detect temporal locations where a phrase given as a query appears in musical pieces. Instead of pursuing perfect transcription of music audio signals, our method used nonnegative matrix factorization (NMF) for calculating the distance between the query and partial components of each musical piece. The experimental results showed that our method performed better than conventional matching methods. We found that our method has a potential to find correct locations in which a query phrase is played by different instruments (timbre change) or in a faster tempo (tempo change).

Future work includes improvement of our method, especially under the timbre-change and tempo-change conditions. One promising solution would be to classify basis spectra of a query into instrument-dependent bases (e.g.,
Acknowledgment: This study was supported in part by the JST OngaCREST project.

6. REFERENCES


BAYESIAN AUDIO ALIGNMENT BASED ON A UNIFIED GENERATIVE MODEL OF MUSIC COMPOSITION AND PERFORMANCE

Akira Maezawa1,2 Katsutoshi Itoyama2 Kazuyoshi Yoshii2 Hiroshi G. Okuno3
1Yamaha Corporation 2Kyoto University 3Waseda University
{amaezaw1,itoyama,yoshii}@kuis.kyoto-u.ac.jp, okuno@aoni.waseda.jp

ABSTRACT

This paper presents a new probabilistic model that can align multiple performances of a particular piece of music. Conventionally, dynamic time warping (DTW) and left-to-right hidden Markov models (HMMs) have often been used for audio-to-audio alignment based on a shallow acoustic similarity between performances. Those methods, however, cannot distinguish latent musical structures common to all performances and temporal dynamics unique to each performance. To solve this problem, our model explicitly represents two state sequences: a top-level sequence that determines the common structure inherent in the music itself and a bottom-level sequence that determines the actual temporal fluctuation of each performance. These two sequences are fused into a hierarchical Bayesian HMM and can be learned at the same time from the given performances. Since the top-level sequence assigns the same state for note combinations that repeatedly appear within a piece of music, we can unveil the latent structure of the piece. Moreover, we can easily compare different performances of the same piece by analyzing the bottom-level sequences. Experimental evaluation showed that our method outperformed the conventional methods.

1. INTRODUCTION

Multiple audio alignment is one of the most important tasks in the field of music information retrieval (MIR). A piece of music played by different people produces different expressive performances, each embedding the unique interpretation of the player. To help a listener better understand the variety of interpretation or discover a performance that matches his/her taste, it is effective to clarify how multiple performances differ by using visualization or playback interfaces [1–3]. Given multiple musical audio signals that play a same piece of music from the beginning to the end, our goal is to find a temporal mapping among different signals while considering the underlying music score.

This paper presents a statistical method of offline multiple audio alignment based on a probabilistic generative model that can integrate various sources of uncertainties in music, such as spectral shapes, temporal fluctuations and structural deviations. Our model expresses how a musical composition gets performed, so it must model how they are generated.1 Such a requirement leads to a conceptual model illustrated in Figure 1, described using a combination of two complementary models.

To represent the generative process of a musical composition, we focus on the general fact that small fragments consisting of multiple musical notes form the basic building blocks of music and are organized into a larger work. For example, the sonata form is based on developing two contrasting fragments known as the “subject groups,” and a song form essentially repeats the same melody. Our model is suitable for modeling the observation that basic melodic patterns are reused to form the sonata or the song.

To represent the generative process of each performance, we focus on temporal fluctuations from a common music composition. Since each performance plays the same musical composition, the small fragments should appear in the same order. On the other hand, each performance can be played by a different set of musical instruments with a unique tempo trajectory.

Since both generative processes are mutually dependent, we integrate a generative model of music composition with that of performance in a hierarchical Bayesian manner. In other words, we separate the characteristics of a given music audio signal into those originating from the underlying music score and those from the unique performance. Inspired by a typical preprocessing step in music structure segmentation [6, 7], we represent a music composition as a sequence generated from a compact, ergodic Markov model (“latent composition”). Each music performance is represented as a left-to-right Markov chain that traverses the latent composition with the state durations unique to each performance.2

1 A generative audio alignment model depends heavily on the model of both how the music is composed and how the composition is performed. This is unlike generative audio-to-score alignment [4, 5], which does not need a music composition model because a music score is already given.

2 Audio samples are available on the website of the first author.
2. RELATED WORK

Audio alignment is typically formulated as a problem of maximizing the similarity or minimizing the cost between a performance and another performance whose time-axis has been “stretched” by a time-dependent factor, using dynamic time warping (DTW) and its variants [8, 9] or other model of temporal dynamics [10]. To permit the use of a simple similarity measure, it is important to design robust acoustic features [11, 12].

Alternatively, tackling alignment by a probabilistic generative model has gathered attention, especially in the context of audio-to-music score alignment [4, 5]. In general, a probabilistic model is formulated to describe how each note in a music score translates to an audio signal. It is useful when one wishes to incorporate, in a unified framework, various sources of uncertainties present in music, such as inclusion of parts [13], mistakes [14], or timbral variations [15–17].

Previous studies in generative audio alignment [13, 18] ignores the organization present in musical composition, by assuming that a piece of music is generated from a left-to-right Markov chain, i.e., a Markov chain whose state appears in the same order for all performances.

3. FORMULATION

We formulate a generative model of alignment that aligns D performances. We provide a conceptual overview, and then mathematically formalize the concept.

3.1 Conceptual Overview

We first extract short-time audio features from each of D performances. Let us denote the feature sequence for the dth performance at frame t ∈ [1, T_d] as x_{d,t}, where T_d is the total number of frames for the dth audio signal. Here, the kind of feature is arbitrary, and depends on the generative model of the short-time audio. Then, we model x_{d,t} as a set of D state sequences. Each state is associated with a unique generative process of short-time audio feature. In other words, each state represents a distinct audio feature, e.g., distinct chord, f_0, and so on, depending on how the generative model of the feature is designed.

For audio alignment, the state sequence must abide by two rules. First, the order in which each state appears is the same for all D feature sequences. In other words, every performance is described by one sequence of distinct audio features, i.e., the musical piece that the performances play in common. We call such a sequence the latent composition. Second, the duration that each performance resides in a given state in the latent composition can be unique to the performance. In other words, each performance traverses the latent composition with a unique “tempo curve.” We call the sequence that each performance traverses over the latent composition sequence as the performance sequence.

The latent composition is a sequence of length N drawn from an ergodic Markov model, which we call the latent common structure. We describe the latent composition as z_n, a sequence of length N and S states, where each state describes a distinct audio feature. In other words, we assume that the musical piece is described by at most N distinct audio events, using at most S distinct sounds. The latent common structure encodes the structure inherent to the music. The transition probabilities of each state sheds light on a “typical” performance, e.g., melody line or harmonic progression. Therefore, the latent common structure provides a generative model of music composition.

The performance sequence provides a generative model of performance. Each audio signal is modeled as an emission from a N-state left-to-right Markov model, where the nth state refers to the generative model associated with the nth position in the latent composition. Specifically, let us denote the performance sequence for audio d as φ_{d,t}, which is a state sequence of length T_d and N states, such that state n refers to the nth element of the latent composition. Each performance sequence is constrained such that (1) it begins in state 1 and ends at state N, and (2) state n may traverse only to itself or state n+1. In other words, we
constrain each performance sequence to traverse the latent composition in the same order but with a unique duration. Such a model conveys the idea that each performance can independently play a piece in any tempo trajectory.

3.1.1 An Example

Let us illustrate our method in Figure 2. In the example, $S = 3$ and $N = 5$, where state “A” corresponds to a combination of notes G, C and F, “B” corresponds to the note C, and so on; moreover, $z_n$ encodes the state sequence “ABCB,” as to reflect the underlying common music composition that the performances play. Note that a single note may be expressed using more than one state in the latent composition, e.g., both $z_2$ and $z_3$ describe the note “C.” Next, each performance aligns to the latent composition, through the performance sequence. Each state of the performance sequence is associated to a position in the latent composition. For example, $\phi_{1,2}$ is associated to position 2 of $z$, $z_2$. Then, at each time, the observation is generated by emitting from the state in latent common structure referred by the current frame of the current audio. This is determined hierarchically by looking up the state $n$ of the performance sequence of audio $d$ at time $t$, and referring to the state $s$ of the $n$th element of the latent composition. In the example, $\phi_{1,2}$ refers to state $n = 2$, so the generative model corresponding to $z_{n=2}$, or “B,” is referred.

3.2 Formulation of the Generative Model

Let us mathematically formalize the above concept using a probabilistic generative model, summarized as a graphical model shown in Fig. 3.

3.2.1 Latent Composition and Common Structure

The latent composition is described as $z_{n=\{1\ldots N\}}$, a $S$-state state sequence of length $N$, generated from the latent common structure. We shall express the latent composition $z_n$ using one-of-$S$ representation; $z_n$ is a $S$-dimensional binary variable where, when the state of $z_n$ is $s$, $z_{n,s} = 1$ and all other elements are 0. Then, we model $z$ as a sequence from the latent common structure, an ergodic Markov chain with initial state probability $\pi$ and transition probability $\tau$:

$$p(z|\pi, \tau) = \prod_{s=1}^{S} \sum_{z_{n=2}}^{N,S} \prod_{n=2}^{S} \sum_{s,s'}^{z_{n-1},s',z_{n,s}} p(z_{n-1}, s', z_{n,s})$$

The latent composition and structure implicitly convey the information about how the music is structured and what its building blocks are. Figure 4 shows a similarity matrix derived from the estimated latent composition of Op. 41-2 by F. Chopin\(^3\) having the ternary form (a.k.a. ABA form). The first “A” section repeats a theme of form “DEDF” repeated twice. The second section is in a modulated key. Finally, the last section repeats the first theme, and ends with a short coda, borrowing from “F” motive from the first theme. Noting that the diagonal lines of a similarity matrix represent strong similarity, we may unveil such a trend by analyzing the matrix. The bottom-left diagonal lines in the first section, for example, shows a theme repeats, and the top-left diagonal suggests that the first theme is repeated at the end. This suggests that the latent composition reflects the organization of music.

Notice that this kind of structure arises because we explicitly model the organization of music, conveyed through an ergodic Markov model; simply aligning multiple performances to a single left-to-right HMM \([13, 18]\) is insufficient because it cannot revisit a previously visited state.

3.2.2 Performance Sequence

Recall that we require the performance sequence such that (1) it traverses in the order of latent composition, and (2) the duration that each performance stays in a particular state in the latent composition is conditionally independent given the latent composition. To satisfy these requirements, we model the performance sequence as a $N$-state left-to-right Markov chain of length $T_d$, $\phi_{d,t}$, where the first state of the chain is fixed to the beginning of the latent

\[^{3}\] The similarity matrix $R_{i,j}$ was determined by removing self-transitions from $z_n$ and assigning it to $z'_n$, and setting $R_{i,j} = 1$ if $z'_n = z'_j$, and 0 otherwise. Next, we convolved $R$ by a two-dimensional filter that emphasizes diagonal lines.
composition and the last state to be the end. This assumes that there are no repeats or unique updates to a performance. Let us define \( \eta_{d,n} \) to be the probability for performance \( d \) to traverse from position \( n \) of the latent composition to \( n + 1 \).

Then, we model the performance sequence as follows:

\[
p(\phi_{d,t} = (1 \cdots T_d)) = \delta(n, 1) \delta(n, S) \phi_{d,T_d, n} \\
\quad \times \prod_{t=1}^{T_d} \prod_{n=1}^{N} \left[ \phi_{d,t-1,n} \phi_{d,t,n+1} \right] (1 - \eta_{d,n}) \phi_{d,t-1,n} \phi_{d,t,n+1}
\]

where \( \delta(x, y) \) indicates the Kronecker Delta, i.e., its value is 1 when \( x = y \) and 0 otherwise. We assume \( \eta_{d,n} \) is drawn from a conjugate Beta distribution, i.e., \( \eta_{d,n} \sim \text{Beta}(a_0, b_0) \).

The ratio \( a_0/b_0 \) controls the likelihood of traversing to next states, and their magnitudes control the influence of the observation on the posterior distribution.

Figure 5 shows excerpts of the feature sequences obtained from two performances, and blue lines indicating the change of the state of the latent composition has changed. The figure suggests that the state changes with a notable change in the feature, such as when new notes are played. Since, by the definition of a left-to-right Markov model, the number of vertical lines is identical for all performances, we can align audio signals by mapping the occurrences of the \( i \)th vertical line for all performances, for each \( i \).

### 3.2.3 Generating Audio Features

Based on the previous expositions, we can see that at time \( t \) of performance \( d \), the audio feature is generated by choosing the state in the latent common structure that is referred at time \( t \) for performance \( d \). This state is extracted by referring to the performance sequence to recover the position of the latent composition. Therefore, the observation likelihood is given as follows:

\[
p(x_{d,t} | z, \phi, \theta) = \prod_{n,t} p(x_{d,t} | \theta_s) z_{n,t} \phi_{d,t,n}
\]

Here, \( p(x | \theta_s) \) is the likelihood of observation feature \( x \) at state \( s \) of the latent common structure, and its parameter \( \theta_s \) is generated from a prior distribution \( p(\theta_s | \theta_0) \).

For the sake of simplicity, we let \( p(x_{d,t} | \theta_s) \) be a \( \text{dim}(x) \)-dimensional Gaussian distribution with its parameters \( \theta_s \), generated from its conjugate distribution, the Gaussian-Gamma distribution. Specifically we let \( \theta_s = \{\mu_s, \lambda_s\} \), \( \theta_0 = \{m_0, v_0, u_0, k_0\} \), and let \( x_{d,t} | \mu_s, \lambda_s \sim N(\mu_s, \lambda_s^{-1}) \), with \( p(\mu_s, \lambda_s) \propto \lambda_s v_0 - \frac{1}{2} e^{-(\mu_s - m_0)^2 / (2 \lambda_s)} + k_0 \). One may incorporate a more elaborate model that better expresses the observation.

### 3.3 Inferring the Posterior Distribution

We derive the posterior distribution to the model described above. Since direct application of Bayes’ rule to arrive at the posterior is difficult, we employ the variational Bayes method [19] and find an approximate posterior of form

\[
q(\phi, z, \theta, \eta, \pi, \tau) = \prod_d q(\phi_{d,\cdot}) q(z | \pi) \prod_{d,n} q(\eta_{d,n}) \prod_s q(\theta_s) q(\tau_s)
\]

that minimizes the Kullback-Leibler (KL) divergence to the true posterior distribution.

\[ q(\phi) \text{ and } q(z) \text{ can be updated in a manner analogous to a HMM. For } q(z), \text{ we perform the forward-backward algorithm, with the state emission probability } g_{s,d} \text{ at position } n \text{ of the latent composition and the transition probability } v_s \text{ from state } s \text{ given as follows:} \]

\[
\log g_{n,s} = \sum_{d,t} (\phi_{d,t,n}) (\log p(x_{d,t} | \theta_s))
\]

\[
\log v_{s,s'} = (\log \tau_{s,s'})
\]

Here, \( \langle f(x) \rangle \) denotes the expectation of \( f(x) \) w.r.t. \( q \). Likewise, for \( q(\phi_{d,\cdot}) \), we perform the forward-backward algorithm, with the state emission probability \( h_{d,n} \) and transition probability \( w_{d,n'} \) given as follows:

\[
\log h_{d,t,n} = \sum_s (z_{n,s}) (\log p(x_{d,t} | \theta_s))
\]

\[
\log w_{d,n,n'} = \begin{cases} (\log \eta_{d,n}) & n = n' \\ (\log (1 - \eta_{d,n})) & n + 1 = n' \end{cases}
\]

We can update \( \pi \) as \( q(\pi) = \text{Dir}(\pi_0 + \langle z_1 \rangle) \), \( \eta \) as \( q(\eta_{d,n}) = \text{Beta}(a_0 + \sum_t (\phi_{d,t-1,n} \phi_{d,t,n}), b_0 + \sum_t (\phi_{d,t-1,n} - \phi_{d,t-1,n-1} \phi_{d,t,n})) \), and \( \tau \) as \( q(\tau_s) = \text{Dir}(\tau_{0,s} + \sum_{n>1} (z_{n-1,s} \tau_n)) \).

Based on these parameters, the generative model of audio features can be updated. Some commonly-used statistics for state \( s \) include the count \( N_s \), the mean \( \mu_s \) and the variance \( \Sigma_s \), which are given as follows:

\[
\bar{N}_s = \sum_{d,n,t} (z_{n,s}) (\phi_{d,t,n})
\]

\[
\bar{\mu}_s = \frac{1}{N_s} \sum_{d,n,t} (z_{n,s}) (\phi_{d,t,n}) x_{d,t}
\]

\[
\bar{\Sigma}_s = \frac{1}{N_s} \sum_{d,n,t} (z_{n,s}) (\phi_{d,t,n}) (x_{d,t} - \bar{\mu}_s)^2
\]

For example, the Gaussian/Gaussian-Gamma model described earlier can be updated as follows:

\[
q(\mu_s, \lambda_s) = N \mathcal{G} \left( \nu_0 + \bar{N}_s, \nu_0 \bar{m}_0 + \bar{N}_s \bar{\mu}_s, \frac{1}{2 \nu_0 + \bar{N}_s} \right)
\]

\[
u_0 + \frac{1}{2} \left( \bar{N}_s + \frac{\nu_0 \bar{N}_s}{\nu_0 + \bar{N}_s} (\mu_s - m_0)^2 \right)
\]

Hyperparameters may be set manually, or optimized by minimizing the KL divergence from \( q \) to the posterior.

### 3.4 Semi-Markov Performance Sequence

The model presented previously implicitly assumes that the state duration of the performance sequence follows the
geometric distribution. In such a model, it is noted, especially in the context of audio-to-score alignment [4], that further improvement is possible by incorporating a more explicit duration probability using an extension of the HMM known as the hidden semi-Markov models [5, 20].

In this paper, we assume that every performance plays a particular position in the music composition with more or less the same tempo. Hence, we incorporate an explicit duration probability to the performance sequence, such that the duration of each state is concentrated about some average state duration common to each performance. To this end, we assume that for each state \( n \) of the performance sequence, the state duration \( l \) follows a Gaussian distribution concentrated about a common mean:

\[
p(l|\gamma_n, c) = N(\gamma_n, \epsilon \gamma_n^2)
\]

We chose the Gaussian distribution due to convenience of inference. By setting \( c \) appropriately, we can provide a trade-off between the tendency for every piece to play in a same tempo sequence, and variation of tempo among different performances.

To incorporate such a duration probability in the performance sequence model, we augment the state space of the left-to-right Markov model of the performance sequence by a “count-down” variable \( l \) that indicates the number of frames remaining in the current state. Then, we assume that the maximum duration of each state is \( L \), and represent each state of the performance \( \phi_{d,t} \) as a tuple \((n, l) \in [1 \cdots N] \times [1 \cdots L] \), i.e., \( \phi_{d,t,n,l} \). In this model, state \((n, 1)\) transitions to \((n + 1, 1)\) with probability \( p(l|\mu_{n+1}, c) \), and state \((n, l)\) for \( l > 1 \) transitions to \((n, l - 1)\) with probability one. Finally, we constrain the terminal state to be \((N, 1)\). Note that \( n \) is no longer used because state duration is now described explicitly. The parameter \( \gamma_n \) can be optimized by maximum likelihood estimation of the second kind, to yield the following:

\[
\gamma_n = \frac{\sum_{d,t,l} l \phi_{d,t-1,n-1,l} \phi_{d,t,n,l} \epsilon \gamma_n^2}{\sum_{d,t,l} \phi_{d,t-1,n-1,l} \phi_{d,t,n,l} \epsilon \gamma_n^2}
\]

\( c \) may be optimized in a similar manner, but we found that the method performs better when \( c \) is fixed to a constant.

4. EVALUATION

We conducted two experiments to assess our method. First, we tested the effectiveness of our method against existing methods that ignore the organization of music [13, 18]. Second, we tested the robustness of our method to the length of the latent composition, which we need to fix in advance.

4.1 Experimental Conditions

We prepared two to five recordings to nine pieces of Chopin’s Mazurka (Op. 6-4, 17-4, 24-2, 30-2, 33-2, 41-2, 63-3, 67-1, 68-3), totaling in 38 audio recordings. For each of the nine pieces, we evaluated the alignment using (1) DTW using path constraints in [21] that minimizes the net squared distance (denoted “DTW”), (2) left-to-right HMM to model musical audio as done in existing methods [13, 18] (denoted “LRHMM”), (3) proposed method (denoted “Proposed”), and (4) proposed method with semi-Markov performance sequence (denoted “Proposed (HSMM)”).

For the feature sequence \( x_{d,t} \), we employed the chroma vector [11] and half-wave rectified difference of the chroma (\( \Delta \) chroma), evaluated using a frame length of 8192 samples and a 20% overlap with a sampling frequency of 44.1kHz.

For the proposed method, the hyperparameters related to the latent common sequence were set to \( \pi_0 = 0.1 \) and \( \tau_{0,s,s'} = 0.9 + 10\delta(s, s') \); these parameters encourages sparsity of the initial state probability and the state transitions, while encouraging self-transitions. The parameters related to the observation were set to \( \nu_0 = k_0 = 1 \), \( \nu_0 = 0.1 \) and \( m_0 = 0 \); such a set of parameters encourages a sparse variance, and assumes that the mean is highly dispersed. Moreover, we used \( S = 100 \) and \( N = 0.3 \min_d T_d \). For the semi-Markov performance sequence model, we set \( c = 0.1 \). This corresponds to having a standard deviation of \( \gamma_n \sqrt{0.1} \), or allowing the notes to deviate by a standard deviation of about 30%.

4.2 Experimental Results

We present below the evaluation of the alignment accuracy and the robustness to the length of the latent composition. On a workstation with Intel Xeon CPU (3.2GHz), our method takes about 3 minutes to process a minute of single musical audio.

4.2.1 Alignment Accuracy

We compared the aligned data to that given by reverse conducting data of the Mazurka Project [1]. Figure 6 shows the absolute error percentile. The figure shows that our method (“Proposed”) performs significantly better than the existing method based on a LRHMM. This suggests that, for a generative model approach to alignment, not only is model of performance difference critical but also that of the common music that the performances play. We also note an improved performance of the semi-Markov model performance sequence (“Proposed (HSMM)”) over the Markovian model (“Proposed”).

Note that when using the same features and squared-error model, the semi-Markovian model performs better.
than DTW. This result suggests that with appropriate structural and temporal models, a generative model approach is a viable alternative to audio alignment. The performance gain from Markov to semi-Markov model illuminates the forte of the generative model approach: temporal, spectral and structural constraints are mixed seamlessly to attain a trade-off among the trichotomy.

We note that our model is weak to compositional deviations, such as added ornaments and repeats because we assume every performance plays an identical composition. We observed that our method deals with an added noise as a noise or a note that gets played very shortly by most of the audio signals, but neither captures the nature of added notes as structural deviations. Moreover, our method sometimes gets “trapped” in local optima, most likely due to the strong mutual dependency between the latent variables.

4.2.2 Robustness to the Length of the Latent Composition

Since our method requires the user to set the length of latent composition $N$, we evaluated the quality of alignment as $N$ is varied. To evaluate the performance of our method with different values of $N$, we evaluated the alignment of the proposed method when $N$ is set to $N = \alpha |T_{d=1}|$, with $\alpha$ ranging from $\alpha = 0.1$ to $\alpha = 0.9$ with an increment of 0.1. Figure 7 shows the median alignment error. We find that when $\alpha$ is too small, when there is an insufficient number of states to describe a composition, the error increases. The error also increases when $\alpha$ is too large, since the maximum total allowed deviation decreases (i.e., to about $(1 - \alpha)|T_{d=1}|$). However, outside such extremities, the performance is relatively stable for moderate values of $\alpha$ around 0.5. This suggests that our method is relatively insensitive to a reasonable choice of $N$.

5. CONCLUSION

This paper presented an audio alignment method based on a probabilistic generative model. Based on the insight that a generative model of musical audio alignment should represent both the underlying musical composition and how it is performed by each audio signal, we formulated a unified generative model of musical composition and performance. The proposed generative model contributed to a significantly better alignment performance than existing methods. We believe that our contribution brings generative alignment on par with DTW-based alignment, opening door to alignment problem settings that require integration of various sources of uncertainties.

Future study includes incorporating better models of composition, performance and observation in our unified framework. In addition, inference over highly coupled hierarchical discrete state models is another future work.

Acknowledgment: This study was supported in part by JSPS KAKENHI 24220006 and 26700020.

6. REFERENCES

AUTOMATIC SET LIST IDENTIFICATION AND SONG SEGMENTATION FOR FULL-LENGTH CONCERT VIDEOS

Ju-Chiang Wang\textsuperscript{1,2}, Ming-Chi Yen\textsuperscript{1}, Yi-Hsuan Yang\textsuperscript{1}, and Hsin-Min Wang\textsuperscript{1}
\textsuperscript{1}Academia Sinica, Taipei, Taiwan
\textsuperscript{2}University of California, San Diego, CA, USA

\texttt{asriver.wang@gmail.com; \{ymchiqq, yang, whm\}@iis.sinica.edu.tw}

ABSTRACT

Recently, plenty of full-length concert videos have become available on video-sharing websites such as YouTube. As each video generally contains multiple songs, natural questions that arise include “what is the set list?” and “when does each song begin and end?” Indeed, many full concert videos on YouTube contain song lists and timecodes contributed by uploaders and viewers. However, newly uploaded content and videos of lesser-known artists typically lack this metadata. Manually labeling such metadata would be labor-intensive, and thus an automated solution is desirable. In this paper, we define a novel research problem, automatic set list segmentation of full concert videos, which calls for techniques in music information retrieval (MIR) such as audio fingerprinting, cover song identification, musical event detection, music alignment, and structural segmentation. Moreover, we propose a greedy approach that sequentially identifies a song from a database of studio versions and simultaneously estimates its probable boundaries in the concert. We conduct preliminary evaluations on a collection of 20 full concerts and 1,152 studio tracks. Our result demonstrates the effectiveness of the proposed greedy algorithm.

1. INTRODUCTION

In recent years, the practice of sharing and watching concert/performance footage on video sharing websites such as YouTube has grown significantly [12]. In particular, we have noticed that many concert videos consist of full-length, unabridged footage, featuring multiple songs. For example, the query “full concert” on YouTube returns a list of more than 2 million relevant videos. For a full concert video, the former is to identify the sequence of song titles in order to locate the studio version.

Ju-Chiang Wang, Ming-Chi Yen, Yi-Hsuan Yang, and Hsin-Min Wang.

To satisfy such a demand, the uploader or some viewers often post the “set list” with the timecode for each song,\textsuperscript{1} so that other viewers can easily fast-forward to the desired song. This metadata can help viewers to navigate a long concert. From a technical point of view, it also helps to extract the live version of a song to enrich a music database. Such a database could be used to analyze performance style, to discover song transition [17], to train classifiers for visual event detection [28], or to generate multi-camera mashups and summaries of concert videos [22, 27].

However, newly uploaded videos and those performed by less known artists typically lack this metadata, because manually identifying songs and song segmentation can be time-consuming even for an expert. One reason for this is because live performances can differ substantially from the studio recordings. Another reason is that live performances often contain covers of songs by other artists. Even if the annotator can readily identify all songs, it is still necessary to go through the entire video to locate the precise times that each song begins and ends. Therefore, an automated method is desirable to annotate the rapidly growing volume of full-length concert videos available online.

The aim of this paper is threefold. First, we define a novel research problem, i.e. automatic set list segmentation of full concert videos, and discuss its challenges. Second, we propose a greedy approach to tackle the problem. Third, we construct a novel dataset designed for this task and suggest several evaluation methods.

1.1 Task Definition and Challenges

There are two sub-tasks for this research problem: set list identification and song segmentation. Given a full concert video, the former is to identify the sequence of song titles played in the concert based on a large collection of studio version tracks, assuming that no prior knowledge on the live performance of the artist(s) of the concert is available. The latter task is to estimate the boundaries of each identified song in the set list. This problem poses some interesting challenges as follows:

- A live song can be played in many different ways, e.g., by changing its timbre, tempo, pitch and structure, corresponding to the comparing studio version.

\textsuperscript{1}A set list refers to a list of songs that a band/artist has played in a concert, and the timecode corresponds to the starting time of a song. Here is an example of full concert video with set list and timecodes on YouTube: \url{https://www.youtube.com/watch?v=qYojin1lltG}
Therefore, certain robustness should be considered.

- Live performances often feature transitions between consecutive songs, or even repeated oscillations between the sections of different songs, suggesting that one should identify songs on a small temporal scale.
- Concerts often feature sections with no reference in the collection of studio versions, such as intros, outros, solos, banter, transitions between songs, big rock endings, and applause, amongst others. Unexpected events such as broken instruments, sound system malfunctions, and interrupted songs can also be found. An ideal system should identify them or mark them as unknown songs/events, avoiding including them in a segmented song when appropriate.
- The artist may play cover songs from other artists partially or entirely throughout the concert, resulting in a much larger search space in the music database.
- The audio quality of user-contributed concert videos can vary significantly due to recording factors such as acoustic environment, position, device and user expertise [14]. The quality degradation can amplify the difficulty of the problem.

To tackle the above challenges, one may consider techniques for several fundamental problems in music information retrieval (MIR), such as audio fingerprinting/matching [3, 7], cover song identification [5, 24], audio quality assessment [14], musical event detection/tracking [32, 33], and music signal alignment and segmentation [18]. Therefore, automatic set list segmentation of full concert videos may present a new opportunity for MIR researchers to link music/audio technology to real-world applications.

### 1.2 Technical Contribution

Our technical contribution lies in the development of a greedy approach that incorporates three components: segmentation, song identification, and alignment (see Section 3). This approach provides a basic view as a baseline towards future advance. Starting from the beginning of the concert, our approach first identifies the candidate songs for a “probe excerpt” of the concert based on segmented music signals. Then, it estimates the most likely song title and boundaries of the probe excerpt based on dynamic time warping (DTW) [18]. This sequential process is repeated until the entire concert video has been processed. To evaluate the proposed algorithm, we collect 20 full concerts and 1,152 studio tracks from 10 artists (see Section 4). Moreover, we suggest three performance metrics for this task (see Section 5). Finally, we demonstrate the effectiveness of the proposed approach and observe that cover song identification works much better than audio fingerprinting for identifying the songs in a live performance (see Section 5).

### 2. RELATED WORK

According to a recent user study, YouTube was the second most preferred online music streaming service by users in 2012, just behind Pandora [12]. These community-contributed concert videos have been extensively studied in the multimedia community. Most existing works focus on handling the visual content of the concert videos [1, 10, 22, 27, 28]. Relatively little attention, however, has been paid in the MIR community to study the audio content of this type of data. Related work mainly focused on low-level audio signal processing for tasks such as audio fingerprint-based synchronization and alignment for concert video organization [9, 11, 29], and audio quality ranking for online concert videos [14]. More recently, Rafii et al. proposed a robust audio fingerprinting system to identify a live music fragment [23], without exploring full-length concert videos and song segmentation. To gain deeper understanding of the context and content of live performance, our work represents an early attempt to use the full concert video data.

We note that our work is also related to PHENICX [6], an ongoing project which aims at enriching the user experience of watching classical music concerts via state-of-the-art multimedia and Internet technologies. With a system for automatic set list segmentation of full concert videos, one could index a large amount of online musical content, extracting information that helps link live performance to the associated video content.

Aside from potential applications, the technical development of our work is highly motivated by several signal matching-based music retrieval problems, which can be categorized into audio fingerprinting (AF) [3, 30], audio matching [21], and cover song identification (CSID) [5, 24], according to their specificities and granularity [4, 7]. An AF system retrieves the exact audio piece that is the source of a query audio fragment. Audio matching is defined as the task of retrieving from a database all the audio fragments that are musically relevant to a query fragment. In contrast, CSID aims at identifying different renditions of a music piece in the track level (instead of fragment-level). Unlike AF which usually holds robustness to any noises that may apply on the same rendition of a song, audio matching and CSID should handle the musically motivated variations occurring in different performances or arrangements of a music piece [7].

### 3. PROPOSED GREEDY APPROACH

The proposed approach is outlined in Algorithm 1. It employs an intuitive greedy strategy that recursively probes an excerpt $X$ from the beginning of the unprocessed concert $Z$, identifies $K$ song candidates ($K = 5$) from the studio database $D$, selects the most probable song title $s^*$, estimates the boundaries $(i, j)$ of $s^*$ in $X$, and finally removes $s^*$ from $D$ and $X(1 : j)$ from $Z$. The process stops when the unprocessed portion of the input concert is shorter than a pre-defined threshold $\tau$. We make the following assumptions while developing Algorithm 1: 1) the performer plays nearly the entire part of a song rather than a certain small portion of the song, 2) a song in the studio database is performed at most once in a concert, and 3) the concert contains only songs from the same artist without covers. In practice, the artist of a concert can be easily known from the video title. Therefore, we only take the studio tracks of the artist to construct $D$. More details are given below.
Algorithm 1: Set list identification & segmentation

Input: A concert \( Z \); studio track database \( D \); probe length \( \ell \); end length \( \tau \); candidate number \( K \);

Output: Song list \( S \); boundary set \( B \);

1. \( S \leftarrow \emptyset; B \leftarrow \emptyset; \)
2. while \( \text{length}(Z) > \tau \) do
3. \( X \leftarrow Z(1 : l), \text{if } l > \text{length}(Z), l = \text{length}(Z); \)
4. \( \{s_k\}_{k=1}^K \leftarrow \text{identify the } K \text{ most probable songs that match } X, \text{based on the thumbnails of } D; \)
5. \( \{s^*, (i, j)\} \leftarrow \text{select the best song from } \{s_k\}_{k=1}^K \text{ and estimate its boundaries on } X, \text{based on the complete track of } D; \)
6. \( S \leftarrow S + s^*; B \leftarrow B + (i, j); \)
7. \( D \leftarrow D - s^*; Z \leftarrow Z - X(1 : j); \)
8. end

3.1 Segmentation

In our original design, we adopt music segmentation techniques to pre-process both the concert and every studio track in the database. This enhances the robustness to variation of song structure for the music matching and identification processes. However, operating on fine-grained segments of the concert significantly increases the computational time of the algorithm. Therefore, we make heuristic modifications to gain more efficiency as follows.

First, we segment a sufficiently long probe excerpt from the beginning of an unprocessed concert that could include the first entire song played in the unprocessed concert, without involving any musically motivated segmentation. Ideally, we hope the probe length \( l \) is longer than the exact song \( s^* \) plus the events prior to \( s^* \) (e.g., banter, applause).

In the experiment, we will compare different settings of \( l = \alpha \times \mu \), where \( \alpha \) is the parameter and \( \mu \) the mean length of all studio tracks in the database.

Second, each studio track in the database is represented by its thumbnail for better efficiency in the later song candidate identification stage. Similar idea has been introduced by Grosche et al. [8]. We develop a simple method analogous to [15] based on structural segmentation. Segmentation [2, 16] is utilized to discover the musically homogeneous sections marked by structure labels such as ‘A,’ ‘B,’ and ‘N’ for each studio track. We compute a weighted factor \( \gamma \) that jointly considers the repetition count and average segment length for each label. The longest segment of the label that has the largest \( \gamma \) is selected as the thumbnail.

3.2 Song Candidate Identification

Song candidate identification uses the probe excerpt as a query and ranks the studio thumbnails in the database. We employ two strategies for the identifier: audio fingerprinting (AF) and cover song identification (CSID). For simplicity, we employ existing AF and CSID methods in this work. For future work, it might be more interesting to integrate the identifier with the subsequent boundary estimator.

For AF, we implement the identifier using the widely-known landmark-based approach proposed in [31]. It extracts prominent peaks (a.k.a. landmarks) from the magnitude spectrogram of a reference track (e.g. a studio version) and characterizes each pair of landmarks by the frequencies of the landmarks and the time in between them, which provide indices to a hash table that allows fast retrieval of similarity information [30]. For a query (e.g. a probe excerpt), we see whether there are sufficient number of matched landmarks between the query and a reference track by looking up the hash table. If the query track is a noisy version of the reference track, this approach is likely to perform fairly well, because the landmarks are most likely to be preserved in noise and distortion.

For CSID, we implement the identifier mainly based on the chroma DCT-reduced log pitch (CRP) features [19] and the cross recurrence quantification (CRQ) approach [25], which correspond to two major components in a state-of-the-art CSID system [26]. Specifically, we first extract the frame-based CRP features for the probe excerpt and each studio track by the Chroma Toolbox [20]. Then, we determine the key transposition using the optimal transposition index (OTI) [25]. To apply CRQ, we follow the standard procedures [25], including constructing the delay coordinate state space vectors, computing the cross recurrence plot, deriving the \( Q_{\text{max}} \) score, and performing normalization on the scores across the database. This CSID system (cf. CYWW1) has led to performance comparable to the state-of-the-art systems in the MIREX audio cover song identification task (e.g., on Sapp’s Mazurka Collection).

3.3 Song Selection and Boundary Estimation

The next step is to select the most probable song \( k^* \) from the top \( K \) studio song candidates, \( \{Y_k\}_{k=1}^K \), and at the same time estimate its boundaries on the probe excerpt \( X \).

Accordingly, our goal is to find a \( Y_k \) and the corresponding subsequence \( X^* = X(i^* : j^*) \) that results in the best matching between \( Y_k \) and \( X^* \), where \( 1 \leq i^* < j^* \leq N \).

Such process is based on the DTW alignment between \( X \) and each \( Y_k \), as presented in Algorithm 2.

Let \( X = \{x_1, \ldots, x_N\} \) and denote the complete track of \( Y_k \) as \( Y' = \{y_1, \ldots, y_N\} \), where \( x_i \) and \( y_i \) represent the frame-based CRP vectors and \( N > M \). We compute the cost by the negative cosine similarity of CRP between two frames after the OTI key transposition. One can observe that Algorithm 2 includes two sub-procedures of one-side boundary estimation (cf. Algorithm 3). It first executes Algorithm 3 to search for the end boundary \( j' \) on \( X \) and then reverses the search from \( j' \) for the start boundary \( i' \) using Algorithm 3 with the cost matrix rotated by 180 degrees.

We follow the standard procedure to compute the accumulated cost matrix \( D \) in [18]. Then, Algorithm 3 searches from \( D(\frac{N}{2} + 1, M) \) to \( D(N, M) \) for the minimum average cost of DTW alignments, denoted by \( \delta_e^* \), where the average cost is defined as the accumulated cost divided by the length of its optimal warping path (OWP). The frame index of \( \delta_e^* \) is set as the boundary.

After the \( K \) candidates are processed, we pick the one

\[ \delta_e^* \]
Algorithm 2: Boundaries & average cost estimation
\begin{algorithm}
\textbf{Input}: Concert excerpt $X$; a studio track $Y'$;
\textbf{Output}: Boundary pair $(i',j')$; average cost $\delta$;
\begin{enumerate}
\item $C \leftarrow N$-by-$M$ cost matrix between $X$ and $Y'$;
\item $(j', \emptyset) \leftarrow$ one-side boundary estimation on $C$;
\item $C \leftarrow$ rotate $C(1:j',1:M)$ by 180 degrees;
\item $(\text{index}, \delta) \leftarrow$ one-side boundary estimation on $C$;
\item $i' \leftarrow j' - \text{index} + 1$;
\end{enumerate}
\end{algorithm}

Algorithm 3: One-side boundary estimation
\begin{algorithm}
\textbf{Input}: Cost matrix $C$;
\textbf{Output}: Boundary $\beta$; average cost $\delta$;
\begin{enumerate}
\item $D \leftarrow$ accumulated cost matrix from $C(1,1)$;
\item for $1 \leftarrow i$ to $\frac{N}{2}$ do
\begin{enumerate}
\item $p^* \leftarrow$ compute the OWP of $D(1:\frac{N}{2} + i, 1:M)$;
\item $\Delta(i) \leftarrow D(\frac{N}{2} + i, M) / \text{length}(p^*)$;
\end{enumerate}
\item end
\item $(\delta, \text{index}) \leftarrow$ the minimum value and its index of $\Delta$;
\item $\beta \leftarrow \text{index} + \frac{N}{2}$;
\end{enumerate}
\end{algorithm}

with the lowest average cost, $k^* = \arg \min_k \{\delta_k\}_{k=1}^K$, and set the boundary pair as $(i^*_k, j^*_k)$. In other words, we re-rank the top $K$ candidates according to the results of Algorithm 2, based on the content of the complete studio tracks.

4. DATA COLLECTION
We collect 20 popular full concert videos (from the first few responses to the query “full concert” to Youtube) and the associated set lists and timecodes from YouTube. Therefore, the music genre is dominated by pop/rock. We manually label the start and end boundaries of each song based on the timecodes, as a timecode typically corresponds to the start time of a song and may not be always accurate. There are 10 artists. For each artist, we collect as many studio tracks as possible including the songs performed in the collected concerts to form the studio database. On average, we have 115.2 studio version tracks for each artist, and each full concert video contains 16.2 live version tracks. Table 1 summarizes the dataset.

5. EVALUATION
5.1 Pilot Study on Set List Identification
We conduct a pilot study to investigate which strategy (i.e., AF or CSID) performs better for set list identification, assuming that the song segmentation is perfect. For simplicity, we extract all the songs from the concert videos according to the manually labeled boundaries and treat each entire live song as a query (instead of thumbnail). We use mean average precision (MAP) and precision@1 with respect to the studio database as the performance metrics. We also perform random permutation ten times for each query to generate a lower bound performance, denoted by ‘Random.’ One can observe from Table 2 that CSID performs significantly better than AF in our evaluation, showing that the landmark-based AF approach does not work well for live version identification. This confirms our intuition as live rendition can be thought of as a cover version of the studio version [5]. In consequence, we use CSID as the song candidate identifier in the following experiments.

5.2 Performance Metrics
We use the following performance metrics for set list identification and song segmentation: edit distance (ED), boundary deviation (BD), and frame accuracy (FA). The first metric ED is originally used to estimate the dissimilarity of two strings and has been adopted in numerous MIR tasks [13]. We compute the ED between an output song sequence (a list of song indices) and the ground truth counterpart via dynamic programming. The weights for insertion, deletion, and substitution are all set to 1. ED can only evaluate the accuracy of set list identification.

The second metric BD directly measures the absolute deviation in second between the estimated boundary and that of the ground truth for only each correctly identified song, ignoring those wrongly inserted songs in the output set list, as they are not presented in the ground truth. Therefore, the average BD of a concert reflects the accuracy of song segmentation but not set list identification.

The last metric, FA, which has been used in tasks such as melody extraction, represents the accuracy at the frame-level (using non-overlapped frame with length 200 ms). Throughout the concert, we mark the frames between the start and end boundaries of each song by its song index and otherwise by ‘x’ (belonging to no song). Then, we calculate the percentage of correct frames (the intersection rate) by comparing the output frame sequence with the ground truth counterpart. Therefore, FA can reflect the accuracy of both set list identification and song segmentation.

5.3 Baseline Approach
To study the effectiveness of the song selection and boundary estimation algorithms (see Section 3.3), we construct a baseline approach using Algorithm 1 without Algorithms 2 and 3. Specifically, we select the song $s^*$ with the largest
Table 3. Result of the greedy approach with $\alpha=1.5$ for the 20 full concerts and their average (AVG) performance. While ‘AVG ($\alpha=1.2$ or $\alpha=1.8$)’ only shows the average performance with different $l$ settings. ‘Baseline’ represents the average performance of the approach in Section 5.3. Additional abbreviations: A (Artist ID), SG (number of Songs in the Ground truth set list), SO (number of Songs in the Output set list), SBD (start BD), and eBD (end BD). Symbol $\star$ marks the metrics that are the smaller the better.

<table>
<thead>
<tr>
<th>ID</th>
<th>A</th>
<th>SG</th>
<th>SO</th>
<th>ED$\star$</th>
<th>sBD$\star$</th>
<th>eBD$\star$</th>
<th>FA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7</td>
<td>20</td>
<td>15</td>
<td>17</td>
<td>6.5</td>
<td>89.1</td>
<td>0.317</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>17</td>
<td>17</td>
<td>4</td>
<td>3.3</td>
<td>12.3</td>
<td>0.786</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>15</td>
<td>15</td>
<td>3</td>
<td>27.2</td>
<td>33.2</td>
<td>0.744</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>23</td>
<td>25</td>
<td>14</td>
<td>8.8</td>
<td>66.8</td>
<td>0.441</td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>19</td>
<td>18</td>
<td>5</td>
<td>11.5</td>
<td>27.8</td>
<td>0.641</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>10</td>
<td>11</td>
<td>1</td>
<td>19.1</td>
<td>22.8</td>
<td>0.875</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>10</td>
<td>10</td>
<td>6</td>
<td>28.2</td>
<td>39.1</td>
<td>0.428</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>22</td>
<td>22</td>
<td>9</td>
<td>28.2</td>
<td>39.6</td>
<td>0.610</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>20</td>
<td>21</td>
<td>7</td>
<td>30.7</td>
<td>35.9</td>
<td>0.653</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>17</td>
<td>15</td>
<td>4</td>
<td>5.3</td>
<td>9.8</td>
<td>0.758</td>
</tr>
<tr>
<td>11</td>
<td>9</td>
<td>22</td>
<td>21</td>
<td>3</td>
<td>6</td>
<td>8.7</td>
<td>0.860</td>
</tr>
<tr>
<td>12</td>
<td>4</td>
<td>17</td>
<td>19</td>
<td>7</td>
<td>32.0</td>
<td>21.9</td>
<td>0.681</td>
</tr>
<tr>
<td>13</td>
<td>2</td>
<td>9</td>
<td>12</td>
<td>5</td>
<td>110</td>
<td>155</td>
<td>0.509</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>17</td>
<td>17</td>
<td>2</td>
<td>20.1</td>
<td>18.4</td>
<td>0.777</td>
</tr>
<tr>
<td>15</td>
<td>2</td>
<td>11</td>
<td>11</td>
<td>7</td>
<td>50.9</td>
<td>72.9</td>
<td>0.393</td>
</tr>
<tr>
<td>16</td>
<td>3</td>
<td>17</td>
<td>20</td>
<td>9</td>
<td>36.9</td>
<td>24.7</td>
<td>0.544</td>
</tr>
<tr>
<td>17</td>
<td>4</td>
<td>13</td>
<td>11</td>
<td>4</td>
<td>48.1</td>
<td>94.3</td>
<td>0.626</td>
</tr>
<tr>
<td>18</td>
<td>3</td>
<td>23</td>
<td>22</td>
<td>10</td>
<td>10</td>
<td>34.8</td>
<td>0.636</td>
</tr>
<tr>
<td>19</td>
<td>5</td>
<td>7</td>
<td>7</td>
<td>3</td>
<td>42.4</td>
<td>13.6</td>
<td>0.584</td>
</tr>
<tr>
<td>20</td>
<td>5</td>
<td>15</td>
<td>13</td>
<td>9</td>
<td>42.4</td>
<td>36.6</td>
<td>0.465</td>
</tr>
<tr>
<td>AVG($\alpha=1.5$)</td>
<td>16.2</td>
<td>16.1</td>
<td>6.5</td>
<td>23.4</td>
<td>42.9</td>
<td>0.616</td>
<td></td>
</tr>
<tr>
<td>AVG($\alpha=1.2$)</td>
<td>16.2</td>
<td>18</td>
<td>7.3</td>
<td>25.7</td>
<td>57.3</td>
<td>0.582</td>
<td></td>
</tr>
<tr>
<td>AVG($\alpha=1.8$)</td>
<td>16.2</td>
<td>14.6</td>
<td>8.4</td>
<td>29.3</td>
<td>45.3</td>
<td>0.526</td>
<td></td>
</tr>
<tr>
<td>Baseline</td>
<td>16.2</td>
<td>19.7</td>
<td>8.9</td>
<td>229</td>
<td>241</td>
<td>0.434</td>
<td></td>
</tr>
</tbody>
</table>

5.4 Result and Discussion

Table 3 shows the quantitative result of each concert, the average performance (AVG) with different values of $l$, and the average performance of Baseline. Figure 1 depicts the qualitative results of three concerts, including the best, medium, and the worst cases according to FA in Table 3.

The following observations can be made. First, the AVG performances of the complete approach are significantly better than those of Baseline in all metrics, demonstrating the effectiveness of the proposed approach via both quantitative and qualitative results. We are currently expanding the size of the dataset and conducting more in-depth signal-level analysis of the dataset. Due to the copyright issue on the studio track collection, however, it is not likely to distribute the dataset.

In short, while there is still much room for improvement, we find that the result of the proposed greedy approach is quite satisfactory in some cases (e.g., Concert 6 in Figure 1). The greedy approach is preliminary in nature. We believe that better result can be obtained by explicitly addressing the challenges described in Section 1.1.

6. CONCLUSION AND FUTURE DIRECTION

In this paper, we have proposed a novel MIR research problem with a new dataset and a new greedy approach to address the problem. We have also validated the effectiveness of the proposed approach via both quantitative and qualitative results. We are currently expanding the size of the dataset and conducting more in-depth signal-level analysis of the dataset. Due to the copyright issue on the studio track collection, however, it is not likely to distribute the dataset. We will propose this task to MIREX to call for more advanced approaches to tackle this problem.

7. ACKNOWLEDGEMENT

This work was supported by Academia Sinica–UCSD Postdoctoral Fellowship to Ju-Chiang Wang, and the Ministry of Science and Technology of Taiwan under Grants NSC 101-2221-E-001-019-MY3 and 102-2221-E-001-004-MY3.

8. REFERENCES


Figure 1. Qualitative result of three concerts, which represent the best (Concert 6, ‘Guns N’ Roses’), medium (Concert 8, ‘Linkin’ Park’), and worst (Concert 1, ‘Metallica’) output cases in the dataset. Black blocks correspond to no song. Different songs are marked by different colors. The number in a song block stands for the song index in the studio database.

Note that Song 42 (‘Numb’) was sung twice in Concert 8, firstly by ‘Linkin’ Park’ and then by ‘featuring Jay-Z’.


ON INTER-RATER AGREEMENT IN AUDIO MUSIC SIMILARITY

Arthur Flexer
Austrian Research Institute for Artificial Intelligence (OFAI)
Freyung 6/6, Vienna, Austria
arthur.flexer@ofai.at

ABSTRACT

One of the central tasks in the annual MIREX evaluation campaign is the "Audio Music Similarity and Retrieval (AMS)" task. Songs which are ranked as being highly similar by algorithms are evaluated by human graders as to how similar they are according to their subjective judgment. By analyzing results from the AMS tasks of the years 2006 to 2013 we demonstrate that: (i) due to low inter-rater agreement there exists an upper bound of performance in terms of subjective gradings; (ii) this upper bound has already been achieved by participating algorithms in 2009 and not been surpassed since then. Based on this sobering result we discuss ways to improve future evaluations of audio music similarity.

1. INTRODUCTION

Probably the most important concept in Music Information Retrieval (MIR) is that of music similarity. Proper modeling of music similarity is at the heart of every application allowing automatic organization and processing of music databases. Consequently, the "Audio Music Similarity and Retrieval (AMS)" task has been part of the annual "Music Information Retrieval Evaluation eXchange" (MIREX) [2] since 2006. MIREX is an annual evaluation campaign for MIR algorithms allowing for a fair comparison in standardized settings in a range of different tasks. As such it has been of great value for the MIR community and an important driving force of research and progress within the community. The essence of the AMS task is to have human graders evaluate pairs of query/candidate songs. The query songs are randomly chosen from a test database and the candidate songs are recommendations automatically computed by participating algorithms. The human graders rate whether these query/candidate pairs "sound similar" using both a BROAD ("not similar", "somewhat similar", "very similar") and a FINE score (from 0 to 10 or from 0 to 100, depending on the year the AMS task was held, indicating degrees of similarity ranging from failure to perfection).

It is precisely this general notion of "sounding similar" which is the central point of criticism in this paper. A recent survey article on the "neglected user in music information retrieval research" [13] has made the important argument that users apply very different, individual notions of similarity when assessing the output of music retrieval systems. It seems evident that music similarity is a multi-dimensional notion including timbre, melody, harmony, tempo, rhythm, lyrics, mood, etc. Nevertheless most studies exploring music similarity within the field of MIR, which are actually using human listening tests, are restricted to overall similarity judgments (see e.g. [10] or [11, p. 82]) thereby potentially blurring the many important dimensions of musical expression. There is very little work on what actually are important dimensions for humans when judging music similarity (see e.g. [19]).

This paper therefore presents a meta analysis of all MIREX AMS tasks from 2006 to 2013 thereby demonstrating that: (i) there is a low inter-rater agreement due to the coarse concept of music similarity; (ii) as a consequence there exists an upper bound of performance that can be achieved by algorithmic approaches to music similarity; (iii) this upper bound has already been achieved years ago and not surpassed since then. Our analysis is concluded by making recommendations on how to improve future work on evaluating audio music similarity.

2. RELATED WORK

In our review on related work we focus on papers directly discussing results of the AMS task thereby addressing the problem of evaluation of audio music similarity.

After the first implementation of the AMS task in 2006, a meta evaluation of what has been achieved has been published [8]. Contrary to all subsequent editions of the AMS task, in 2006 each query/candidate pair was evaluated by three different human graders. Most of the study is concerned with the inter-rater agreement of the BROAD scores of the AMS task as well as the "Symbolic Melodic Similarity (SMS)" task, which followed the same evaluation protocol. To access the amount of agreement, the authors use Fleiss's Kappa [4] which ranges between 0 (no agreement) and 1 (perfect agreement). Raters in the AMS task achieved a Kappa of 0.21 for the BROAD scores of the AMS task, which can be seen as a "fair" level of agreement. Such a "fair" level of agreement [9] is given if the Kappa result is between 0.21 and 0.40, therefore positioning the
BROAD result at the very low end of the range. Agreement in SMS is higher (Kappa of 0.37), which is attributed to the fact that the AMS task is "less well-defined" since graders are only informed that "works should sound similar" [8]. The authors also note that the FINE scores for query/candidate pairs, which have been judged as "somewhat similar", show more variance than the one judged as "very" or "not" similar. One of the recommendations of the authors is that "evaluating more queries and more candidates per query would more greatly benefit algorithm developers" [8], but also that a similar analysis of the FINE scores is also necessary.

For the AMS task 2006, the distribution of differences between FINE scores of raters judging the same query/candidate pair has already been analysed [13]. For over 50%, the difference between rater FINE scores is larger than 20. The authors also note that this is very problematic since the difference between the best and worst AMS 2012 systems was just 17.

In yet another analysis of the AMS task 2006, it has been reported [20] that a range of so-called "objective" measures of audio similarity are highly correlated with subjective ratings by human graders. These objective measures are based on genre information, which can be used to automatically rank different algorithms producing lists of supposedly similar songs. If the genre information of the query and candidate songs are the same, a high degree of audio similarity is achieved since songs within a genre are supposed to be more similar than songs from different genres. It has therefore been argued that, at least for large-scale evaluations, these objective measures can replace human evaluation [20]. However, this is still a matter of controversy within the music information retrieval community, see e.g. [16] for a recent and very outspoken criticism of this position.

A meta study of the 2011 AMS task explored the connection between statistical significance of reported results and how this relates to actual user satisfaction in a more realistic music recommendation setting [17]. The authors made the fundamental clarification that the fact of observing statistically significant differences is not sufficient. More important is whether this difference is noticeable and important to actual users interacting with the systems. Whereas a statistically significant difference can always be achieved by enlarging the sample size (i.e. the number of query/candidate pairs), the observed difference can nevertheless be so small that it is of no importance to users. Through a crowd-sourced user evaluation, the authors are able to show that there exists an upper bound of user satisfaction with music recommendation systems of about 80%. More concretely, in their user evaluation the highest percentage of users agreeing that two systems "are equally good" never exceeded 80%. This upper bound cannot be surpassed since there will always be users that disagree concerning the quality of music recommendations. In addition the authors are able to demonstrate that differences in FINE scores, which are statistically significant, are so small that they make no practical difference for users.

3. DATA

For our meta analysis of audio music similarity (AMS) we use the data from the "Audio Music Similarity and Retrieval" tasks from 2006 to 2013 within the annual MIREX [2] evaluation campaign for MIR algorithms.

For the AMS 2006 task, 5000 songs were chosen from the so-called "uspop", "uscrap" and "cover song" collections. Each of the participating 6 system then returned a 5000x5000 AMS distance matrix. From the complete set of 5000 songs, 60 songs were randomly selected as queries and the first 5 most highly ranked songs out of the 5000 were extracted for each query and each of the 6 systems (according to the respective distance matrices). These 5 most highly ranked songs were always obtained after filtering out the query itself, results from the same artist (i.e. a so-called artist filler was employed [5]) and members of the cover song collection (since this was essentially a separate task run together with the AMS task). The distribution for the 60 chosen random songs is highly skewed towards rock music: 22 ROCK songs, 6 JAZZ, 6 RAP&HIPHOP, 5 ELECTRONICA&DANCE, 5 R&B, 4 REGGAE, 4 COUNTRY, 4 LATIN, 4 NEWAGE. Unfortunately the distribution of genres across the 5000 songs is not available, but there is some information concerning the "excessively skewed distribution of examples in the database (roughly 50% of examples are labeled as Rock/Pop, while a further 25% are Rap & Hip-Hop)". For each query song, the returned results (candidates) from all participating systems were evaluated by human graders. For each individual query/candidate pair, three different human graders provided both a FINE score (from 0 (failure) to 10 (perfection)) and a BROAD score (not similar, somewhat similar, very similar) indicating how similar the songs are in their opinion. This altogether gives $6 \times 60 \times 5 \times 3 = 5400$ human FINE and BROAD gradings. Please note that since some of the query/candidate pairs are identical for some algorithms (i.e. different algorithms returned identical candidates) and since such identical pairs were not graded repeatedly, the actual number of different FINE and BROAD gradings is somewhat smaller.

Starting with the AMS task 2007, a number of small changes to the overall procedure was introduced. Each participating algorithm was given 7000 songs chosen from the "uspop", "uscrap" and "american" "classical" and "sundry" collections. Therefore there is only a partial overlap in music collections ("uspop" and "uscrap") compared to AMS 2006. From now on 30 second clips instead of the full songs were being used both as input to the algorithms and as listening material for the human graders. For the subjective evaluation of music similarity, from now on 100 query songs were randomly chosen representing the 10 genres found in the database (i.e., 10 queries per genre). The whole database consists of songs from equally sized genre groups: BAROQUE, COUNTRY, EDANCE,

---

3 The results and details can be found at: [http://www.music-ir.org/mirex/wiki/MIREX_HOME](http://www.music-ir.org/mirex/wiki/MIREX_HOME)

3 This is stated in the 2006 MIREX AMS results: [http://www.music-ir.org/mirex/wiki/2006\_Audio\_Music\_Similarity\_and\_Retrieval\_Results](http://www.music-ir.org/mirex/wiki/2006\_Audio\_Music\_Similarity\_and\_Retrieval\_Results)
JAZZ, METAL, RAP/HIPHOP, ROCKROLL, ROMANTIC, BLUES, CLASSICAL. Therefore there is only a partial overlap of genres compared to AMS 2006 (COUNTRY, EDANCE, JAZZ, RAP/HIPHOP, ROCKROLL). As with AMS 2006, the 5 most highly ranked songs were then returned per query as candidates (after filtering for the query song and songs from the same artist). For AMS tasks 2012 and 2013, 50 instead of 100 query songs were chosen and 10 instead of 5 most highly ranked songs returned as candidates.

Probably the one most important change to the AMS 2006 task is the fact that from now on every query/candidate pair was only being evaluated by a single user. Therefore the degree of inter-rater agreement cannot be analysed anymore. For every AMS task, the subjective evaluation therefore results in $a \times 100 \times 5$ human FINE and BROAD gradings, with $a$ being the number of participating algorithms, 100 the number of query songs and 5 the number of candidate songs. For AMS 2012 and 2013 this changed to $a \times 50 \times 10$, which yields the same overall number. These changes are documented on the respective MIREX websites, but also in a MIREX review article covering all tasks of the campaign [3]. For AMS 2007 and 2009, the FINE scores range from 0 to 10, from AMS 2010 onwards from 0 to 100. There was no AMS task in MIREX 2008.

4. RESULTS

In our meta analysis of the AMS tasks from years 2006 to 2013, we will focus on the FINE scores of the subjective evaluation conducted by the human graders. The reason is that the FINE scores provide more information than the BROAD scores which only allow for three categorical values. It has also been customary for the presentation of AMS results to mainly compare average FINE scores for the participating algorithms.

4.1 Analysis of inter-rater agreement

Our first analysis is concerned with the degree of inter-rater agreement achieved in the AMS task 2006, which is the only year every query/candidate pair has been evaluated by three different human graders. Previous analysis of AMS results has concentrated on BROAD scores and used Fleiss’s Kappa as a measure of agreement (see Section 2). Since the Kappa measure is only defined for the categorical scale, we use the Pearson correlation $\rho$ between FINE scores of pairs of graders. As can be seen in Table 1, the average correlations range from 0.37 to 0.43. Taking the square of the observed values of $\rho$, we can see that only about 14 to 18 percent of the variance of FINE scores observed in one grader can be explained by the values observed for the respective other grader (see e.g. [11] on $\rho^2$ measures). Therefore, this is the first indication that agreement between raters in the AMS task is rather low.

Next we plotted the average FINE score of a rater $i$ for all query/candidate pairs, which he or she rated within a certain interval of FINE scores $v$, versus the average FINE scores achieved by the respective other graders. The main diagonal gives the average FINE scores of the AMS task 2006. This upper bound inter-rater agreement achieved in the AMS task 2006, which is an upper bound for the average FINE scores of the AMS task 2006. This upper bound is the maximum of average FINE scores that can be achieved within such an evaluation setting. This upper bound is due to the fact that there is a considerable lack of agreement between human graders. What sounds very similar to one of the graders will on average not receive equally high scores by other graders.

The average FINE score achieved by the best participating system in AMS 2006 (algorithm EP) is $4.30 \pm 8.8$ (mean ± variance). The average upper bound inter-rater grading is $6.54 \pm 6.96$. The difference between the best FINE scores achieved by the system EP and the upper bound is significant according to a t-test: $|t| = |z| = 12.0612 > t_{0.05, df=1231} = 1.96$ (confidence level of 95%, degrees of freedom = 1231). We can therefore conclude that for the AMS 2006 task, the upper bound on the av-

<table>
<thead>
<tr>
<th></th>
<th>grader1</th>
<th>grader2</th>
<th>grader3</th>
</tr>
</thead>
<tbody>
<tr>
<td>grader1</td>
<td>1.00</td>
<td>0.43</td>
<td>0.37</td>
</tr>
<tr>
<td>grader2</td>
<td>1.00</td>
<td>1.00</td>
<td>0.40</td>
</tr>
<tr>
<td>grader3</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Table 1. Correlation of FINE scores between pairs of human graders.

<table>
<thead>
<tr>
<th></th>
<th>grader1</th>
<th>grader2</th>
<th>grader3</th>
</tr>
</thead>
<tbody>
<tr>
<td>grader1</td>
<td>9.57</td>
<td>6.66</td>
<td>5.99</td>
</tr>
<tr>
<td>grader2</td>
<td>6.60</td>
<td>9.55</td>
<td>6.67</td>
</tr>
<tr>
<td>grader3</td>
<td>6.62</td>
<td>6.87</td>
<td>9.69</td>
</tr>
</tbody>
</table>

Table 2. Pairwise inter-rater agreement for FINE scores from interval $v = [9, 10]$. FINE scores achieved by the other two raters $j \neq i$ for the same query/candidate pairs. We therefore explore how human graders rate pairs of songs which another human grader rated at a specific level of similarity. The average results across all raters and for intervals $v$ ranging from $[0, 1]$, $[1, 2]$... to $[9, 10]$ are plotted in Figure 1. It is evident that there is a considerable deviation from the theoretical perfect agreement which is indicated as a dashed line. Pairs of query/candidate songs which are rated as being very similar (FINE score between 9 and 10) by one grader are on average only rated at around 6.5 by the two other raters. On the other end of the spectrum, query/candidate pairs rated as being not similar at all (FINE score between 0 and 1) receive average FINE scores of almost 3 by the respective other raters. The degree of inter-rater agreement for pairs of raters at the interval $v = [9, 10]$ is given in Table 2. There are 333 pairs of songs which have been rated within this interval. The main diagonal gives the average rating one grader gave to pairs of songs in the interval $v = [9, 10]$. The off-diagonal entries show the level of agreement between different raters. As an example, query/candidate pairs that have been rated between 9 and 10 by grader1 have received an average rating of 6.66 by grader2. The average of these pairwise inter-rater agreements given in Table 2 is 6.54 and is an upper bound for the average FINE scores of the AMS task 2006. This upper bound is the maximum of average FINE scores that can be achieved within such an evaluation setting. This upper bound is due to the fact that there is a considerable lack of agreement between human graders. What sounds very similar to one of the graders will on average not receive equally high scores by other graders.
average FINE score had not yet been reached and that there still was room for improvement for future editions of the AMS task.

4.2 Comparison to the upper bound

We will now compare the performance of the respective best participating systems in AMS 2007, 2009 to 2013 to the upper bound of average FINE scores we have retrieved in Section 4.1. This upper bound can that possibly be achieved due to the low inter-rater agreement results from the analysis of the AMS 2006 task. Although the whole evaluation protocol in all AMS tasks over the years is almost identical, AMS 2006 did use a song database that is only overlapping with that of subsequent years. It is therefore of course debatable how strictly the upper bound from AMS 2006 applies to the AMS results of later years. As outlined in Section 3, AMS 2006 has a genre distribution that is skewed to about 50% of rock music whereas all other AMS databases consist of equal amounts of songs from 10 genres. One could make the argument that in general songs from the same genre are being rated as being more similar than songs from different genres. As a consequence, agreement of raters for query/candidate pairs from identical genres might also be higher. Therefore inter-rater agreement within such a more homogeneous database should be higher than in a more diverse database and it can be expected that an upper bound of inter-rater agreement for AMS 2007 to 2013 is even lower than the one we obtained in Section 4.1. Of course this line of argument is somewhat speculative and needs to be further investigated.

In Figure 2 we have plotted the average FINE score of the highest performing participants of AMS tasks 2007, 2009 to 2013. These highest performing participants are the ones that achieved the highest average FINE scores in the respective years. In terms of statistical significance, the performance of these top algorithms is often at the same level as a number of other systems. We have also plotted the upper bound (dashed line) and a 95% confidence interval (dot-dashed lines). As can be seen the performance peaked in the year 2009 where the average FINE score reached the confidence interval. Average FINE scores in all other years are always a little lower. In Table 3 we show the results of a number of t-tests always comparing the performance to the upper bound. Table 3 gives the AMS year, the abbreviated name of the winning entry, the mean performance, its variance and the resulting t-value (with 831 degrees of freedom and 95% confidence). Only the best entry from year 2009 (PS2) reaches the performance of the upper bound, the best entries from all other years are statistically significant below the upper bound (critical value for all t-tests is again 1.96).

Interestingly, this system PS2 which gave the peak performance of all AMS years has also participated in 2010 to 2013. In terms of statistical significance (as measured via Friedman tests as part of the MIREX evaluation), PS2 has performed on the same level with the top systems of all following years. The systems PS2 has been submitted by Tim Pohle and Dominik Schnitzer and essentially consists of a timbre and a rhythm component [12]. Its main ingredients are MFCCs modeled via single Gaussians and Fluctuation patterns. It also uses the so-called P-norm normalization of distance spaces for combination of timbre and rhythm and to reduce the effect of hubness (anormal behavior of
distance spaces due to high dimensionality, see [6] for a discussion related to the AMS task and [14] on re-scaling of distance spaces to avoid these effects).

As outlined in Section 3, from 2007 on the same database of songs was used for the AMS tasks. However, each year a different set of 100 or 50 songs was chosen for the human listening tests. This fact can explain that the one algorithm participating from 2009 to 2013 did not always perform at the exact same level. After all, not only the choice of different human graders is a source of variance in the obtained FINE scores, but also the choice of different song material. However, the fact that the one algorithm that reached the upper bound has so far not been outperformed adds additional evidence that the upper bound that we obtained indeed is valid.

5. DISCUSSION

Our meta analysis of all editions of the MIREX "Audio Music Similarity and Retrieval" tasks conducted so far has produced somewhat sobering results. Due to the lack of inter-rater agreement there exists an upper bound of performance in subjective evaluation of music similarity. Such an upper bound will always exist when a number of different people have to agree on a concept as complex as that of music similarity. The fact that in the MIREX AMS task the notion of similarity is not defined very clearly adds to this general problem. After all, "sound similar" does mean something quite different to different people listening to diverse music. As a consequence, an algorithm that has reached this upper bound of performance already in 2009 has not been outperformed ever since. Following our argumentation, this algorithm cannot be outperformed since any additional performance will be lost in the variance of the different human graders.

We now like to discuss a number of recommendations for future editions of the AMS task. One possibility is to go back to the procedure of AMS 2006 and again have more than one grader rate the same query/candidate pairs. This would allow to also always quantify the degree of inter-rater agreement and obtain upper bounds specific to the respective test songs. As we have argued above, we believe that the upper bound we obtained for AMS 2006 is valid for all AMS tasks. Therefore obtaining specific upper bounds would make much more sense if future AMS tasks would use an entirely different database of music. Such a change of song material would be a healthy choice in any case. Re-introducing multiple ratings per query/candidate pair would of course multiply the work load and effort if the number of song pairs to be evaluated should stay the same. However, using so-called "minimal test collections"-algorithms allows to obtain accurate estimates on much reduced numbers of query/candidate pairs as has already been demonstrated for the AMS task [18]. In addition rater-specific normalization should be explored. While some human graders use the full range of available FINE scores when grading similarity of song pairs, others might e.g. never rate song pairs as being very similar or not similar at all, thereby staying away from the extremes of the scale. Such differences in rating style could add even more variance to the overall task and should therefore be taken care of via normalization.

However, all this would still not change the fundamental problem that the concept of music similarity is formulated in such a diffuse way that high inter-rater agreement cannot be expected. Therefore, it is probably necessary to research what the concept of music similarity actually means to human listeners. Such an exploration of what perceptual qualities are relevant to human listeners has already been conducted in the MIR community for the specific case of textural sounds [7]. Textural sounds are sounds that appear stationary as opposed to evolving over time and are therefore much simpler and constrained than real songs. By conducting mixed qualitative-quantitative interviews the authors were able to show that qualities like "high-low", "smooth-coarse" or "tonal-noisy" are important to humans discerning textural sounds. A similar approach could be explored for real song material, probably starting with a limited subset of genres. After such perceptual qualities have then been identified, future AMS tasks could ask human graders how similar pairs of songs are according to a specific quality of the music. Such qualities might not necessarily be straight forward musical concepts like melody, rhythm, or tempo, but rather more abstract notions like instrumentation, genre or specific recording effects signifying a certain style. Such a more fine-grained approach to music similarity would hopefully raise inter-rater agreement and make more room for improvements in modeling music similarity.

Last but not least it has been noted repeatedly that evaluation of abstract music similarity detached from a specific user scenario and corresponding user needs might not be meaningful at all [13]. Instead the MIR community might have to change to evaluation of complete music retrieval systems, thereby opening a whole new chapter for MIR research. Such an evaluation of a complete real life MIR system could center around a specific task for the users (e.g. building a playlist or finding specific music) thereby making the goal of the evaluation much clearer. Incidentally, this has already been named as one of the grand challenges for future MIR research [15]. And even more importantly, exactly such a user centered evaluation will happen at this year’s tenth MIREX anniversary: the "MIREX Grand Challenge 2014: User Experience (GC14UX)" \(^4\). The task for participating teams is to create a web-based interface that supports users looking for background music for a short video. Systems will be rated by human evaluators on a number of important criteria with respect to user experience.

6. CONCLUSION

In our paper we have raised the important issue of the lack of inter-rater agreement in human evaluation of music information retrieval systems. Since human appraisal of phenomena as complex and multi-dimensional as music similarity
ilarity is highly subjective and depends on many factors such as personal preferences and past experiences, evaluation based on human judgments naturally shows high variance across subjects. This lack of inter-rater agreement presents a natural upper bound for performance of automatic analysis systems. We have demonstrated and analysed this problem in the context of the MIREX "Audio Music Similarity and Retrieval" task, but any evaluation of MIR systems that is based on ground truth annotated by humans has the same fundamental problem. Other examples from the MIREX campaign include such diverse tasks as "Structural Segmentation", "Symbolic Melodic Similarity" or "Audio Classification", which are all based on human annotations of varying degrees of ambiguity. Future research should explore upper bounds of performance for these many other MIR tasks based on human annotated data.

7. ACKNOWLEDGEMENTS

We would like to thank all the spiffy people who have made the MIREX evaluation campaign possible over the last ten years, including of course J. Stephen Downie and his people at IMIRSEL. This work was supported by the Austrian Science Fund (FWF, grants P27082 and Z159).

8. REFERENCES


Poster Session 2
EMOTIONAL PREDISPOSITION OF MUSICAL INSTRUMENT TIMBRES WITH STATIC SPECTRA

Bin Wu
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Hong Kong
bwuaa@cse.ust.hk

Andrew Horner
Department of Computer Science and Engineering
Hong Kong University of Science and Technology
Hong Kong
horner@cse.ust.hk

Chung Lee
The Information Systems Technology and Design Pillar
Singapore University of Technology and Design
20 Dover Drive
Singapore 138682
im.lee.chung@gmail.com

ABSTRACT

Music is one of the strongest triggers of emotions. Recent studies have shown strong emotional predispositions for musical instrument timbres. They have also shown significant correlations between spectral centroid and many emotions. Our recent study on spectral centroid-equalized tones further suggested that the even/odd harmonic ratio is a salient timbral feature after attack time and brightness. The emergence of the even/odd harmonic ratio motivated us to go a step further: to see whether the spectral shape of musical instruments alone can have a strong emotional predisposition. To address this issue, we conducted follow-up listening tests of static tones. The results showed that the even/odd harmonic ratio again significantly correlated with most emotions, consistent with the theory that static spectral shapes have a strong emotional predisposition.

1. INTRODUCTION

Music is one of the most effective media for conveying emotion. A lot of work has been done on emotion recognition in music, especially addressing melody [4], harmony [18], rhythm [23, 25], lyrics [15], and localization cues [11].

Some recent studies have shown that emotion is also closely related to timbre. Scherer and Oshinsky found that timbre is a salient factor in the rating of synthetic tones [24]. Peretz et al. showed that timbre speeds up discrimination of emotion categories [22]. Bigand et al. reported similar results in their study of emotion similarities between one-second musical excerpts [7]. It was also found that timbre is essential to musical genre recognition and discrimination [3, 5, 27].

Even more relevant to the current study, Eerola carried out listening tests to investigate the correlation of emotion with temporal and spectral sound features [10]. The study confirmed strong correlations between features such as attack time and brightness and the emotion dimensions valence and arousal for one-second isolated instrument tones. Valence and arousal are measures of how pleasant and energetic the music sounds [31]. Asutay et al. also studied valence and arousal responses to 18 environmental sounds [2]. Despite the widespread use of valence and arousal in music research, composers may find them rather vague and difficult to interpret for composition and arrangement, and limited in emotional nuance. Using a different approach than Eerola, Ellermeier et al. investigated the unpleasantness of environmental sounds using paired comparisons [12].

Recently, we investigated the correlations between emotion and timbral features [30]. In our previous study, listening test subjects compared tones in terms of emotion categories such as Happy and Sad. We equalized the stimuli attacks and decays so that temporal features would not be factors. This modification isolated the effects of spectral features such as spectral centroid. Average spectral centroid significantly correlated for all emotions, and spectral centroid deviation significantly correlated for all emotions. This correlation was even stronger than average spectral centroid for most emotions. The only other correlation was spectral incoherence for a few emotions.

However, since average spectral centroid and spectral centroid deviation were so strong, listeners did not notice other spectral features much. This raised the question: if average spectral centroid was equalized in the tones, would spectral incoherence be more significant? Would other spectral characteristics emerge as significant? We tested this idea on spectral centroid normalized tones, and found that even/odd harmonic ratio was significant. This made us even more curious: if musical instruments tones only differed from one another in their spectral shapes, would they still have strong emotional predispositions? To answer this question, we conducted the follow-up experiment described in this paper using emotion responses for static spectra tones.
2. LISTENING TEST

In our listening test, listeners compared pairs of eight instruments for eight emotions, using tones that were equalized for attack, decay, and spectral centroid.

2.1 Stimuli

2.1.1 Prototype instrument sounds

The stimuli consisted of eight sustained wind and bowed string instrument tones: bassoon (Bs), clarinet (Cl), flute (Fl), horn (Hn), oboe (Ob), saxophone (Sx), trumpet (Tp), and violin (Vn). They were obtained from the McGill and Prosonus sample libraries, except for the trumpet, which had been recorded at the University of Illinois at Urbana-Champaign School of Music. The original of all these tones were used in a discrimination test carried out by Horner et al. [14], six of them were also used by McAdams et al. [20], and all of them used in our emotion-timbre test [30].

The tones were presented in their entirety. The tones were nearly harmonic and had fundamental frequencies close to 311.1 Hz (Eb4). The original fundamental frequencies deviated by up to 1 Hz (6 cents), and were synthesized by additive synthesis at 311.1 Hz.

Since loudness is potential factor in emotion, amplitude multipliers were determined by the Moore-Glasberg loudness program [21] to equalize loudness. Starting from a value of 1.0, an iterative procedure adjusted an amplitude multiplier until a standard loudness of 87.3 \pm 0.1 phons was achieved.

2.2 Stimuli Analysis and Synthesis

2.2.1 Spectral Analysis Method

Instrument tones were analyzed using a phase-vocoder algorithm, which is different from most in that bin frequencies are aligned with the signal’s harmonics (to obtain accurate harmonic amplitudes and optimize time resolution) [6]. The analysis method yields frequency deviations between harmonics of the analysis frequency and the corresponding frequencies of the input signal. The deviations are approximately harmonic relative to the fundamental and within \pm 2% of the corresponding harmonics of the analysis frequency. More details on the analysis process are given by Beauchamp [6].

2.2.2 Spectral Centroid Equalization

Different from our previous study [30], the average spectral centroid of the stimuli was equalized for all eight instruments. The spectra of each instrument was modified to an average spectral centroid of 3.7, which was the mean average spectral centroid of the eight tones. This modification was accomplished by scaling each harmonic amplitude by its harmonic number raised to a to-be-determined power:

\[ A_k(t) \leftarrow k^p A_k(t) \quad (1) \]

For each tone, starting with \( p = 0 \), \( p \) was iterated using Newton’s method until an average spectral centroid was obtained within \pm 0.1 of the 3.7 target value.

2.2.3 Static Tone Preparation

The static tones were 0.5s in duration and were generated using the average steady-state spectrum of each spectral centroid equalized tone with linear 0.05s attacks and decays, and 0.4 sustains.

2.2.4 Resynthesis Method

Stimuli were resynthesized from the time-varying harmonic data using the well-known method of time-varying additive sinewave synthesis (oscillator method) [6] with frequency deviations set to zero.

2.3 Subjects

32 subjects without hearing problems were hired to take the listening test. They were undergraduate students and ranged in age from 19 to 24. Half of them had music training (that is, at least five years of practice on an instrument).

2.4 Emotion Categories

As in our previous study [30], the subjects compared the stimuli in terms of eight emotion categories: Happy, Sad, Heroic, Scary, Comic, Shy, Joyful, and Depressed.

2.5 Listening Test Design

Every subject made pairwise comparisons of all eight instruments. During each trial, subjects heard a pair of tones from different instruments and were prompted to choose which tone more strongly aroused a given emotion. Each combination of two different instruments was presented in four trials for each emotion, and the listening test totaled \( C_2^8 \times 4 \times 8 = 896 \) trials. For each emotion, the overall trial presentation order was randomized (i.e., all the Happy comparisons were first in a random order, then all the Sad comparisons were second, ...).

Before the first trial, the subjects read online definitions of the emotion categories from the Cambridge Academic Content Dictionary [1]. The listening test took about 1.5 hours, with breaks every 30 minutes.

The subjects were seated in a “quiet room” with less than 40 dB SPL background noise level. Residual noise was mostly due to computers and air conditioning. The noise level was further reduced with headphones. Sound signals were converted to analog by a Sound Blaster X-Fi Xtreme Audio sound card, and then presented through Sony MDR-7506 headphones at a level of approximately 78 dB SPL, measured with a sound-level meter. The Sound Blaster DAC utilized 24 bits with a maximum sampling rate of 96 kHz and a 108 dB S/N ratio.
3. RESULTS

3.1 Quality of Responses

The subjects’ responses were first screened for inconsistencies, and two outliers were filtered out. Consistency was defined based on the four comparisons of a pair of instruments A and B for a particular emotion the same with our previous work [30]:

$$\text{consistency}_{A,B} = \frac{\max(v_A, v_B)}{4}$$  

(2)

where $v_A$ and $v_B$ are the number of votes a subject gave to each of the two instruments. A consistency of 1 represents perfect consistency, whereas 0.5 represents approximately random guessing. The mean average consistency of all subjects was 0.74. Also, as in our previous work [30], we found that the two least consistent subjects had the highest outlier coefficients using White et al.’s method [28]. Therefore, they were excluded from the results.

We measured the level of agreement among the remaining 30 subjects with an overall Fleiss’ Kappa statistic [16]. Fleiss’ Kappa was 0.026, indicating a slight but statistically significant agreement among subjects. From this, we observed that subjects were self-consistent but less agreed in their responses than our previous study [30], since spectrally significant agreement among subjects. Therefore, they were excluded from the results.

We also performed a $\chi^2$ test [29] to evaluate whether the number of circular triads significantly deviated from the number to be expected by chance alone. This turned out to be insignificant for all subjects. The approximate likelihood ratio test [29] for significance of weak stochastic transitivity violations [26] was tested and showed no significance for all emotions.

3.1.1 Emotion Results

Same with our previous work, we ranked the spectral centroid equalized instrument tones by the number of positive votes they received for each emotion, and derived scale values using the Bradley-Terry-Luce (BTL) model [8, 29] as shown in Figure 1. The likelihood-ratio test showed that the BTL model describes the paired-comparisons well for all emotions. We observe that: 1) The distribution of emotion ratings were much narrower than the original tones in our previous study [30]. The reason is that spectral shape was the only factor that could possibly affect emotion, which made it more difficult for subjects to distinguish. 2) Opposite of our previous study [30], the horn evoked positive emotions. It was ranked as the least Shy and Depressed, and among the most Heroic and Comic. 3) The clarinet and the saxophone were contrasting outliers for all emotions (except Scary).

Figure 2 shows BTL scale values and the corresponding 95% confidence intervals of the instruments for each emotion. The confidence intervals cluster near the line of indifference since it was difficult for listeners to make emotional distinctions. Table 1 shows the spectral characteristics of the static tones (time-domain spectral characteristics are omitted since the tones are static). With all time-domain spectral characteristics removed, spectral shape features such as even/odd harmonic ratio became more salient. Specifically, even/odd ratio was calculated according to Caclin et al.’s method [9]. Pearson correlation between emotion and spectral characteristics are shown in Table 2. Both spectral irregularity and even/odd harmonic ratio are measures of spectral jaggedness, where even/odd harmonic ratio measures a particular, extreme type of spectral irregularity that is typical of the clarinet. In Table 2, even/odd harmonic ratio significantly correlated with nearly all emotions. The correlations were much stronger than in the original tones [30], and indicate that spectral shape by itself can arouse strong emotional responses.

4. DISCUSSION

These results are consistent with our previous results [30] and Eerola’s Valence-Arousal results [10]. All these studies indicate that musical instrument timbres carry cues about emotional expression that are easily and consistently recognized by listeners. They show that spectral centroid/brightness is a significant component in music emotion. Beyond Eerola’s and our previous findings, we have found that spectral shape by itself can have strong emotional predispositions, and even/odd harmonic ratio is the most salient timbral feature after attack time and brightness in static tones.

In hindsight, perhaps it is not so surprising that static spectra tones have emotional predispositions just as dynamic musical instrument tones do. It is somewhat analogous to viewers’ emotional dispositions to primary colors [13, 17, 19]. Of course, just because static tones have emotional predispositions, it does not mean they are interesting to listen to. The dynamic spectra of real acoustic instruments are much more natural and life-like than any static tones, regardless of emotional predisposition. This is reflected in the wider range of emotion rankings of the original dynamic tones compared to the static tones.

For future work, it will be fascinating to see how emotion varies with pitch, dynamic level, brightness, articulation, and cultural backgrounds.

5. ACKNOWLEDGMENT

This work has been supported by Hong Kong Research Grants Council grants HKUST613112.

6. REFERENCES


Figure 1. Bradley-Terry-Luce scale values of the static tones for each emotion.


Figure 2. BTL scale values and the corresponding 95% confidence intervals of the static tones for each emotion. The dotted line represents no preference.

Table 1. Spectral characteristics of the static instrument tones.

<table>
<thead>
<tr>
<th>Features</th>
<th>Bs</th>
<th>Cl</th>
<th>Fl</th>
<th>Hn</th>
<th>Ob</th>
<th>Sx</th>
<th>Tp</th>
<th>Vn</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spectral Irregularity</td>
<td>0.0971</td>
<td>0.1818</td>
<td>0.143</td>
<td>0.0645</td>
<td>0.119</td>
<td>0.1959</td>
<td>0.0188</td>
<td>0.1176</td>
</tr>
<tr>
<td>Even/odd Ratio</td>
<td>1.2565</td>
<td>0.1775</td>
<td>0.9493</td>
<td>0.9694</td>
<td>0.4308</td>
<td>1.7719</td>
<td>0.7496</td>
<td>0.8771</td>
</tr>
</tbody>
</table>

Table 2. Pearson correlation between emotion and spectral characteristics for static tones. **: p < 0.05; *: 0.05 < p < 0.1.

<table>
<thead>
<tr>
<th>Features</th>
<th>Happy</th>
<th>Sad</th>
<th>Heroic</th>
<th>Scary</th>
<th>Comic</th>
<th>Shy</th>
<th>Joyful</th>
<th>Depressed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spectral Irregularity</td>
<td>-0.1467</td>
<td>0.1827</td>
<td>-0.4859</td>
<td>-0.0897</td>
<td>-0.3216</td>
<td>0.1565</td>
<td>-0.509</td>
<td>0.3536</td>
</tr>
<tr>
<td>Even/odd Ratio</td>
<td><strong>0.8901</strong></td>
<td><em>-0.8441</em>*</td>
<td><strong>0.7468</strong></td>
<td>*-0.3398</td>
<td><strong>0.8017</strong></td>
<td><strong>-0.7942</strong></td>
<td><strong>0.6524</strong></td>
<td><strong>-0.7948</strong></td>
</tr>
</tbody>
</table>
PANAKO - A SCALABLE ACOUSTIC FINGERPRINTING SYSTEM HANDLING TIME-SCALE AND PITCH MODIFICATION

Six Joren, Marc Leman
Institute for Psychoacoustics and Electronic Music (IPEM), Department of Musicology, Ghent University
Ghent, Belgium
joren.six@ugent.be

ABSTRACT

This paper presents a scalable granular acoustic fingerprinting system. An acoustic fingerprinting system uses condensed representation of audio signals, acoustic fingerprints, to identify short audio fragments in large audio databases. A robust fingerprinting system generates similar fingerprints for perceptually similar audio signals. The system presented here is designed to handle time-scale and pitch modifications. The open source implementation of the system is called Panako and is evaluated on commodity hardware using a freely available reference database with fingerprints of over 30,000 songs. The results show that the system responds quickly and reliably on queries, while handling time-scale and pitch modifications of up to ten percent.

The system is also shown to handle GSM-compression, several audio effects and band-pass filtering. After a query, the system returns the start time in the reference audio and how much the query has been pitch-shifted or time-stretched with respect to the reference audio. The design of the system that offers this combination of features is the main contribution of this paper.

1. INTRODUCTION

The ability to identify a small piece of audio by comparing it with a large reference audio database has many practical use cases. This is generally known as audio fingerprinting or acoustic fingerprinting. An acoustic fingerprint is a condensed representation of an audio signal that can be used to reliably identify identical, or recognize similar, audio signals in a large set of reference audio. The general process of an acoustic fingerprinting system is depicted in Figure 1. Ideally, a fingerprinting system only needs a short audio fragment to find a match in large set of reference audio. One of the challenges is to design a system in a way that the reference database can grow to contain millions of entries. Another challenge is that a robust fingerprinting should handle noise and other modifications well, while limiting the amount of false positives and processing time [5]. These modifications typically include dynamic range compression, equalization, added background noise and artifacts introduced by audio coders or A/D-D/A conversions.

Over the years several efficient acoustic fingerprinting methods have been introduced [1, 6, 8, 13]. These methods perform well, even with degraded audio quality and with industrial sized reference databases. However, these systems are not designed to handle queries with modified time-scale or pitch although these distortions can be present in replayed material. Changes in replay speed can occur either by accident during an analog to digital conversion or they are introduced deliberately.

Accidental replay speed changes can occur when working with physical, analogue media. Large music archive often consist of wax cylinders, magnetic tapes and gramophone records. These media are sometimes digitized using an incorrect or varying playback speed. Even when calibrated mechanical devices are used in a digitization process, the media could already have been recorded at an undesirable or undocumented speed. A fingerprinting system should therefore allow changes in replay speed to correctly detect duplicates in such music archives.

Deliberate time-scale manipulations are sometimes introduced as well. During radio broadcasts, for example, songs are occasionally played a bit faster to make them fit into a time slot. During a DJ-set pitch-shifting and time-stretching are present almost continuously. To correctly identify audio in these cases as well, a fingerprinting system robust against pitch-shifting and time-stretching is desired.

Some fingerprinting systems have been developed that take pitch-shifts into account [3, 7, 11] without allowing time-scale modification. Others are designed to handle both pitch and time-scale modification [10, 14]. The system by Zhu et al [14] employs an image processing algorithm on an auditory image to counter time-scale modification and pitch-shifts. Unfortunately, the system is computationally expensive, it iterates the whole database to find a match. The system by Malekesmaeili et al [10] allows extreme pitch-shifting and time-stretching, but has the same problem. To the best of our knowledge, a description of a practical acoustic fingerprinting system that allows sub-
substantial pitch-shift and time-scale modification is nowhere to be found in the literature. This description is the main contribution of this paper.

2. METHOD

The proposed method is inspired by three works. Combining key components of those works results in a design of a granular acoustic fingerprinter that is robust to noise and substantial compression, has a scalable method for fingerprint storage and matching, and allows time-scale modification and pitch-shifting.

Firstly, the method used by Wang [13] establishes that local maxima in a time-frequency representation can be used to construct fingerprints that are robust to quantization effects, filtering, noise and substantial compression. The described exact-hashing method for storing and matching fingerprints has proven to be very scalable. Secondly, Artz et al. [2] describe a method to align performances and scores. Especially interesting is the way how triplets of events are used to search for performances with different timings. Thirdly, the method by Fenet et al. [7] introduces the idea to extract fingerprints from a Constant-Q [4] transform, a time-frequency representation that has a constant amount of bins for every octave. In their system a fingerprint remains constant when a pitch-shift occurs. However, since time is encoded directly within the fingerprint, the method does not allow time-scale modification.

Considering previous works, the method presented here uses local maxima in a spectral representation. It combines three event points, and takes time ratios to form time-scale invariant fingerprints. It leverages the Constant-Q transform, and only stores frequency differences for pitch-shift invariance. The fingerprints are designed with an exact hashing matching algorithm in mind. Below each aspect is detailed.

2.1 Finding Local Maxima

Suppose a time-frequency representation of a signal is provided. To locate the points where energy reaches a local maximum, a tiled two-dimensional peak picking algorithm is applied. First the local maxima for each spectral analysis frame are identified. Next each of the local maxima are iterated and put in the center of a tile with \( \Delta T \times \Delta F \) as dimensions. If the local maximum is also the maximum within the tile it is kept, otherwise it is discarded. Thus, making sure only one point is identified for every tile of \( \Delta T \times \Delta F \). This approach is similar to [7,13]. This results in a list of event points each with a frequency component \( f \), expressed in bins, and a time component \( t \), expressed in time steps. \( \Delta T \) and \( \Delta F \) are chosen so that there are between 24 and 60 event points every second.

A spectral representation of an audio signal has a certain granularity; it is essentially a grid with bins both in time as in frequency. When an audio signal is modified, the energy that was originally located in one single bin can be smeared over two or more bins. This poses a problem, since the goal is to be able to locate event points with maximum energy reliably. To improve reliability, a post processing step is done to refine the location of each event point by taking its energy and mixing it with the energy of the surrounding bins. The same thing is done for the surrounding bins. If a new maximum is found in the surroundings of the initial event point, the event point is relocated accordingly. Effectively, a rectangular blur with a \( 3 \times 3 \) kernel is applied at each event point and its surrounding bins.

Once the event points with local maximum energy are identified, the next step is to combine them to form a fingerprint. A fingerprint consists of three event points, as seen in Figure 2. To construct a fingerprint, each event point is combined with two nearby event points. Each event point can be part of multiple fingerprints. Only be-

![Figure 1: A generalized audio fingerprinter scheme. Audio is fed into the system, features are extracted and fingerprints constructed. The fingerprints are consecutively compared with a database containing the fingerprints of the reference audio. The original audio is either identified or, if no match is found, labeled as unknown.](image1)

![Figure 2: The effect of time-scale and pitch modifications on a fingerprint. It shows a single fingerprint extracted from reference audio ( — ) and the same fingerprint extracted from audio after pitch-shifting ( — ), time-stretching ( — ) and time-scale modification ( — ).](image2)
tween 8 and 20 fingerprints are kept every second. Fingerprints with event points with the least cumulative energy are discarded. Now that a list of fingerprints has been created a method to encode time information in a fingerprint hash is needed.

2.2 Handling Time Stretching: Event Triplets

Figure 2 shows the effect of time stretching on points in the time-frequency domain. There, a fingerprint extracted from reference audio (Fig. 2, 1) is compared with a fingerprint from time stretched audio (Fig. 2, 2). Both fingerprints are constructed using three local maxima \( e_1, e_2, e_3 \) and \( e'_1, e'_2, e'_3 \). While the frequency components stay the same, the time components do change. However, the ratios between the time differences are constant as well. The following equation holds:

\[
\frac{t_2 - t_1}{t_3 - t_1} = \frac{t'_2 - t'_1}{t'_3 - t'_1} \quad (1)
\]

With event point \( e_n \) having a time and frequency component \((t_n, f_n)\) and the corresponding event points \( e'_n \) having the components \((t'_n, f'_n)\). Since \( t_3 - t_1 \geq t_2 - t_1 \), the ratio always resolves to a number in the range \([0, 1]\). This number, scaled and rounded, is a component of the eventual fingerprint hash (an approach similar to [2]).

Now that a way to encode time information, indifferent of time-stretching, has been found, a method to encode frequency, indifferent to pitch-shifting is desired.

2.3 Handling Pitch-Shifts: Constant-Q Transform

Figure 2 shows a comparison between a fingerprint from pitch shifted audio (1) with a fingerprint from reference audio (2). In the time-frequency domain pitch shifting is a vertical translation and time information is preserved. Since every octave has the same number of bins [4] a pitch shift on event \( e_1 \) will have the following effect on its frequency component \( f_1 \), with \( K \) being a constant, \( f'_1 = f_1 + K \). It is clear that the difference between the frequency components remains the same, before and after pitch shifting: \( f_1 - f_2 = (f'_1 + K) - (f'_2 + K) \) [7].

In the proposed system three event points are available, the following information is stored in the fingerprint hash:

\[
\left(f_1 - f_2; f_2 - f_3; f_1; f_3; t_2 - t_1; t_3 - t_1; id\right) \quad (2)
\]

The hash, the first element between brackets, can be packed into a 32bit integer. To save space, \( f_1 \) and \( t_3 - t_1 \) can be combined in one 32bit integer. An integer of 32bit is also used to store \( t_1 \). The reference audio identifier is also a 32bit identifier. A complete fingerprint consists of \( 4 \times 32 \text{bit} = 128 \text{bit} \). At eight fingerprints per second a song of four minutes is reduced to \( 128 \text{bit} \times 8 \times 60 \times 4 = 30 \text{KB} \). An industrial size data set of one million songs translates to a manageable 28GB.

2.5 Fingerprint Hash

A fingerprint with a corresponding hash needs to be constructed carefully to maintain aforementioned properties. The result of a query should report the amount of pitch-shift and time-stretching that occurred. To that end, the absolute value of \( f_1 \) and \( t_3 - t_1 \) is stored, they can be used to compare with \( f'_1 \) and \( t'_3 - t'_1 \) from the query. The time offset at which a match was found should be returned as well, so \( t_1 \) needs to be stored. The complete information to store for each fingerprint is:

\[
\left(f_1 - f_2; f_2 - f_3; f_1; f_3; t_2 - t_1; t_3 - t_1; id\right) \quad (2)
\]

The hash, the first element between brackets, can be packed into a 32bit integer. To save space, \( f_1 \) and \( t_3 - t_1 \) can be combined in one 32bit integer. An integer of 32bit is also used to store \( t_1 \). The reference audio identifier is also a 32bit identifier. A complete fingerprint consists of \( 4 \times 32 \text{bit} = 128 \text{bit} \). At eight fingerprints per second a song of four minutes is reduced to \( 128 \text{bit} \times 8 \times 60 \times 4 = 30 \text{KB} \). An industrial size data set of one million songs translates to a manageable 28GB.

2.6 Matching Algorithm

The matching algorithm is inspired by [13], but is heavily modified to allow time stretched and pitch-shifted matches. It follows the scheme in Figure 1 and has seven steps.

1. Local maxima are extracted from a constant-Q spectrogram from the query. The local maxima are combined by three to form fingerprints, as explained in Sections 2.1, 2.3 and 2.4.

2. For each fingerprint a corresponding hash value is calculated, as explained in Section 2.5.

3. The set of hashes is matched with the hashes stored in the reference database, and each exact match is returned.

4. The matches are iterated while counting how many times each individual audio identifier occurs in the result set.

5. Matches with an audio identifier count lower than a certain threshold are removed, effectively dismissing random chance hits. In practice there is almost always only one item with a lot of matches, the rest being random chance hits. A threshold of three or four suffices.

1 It is assumed that the time stretch factor is constant in the time interval \( t'_2 - t'_1 \). A reasonable assumption since \( t'_2 - t'_1 \) is small.

2 Depending on the storage engine used, storage of fingerprints together with an index of sorts introduces a storage overhead. Since the data to store is small, the index can be relatively large.
6. The residual matches are checked for alignment, both in frequency and time, with the reference fingerprints using the information that is stored along with the hash.

7. A list of audio identifiers is returned ordered by the amount of fingerprints that align both in pitch and frequency.

In step six, frequency alignment is checked by comparing the \( f_1 \) component of the stored reference with \( f'_1 \), the frequency component of the query. If, for each match, the difference between \( f_1 \) and \( f'_1 \) is constant, the matches align.

Alignment in time is checked using the reference time information \( t_1 \) and \( t_3 - t_1 \), and the time information of the corresponding fingerprint extracted from the query fragment \( t'_1, t'_3 - t'_1 \). For each matching fingerprint the time offset \( t_o \) is calculated. The time offset \( t_o \) resolves to the amount of time steps between the beginning of the query and the beginning of the reference audio, even if a time modification took place. It stands to reason that \( t_o \) is constant for matching audio.

\[
t_o = t_1 - t'_1 \times \frac{(t_3 - t_1)}{(t'_3 - t'_1)}
\]  
(3)

The matching algorithm also provides information about the query. The time offset tells at which point in time the query starts in the reference audio. The time difference ratio \( (t_3 - t_1)/(t'_3 - t'_1) \) represents how much time is modified, in percentages. How much the query is pitch-shifted with respect to the reference audio can be deduced from \( f'_1 - f_1 \), in frequency bins. To convert a difference in frequency bins to a percentage the following equation is used, with \( n \) the number of cents per bin, \( e \) Eulers number, and \( \ln \) the natural logarithm: \[e((f'_1-f_1)\times n \times \ln(2))/1200\]

The matching algorithm ensures that random chance hits are very uncommon, the number of false positives can be effectively reduced to zero by setting a threshold on the number of aligned matches. The matching algorithm also provides the query time offset and the percentage of pitch-shift and time-scale modification of the query with respect to the reference audio.

3. RESULTS

To test the system, it was implemented in the Java programming language. The implementation is called Panako and is available under the GNU Affero General Public License on http://panako.be. The DSP is also done in Java using a DSP library [12]. To store and retrieve hashes, Panako uses a key-value store. Kyoto Cabinet, BerkeleyDB, Redis, LevelDB, RocksDB, Voldemort, and MapDB were considered.

MapDB is an implementation of a storage backed B-Tree with efficient concurrent operations [9] and was chosen for its simplicity, performance and good Java integration. Also, the storage overhead introduced when storing fingerprints on disk is minimal. Panako is compared with Audfprint by Dan Ellis, an implementation of a fingerprinter system based on [13].

The test data set consists of freely available music downloaded from Jamendo. A reference database of about 30,000 songs, about 10^6 seconds of audio, was created. From this data set random fragments were selected, with a length of 20, 40 and 60 seconds. Each fragments was modified 54 times. The modifications included: pitch-shifting (−200 to 200 cents in steps of 25 cents), time-stretching (−16% to +16%, in steps of 2%), time-scale modification (−16% to +16%, in steps of 2%), echo, flanger, chorus and a band-pass filter. Another set of fragments were created from audio not present in the reference database, in order to measure the number of correctly unidentified fragments. In total 3 (durations) × 600 (excerpts) × 54 (modifications) = 97,200 fragments were created.

Each fragment is presented to both Panako and Audfprint and the detection results are recorded. The systems are regarded as binary classifiers of which the amount of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN) are counted. During the experiment with Panako no false positives (FP) were detected. Also, all fragments that are not present in the reference database were rejected correctly (TN). So Panako’s specificity is \( TN/(TN + FP) = 100\% \). This can be explained by the design of the matching algorithm. A match is identified as such if a number of hashes, each consisting of three points in a spectrogram, align in time. A random match between hashes is rare, the chances of a random match between consecutively aligned hashes is almost non-existent, resulting in \( 100\% \) specificity.

The sensitivity \( FP/(TP + FN) \) of the system, however, depends on the type of modification on the fragment. Figure 3 shows the results after pitch-shifting. It is clear that the amount of pitch-shift affects the performance, but

![Pitch shift vs. True positive rate](http://panako.be)

The scripts used to generate the queries can be found at the website http://panako.be.

---

3. The effects were applied using SoX, a command line audio editor. The scripts used to generate the queries can be found at the website http://panako.be.
Figure 4: The true positive rate after time-stretching.

Figure 5: True positive rate after time-scale modification in a fluctuating pattern. The effect can be explained by taking into account the Constant-Q bins. Here, a bin spans 33 cents, a shift of $n \times \frac{33}{2}$ cents spreads spectral information over two bins, if $n$ is an odd number. So performance is expected to degrade severely at $\pm 49.5$ cents (3%) and $\pm 148.5$ cents (9%) an effect clearly visible in figure 3. The figure also shows that performance is better if longer fragments are presented to the system. The performance of Audfprint, however, does not recover after pitch-shifts of more than three percent.

Figure 4 shows the results after time stretching. Due to the granularity of the time bins, and considering that the step size stays the same for each query type, time modifications have a negative effect on the performance. Still, a more than a third of the queries is resolved correctly after a time stretching modification of 8%. Performance improves with the length of a fragment. Surprisingly, Audfprint is rather robust against time-stretching, thanks to the way time is encoded into a fingerprint.

Figure 5 shows the results after time-scale modification. The performance decreases severely above eight percent. The figure shows that there is some improvement when comparing the results of 20s fragments to 40s fragments, but going from 40s to 60s does not change much. Audfprint is unable to cope with time-scale modification due to the changes in both frequency and time.

In Figure 6, the results for other modifications like echo, chorus, flanger, tremolo, and a band pass filter can be seen. The parameters of each effect are chosen to represent typical use, but on the heavy side. For example the echo effect applied has a delay line of 0.5 seconds and a decay of 30%. The system has the most problems with the chorus effect. Chorus has a blurring effect on a spectrogram, which makes it hard for the system to find matches. Still it can be said that the algorithm is rather robust against very present, clearly audible, commonly used audio effects. The result of the band pass filter with a center of 2000Hz is especially good. To test the systems robustness to severe audio compression a test was executed with GSM-compressed queries. The performance on 20s fragments is about 30% but improves a lot with query length, the 60s fragment yields 65%. The results for Audfprint show that there is room for improvement for the performance of Panako.

A practical fingerprinting system performs well, in terms of speed, on commodity hardware. With Panako extracting and storing fingerprints for 25s of audio is done in one second using a single core of a dated processor. The test data set was constructed in $30,000 \times 4 \times 60s / 25 = 80$ processor hours. Since four cores were used, it took less than a full day. After the feature extraction, matching a 40s query with the test database with 30,000 songs is done within 75ms. The complete matching process for a 40s fragment takes about one second. Monitoring multiple streams in real-time poses no problem for the system. Building a fingerprint dataset with Audfprint is faster since fingerprints are extracted from an FFT which is less demanding than a Constant-Q transform. The matching step performance, however, is comparable.

5 The testing machine has an Intel Core2 Quad CPU Q9650 @ 3.00GHz introduced in 2009. The processor has four cores.
Failure analysis shows that the system does not perform well on music with spectrograms either with very little energy or energy evenly spread across the range. Also extremely repetitive music, with a spectrogram similar to a series of dirac impulses, is problematic. Also, performance drops when time modifications of more than 8% are present. This could be partially alleviated by redesigning the time parameters used in the fingerprint hash, but this would reduce the discriminative power of the hash.

4. CONCLUSION

In this paper a practical acoustic fingerprinting system was presented. The system allows fast and reliable identification of small audio fragments in a large set of audio, even when the fragment has been pitch-shifted and time-stretched with respect to the reference audio. If a match is found the system reports where in the reference audio a query matches, and how much time/frequency has been modified. To achieve this, the system uses local maxima in a Constant-Q spectrogram. It combines event points into groups of three, and uses time ratios to form a time-scale invariant fingerprint component. To form pitch-shift invariant fingerprint components only frequency differences are stored. For retrieval an exact hashing matching algorithm is used.

The system has been evaluated using a freely available data set of 30,000 songs and compared with a baseline system. The results can be reproduced entirely using this data set and the open source implementation of Panako. The scripts to run the experiment are available as well. The results show that the system’s performance decreases with time-scale modification of more than eight percent. The system is shown to cope with pitch-shifting, time-stretching, severe compression, and other modifications as echo, flanger and band pass.

To improve the system further the constant-Q transform could be replaced by an efficient implementation of the non-stationary Gabor transform. This is expected to improve the extraction of event points and fingerprints without effecting performance. Panako could also benefit from a more extensive evaluation and detailed comparison with other techniques. An analysis of the minimum, most discriminative, information needed for retrieval purposes could be especially interesting.

5. REFERENCES

PERCEPTUAL ANALYSIS OF THE F-MEASURE FOR EVALUATING SECTION BOUNDARIES IN MUSIC

Oriol Nieto\textsuperscript{1}, Morwaread M. Farbood\textsuperscript{1}, Tristan Jehan\textsuperscript{2}, and Juan Pablo Bello\textsuperscript{1}

\textsuperscript{1}Music and Audio Research Lab, New York University, \{oriol, mfarbood, jpbello\}@nyu.edu
\textsuperscript{2}The Echo Nest, tristan@echonest.com

ABSTRACT

In this paper, we aim to raise awareness of the limitations of the F-measure when evaluating the quality of the boundaries found in the automatic segmentation of music. We present and discuss the results of various experiments where subjects listened to different musical excerpts containing boundary indications and had to rate the quality of the boundaries. These boundaries were carefully generated from state-of-the-art segmentation algorithms as well as human-annotated data. The results show that humans tend to give more relevance to the precision component of the F-measure rather than the recall component, therefore making the classical F-measure not as perceptually informative as currently assumed. Based on the results of the experiments, we discuss the potential of an alternative evaluation based on the F-measure that emphasizes precision over recall, making the section boundary evaluation more expressive and reliable.

1. INTRODUCTION

Over the past decade, significant effort has been made toward developing methods that automatically extract large-scale structures in music. In this paper, we use the term musical structure analysis to refer to the task that identifies the different sections (or segments) of a piece. In Western popular music, these sections are commonly labeled as verse, chorus, bridge, etc. Given that we now have access to vast music collections, this type of automated analysis can be highly beneficial for organizing and exploring these collections.

Musical structure analysis is usually divided into two subtasks: the identification of section boundaries and the labeling of these sections based on their similarity. Here, we will only focus on the former. Section boundaries usually occur when salient changes in various musical qualities (such as harmony, timbre, rhythm, or tempo) take place. See [9] for a review of some of the state of the art in musical structure analysis.

Typically, researchers make use of various human-annotated datasets to measure the accuracy of their analysis algorithms. The standard methodology for evaluating the accuracy of estimated section boundaries is to compare those estimations with ground truth data by means of the F-measure (also referred to as the hit rate), which gives equal weight to the values of precision (proportion of the boundaries found that are correct) and recall (proportion of correct boundaries that are located). However, it is not entirely clear that humans perceive the type of errors those two metrics favor or the penalties they impose as equally important, calling into question the perceptual relevance of the F-measure for evaluating long-term segmentation. To the best of our knowledge, no empirical evidence or formal study exists that can address such a question in the context of section boundary identification. This work is an effort to redress that.

Our work is motivated by a preliminary study we ran on two subjects showing a preference for high precision results, thus making us reconsider the relevance of precision and recall for the evaluation of section boundary estimations. As a result, in this paper we present two additional experiments aimed at validating and expanding those preliminary findings including a larger subject population and more controlled conditions. In our experiments, we focus on the analysis of Western popular songs since this is the type of data most segmentation algorithms in the MIR literature operate on, and since previous studies have shown that most listeners can confidently identify structure in this type of music [1].

The rest of this paper is organized as follows. We present a review of the F-measure and a discussion of the preliminary study in section 2. We describe the design of two experiments along with discussions of their results in sections 3 and 4. We explore an alternative F-measure based on our experimental findings that could yield more expressive and perceptually relevant outcomes in section 5. Finally, we draw conclusions and discuss future work in section 6.

2. THE F-MEASURE FOR MUSIC BOUNDARIES

2.1 Review of the F-measure

In order to evaluate automatically computed music boundaries, we have to define how we accept or reject an estimated boundary given a set of annotated ones (i.e., find the intersection between these two sets). Traditionally, re-
searchers consider an estimated boundary correct as long as its maximum deviation to its closest annotated boundary is $\pm 3$ seconds [8] (in MIREX,[1] inspired by [16], an evaluation that uses a shorter window of $\pm 0.5$ seconds is also performed). Following this convention, we use a $\pm 3$-second window in our evaluation.

Let us assume that we have a set of correctly estimated boundaries given the annotated ones (hits), a set of annotated boundaries that are not estimated (false negatives), and a set of estimated boundaries that are not in the annotated dataset (false positives). Precision is the ratio between hits and the total number of estimated elements (e.g., we could have 100% precision with an algorithm that only returns exactly one boundary and this boundary is correct). Recall is the ratio between hits and the total number of annotated elements (e.g. we could have a 100% recall with an algorithm that returns one boundary every 3 seconds, since all the annotated boundaries will be sufficiently close to an estimated one). Precision and recall are defined formally as

$$
    P = \frac{|\text{hits}|}{|\text{bounds}_e|}; \quad R = \frac{|\text{hits}|}{|\text{bounds}_a|}
$$

where $|\cdot|$ represents the cardinality of the set $\cdot$, bounds$_e$ is the set of estimated boundaries and bounds$_a$ is the set of annotated ones. Finally, the F-measure is the harmonic mean between $P$ and $R$, which weights these two values equally, penalizes small outliers, and mitigates the impact of large ones:

$$
    F = \frac{2 \cdot P \cdot R}{P + R}
$$

When listening to the output of music segmentation algorithms, it is immediately apparent that false negatives and false positives are perceptually very different (an initial discussion about assessing a synthetic precision of 100% when evaluating boundaries can be found in [14]). Thus, in the process of developing novel methods for structure segmentation, we decided to informally assess the relative effect that different types of errors had on human evaluations of the accuracy of the algorithms’ outputs. The following section describes the resulting preliminary study.

### 2.2 Preliminary Study

For this study we compared three algorithms, which we will term $A$, $B$ and $C$. $A$ is an unpublished algorithm currently in development that relies on homogeneous repeated section blocks; $B$ is an existing algorithm that uses novelty in audio features to identify boundaries; and $C$ combines the previous two methods. All three methods were optimized to maximize their F-measure performance on the structure-annotated Levy dataset [5]. Table 1 shows each method’s average F-measure, precision, and recall values across the entire set. Note how $C$ maximizes the F-measure, mostly by increasing recall, while $A$ shows maximum precision.

We asked two college music majors to rank the three algorithms for every track in the Levy set. The goal was not to compare the results of the algorithms to the annotated ground truth, but to compare the algorithms with each other and determine the best one from a perceptual point of view. The participants were asked to listen to each of the algorithm outputs for all the songs and rank the algorithms by the quality of their estimated section boundaries; no particular constraints were given on what to look for. We used Sonic Visualiser [3] to display the waveform and three section panels for each of the algorithms in parallel (see Figure 1). While playing the audio, listeners could both see the sections and hear the boundaries indicated by a distinctive percussive sound. The section panels were organized at random for each song so listeners could not easily tell which algorithm they were choosing.

![Figure 1. Screenshot of Sonic Visualiser used in the preliminary experiment. The song is “Smells Like Teen Spirit” by Nirvana. In this case, algorithms are ordered as $A$, $B$, and $C$ from top to bottom.](image)

Analysis of the results showed that 68.3% of the time, the two participants chose the same best algorithm. In 23.3% of the cases, they disagreed on the best, and in just 8.3% of the cases, they chose opposite rankings. When they actually agreed on the best algorithm, they chose $A$ 58.5% of the time. $A$ did not have the highest F-measure but the highest precision. Perhaps more surprising, they chose $C$ only 14.6% of the time even though that algorithm had the highest F-measure.

These results raised the following questions: Is the F-measure informative enough to evaluate the accuracy of automatically estimated boundaries in a perceptually-meaningful way? Is precision more important than recall when assessing music boundaries? Would the observed trends remain when tested on a larger population of subjects? Can these results inform more meaningful evaluation measures? We decided to address these questions by running two more formal experiments in order to better understand this apparent problem and identify a feasible solution.

### Table 1

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$F$</th>
<th>$P$</th>
<th>$R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>49%</td>
<td>57%</td>
<td>47%</td>
</tr>
<tr>
<td>$B$</td>
<td>44%</td>
<td>46%</td>
<td>46%</td>
</tr>
<tr>
<td>$C$</td>
<td>51%</td>
<td>47%</td>
<td>64%</td>
</tr>
</tbody>
</table>

Table 1. Algorithms and their ratings used to generate the input for the preliminary study. These ratings are averaged across the 60 songs of the Levy dataset.
3. EXPERIMENT 1: RATING BOUNDARIES

3.1 Motivation

The results of the preliminary study suggested that precision is more relevant than recall when perceiving boundaries. However, to fully explore this hypothesis, these two values had to be carefully manipulated. For this experiment, a set of boundaries was synthesized by setting specific values for precision and recall while maintaining a near-constant F-measure. Moreover, we wanted to ensure that the findings were robust across a larger pool of subjects. With these considerations in mind, the experiment was designed to be both shorter in time and available online.

3.2 Methodology

We selected five track excerpts from the Levy catalog by finding the one-minute segments containing the highest number of boundaries across the 60 songs of the dataset. By having short excerpts instead of full songs, we could reduce the duration of the entire experiment with negligible effect on the results—past studies have shown that boundaries are usually perceived locally instead of globally [15]. We decided to use only five excerpts with the highest number of boundaries in order to maintain participants’ attention as much as possible. For each track excerpt, we synthesized three different segmentations: ground truth boundaries (GT) with an F-measure of 100%; high precision (HP) boundaries with a precision of 100% and recall of around 65%; and high recall (HR) boundaries with a recall of 100% and precision of around 65%. The extra boundaries for the HR version were randomly distributed (using a normal distribution) across a 3 sec window between the largest regions between boundaries. For the HP version, the boundaries that were most closely spaced were removed. Table 2 presents F-measure, precision, and recall values for the five tracks along with the average values across excerpts. Note the closeness between F-measure values for HP and HR.

A total number of 48 participants took part in the experiment; subjects had an average of 3.1 ± 1.6 years of musical training and 3.7 ± 3.3 years of experience playing an instrument.

3.3 Results and Discussion

Box plots of accuracy ratings across versions can be seen in Figure 2. These experimental results show that higher accuracy ratings were assigned to GT followed by HP, and then HR.

A two-way, repeated-measures ANOVA was performed on the accuracy ratings with type (ground truth, high precision, high recall) and excerpt (the five songs) as factors. There were 48 data points in each Type × Excerpt category. The main effects of type, F(2, 94) = 90.74, MSE = 1.10, p < .001, and excerpt, F(4, 188) = 59.84, MSE = 0.88, p < .001, were significant. There was also an interaction effect, F(6, 290) = 9.42, MSE = 0.74, p < .001 (Greenhouse-Geisser corrected), indicating that rating profiles differed based on excerpt. Mean ratings by type and excerpt are shown in Figure 3.

Looking at the data for each excerpt, there was a clear pattern showing that subjects preferred segmentations with

Table 2. Excerpt list with their evaluations for experiment 1. The F-measure of GT is 100% (not shown in the table).

<table>
<thead>
<tr>
<th>Song Name (Artist)</th>
<th>HP F</th>
<th>HP P</th>
<th>HP R</th>
<th>HR F</th>
<th>HR P</th>
<th>HR R</th>
</tr>
</thead>
<tbody>
<tr>
<td>Black &amp; White (Michael Jackson)</td>
<td>.809</td>
<td>1</td>
<td>.68</td>
<td>.794</td>
<td>.658</td>
<td>1</td>
</tr>
<tr>
<td>Drive (R.E.M.)</td>
<td>.785</td>
<td>1</td>
<td>.647</td>
<td>.791</td>
<td>.654</td>
<td>1</td>
</tr>
<tr>
<td>Intergalactic (Beastie Boys)</td>
<td>.764</td>
<td>1</td>
<td>.619</td>
<td>.792</td>
<td>.656</td>
<td>1</td>
</tr>
<tr>
<td>Suds And Soda (Deus)</td>
<td>.782</td>
<td>1</td>
<td>.653</td>
<td>8</td>
<td>.666</td>
<td>1</td>
</tr>
<tr>
<td>Tubthumping (Chumbawamba)</td>
<td>.744</td>
<td>1</td>
<td>.593</td>
<td>.794</td>
<td>.659</td>
<td>1</td>
</tr>
<tr>
<td>Average</td>
<td>.777</td>
<td>1</td>
<td>.636</td>
<td>.794</td>
<td>.659</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 2. Average ratings across excerpts for Experiment 1; GT = ground truth; HP = high precision; HR = high recall.

2 http://urinieto.com/NYU/BoundaryExperiment/
high precision over high recall (Figure 3). Post-hoc multiple comparisons indicated that differences between means of all three types were significant. The only excerpt where precision was not rated more highly than recall was in Excerpt 5 (Tubthumping), a difference that contributed primarily to the interaction. In this case, the excerpt contains a distinctive chorus where the lyrics ‘I get knocked down’ keep repeating. This feature is likely the reason some subjects were led to interpret every instance of this refrain as a possible section beginning even though the harmony underneath follows a longer sectional pattern that is annotated in the ground truth. On the other hand, Excerpt 3 (Intergalactic) obtained similar ratings for ground truth and high precision, likely due to the high number of different sections and silences it contains. This can become problematic when extra boundaries are added (therefore obtaining poor ratings for the high-recall version). Nevertheless, given the subjectivity of this task [2] and the multi-layer organization of boundaries [10], it is not surprising that this type of variability appears in the results.

4. EXPERIMENT 2: CHOOSING BOUNDARIES

4.1 Motivation

The results of Experiment 1 show the relative importance of precision over recall for the evaluation of boundaries, validating the preliminary findings (Section 2.2) in a controlled scenario and with a much larger population of subjects. Nevertheless, the number of tracks employed in this experiment was limited. As a follow-up, we explored these findings using a larger dataset in Experiment 2.

4.2 Methodology

The analysis methods used to compute the boundaries included structural features (SF, [12]), convex non-negative matrix factorization (C-NMF, [7]), and shift-invariant probabilistic latent component analysis (SI-PLCA, [17]). These three algorithms yield ideal results for our experimental design since SF provides one of the best results reported so far on boundaries recognition (high precision and high recall) footnoteRecently challenged by Ordinal Linear Discriminant Analysis [6]. C-NMF tends to over segment (higher recall than precision), and SI-PLCA, depending on parameter choices, tends to under segment (higher precision than recall).

We ran these three algorithms on a database of 463 songs composed of the conjunction of the TUT Beatles dataset, the Levy catalogue [5], and the freely available songs of the SALAMI dataset [13]. Once computed, we filtered the results based on the following criteria for each song: (1) at least two algorithm outputs have a similar F-measure (within a 5% threshold); (2) the F-measure of both algorithms must be at least 45%; (3) at least a 10% difference between the precision and recall values of the two selected algorithm outputs exists.

We found 41 out of 463 tracks that met the above criteria. We made a qualitative selection of these filtered tracks (there are many free tracks in the SALAMI dataset that are live recordings with poor audio quality or simply speech), resulting in a final set of 20 songs. The number of these carefully selected tracks is relatively low, but we except it to be representative enough to address our research questions. Given the two algorithmic outputs maximizing the difference between precision and recall, two differently segmented versions were created for each track: high precision (HP) and high recall (HR). Moreover, similar to Experiment 1, only one minute of audio from each track was utilized, starting 15 seconds into the song.

Table 3 shows average metrics across the 20 selected tracks. The F-measures are the same, while precision and recall vary.

<table>
<thead>
<tr>
<th>Boundaries Version</th>
<th>F</th>
<th>P</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>HP</td>
<td>.63</td>
<td>.32</td>
<td>.56</td>
</tr>
<tr>
<td>HR</td>
<td>.65</td>
<td>.54</td>
<td>.83</td>
</tr>
</tbody>
</table>

Table 3. Average F-measure, precision, and recall values for the two versions of excerpts used in Experiment 2.

As in Experiment 1, the interface for Experiment 2 was on line\(^4\) to facilitate participation. Each participant was presented with five random excerpts selected from the set of 20. Instead of assessing the accuracy on a scale, listeners had to choose the version they found more accurate. In order uniformly distribute excerpts across total trials, selection of excerpts was constrained by giving more priority to those excerpts with fewer collected responses. We obtained an average of 5.75 results per excerpt. The two versions were presented in random order, and subjects had

---

\(^2\) http://www.cs.tut.fi/~sgn/arg/paulus/beatles_sections_TUT.zip

\(^4\) http://cognition.smusic.nyu.edu/boundaryExperiment2/
to listen to the audio at least once before submitting the results. Boundaries were marked with a salient sound like in the prior experiments.

A total 23 subjects, recruited from professional mailing lists, participated in the experiment. Participants had an average of 2.8 ± 1.4 years of musical training and 3.2 ± 2.9 years of experience playing an instrument.

4.3 Results and Discussion

We performed binary logistic regression analysis [11] on the results with the goal of understanding what specific values of the F-measure were actually useful in predicting subject preference (the binary values representing the versions picked by the listeners). Logistic regression enables us to compute the following probability:

\[
P(Y | X_1, \ldots, X_n) = \frac{e^{k+\beta_1 X_1 + \ldots + \beta_n X_n}}{1 + e^{k+\beta_1 X_1 + \ldots + \beta_n X_n}}
\] (3)

where \( Y \) is the dependent, binary variable, \( X_i \) are the predictors, \( \beta_i \) are the weights for these predictors, and \( k \) is a constant value. Parameters \( \beta_i \) and \( k \) are learned through the process of training the regressor. In our case, \( Y \) tells us whether a certain excerpt was chosen or not according to the following predictors: the F-measure (\( X_1 \)), the signed difference between precision and recall (\( X_2 \)), and the absolute difference between precision and recall (\( X_3 \)).

Since 23 subjects took part in the experiment and there were five different tracks with two versions per excerpt, we had a total of 23 \( \times \) 5 \( \times \) 2 = 230 observations as input to the regression with the parameters defined above. We ran the Hosmer & Lemeshow test [4] in order to understand the predictive ability of our input data. If this test is not statistically significant (\( p > 0.05 \)), we know that logistic regression can indeed help us predict \( Y \). In our case, we obtain a value of \( p = .763 (\chi^2 = 4.946, \text{with 8 degrees of freedom}) \) which tells us that the data for this type of analysis fits well, and that the regressor has predictive power.

The analysis of the results of the learned model is shown in Table 4. As expected, the F-measure is not able to predict the selected version (\( p = .992 \)), providing clear evidence that the metric is inexpressive and perceptually irrelevant for the evaluation of segmentation algorithms. Furthermore, we can see that \( P - R \) can predict the results in a statistically significant manner (\( p = .000 \)), while the absolute difference \( |P - R| \), though better than the F-measure, has low predictive power (\( p = .482 \)). This clearly illustrates the asymmetrical relationship between \( P \) and \( R \): it is not sufficient that \( P \) and \( R \) are different, but the sign matters: \( P \) has to be higher than \( R \).

Based on this experiment we can claim that, for these set of tracks, (1) the F-measure does not sufficiently characterize the perception of boundaries, (2) precision is clearly more important than recall, and (3) there might be a better parameterization of the F-measure that encodes relative importance. We attempt to address this last point in the next section.

<table>
<thead>
<tr>
<th>Predictor</th>
<th>( \beta )</th>
<th>S.E.</th>
<th>( \beta )</th>
<th>Wald’s ( \chi^2 )</th>
<th>df</th>
<th>( p )</th>
<th>( e^{\beta} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>F-measure</td>
<td>-0.012</td>
<td>1.153</td>
<td></td>
<td>992</td>
<td>1</td>
<td>.998</td>
<td></td>
</tr>
<tr>
<td>( P - R )</td>
<td>2.268</td>
<td>.471</td>
<td>23.226</td>
<td>.000</td>
<td>1</td>
<td>1.023</td>
<td></td>
</tr>
<tr>
<td>(</td>
<td>P - R</td>
<td>)</td>
<td>-0.669</td>
<td>.951</td>
<td>.495</td>
<td>.838</td>
<td>1</td>
</tr>
</tbody>
</table>

5. ENHANCING THE F-MEASURE

Based on our experiments, we have empirical evidence that high precision is perceptually more relevant than high recall for the evaluation of segmentation algorithms. We can then leverage these findings to obtain a more expressive and perceptually informative version of the F-measure for benchmarking estimated boundaries.

The F-measure is, in fact, a special case of the \( F_\alpha \)-measure:

\[
F_\alpha = (1 + \alpha^2) \frac{P \cdot R}{\alpha^2 P + R} \tag{4}
\]

where \( \alpha = 1 \), resulting in \( P \) and \( R \) having the same weight. However, it is clear from the equation that we should impose \( \alpha < 1 \) in order to give more importance to \( P \) and make the F-measure more perceptually relevant. Note that an algorithm that outputs fewer boundaries does not necessarily increase its \( F_{\alpha} \)-measure, since the fewer predicted boundaries could still be incorrect. Regardless, the question remains: how is the value of \( \alpha \) determined?

A possible method to answer this question is to sweep \( \alpha \) from 0 to 1 using a step size of 0.05 and perform logistic regression analysis at each step using the \( F_{\alpha} \)-measure as the only predictor (\( X_1 = F_{\alpha}, n = 1 \)). The \( p \)-value of the \( F_{\alpha} \)-measure predicting subject preference in Experiment 2 across all \( \alpha \) is shown in Figure 4.

Figure 4. Statistical significance of the \( F_{\alpha} \)-measure predicting the perceptual preference of a given evaluation for \( \alpha \in [0, 1] \).

It is important to note that the data from Experiment 2 is limited as it does not include information at the limits of the difference between precision and recall. As a result, our model predicts that decreases of \( \alpha \) always lead to highest predictive power. Naturally, this is undesirable since we will eventually remove all influence from recall in the measure and favor the trivial solutions discussed at
the beginning of this paper. At some point, as \( P - R \) increases, we expect subject preference to decrease, as preserving a minimum amount of recall becomes more important. Therefore, we could choose the first value of \( \alpha \) (0.58) for which \( F_{\alpha} \)-based predictions of subject preference become accurate at the statistically significant level of 0.01.

We can re-run the evaluation of Experiments 1 and 2 using the \( F_{0.58} \)-measure (i.e. \( \alpha = 0.58 \)) to illustrate that it behaves as expected. For Experiment 1, we obtain 83.3% for HP and 72.1% for HR (instead of 77.7% and 79.4% respectively). For Experiment 2, the values of HP and HR become 71.8% and 58.9% respectively, whereas they were both 65.0% originally. This shows how the new approximated measure is well coordinated with the preferences of the subjects from Experiments 1 and 2, therefore making this evaluation of section boundaries more expressive and perceptually relevant.

This specific \( \alpha \) value is highly dependent on the empirical data, and we are aware of the limitations of using reduced data sets as compared to the real world—in other words, we are likely overfitting to our data. Nonetheless, based on our findings, there must be a value of \( \alpha < 1 \) that better represents the relative importance of precision and recall. Future work, utilizing larger datasets and a greater number of participants, should focus on understanding the upper limit of the difference between precision and recall in order to find the specific inflection point at which higher precision is not perceptually relevant anymore.

6. CONCLUSIONS

We presented a series of experiments concluding that precision is perceived as more relevant than recall when evaluating boundaries in music. The results of the two main experiments discussed here are available on line. Moreover, we have noted the shortcomings of the current F-measure when evaluating results in a perceptually meaningful way. By using the general form of the F-measure, we can obtain more relevant results when precision is emphasized over recall (\( \alpha < 1 \)). Further steps should be taken in order to determine a more specific and generalizable value of \( \alpha \).

7. ACKNOWLEDGMENTS

This work was partially funded by Fundación Caja Madrid and by the National Science Foundation, under grant IIS-0844654.

8. REFERENCES


KEYWORD SPOTTING IN A-CAPELLA SINGING

Anna M. Kruspe
Fraunhofer IDMT, Ilmenau, Germany
Johns Hopkins University, Baltimore, MD, USA
kpe@idmt.fhg.de

ABSTRACT
Keyword spotting (or spoken term detection) is an interesting task in Music Information Retrieval that can be applied to a number of problems. Its purposes include topical search and improvements for genre classification. Keyword spotting is a well-researched task on pure speech, but state-of-the-art approaches cannot be easily transferred to singing because phoneme durations have much higher variations in singing. To our knowledge, no keyword spotting system for singing has been presented yet.

We present a keyword spotting approach based on keyword-filler Hidden Markov Models (HMMs) and test it on a-capella singing and spoken lyrics. We test Mel-Frequency Cepstral Coefficients (MFCCs), Perceptual Linear Predictive Features (PLPs), and Temporal Patterns (TRAPs) as front ends. These features are then used to generate phoneme posteriors using Multilayer Perceptrons (MLPs) trained on speech data. The phoneme posteriors are then used as the system input. Our approach produces useful results on a-capella singing, but depend heavily on the chosen keyword. We show that results can be further improved by training the MLP on a-capella data.

We also test two post-processing methods on our phoneme posteriors before the keyword spotting step. First, we average the posteriors of all three feature sets. Second, we run the three concatenated posteriors through a fusion classifier.

1. INTRODUCTION
Keyword spotting is the task of searching for certain words or phrases (spoken term detection) in acoustic data. In contrast to text data, we cannot directly search for these words, but have to rely on the output of speech recognition systems in some way.

In speech, this problem has been a topic of research since the 1970’s [1] and has since seen a lot of development and improvement [11]. For singing, however, we are not aware of any fully functional keyword spotting systems. Music collections of both professional distributors and private users have grown exponentially since the switch to a digital format. For these large collections, efficient search methods are necessary. Keyword spotting in music collections has beneficial applications for both user groups. Using keyword spotting, users are able to search their collections for songs with lyrics about certain topics. As an example, professional users might use this in the context of synch licensing [4] (e.g., “I need a song containing the word ‘freedom’ for a car commercial.”) Private users could, for example, use keyword spotting for playlist generation (“Generate a playlist with songs that contain the word ‘party’.”)

In this paper, we present our approach to a keyword spotting system for a-capella singing. We will first look at the current state of the art in section 2. We then present our data set in section 3. In section 4, we describe our own keyword spotting system. A number of experiments on this system and their results are presented in section 5. Finally, we draw conclusions in section 6 and give an outlook on future work in section 7.

2. STATE OF THE ART
2.1 Keyword spotting principles
As described in [13], there are three basic principles that have been developed over the years for keyword spotting in speech:

LVCSR-based keyword spotting For this approach, full Large Vocabulary Continues Speech Recognition (LVCSR) is performed on the utterances. This results in a complete text transcription, which can then be searched for the required keywords. LVCSR-based systems lack tolerance for description errors - i.e., if a keyword is not correctly transcribed from the start, it cannot be found later. Additionally, LVCSR systems are complex and expensive to implement.

Acoustic keyword spotting As in LVCSR-based key- word spotting, acoustic keyword spotting employs Viterbi search to find the requested keyword in a given utterance. In this approach, however, the system does not attempt to transcribe each word, but only searches for the specific keyword. Everything else is treated as “filler”. This search can be performed directly on the audio features using an acoustic example, or on phoneme posteriorgrams generated by an acoustic model. In the second case, the algorithm searches for the word’s phonemes. This approach is easy to implement and provides some pronunciation tolerance. Its disadvantage is
the lack of integration of a-priori language knowledge (i.e., knowledge about plausible phoneme and word sequences) that could improve performance.

**Phonetic search keyword spotting** Phonetic search keyword spotting starts out just like LVCSR-based keyword spotting, but does not generate a word transcription of the utterance. Instead, phoneme lattices are saved. Phonetic search for the keyword is then performed on these lattices. This approach combines the advantages of LVCSR-based keyword spotting (a-priori knowledge in the shape of language models) and acoustic keyword spotting (flexibility and robustness).

### 2.2 Keyword spotting in singing

The described keyword spotting principles cannot easily be transferred to music. Singing, in contrast to speech, presents a number of additional challenges, such as larger pitch fluctuation, more pronunciation variation, and different vocabulary (which means existing models cannot easily be transferred).

Another big difference is the higher variation of phoneme durations in singing. Both LVCSR-based keyword spotting and Phonetic search keyword spotting depend heavily on predictable phoneme durations (within certain limits). When a certain word is pronounced, its phonemes will usually have approximately the same duration across speakers. The language model employed in both approaches will take this information into account.

We compared phoneme durations in the TIMIT speech database [7] and our own a-capella singing database (see section 3). The average standard deviations for vowels and consonants are shown in figure 1. It becomes clear that the phoneme durations taken from TIMIT do not vary a lot, whereas some the a-capella phonemes show huge variations. It becomes clear that this especially concerns vowels (AA, AW, EH, IY, AE, AH, AO, EY, AY, ER, UW, OW, UH, IH, OY). This observation has a foundation in music theory: Drawn-out notes are usually sung on vowels. For this reason, acoustic keyword spotting appears to be the most feasible approach to keyword spotting in singing.

To our knowledge, no full keyword spotting system for singing has been presented yet. In [2], an approach based on sub-sequence Dynamic Time Warping (DTW) is suggested. This is similar to the acoustic approach, but does not involve a full acoustic model. Instead, example utterances of the keyword are used to find similar sequences in the tested utterance.

In [5], a phoneme recognition system for singing is presented. It extracts Mel-Frequency Cepstral Coefficients (MFCCs) and Temporal Patterns (TRAPs) which are then used as inputs to a Multilayer Perceptron (MLP). The phonetic output of such a system could serve as an input to a keyword spotting system.

There are also some publications where similar principles are applied to lyrics alignment and Query by Humming [12][3].

![Figure 1: Average standard deviations for vowels and consonants in the TIMIT speech databases (blue) and our a-capella singing data set (green).](http://cmusphinx.sourceforge.net/)

### 3. DATA SET

Our data set is the one presented in [5]. It consists of the vocal tracks of 19 commercial pop songs. They are studio quality with some post-processing applied (EQ, compression, reverb). Some of them contain choir singing. These 19 songs are split up into clips that roughly represent lines in the song lyrics.

Twelve of the songs were annotated with time-aligned phonemes. The phoneme set is the one used in CMU Sphinx 1 and TIMIT [7] and contains 39 phonemes. All of the songs were annotated with word transcriptions. For comparison, recordings of spoken recitations of all song lyrics were also made. These were all performed by the same speaker.

We selected 51 keywords for testing our system. Most of them were among the most frequent words in the provided lyrics. A few were selected because they had a comparatively large number of phonemes. An overview is given in table 1.

### 4. PROPOSED SYSTEM

Figure 2 presents an overview of our system.

1. **Feature extraction** We extract Mel-Frequency Cepstral Coefficients (MFCCs), Perceptual Linear Predictive features (PLPs), and Temporal Patterns (TRAPs) [6]. We keep 20 MFCC coefficients and 39 PLP coefficients (13 direct coefficients plus deltas and double-deltas). For the TRAPs, we use 8 linearly spaced spectral bands and a temporal context of 20 frames and keep 8 DCT coefficients.

2. **MLP training and phoneme recognition** Using each feature data set, we train Multi-Layer Perceptrons (MLPs). MLPs are commonly used to train acoustic models for the purpose of phoneme recognition. We chose a structure with two hidden layers and tested three different dimension settings: 50, 200, and 1000 dimensions per layer. MLPs were trained solely on TIMIT data first, then on a mix of TIMIT and a-capella in a second experiment. The resulting MLPs are then used to recognize phonemes in our a-capella dataset, thus generating phoneme posteriorgrams.
Table 1: All 51 tested keywords, ordered by number of phonemes.

<table>
<thead>
<tr>
<th>Number of Phonemes</th>
<th>Keywords</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>love, girl, away, time, over, home, sing, kiss, play, other</td>
</tr>
<tr>
<td>3</td>
<td>hello, trick, never, hand, baby, times, under, things, world, think, heart, tears, lights</td>
</tr>
<tr>
<td>4</td>
<td>always, inside, drink, nothing, rehab, forever, rolling, feeling, waiting, alright, tonight</td>
</tr>
<tr>
<td>5</td>
<td>something, denial, together, morning, friends, leaving, sunrise</td>
</tr>
<tr>
<td>6</td>
<td>umbrella, afternoon, stranger, somebody, entertain, everyone</td>
</tr>
<tr>
<td>7</td>
<td>beautiful, suicidal</td>
</tr>
</tbody>
</table>

Figure 2: Overview of our keyword spotting system. Variable parameters are shown in italics.

The following two points described optional post-processing steps on the phoneme posteriorgrams.

3a. Posteriorgram merging For this post-processing step, we take the phoneme posterior results that were obtained using different feature sets and average them. We tested both the combinations of PLP+MFCC, PLP+TRAP, and PLP+MFCC+TRAP.

3b. Fusion MLP classifier As a second post-processing option, we concatenate phoneme posteriors obtained by using different feature sets and run them through a fusion MLP classifier to create better posteriors. We again tested the combinations PLP+MFCC, PLP+TRAP, and PLP+MFCC+TRAP.

4. Keyword spotting The resulting phoneme posteriors are then used to perform the actual keyword spotting. As mentioned above, we employ an acoustic approach. It is based on keyword-filler Hidden Markov Models (HMMs) and has been described in [14] and [8].

In general, two separate HMMs are created: One for the requested keyword, and one for all non-keyword regions (=filler). The keyword HMM is generated using a simple left-to-right topology with one state per keyword phoneme, while the filler HMM is a fully connected loop of states for all phonemes. These two HMMs are then joined. Using this composite HMM, a Viterbi decode is performed on the phoneme posteriors. Whenever the Viterbi path passes through the keyword HMM, the keyword is detected. The likelihood of this path can then be compared to an alternative path through the filler HMM, resulting in a detection score. A threshold can be employed to only return highly scored occurrences. Additionally, the parameter $\beta$ can be tuned to adjust the model. It determines the likelihood of transitioning from the filler HMM to the keyword HMM. The whole process is illustrated in figure 3.

We use the $F_1$ measure for evaluation. Results are considered to be true positives when a keyword is spotted somewhere in an expected utterance. Since most utterances contain one to ten words, we consider this to be sufficiently exact. Additionally, we evaluate the precision of the results. For the use cases described in section 1, users will usually only require a number of correct results, but not necessarily all the occurrences of the keyword in the whole database. We consider a result to be correct when the keyword is found as part of another word with the same pronunciation. The reasoning behind this is that a user who searched
for the keyword “time” might also accept occurrences of the word “times” as correct.

5. EXPERIMENTS

5.1 Experiment 1: Oracle search
As a precursor to the following experiments, we first tested our keyword spotting approach on oracle posteriorgrams for the a-capella data. This was done to test the general feasibility of the algorithm for keyword spotting on singing data with its highly variable phoneme durations. The oracle posteriorgrams were generated by converting the phoneme annotations to posteriorgram format by setting the likelihoods of the annotated phonemes to 1 during the corresponding time segment and everything else to 0. A keyword search on these posteriorgrams resulted in $F_1$ measures of 1 for almost all keywords. In cases where the result was not 1, we narrowed the reasons down to annotation errors and pronunciation variants that we did not account for. We conclude that our keyword-filler approach is generally useful for keyword spotting on a-capella data, and our focus in the following experiments is on obtaining good posteriorgrams from the audio data.

5.2 Experiment 2: A-Capella vs. Speech
For our first experiment, we run our keyword spotting system on the a-capella singing data, and on the same utterances spoken by a single speaker. We evaluate all three feature datasets (MFCC, PLP, TRAP) separately. The recognition MLP is trained on TIMIT speech data only. We also test three different sizes for the two hidden MLP layers: 50 nodes, 200 nodes, and 1000 nodes in each layer. The results are shown in figure 4.

As described in section 2.2, we expected keyword spotting on singing to be more difficult than on pure speech because of a larger pitch range, more pronunciation variations, etc. Our results support this assumption: In speech, keywords are recognized with an average $F_1$ measure of 33% using only PLP features, while the same system results in an average $F_1$ of only 10% on a-capella singing.

For both data sets, an MLP with 200 nodes in the hidden layers shows a notable improvement over one with just 50. When using 1000 nodes, the result still improves by a few percent in most cases.

When looking at the features, PLP features seem to work best by a large margin, with TRAPs coming in second. It is notable, however, that some keywords can be detected much better when using MFCCs or TRAPs than PLPs (e.g. “sing”, “other”, “hand”, “world”, “tears”, “alright”). As described in [5] and [10], certain feature sets represent some phonemes better than others and can therefore balance each other out. A combination of the features might therefore improve the whole system.

Evaluation of the average precision (instead of $F_1$ measure) shows the same general trend. The best results are again obtained when using PLP features and the largest MLP. The average precision in this configuration is 16% for a-capella singing and 37% for speech. (While the difference is obvious, the result is still far from perfect for speech. This demonstrates the difficulty of the recognition process without a-priori knowledge."

5.3 Experiment 3: Training including a-capella data
As a measure to improve the phoneme posteriorgrams for a-capella singing, we next train our recognition MLP with both TIMIT and a part of the a-capella data. We mix in about 50% of the a-capella clips with the TIMIT data. They make up about 10% of the TIMIT speech data. The results are shown in figure 5 (only the results for the largest MLP are shown).

This step improves the keyword recognition on a-capella data massively in all feature and MLP configurations. The best result still comes from the biggest MLP when using PLP features and is now an average $F_1$ of 24%. This step makes the recognition MLP less specific to the properties of pure speech and therefore does not improve the results for the speech data very much. It actually degrades the best result somewhat.

The effect on the average precision is even greater. The a-capella results are improved by 10 to 15 percentage points for each feature set. On speech data, the PLP precision decreases by 7 percentage points.

5.4 Experiment 4: Posterior merging
As mentioned in experiment 2, certain feature sets seem to represent some keywords better than others. We therefore concluded that combining the results for all features could improve the recognition result.
The fusion classifier improves the and PLP+MFCC+TRAP. The results are shown in figure 7.
Again tested the configurations PLP+MFCC, PLP+TRAP, and MLP+MFCC+TRAP are shown and compared to the PLP only result.

To this end, we tested merging the phoneme posteriorgrams between the MLP phoneme recognition step and the HMM keyword spotting step. In order to do this, we simply calculated the average values across the posteriors obtained using the three different feature data sets. This was done for all phonemes and time frames. Keyword spotting was then performed on the merged posteriorgrams. We tested the configurations PLP+MFCC, PLP+TRAP, and PLP+MFCC+TRAP. The results are shown in figure 6.

Posterior merging seems to improve the results for a-capella data (left) and speech (right) when posteriorgrams for two or three features are merged. The configurations PLP+MFCC, PLP+TRAP, and PLP+MFCC+TRAP are shown and compared to the PLP only result.

5.5 Experiment 5: Fusion classifier

After the posterior merging, we tested a second method of combining the feature-wise posteriorgrams. In this second method, we concatenated the posteriorgrams obtained from two or all three of the feature-wise MLP recognizers and ran them through a second MLP classifier. This fusion MLP was trained on a subset of the a-capella data. This fusion classifier generates new, hopefully improved phoneme posteriorgrams. HMM keyword spotting is then performed on these new posteriorgrams. We again tested the configurations PLP+MFCC, PLP+TRAP, and PLP+MFCC+TRAP. The results are shown in figure 7.

The fusion classifier improves the $F_1$ measure for a-capella singing by 5 percentage points. The best result of 29% is obtained when all three feature sets are used. Precision improves from 24% to 31%. However, the fusion classifier makes the system less specific towards speech and therefore decreases the performance on speech data.

5.6 Variation across keywords

The various results we presented in the previous experiments varies widely across the 51 keywords. This is a common phenomenon in keyword spotting. In many approaches, longer keywords are recognized better than shorter ones because the Viterbi path becomes more reliable with each additional phoneme. This general trend can also be seen in our results, but even keywords with the same number of phonemes vary a lot. The precisions vary similarly, ranging between 2% and 100%. When taking just the 50% of the keywords that can be recognized best, the average $F_1$ measure for the best approach (fusion MLP) jumps from 20% to 44%. Its precision increases from 31% to 46%. We believe the extremely bad performance of some keywords is in part due to the small size of our data set. Some keywords occurred in just one of the 19 songs and were, for example, not recognized because the singer used an unusual pronunciation in each occurrence or had an accent that the phoneme recognition MLP was not trained with. We therefore believe these results could improve massively when more training data is used.

6. CONCLUSION

In this paper, we demonstrated a first keyword spotting approach for a-capella singing. We ran experiments for 51 keywords on a database of 19 a-capella pop songs and recordings of the spoken lyrics. As our approach, we selected acoustic keyword spotting using keyword-filler HMMs. Other keyword spotting approaches depend on learning average phoneme durations, which vary a lot more in a-capella singing than in speech. These approaches therefore cannot directly be transferred.

As a first experiment, we tested our approach on oracle phoneme posteriorgrams and obtained almost perfect results. We then produced “real world” posteriorgrams using MLPs with two hidden layers which had been trained on TIMIT speech data. We tested PLP, MFCC, and TRAP features. The training yielded MLPs with 50, 200, and 1000 nodes per hidden layer. We observed that the 200 node MLP produced significantly better results than the 50 node MLPs in all cases ($p < 0.0027$), while the 1000 node MLPs only improved upon this result somewhat. PLP features performed significantly better than the two other feature sets. Finally, keywords were detected much better in a-capella singing than in speech. These approaches therefore cannot directly be transferred.

We noticed that some keywords were recognized better when MFCCs or TRAPs were used instead of PLPs. We therefore tried two approaches to combine the results for...

Figure 6: $F_1$ measures for a-capella data (left) and speech (right) when posteriorgrams for two or three features are merged. The configurations PLP+MFCC, PLP+TRAP, and PLP+MFCC+TRAP are shown and compared to the PLP only result.

Figure 7: $F_1$ measures for a-capella data (left) and speech (right) when posteriorgrams for two or three features are merged. The configurations PLP+MFCC, PLP+TRAP, and PLP+MFCC+TRAP are shown and compared to the PLP only result.
all three features: Posterior merging and fusion classifiers. Both approaches improved the results on the a-capella data. The best overall result for a-capella data was produced by a fusion classifier that combined all three features (29%).

As expected, keyword spotting on a-capella singing proved to be a harder task than on speech. The results varied widely between keywords. Some of the very low results arise because the keyword in question only occurred in one song where the singer used an unusual pronunciation or had an accent. The small size of our data set also poses a problem when considering the limited number of singers. The acoustic model trained on speech data and a part of the a-capella data might be subject to overfitting to the singers’ vocal characteristics.

In contrast, the recognition worked almost perfectly for keywords with more training data. Keyword length also played a role. When using only the 50% best keywords, the average $F_1$ measure increased by 15 percentage points. Finally, there are many applications where precision plays a greater role than recall, as described in section 4. Our system can be tuned to achieve higher precisions than $F_1$ measures and is therefore also useful for these applications. We believe that the key to better keyword spotting results lies in better phoneme posteriorgrams. A larger a-capella data set would therefore be very useful for further tests and would provide more consistent results.

7. FUTURE WORK

As mentioned in section 2, more sophisticated keyword spotting systems for speech incorporate knowledge about plausible phoneme durations (e.g. [9]). In section 2.2, we showed why this approach is not directly transferable to singing: The vowel durations vary too much. However, consonants are not affected. We would therefore like to start integrating knowledge about average consonant durations in order to improve our keyword spotting system. In this way, we hope to improve the results for the keywords that were not recognized well by our system.

Following this line of thought, we could include even more language-specific knowledge in the shape of a language model that also contains phonotactic information, word frequencies, and phrase frequencies. We could thus move from a purely acoustic approach to a phonetic (lattice-based) approach.

We will also start applying our approaches to polyphonic music instead of a-capella singing. To achieve good results on polyphonic data, pre-processing will be necessary (e.g. vocal activity detection and source separation).

8. REFERENCES


THE IMPORTANCE OF F0 TRACKING IN QUERY-BY-SINGING-HUMMING

Emilio Molina, Lorenzo J. Tardón, Isabel Barbancho, Ana M. Barbancho
Universidad de Málaga, ATIC Research Group, Andalucía Tech,
ETSI Telecomunicación, Campus de Teatinos s/n, 29071 Málaga, SPAIN
emm@ic.uma.es, lorenzo@ic.uma.es, ibp@ic.uma.es, abp@ic.uma.es

ABSTRACT

In this paper, we present a comparative study of several state-of-the-art F0 trackers applied to the context of query-by-singing-humming (QBSH). This study has been carried out using the well known, freely available, MIR-QBSH dataset in different conditions of added pub-style noise and smartphone-style distortion. For audio-to-MIDI melodic matching, we have used two state-of-the-art systems and a simple, easily reproducible baseline method. For the evaluation, we measured the QBSH performance for 189 different combinations of F0 tracker, noise/distortion conditions and matcher. Additionally, the overall accuracy of the F0 transcriptions (as defined in MIREX) was also measured. In the results, we found that F0 tracking overall accuracy correlates with QBSH performance, but it does not totally measure the suitability of a pitch vector for QBSH. In addition, we also found clear differences in robustness to F0 transcription errors between different matchers.

1. INTRODUCTION

Query-by-singing-humming (QBSH) is a music information retrieval task where short hummed or sung audio clips act as queries. Nowadays, several successful commercial applications for QBSH have been released, such as MusicRadar 1 or SoundHound 2, and it is an active field of research. Indeed, there is a task for QBSH in MIREX since 2006, and every year novel and relevant approaches can be found.

Typically, QBSH approaches firstly extract the F0 contour and/or a note-level transcription for a given vocal query, and then a set of candidate melodies are retrieved from a large database using a melodic matcher module. In the literature, many different approaches for matching in QBSH can be found: statistical, note vs. note, frame vs. note, frame vs. frame. Generally, state-of-the-art systems for QBSH typically combines different approaches in order to achieve more reliable results [3, 12].

However, even state-of-the-art systems for QBSH have not a totally satisfactory performance in many real-world cases [1], so there is still room for improvement. Nowadays, some challenges related to QBSH are [2]: reliable pitch tracking in noisy environments, automatic song database preparation (predominant melody extraction and transcription), efficient search in very large music collections, dealing with errors of intonation and rhythm in amateur singers, etc.

In this paper, we analyse the performance of various state-of-the-art F0 trackers for QBSH in different conditions of background noise and smartphone-style distortion. For this study, we have considered three different melodic matchers: two state-of-the-art systems (one of which obtained the best results in MIREX 2013), and a simple, easily reproducible baseline method based on frame-to-frame matching using dynamic time warping (DTW). In Figure 1, we show a scheme of our study.

Figure 1. Overall scheme of our study

This paper is organized as follows: Section 2 and Section 3 present the studied algorithms for F0 tracking and melodic matching, respectively. The evaluation strategy is presented in Section 4. Section 5 presents the obtained results and Section 6 draws some conclusions about the present study.

2. F0 TRACKERS

In this section, we describe the F0 trackers considered in our study, together with their specific set of parameters. The literature reports a wide set of algorithms oriented to either monophonic or polyphonic audio, so we have focused on well-known, commonly used algorithms (e.g. Yin [4] or Praat-AC [8]), and some recently published algorithms for F0 estimation (e.g. pYin [6] or MELODIA [15]). Most of the algorithms analysed address F0 estimation in monophonic audio, but we have also studied the performance of MELODIA, which is a method for predominant melody extraction in polyphonic audio, using monophonic audio in noisy conditions. Regarding the used set of parameters, when possible, they have been adjusted by trial and error using ten audio queries. The considered methods for F0 tracking are the following ones:
2.1 YIN

The YIN algorithm was developed by de Cheveigné and Kawahara in 2002 [4]. It resembles the idea of the autocorrelation method [5] but it uses the cumulative mean normalized difference function, which peaks at the local period with lower error rates than the traditional autocorrelation function. In our study, we have used Matthias Mauch’s VAMP plugin\(^3\) in Sonic Annotator tool\(^4\).

*Parameters used in YIN:* step size = 80 samples (0.01 seconds), Block size = 512 samples, Yin threshold = 0.15.

2.2 pYIN

The pYin method has been published by Mauch in 2014 [6], and it basically adds a HMM-based F0 tracking stage in order to find a “smooth” path through the fundamental frequency candidates obtained by Yin. Again, we have used the original Matthias Mauch’s VAMP plugin\(^3\) in Sonic Annotator tool\(^4\).

*Parameters used in pYIN:* step size = 80 samples (0.01 seconds), Block size = 512 samples, Yin threshold distribution = Beta (mean 0.15).

2.3 AC-DEFAULT and AC-ADJUSTED (Praat)

Praat is a well-known tool for speech analysis [7], which includes several methods for F0 estimation. In our case, we have chosen the algorithm created by P. Boersma in 1993 [8]. It is based on the autocorrelation method, but it improves it by considering the effects of the window during the analysis and by including a F0 tracking stage based on dynamic programming. This method has 9 parameters that can be adjusted to achieve a better performance for a specific application. According to [9], this method significantly improves its performance when its parameters are adapted to the input signal. Therefore, we have experimented not only with the default set of parameters (AC-DEFAULT), but also with an adjusted set of parameters in order to limit octave jumps and false positives during the voicing process (AC-ADJUSTED). In our case, we have used the implementation included in the console Praat tool.

*Parameters used in AC-DEFAULT:* Time step = 0.01 seconds, Pitch floor = 75Hz, Max. number of candidates = 15, Very accurate = off, Silence threshold = 0.03, Voicing threshold = 0.45, Octave cost = 0.01, Octave-jump cost = 0.35, Voiced / unvoiced cost = 0.15, Pitch ceiling = 600 Hz.

*Parameters used in AC-ADJUSTED:* Time step = 0.01 seconds, Pitch floor = 50Hz, Max. number of candidates = 15, Very accurate = off, Silence threshold = 0.03, Voicing threshold = 0.45, Octave cost = 0.1, Octave-jump cost = 0.5, Voiced / unvoiced cost = 0.5, Pitch ceiling = 700 Hz.

2.4 AC-LEIWANG

In our study we have also included the exact F0 tracker used in Lei Wang’s approach for QBSH [3], which obtained the best results for most of the datasets in MIREX 2013. It is based on P. Boersma’s autocorrelation method\(^[8]\), but it uses a finely tuned set of parameters and a post-processing stage in order to mitigate spurious and octave errors. This F0 tracker is used in the latest evolution of a set of older methods [11, 12] also developed by Lei Wang (an open source C++ implementation is available\(^5\)).

2.5 SWIPE’

The Swipe’ algorithm was published by A. Camacho in 2007 [10]. This algorithm estimates the pitch as the fundamental frequency of the sawtooth waveform whose spectrum best matches the spectrum of the input signal. The algorithm proved to outperform other well-known F0 estimation algorithms, and it is used in the F0 estimation stage of some state-of-the-art query-by-humming systems [13]. In our study, we have used the original author’s Matlab implementation\(^6\). The Matlab code does not provide a voiced / unvoiced classification of frames, but it outputs a strength vector \(s\) which has been used for it. Specifically, a frame is considered voiced if its strength is above a threshold \(S_{th}\), otherwise they are considered unvoiced.

*Parameters used in SWIPE’:* DT (hop-size) = 0.01 seconds, \(p_{min} = 50\) Hz, \(p_{max} = 700\)Hz, \(d\log_{2}p = 1/4\) (default), \(d\text{ERB}_{s} = 0.1\) (default), \(\text{wovlerap} = 0.5\) (default), voicing threshold \(S_{th} = 0.3\).

2.6 MELODIA-MONO and MELODIA-POLY

MELODIA is a system for automatic melody extraction in polyphonic music signals developed by Salamon in 2012 [15]. This system is based on the creation and characterisation of pitch contours, which are time continuous sequences of pitch candidates grouped using auditory streaming cues. Melodic and non-melodic contours are distinguished depending on the distributions of its characteristics. The used implementation is MELODIA VAMP plugin\(^7\) in Sonic Annotator tool\(^4\). This plugin has two default sets of parameters, adapted to deal with monophonic or polyphonic audio. We have experimented with both of them, and therefore we have defined two methods: MELODIA-MONO and MELODIA-POLY.

*Parameters used in MELODIA-MONO:* Program = Monophonic, Min Frequency = 55Hz, Max Frequency = 700Hz, Voicing Tolerance = 3.00, Monophonic Noise Filter = 0.00, Audio block size = 372 (not configurable), Window increment = 23 (not configurable).

*Parameters used in MELODIA-POLY:* Program = Polyphonic, Min Frequency = 55Hz, Max Frequency = 700Hz, Voicing Tolerance = 0.20, Monophonic Noise Filter = 0.00, Audio block size = 372 (not configurable), Window increment = 23 (not configurable).

Note that the time-step in this case can not be directly set to 0.01 seconds. Therefore, we have linearly interpolated the pitch vector in order to scale it to a time-step of 0.01 seconds.

---

\(^3\) http://code.soundsoftware.ac.uk/projects/pyin

\(^4\) http://www.vamp-plugins.org/sonic-annotator/

\(^5\) http://www.atic.uma.es/ismir2014qbsh/

\(^6\) http://www.csee.uff.edu/ acamacho/publications/swipep.m

\(^7\) http://mitg.upf.edu/technologies/melodia
3. AUDIO-TO-MIDI MELODIC MATCHERS

In this section, we describe the three considered methods for audio-to-MIDI melodic matching: a simple baseline (Section 3.1) and two state-of-the-art matchers (Sections 3.2 and 3.3).

3.1 Baseline approach

We have implemented a simple, freely available basis-line approach based on dynamic time warping (DTW) for melodic matching. Our method consists of four steps (a scheme is shown in Figure 2):

1. Model building: We extract one pitch vector \( P^k \) (in MIDI number) for every target MIDI song \( k \in 1 \ldots N_{\text{songs}} \) using a hop-size of 0.01 seconds. Then we replace unvoiced frames (rests) in \( P^k \) by the pitch value of the previous note, except for the case of initial unvoiced frames, which are directly removed (these processed pitch vectors are labelled as \( P^{k} \)). Then, each pitch vector \( P^{k} \forall k \in 1 \ldots N_{\text{songs}} \) is truncated to generate 7 pitch vectors with lengths \( [500, 600, 700, 800, 900, 1000, 1100] \) frames (corresponding to the first 5, 6, 7, 8, 9, 10 and 11 seconds of the target MIDI song, which are reasonable durations for an user query). We label these pitch vectors as \( P_{5s}^k, P_{6s}^k \ldots P_{11s}^k \). Finally, all these pitch vectors are resampled (through linear interpolation) to a length of 50 points, and then zero-mean normalized (for a common key transposition), leading to \( P_{50}^{50} \forall \text{Duration} \in 5s \ldots 11s \) and \( \forall k \in 1 \ldots N_{\text{songs}} \). These vectors are then stored for later usage. Note that this process must be done only once.

2. Query pre-processing: The pitch vector \( P^Q \) of a given .wav query is loaded (note that all pitch vectors are computed with a hopsize equal to 0.01 seconds). Then, as in step (1), unvoiced frames are replaced by the pitch value of the previous note, except for the case of initial unvoiced frames, which are directly removed. This processed vector is then converted to MIDI numbers with 1 cent resolution, and labelled as \( P^Q \). Finally, \( P^Q \) is resampled (using linear interpolation) to a length \( L = 50 \) and zero-mean normalized (for a common key transposition), leading to \( P_{50}^Q \).

3. DTW-based alignment: Now we find the optimal alignment between \( P_{50}^Q \) and all pitch vectors \( P_{50}^{k} \forall \text{Duration} \in 5s \ldots 11s \) and \( \forall k \in 1 \ldots N_{\text{songs}} \) using dynamic time warping (DTW). In our case, each cost matrix \( C_{\text{Duration},k} \) is built using the squared difference:

\[
C_{\text{Duration},k}(i, j) = (P_{50}^{Q}(i) - P_{50}^{k}(j))^2
\]

Where \( k \) is the target song index, \( \text{Duration} \) represents the truncation level (from 5s to 11s), and \( i, j \) are the time indices of the query pitch vector \( P_{50}^Q \) and the target pitch vector \( P_{50}^{k} \), respectively. The optimal path is now found using Dan Ellis’ Matlab implementation for DTW [16] (dpfast.m function), with the following allowed steps and associated cost weights \( [\Delta i, \Delta j, W]: [1, 1, 1], [1, 0, 30], [0, 1, 30], [1, 2, 5], [2, 1, 5] \). The allowed steps and weights have been selected in order to penalize 0 or 90 angles in the optimal path (associated to unnatural alignments), and although they lead to acceptable results, they may not be optimal.

3.2 Music Radar’s approach

MusicRadar [3] is a state-of-the-art algorithm for melodic matching, which participated in MIREX 2013 and obtained the best accuracy in all datasets, except for the case of IO-CA-S [8]. It is the latest evolution of a set of systems developed by Lei Wang since 2007 [11, 12]. The system takes advantage of several matching methods to improve its accuracy. First, Earth Mover’s Distance (EMD), which is note-based and fast, is adopted to eliminate most unlikely candidates. Then, Dynamic Time Warping (DTW), which is frame-based and more accurate, is executed on these surviving candidates. Finally, a weighted voting fusion strategy is employed to find the optimal match. In our study, we have used the exact melody matcher tested in MIREX 2013, provided by its original author.

3.3 NetEase’s approach

NetEase’s approach [13] is a state-of-the-art algorithm for melodic matching, which participated in MIREX 2013 and
obtained the first position for IOACAS dataset\(^8\), as well as relevant results in the rest of datasets. This algorithm adopts a two-stage cascaded solution based on Locality Sensitive Hashing (LSH) and accurate matching of frame-level pitch sequence. Firstly, LSH is employed to quickly filter out songs with low matching possibilities. In the second stage, Dynamic Time Warping is applied to find the N (set to 10) most matching songs from the candidate list. Again, the original authors of NetEase’s approach (who also authored some older works on query-by-humming [14]) collaborated in this study, so we have used the exact melody matcher tested in MIREX 2013.

4. EVALUATION STRATEGY

In this section, we present the datasets used in our study (Section 4.1), the way in which we have combined F0 trackers and melody matchers (Section 4.2) and the chosen evaluation measures (Section 4.3).

4.1 Datasets

We have used the public corpus MIR-QBSH\(^8\) (used in MIREX since 2005), which includes 4431 .wav queries corresponding to 48 different MIDI songs. The audio queries are 8 seconds length, and they are recorded in mono 8 bits, with a sample rate of 8kHz. In general, the audio queries are monophonic with no background noise, although some of them are slightly noisy and/or distorted. This dataset also includes a manually corrected pitch vector for each .wav query. Although these annotations are fairly reliable, they may not be totally correct, as stated in MIR-QBSH documentation.

In addition, we have used the Audio Degradation Toolbox [17] in order to recreate common environments where a QBSH system could work. Specifically, we have combined three levels of pub-style added background noise (PubEnvironment1 sound) and smartphone-style distortion (smartphoneRecording degradation) leading to a total of seven evaluation datasets: (1) Original MIR-QBSH corpus (2) 25 dB SNR (3) 25 dB SNR + smartphone distortion (4) 15 dB SNR (5) 15 dB SNR + smartphone distortion (6) 5 dB SNR (7) 5 dB SNR + smartphone distortion. Note that all these degradations have been checked in order to ensure perceptually realistic environments.

Finally, in order to replicate MIREX conditions, we have included 2000 extra MIDI songs (randomly taken from ESENE collection\(^9\) ) to the original collection of 48 MIDI songs, leading to a songs collection of 2048 MIDI songs. Note that, although these 2000 extra songs fit the style of the original 48 songs, they do not correspond to any .wav query of Jang’s dataset.

4.2 Combinations of F0 trackers and melody matchers

For each of the 7 datasets, the 4431 .wav queries have been transcribed using the 8 different F0 trackers mentioned in Section 2. Additionally, each dataset also includes the 4431 manually corrected pitch vectors of MIR-QBSH as a reference, leading to a total of 7 datasets \(\times (8 + 1)\) manual annotation) \(\times 4431\) queries = 63 \(\times 4431\) queries = 279153 pitch vectors. Then, all these pitch vectors have been used as input to the 3 different melody matchers mentioned in Section 3, leading to 930510 lists of top-10 matched songs. Finally, these results have been used to compute a set of meaningful evaluation measures.

4.3 Evaluation measures

In this section, we present the evaluation measures used in this study:

1. Mean overall accuracy of F0 tracking (Accov): For each pitch vector we have computed an evaluation measure defined in MIREX Audio Melody Extraction task: overall accuracy (Accov) (a definition can be found in [15]). The mean overall accuracy is then defined as

\[
\text{Accov} = \frac{1}{N} \sum_{i=1}^{N} \text{Accov}_i,
\]

where Accov is the accuracy of the i-th pitch vector, and N is the total number of queries considered.

2. Reciprocal Rank (MRR): This measure is commonly used in MIREX Query By Singing/Humming task\(^8\), and it is defined as

\[
\text{MRR} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{r_i},
\]

where N is the total number of queries considered and ri is the rank of the correct answer in the retrieved melodies for i-th query.

5. RESULTS & DISCUSSION

In this section, we present the obtained results and some relevant considerations about them.

5.1Accov and MRR for each F0 tracker - Dataset - Matcher

In Table 1, we show the Accov and the MRR obtained for the whole dataset of 4431 .wav queries in each combination of F0 tracker-dataset-matcher (189 combinations in total). Note that these results are directly comparable to MIREX Query by Singing/Humming task\(^8\) (Jang Dataset). As expected, the manually corrected pitch vectors produce the best MRR in most cases (the overall accuracy is 100% because it has been taken as the ground truth for such measure). Note that, despite manual annotations are the same in all datasets, NetEase and MusicRadar matchers do not produce the exact same results in all cases. It is due to the generation of the indexing model (used to reduce the time search), which is not a totally deterministic process.

Regarding the relationship between Accov and MRR in the rest of F0 trackers, we find a somehow contradictory result: the best Accov does not always correspond with the best MRR. This fact may be due to different reasons. On the one hand, the meaning of Accov may be distorted due to annotation errors in the ground truth (as mentioned in Section 4.1), or to eventual intonation errors in the dataset. However, the manual annotations produce the best MRR, what suggests that the amount of these types

\(^8\) www.esac-data.org/
Table 1: F0 overall accuracy and MRR obtained for each case. F0 trackers: (A) MANUALLY CORRECTED (B) AC-LEIWANG (C) AC-ADJUSTED (D) PYIN (E) SWIPE’ (F) YIN (G) AC-DEFAULT (H) MELODIA-MONO (I) MELODIA-POLY. The format of each cell is: Acc\(_{cv}\) (%) / MRR-baseline / MRR-NetEase / MRR-MusicRadar.

![Figure 3](https://via.placeholder.com/150)

Figure 3. According to MIREX measures, these two pitch vectors (manually manipulated) are equally accurate; however, they are not equally suitable for QBSH.

Regarding the obtained results (shown in Figure 4), we observe clear differences in the robustness to F0 estimation errors between matchers, which is coherent with the results presented in Table 1. The main difference is found in the baseline matcher with respect to both NetEase and Music Radar. Given that the baseline matcher only uses DTW, whereas the other two matchers use a combination of various searching methods (see Sections 3.2 and 3.3), we hypothesise that such combination may improve their robustness to F0 tracking errors. However, further research is needed to really test this hypothesis.

6. CONCLUSIONS

In this paper, eight different state-of-the-art F0 trackers were evaluated for the specific application of query-by-humming-singing in different conditions of pub-style added noise and smartphone-style distortion. This study was carried out using three different matching methods: a simple, freely available baseline (a detailed description has been provided in Section 3.1) and two state-of-the-art matchers. In our results, we found that Boersma’s AC method [8], with an appropriate adjustment and a smoothing stage...
achieves the best results when the audio is not very degraded. In contrast, when the audio is highly degraded, the best results are obtained with pYIN [6], even without further smoothing. Considering that pYIN is a very recent, open source approach, this result is promising in order to improve the noise robustness of future QBSH systems. Additionally, we found that F0 trackers perform differently on QBSH depending on the type of F0 tracking errors made. Due to this, MIREX measures do not fully represent the suitability of a pitch vector for QBSH purposes, so the development of novel evaluation measures in MIREX is encouraged to really measure the suitability of MIR systems for specific applications. Finally, we observed clear differences between matchers regarding their robustness to F0 estimation errors. However, further research is needed for a deeper insight into these differences.

7. ACKNOWLEDGEMENTS
Special thanks to Doreso\(^1\) team (especially to Lei Wang and Yuhang Cao) and to Peng Li for their active collaboration in this study. This work has been funded by the Ministerio de Economía y Competitividad of the Spanish Government under Project No. TIN2013-47276-C6-2-R and by the Junta de Andalucía under Project No. P11-TIC-7154. The work has been done at Universidad de Málaga. Campus de Excelencia Internacional Andalucía Tech.

8. REFERENCES
VOCAL SEPARATION USING SINGER-VOWEL PRIORS OBTAINED FROM POLYPHONIC AUDIO

Shrikant Venkataramani¹, Nagesh Nayak², Preeti Rao¹, and Rajbabu Velmurugan¹

¹Department of Electrical Engineering, IIT Bombay, Mumbai 400076
²Sensibol Audio Technologies Pvt. Ltd.
¹{vshrikant, prao, rajbabu}@ee.iitb.ac.in
²nageshsnayak@sensibol.com

ABSTRACT

Single-channel methods for the separation of the lead vocal from mixed audio have traditionally included harmonic-sinusoidal modeling and matrix decomposition methods, each with its own strengths and shortcomings. In this work we use a hybrid framework to incorporate prior knowledge about singer and phone identity to achieve the superior separation of the lead vocal from the instrumental background. Singer specific dictionaries learned from available polyphonic recordings provide the soft mask that effectively attenuates the bleeding-through of accompanying melodic instruments typical of purely harmonic-sinusoidal model based separation. The dictionary learning uses NMF optimization across a training set of mixed signal utterances while keeping the vocal signal bases constant across the utterances. A soft mask is determined for each test mixed utterance frame by imposing sparseness constraints in the NMF partial co-factorization. We demonstrate significant improvements in reconstructed signal quality arising from the more accurate estimation of singer-vowel spectral envelope.

1. INTRODUCTION

Source separation techniques have been widely applied in the suppression of the lead vocal in original songs to obtain the orchestral background for use in karaoke and remix creation. In stereo and multichannel recordings, spatial cues can contribute significantly to vocal separation from the original mixtures. However this separation is not complete, depending on the manner in which the multiple instruments are panned in the mix. Further, an important category of popular music recordings, dating until the 1950s in the West and even later in the rest of the world, are purely monophonic. Single-channel methods for the separation of the lead vocal from the instrumental background include harmonic sinusoidal modeling and matrix decomposition methods. Of these, harmonic sinusoidal modeling has found success in situations where no clean data is available for supervised learning [6], [10]. Based on the assumption that the vocal is dominant in the mixture, predominant pitch detection methods are applied to obtain the vocal pitch and hence the predicted vocal harmonic locations at each instant in time. Harmonic sinusoidal modeling is then applied to reconstruct the vocal component based on assigning a magnitude and phase to each reconstructed harmonic from a detected sinusoidal peak in the corresponding spectral neighborhood of the mixed signal short-time Fourier transform (STFT). The vocal signal is reconstructed by the amplitude and phase interpolation of the harmonic component tracks. The instrumental background is obtained by the subtraction of the reconstructed vocal from the original mixture. A high degree of vocal separation is obtained when the assumption of vocal dominance holds for the mixture. However some well-known artifacts remain viz. (i) “bleeding through” of some of the melodic instrumentation due to the blind assignment of the total energy in the mixed signal in the vocal harmonic location to the corresponding reconstructed harmonic; this artifact is particularly perceptible in the sustained vowel regions of singing, (ii) improper cancellation of the unvoiced consonants and breathy voice components due to the limitations of sinusoidal modeling of noise and (iii) residual of vocal reverb if present in the original [14]. To address the first shortcoming, recent methods rely on the availability of non-overlapping harmonics of the same source anywhere in the entire audio [3]. We propose to replace the binary mask (implicit in the harmonic-sinusoidal modeling) applied to the vocal harmonics before reconstruction by a soft-mask (a form of Wiener filtering). An effective soft mask would be based on an accurate estimate of the vocal signal spectrum at any time-instant [2], [14]. This would improve the reconstructed vocal signal and lead to more complete suppression in the estimated background.

The vocal signal spectrum depends on several factors such as the singer’s voice, the phone being uttered, the pitch and the vocal effort. We cannot assume the availability of clean data for supervised training (i.e., unaccompanied voice of the particular singer). However popular singers typically have a large number of songs to their
credit, and therefore a method for learning a dictionary of soft masks for the singer from such a training data set could be useful. The training set thus has original single-channel polyphonic songs where the vocal characteristics correspond to the singer but the background orchestration is diverse. We apply non-negative matrix factorization (NMF) methods to estimate the invariant set of basis vectors across multiple instances of the singer’s phones in different songs. In the recent past, several systems have been proposed that qualify as modifications of NMF for improved performance in various scenarios where specific prior knowledge about the data are available [5] (and references therein). In the present work, we attempt to formulate a NMF approach to obtain basis elements corresponding to the singer’s utterances by providing audio corresponding to a particular singer. Given the very diverse spectra of the different phones in a language, the quality of the decomposition can be improved by restricting the optimization to within a phone class [11]. We exploit the availability of song-synchronized lyrics data available in karaoke applications to achieve this. Our main contribution is to combine the advantages of harmonic-sinusoidal modeling in localizing the vocal components in time-frequency with that of soft-masking based on spectral envelope estimates from a NMF decomposition on polyphonic audio training data. Prior knowledge about singer identity and underlying phone transcription of the training and test audio are incorporated in the proposed framework. We develop and evaluate the constrained NMF optimization required for the training across instances where a common basis function set corresponds to the singer-vowel. On the test data, partial co-factorization with a sparseness constraint helps obtain the correct basis decomposition for the mixed signal at any time instant, and thus a reliable spectral envelope estimate of the vowel for use in the soft mask. Finally, the overall system is evaluated based on the achieved vocal and orchestral background separation using objective measures and informal listening. In the next sections, we present the overall system for vocal separation, followed by the proposed NMF-based singer-vowel dictionary learning, estimation of the soft mask for test mixed polyphonic utterances and experimental evaluation of system performance.

2. PROPOSED HYBRID SYSTEM

A block diagram of the proposed hybrid system for vocal separation is shown in Figure 1. The single-channel audio mixture considered for vocal separation is assumed to have the singing voice, when present, as the dominant source in the mix. We assume that the sung regions are annotated at the syllable level, as expected from music audio prepared for karaoke use. A predominant pitch tracker [9] is applied to the sung regions to detect vocal pitch at 10 ms intervals throughout the sung regions of the audio. Sinusoidal components are tracked in the computed short-time magnitude spectrum after biasing trajectory information towards the harmonic locations based on the detected pitch [8]. The pitch salience and total harmonic energy are used to locate the vowel region within the syllable. The vocal signal can be reconstructed from the harmonic sinusoidal component trajectories obtained by amplitude and phase interpolation of the frame-level estimates from the STFT. An estimate of the instantaneous spectral envelope of the singer’s voice provides a soft mask to re-shape the harmonic amplitudes before vocal reconstruction. The mel-filtered spectral envelope (MFS) is computed by applying a 40-band mel-filter bank to the log-linearly interpolated envelope of the mixture harmonic amplitudes. By using the spectral envelope, we eliminate pitch dependence in the soft mask to a large extent. The phoneme dictionary consists of a set of basis vectors for each vowel, at various pitches. A linear combination of these basis vectors may be used to estimate the MFS envelope of the vocal component of the mixture, from the MFS envelope of the mixture. These spectral envelope vectors are learnt from multiple polyphonic mixtures of the phoneme as explained in Section 3. The MFS is used as a low-dimensional perceptually motivated representation. The reconstructed vocal signal is subtracted in the time-domain from the polyphonic mixture to obtain the vocal-suppressed music background.

3. SPECTRAL ENVELOPE DICTIONARY LEARNING USING NMF

To obtain the singer specific soft mask mentioned in the previous section, we create a dictionary of basis vectors corresponding to each of the vowels of the language. This
dictionary is created from polyphonic segment compositions, containing the vowel of the singer under consideration. While spectral envelope of a vowel depends on the vowel identity, there are prominent dependencies on (i) the singer, whose physiological characteristics and singing style affect the precise formant locations and bandwidths for a given vowel. This is especially true of the higher formants (4th and 5th), which depend primarily on the singer rather than on the vowel; (ii) pitch, specifically in singing where the articulation can vary with large changes in pitch due to the “formant tuning” phenomenon [12]; (iii) loudness or vocal effort. Raising the vocal effort reduces spectral tilt, increasing the relative amplitudes of the higher harmonics and consequently the brightness of the voice.

In the proposed dictionary learning, pitch dependence is accounted for by separate dictionary entries corresponding to 2 or 3 selected pitch ranges across the 2 octaves span of a singer. Since the pitch and vowel identity are known for the test song segment, the correct dictionary can be selected at any time. The basis vectors for any pitch range of the singer-vowel capture the variety of spectral envelopes of vocal and instrumental background, there is a common set of bases across the mixtures with changing basis vectors for the accompaniment.

We use NMF to extract common features (singer-vowel spectra) across multiple song segments. The conventional use of NMF is similar to the phoneme-dependent NMF used for speech separation in [7] where the bases are estimated from clean speech. We extend the scope of NMF further, using non-negative matrix partial co-factorization (NMPCF) [4] equivalent to NMF for multiblock data [15]. NMPCF and its variants have been used in drum source separation [4], where one of the training signals is the solo drums audio. Here, we use NMPCF for multiple MFS matrices of mixed signals across segments of the polyphonic audio of the singer, without the use of clean vocal signal. This will yield a common set of bases representing the singer-vowel and other varying bases representative of the accompaniment.

We now describe the NMPCF algorithm for learning the singer-vowel basis. The MFS representation for one specific annotated segment of a polyphonic music is represented as \( \mathbf{V}_i \). This section has the vowel of interest and instrumental accompaniments. We have MFS of \( M \) such mixtures for \( i = 1, \ldots, M \) represented as [15],

\[
\mathbf{V}_i = \mathbf{W}_c \mathbf{H}_{c,i} + \mathbf{W}_{a,i} \mathbf{H}_{a,i}, \quad i = 1, \ldots, M. \tag{1}
\]

where \( \mathbf{V}_{c,i} \) and \( \mathbf{V}_{a,i} \) denote the MFS of the common singer-vowel and accompaniment, respectively. Using NMF decomposition for the MFS spectra we have,

\[
\mathbf{V}_i = \mathbf{W}_c \mathbf{H}_{c,i} + \mathbf{W}_{a,i} \mathbf{H}_{a,i}, \quad i = 1, \ldots, M. \tag{2}
\]

where \( \mathbf{W}_c \in \mathbb{R}^{F \times N_c} \) denotes the basis vectors corresponding to the common vowel and \( \mathbf{W}_{a,i} \in \mathbb{R}^{F \times N_a} \) are the basis vectors corresponding to the accompaniments. Here \( F \) is the number of mel-filters (40) used, \( N_c \) and \( N_a \) are the number of basis vectors for the vowel and accompaniments, respectively. The matrices \( \mathbf{H}_{c,i} \) and \( \mathbf{H}_{a,i} \) are the activation matrices for the vowel and accompaniment basis vectors, respectively. Our objective is to obtain the basis vectors \( \mathbf{W}_c \) corresponding to the common vowel across these \( M \) mixtures. We achieve this by minimizing the Frobenius norm \( || \cdot ||_F^2 \) of the discrepancy between the given mixtures and their factorizations, simultaneously. Accordingly, the cost function,

\[
D = \sum_{i=1}^{M} \frac{1}{2} || \mathbf{V}_i - \mathbf{W}_c \mathbf{H}_{c,i} - \mathbf{W}_{a,i} \mathbf{H}_{a,i} ||_F^2 + \frac{\lambda_1}{2} || \mathbf{W}_{a,i} ||_F^2, \tag{3}
\]

is to be minimized with respect to \( \mathbf{W}_c, \mathbf{W}_{a,i}, \mathbf{H}_{c,i}, \) and \( \mathbf{H}_{a,i} \). The regularizer \( || \mathbf{W}_{a,i} ||_F^2 \) and \( \lambda_1 \) the Lagrange multiplier to denote \( \mathbf{W}_{a,i} \) and \( \mathbf{W}_c \) matrices [15]. The basis vectors thus obtained are a good representation of both the common vowel and the accompaniments, across the mixture. In this work, we choose \( \lambda_1 = 10 \) for our experimentation as it was found to result in the sparsest \( \mathbf{H}_{c,i} \) matrix for varying values of \( \lambda_1 \). We solve (3) using the multiplicative update algorithm. The multiplicative update for a parameter \( \mathbf{P} \) in solving the NMF problem takes the general form,

\[
\mathbf{P} = \mathbf{P} \odot \frac{\nabla \mathbf{X}(\mathbf{D})}{\nabla \mathbf{X}(\mathbf{D})}, \tag{4}
\]

where \( \nabla \mathbf{X}(\mathbf{D}) \) and \( \nabla \mathbf{X}(\mathbf{D}) \) represent the negative and positive parts of the derivative of the cost \( \mathbf{D} \) w.r.t. the parameter \( \mathbf{X} \), respectively, \( \odot \) represents the Hadamard (element-wise) product and the division is also element-wise. Correspondingly, the multiplicative update for the parameter \( \mathbf{W}_c \) in (3) is,

\[
\mathbf{W}_c = \mathbf{W}_c \odot \frac{\nabla \mathbf{W}_c(\mathbf{D})}{\nabla \mathbf{W}_c(\mathbf{D})}. \tag{5}
\]

where,

\[
\nabla \mathbf{W}_c(\mathbf{D}) = \sum_{i=1}^{M} (\mathbf{W}_c \mathbf{H}_{c,i} + \mathbf{W}_{a,i} \mathbf{H}_{a,i} - \mathbf{V}_i) \mathbf{H}_{c,i}^T. \tag{6}
\]

Similarly, the update equation for other terms in (3) are,

\[
\mathbf{H}_{c,i} = \mathbf{H}_{c,i} \odot \frac{\mathbf{W}_c^T \mathbf{V}_i}{\mathbf{W}_c^T \mathbf{W}_c \mathbf{H}_{c,i} + \mathbf{W}_{a,i}^T \mathbf{W}_{a,i} \mathbf{H}_{a,i}}, \tag{7}
\]

\[
\mathbf{H}_{a,i} = \mathbf{H}_{a,i} \odot \frac{\mathbf{W}_c^T \mathbf{V}_i}{\mathbf{W}_{a,i}^T \mathbf{W}_c \mathbf{H}_{c,i} + \mathbf{W}_{a,i}^T \mathbf{W}_{a,i} \mathbf{H}_{a,i}}. \tag{8}
\]
\[ W_{a,i} = \frac{V_i H_{a,i}^T}{W_i H_{c,i} H_{a,i}^T + W_{a,i} H_{a,i} H_{a,i}^T + \lambda_1 \times W_{a,i}} \]

for \( i = 1, \ldots, M \). The basis vectors \( W_c \) for the various phonemes form the dictionary and act as a prior in the spectral envelope estimation. Each dictionary entry is associated with a vowel and pitch range. We denote each entry in the dictionary as \( W_c(/p/, f_0) \) for each vowel /p/ at pitch \( f_0 \).

### 4. SOFT MASK ESTIMATION USING SINGER-VOWEL DICTIONARY

In this section, we describe the approach to estimate the frame-wise soft mask for a test polyphonic vowel mixture segment. We first obtain the MFS envelope for the mixture as mentioned in Section 2. With the vowel label and segment. We first obtain the MFS envelope for the mixture. We do this by minimizing the cost function

\[ D_T = \frac{1}{2} \| V_T - W_c H_{c,T} - W_{a,T} H_{a,T} \|^2_F + \lambda_2 \frac{1}{2} \| H_{c,T} \|^2_F, \]

where the subscript \( T \) refers to the test case. The minimization is done with the dictionary bases \( W_c \) kept fixed and using multiplicative updates for \( H_{c,T} \), \( W_{a,T} \) and \( H_{a,T} \). The sparsity constraint on \( H_{c,T} \) in (10) accounts for the fact that the best set of bases representing the vowel would result in the sparsest temporal matrix \( H_{c,T} \). Under this formulation, \( W_c H_{c,T} \) will give an estimate of the vowel’s MFS envelope \( V_c \) (as in (1)) for the mixture. An alternate way is to use Wiener filtering to estimate \( V_c \), as

\[ \hat{V}_c = \frac{W_c H_{c,T}}{W_c H_{c,T} + W_{a,T} H_{a,T}} \odot V_T. \]

This estimated vowel MFS can be used to reconstruct the spectral envelope of the vowel \( \hat{C} \). This is done by multiplying \( \hat{V}_c \) with the pseudoinverse of the DFT matrix \( M \) of the mel filter bank [1] as \( \hat{C} = M^{+} \hat{V}_c \). A soft mask corresponding to this spectral envelope can be obtained using the Gaussian radial basis function [2],

\[ G_b(f, t) = \exp \left( - \frac{(\log X(f, t) - \log \hat{C}(f, t))^2}{2\sigma^2} \right) \]

where, \( \sigma \) is the Gaussian spread, \( X \) is the magnitude spectrum of the mixed signal. The soft mask (12) is evaluated with \( \sigma = 1 \), in a 50 Hz band around the pitch \( f(0) \) and its harmonics [14].

Having obtained the soft mask, the vocals track is reconstructed by multiplying the soft mask with the harmonic amplitudes of the sinusoidally modeled signal. The synthesized signal then corresponds to the reconstructed vocals. The accompaniment can be obtained by performing a time-domain subtraction of the reconstructed vocals from the original mixture.

### 5. EXPERIMENTS AND PARAMETER CHOICES

Given a polyphonic vowel segment, the vocal is separated by applying the generated soft mask corresponding to the given mixture. We compare the separated vocal with the ground truth to evaluate the performance. The performance evaluation of the proposed system is carried out in two steps. The first step is to choose the parameters of the system using the distance in the MFS space between the estimated and ground-truth MFS vectors obtained from the clean utterance. The second step is the computation of signal-to-distortion (SDR) measure (in dB) on the separated vocal and instrumental time-domain signals which will be given in Section 6. We present the training and test data used in the experiments next.

#### 5.1 Description of the Dataset

The training dataset comprised of nine instances of three vowels viz., /a/, /i/, /o/ at two average pitches of 200 Hz and 300 Hz and sung by a male singer over three different songs with their accompaniments, annotated at the phoneme level. The training data was chosen so as to have different accompaniments across all the instances of a vowel. The training audios thus contained the vowel utterances throughout in the presence of background accompaniments. The training mixtures were pre-emphasised using a filter with a zero located at 0.7 to better represent the higher formants. A dictionary of bases was created for all the vowels for the two pitch ranges using the NMCPF optimization procedure discussed in Section 3. The performance was evaluated over a testing dataset of 45 test mixtures with 15 mixtures for each vowel over the two pitch ranges. The mixtures used for testing were distinct from the training mixtures. Since the audios were obtained directly from full songs, there was a significant variation in terms of the pitch of the vowel utterances around the average pitch ranges and in terms of coarticulation. The training and testing mixtures were created in a karaoke singing context and hence, we had available, the separate vocal and accompaniment tracks to be used as ground truth in the performance evaluation. All the mixtures had durations in the range of 400 ms - 2.2 s and were sampled at a frequency of 16 kHz. The window size and hop size used for the 1024 point STFT were 40 ms and 10 ms, respectively.

#### 5.2 Choice of Parameters

There are several training parameters likely to influence the performance of the system. These include the ranks of the matrices in the decomposition \( W_c, W_{a,i} \) and the number of mixtures \( M \). We obtain these parameters experimentally using a goodness-of-fit measure. The goodness-of-fit is taken to be the normalised Frobenius norm of the
difference between the ideal envelope of a vowel in the MFS domain \( V_c \) and its best estimate \( \hat{V}_c \) obtained as a linear combination of the bases, for a mixture containing the vowel and accompaniment. This estimate can be calculated as explained in Section 4. Lower the value of this distance measure, closer is the envelope estimate to the ideal estimate. The various bases may be compared by calculating \( D_i \) for different bases \( W_{c_i} \) using

\[
D_i = \frac{\| V_c - \hat{V}_{c_i} \|^2_{\text{F}}}{\| V_c \|^2_{\text{F}}}
\]

\[
= \frac{\| V_c - \frac{W_{c_i} H_{c_i}}{W_{c_i} H_{c_i} + W_{c} H_{c}} \odot V_T \|^2_{\text{F}}}{\| V_c \|^2_{\text{F}}},
\]

(13)

and comparing the same. To account for variabilities in the evaluation mixtures, the distance measure is evaluated and averaged over a number of mixtures and combinations, for each set of bases \( W_{c_i} \). The goodness-of-fit is used only to choose the appropriate parameter values for the system. The performance of the overall system, however, is evaluated in terms of SDR.

As shown in Figure 2, the goodness-of-fit measure decreases with increasing rank of the decomposition (number of vocal basis vectors) for a given \( M \). The decreasing trend flattens out and then shows a slight increase beyond rank 35. For a fixed rank, the goodness-of-fit improves with increasing number of mixtures. Of the configurations tried, the distance measure is minimum when four mixtures (\( M = 4 \)) are used in the NMPFCF optimization to obtain the dictionary. Thus, a rank 35 decomposition with \( M = 4 \) is chosen for each singer-vowel dictionary for system performance evaluation.

As for the rank of the accompaniment basis, it is observed that the regularization term in the joint optimization (3) seems to make the algorithm robust to choice of number of basis vectors for the accompaniment. Eight basis vectors were chosen for each mixture term in the joint optimization. Although the number of accompaniment bases seems to be comparatively low, eight bases were sufficient to reconstruct the accompaniment signals from the mixtures. A high value for \( \lambda_2 \) in the test optimization problem of (10) results in a decomposition involving a linear combination of the least number of bases per time frame. This sparse decomposition may not necessarily lead to the best reconstruction in more challenging scenarios involving articulation variations. Thus a small value of \( \lambda_2 = 0.1 \) was chosen.

### 6. RESULTS AND DISCUSSION

We evaluate the performance of the system using the SDR. The SDR is evaluated using the BSS_eval toolbox [13]. The SDR values averaged across 45 vowel test mixtures, separately for the reconstructed vocals and instrumental background are given in Table 1. To appreciate the improvement, if any, the SDR is also computed for the harmonic sinusoidal model without soft masking (i.e., binary masking only). While the proposed soft masking shows an increase in SDR, closer examination revealed that the improvements were particularly marked for those mixtures with overlapping vocal and instrumental harmonics (accompaniments) in some spectral regions. This was also borne out by informal listening. When we isolated these samples, we observed SDR improvements of up to 4 dB in several instances. This is where the selective attenuation of the harmonic amplitudes in accordance with the estimated vowel spectral envelope is expected to help most. The harmonics in the non-formant regions are retained in the instrumental background rather than being canceled out as in binary masking, contributing to higher SDR.

To understand the singer dependence of the dictionary, we carried out soft mask estimation from the polyphonic test mixtures using the basis vectors of an alternate singer. This basis was a set of clean vowel spectral envelopes obtained from another male singer’s audio with the same vowels and pitches corresponding to our training dataset. We observe from Table 1 that the alternate singer soft mask does better than the binary mask, since it brings in the vowel dependence of the soft mask. However, it does not perform as well as the original singer’s soft mask even though the latter is obtained from clean vowel utterances. As depicted in Figure 3 (for a sample case), the envelope obtained using the original singer’s data closely follows the

<table>
<thead>
<tr>
<th></th>
<th>Separated track</th>
<th>Binary mask</th>
<th>Soft mask</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vocal</td>
<td>8.43</td>
<td>9.06</td>
<td>8.66</td>
</tr>
<tr>
<td>Instrumental</td>
<td>13.63</td>
<td>14.16</td>
<td>14.10</td>
</tr>
</tbody>
</table>

Table 1. SDR values (in dB) for separated vocals and instruments obtained using a binary mask, soft mask from the original singer’s training mixtures and soft mask from an alternate singer’s vocals.

---

1 Audio examples are available at [http://www.ee.iitb.ac.in/student/ndlab/ISMIR_webpage/webpageISMIR.html](http://www.ee.iitb.ac.in/student/ndlab/ISMIR_webpage/webpageISMIR.html).
ideal envelope of the phoneme.

Although the NMF optimization converges slowly, the number of iterations to be carried out to obtain the envelope is low, for both training and testing procedures. It is observed that the bases and envelopes attain their final structure after 4000 and 1000 iterations, respectively.

7. CONCLUSION

Soft masks derived from a dictionary of singer-vowel spectra are used to improve upon the vocal-instrumental music separation achieved by harmonic sinusoidal modeling for polyphonic music of the particular singer. The main contribution of this work is an NMF based framework that exploits the amply available original polyphonic audios of the singer as training data for learning the dictionary of singer spectral envelopes. Appropriate constraints are introduced in the NMF optimization for training and test contexts. The availability of lyric-aligned audio (and therefore phone labels) helps to improve the homogeneity of the training data and have a better model with fewer basis vectors. Significant improvements in reconstructed signal quality are obtained over binary masking. Further it is demonstrated that a vowel-dependent soft mask obtained from clean data of a different available singer is not as good as the singer-vowel dependent soft mask even if the latter is extracted from polyphonic audio.

8. ACKNOWLEDGEMENT

Part of the work was supported by Bharti Centre for Communication in IIT Bombay.

9. REFERENCES


ABSTRACT

Query by tapping (QBT) is a content-based music retrieval method that can retrieve a song by taking the user’s tapping or clapping at the note onsets of the intended song in the database for comparison. This paper proposes a new query-by-tapping algorithm that aligns the IOI (inter-onset interval) vector of the query sequence with songs in the dataset by building an IOI ratio matrix, and then applies a dynamic programming (DP) method to compute the optimum path with minimum cost. Experiments on different datasets indicate that our algorithm outperforms other previous approaches in accuracy (top-10 and MRR), with a speedup factor of 3 in computation. With the advent of personal handheld devices, QBT provides an interesting and innovative way for music retrieval by shaking or tapping the devices, which is also discussed in the paper.

1. INTRODUCTION

QBT is a mechanism for content-based music retrieval which extracts the note onset time from recordings of users’ input tapping or symbolic signals, which it then compares against a song database to retrieve the correct song. Unlike query-by-singing/humming (QBSH) [1, 2], which takes the user's melody pitch for comparison, QBT only uses the note duration for comparison, with no pitch information. This makes QBT more difficult to implement than QBSH, because the note onset in QBT contains less information than the musical pitch in QBSH, raising the likelihood of collision. For example, musical pieces with different melodies but similar rhythmic patterns may be characterized by the same onset sequence.

One may argue that QBT is not a popular way of music retrieval. Some people may even think it is not useful. However, with the advent of personal handheld devices, we can think QBT as a novel way of human-computer interface. For instance, with the use of QBT, one may shake or click his/her mobile phones in order to retrieve a song. Moreover, one can even use a personal style of shaking or clicking as the password to unlock a phone. These innovative ways of human-machine interface indicate that QBT, though not the most popular way of music retrieval, is itself interesting and could pave the way for other innovative applications [10].

QBT system algorithms are based on the estimation of the similarity between two onset sequences. For example, G. Eisenberg proposed a simple algorithm called "Direct Measure" to accomplish such comparisons [3, 4]. R. Typke presented a variant of the earth mover's distance appropriate for searching rhythmic patterns [5]. Among these algorithms, the techniques of dynamic programming (DP) have been widely used, such as R. Jang's Dynamic Time Warping (DTW) [6], G. Peters’s edit distance algorithm [7, 8], and P. Hanna’s adaptation of local alignment algorithm [9].

In this paper, we propose and test a new QBT algorithm. In Section 2, we discuss the general frameworks of QBT and existing QBT methods. Section 3 describes the proposed method. Experiments with different QBT techniques are described in Section 4. Finally, Section 5 concludes this paper.

2. THE QBT SYSTEM

Fig. 1 illustrates the flowchart of our query-by-tapping system. In general, there are 2 kinds of inputs to a QBT system:

- Symbolic input: The onset time of the tapping event is provided symbolically with little or no ambiguity. For instance, the user may tap on a PC’s keyboard or an iPad’s touch panel to give the onset time.
- Acoustic input: The onset time is extracted from acoustic input of the user’s tapping on a microphone. This input method requires additional onset detection to extract the onset time of the acoustic input. For example, we can estimate the onset time by local-maximum-picking of the input audio’s intensity as in [5], or by detecting the transients of kurtosis variation as in [7].

The input onset sequence can be obtained as the inter-onset interval (IOI) vector whose elements are the difference between two successive onset times. The note onset sequences extracted from the monophonic MIDIs (or the melody track of polyphonic MIDIs) in the song database are also converted into IOIs in advance. We can then apply a QBT algorithm to compare the query IOI vector to those in the database in order to retrieve the most similar song from the database. A QBT algorithm usually needs to perform IOI vector normalization before similarity comparison. Normalization can take care of tempo devia-
tion, which similarity comparison can handle possible insertion/deletion errors. Once normalization is performed, we can apply similarity comparison to find the similarity between the IOI query vector and that of each database song. The system can then return a ranked list of all database songs according to their similarity to the query input.

Normalization and the similarity comparison are detailed in the following sections.

2.1 Normalization of IOI Vectors

In most cases, the tempo of the user’s query input is different from those of the candidate songs in the database. To deal with this problem, we need to normalize the IOI vectors of the input query and the candidate songs. There are 2 common methods for normalization. The first one is to convert the summation of all IOI to a constant value [5]:

$$\tilde{q}_i = q_i / \sum_{k=1}^{n} q_k$$  \hspace{0.5cm} (1)

$$\tilde{r}_j = r_j / \sum_{k=1}^{n} r_k$$

where \{q_i, i=1~m\} is the input query IOI vector, and \{r_j, j=1~n\} is a reference IOI vector from the song database. Note that the reference IOI vector of a song is truncated to a variety of lengths in order to match the query IOI. For instance, \(n\) may be set to a value from \(m-2\) to \(m+2\) in order to deal with possible insertions and deletions in the query input. Thus all these variant normalized versions of the IOI vectors for a song must be compared for similarity with the query IOI vector. The second method is to represent the normalized IOI vector as the ratio of the current IOI element to its preceding element [7]. That is:

$$\tilde{s}_i = 1$$

$$\tilde{s}_i = s_i / s_{i-1}, \text{if } i \geq 2$$  \hspace{0.5cm} (2)

where \{s_i\} is the original input query or reference IOI vector, and \{\tilde{s}_i\} is its normalized version. The advantage of this method is that computation-wise it is much simpler than the first one. However, this method is susceptible to the problem of magnified insertion and deletion errors of the original IOI vectors, if any. For example, an IOI vector is \([1, 2, 1]\), then its normalized vector is \([1, 2, 0.5]\). If this IOI vector is wrongly tapped as \([1, 1, 1, 1]\) (i.e., with one insertion in the second IOI), the normalized will become \([1, 1, 1, 1]\), which has a larger degree of difference from the groundtruth after normalization. This kind of amplified difference is harder to recover in the step of similarity comparison.

2.2 Similarity Comparison

A robust QBT system should be able to handle insertion and deletion errors since most of the common users are not likely to tap the correct note sequence of the intended song precisely. In particular, a common user is likely to lose one or several notes when the song has a fast tempo, which leads to deletion errors. On the other hand, though less likely, a user may have a wrong impression of the intended song and taps more notes instead, which lead to insertion errors. Several methods have been proposed to compare IOI vectors for QBT, including the earth mover’s distance [4] and several DP-based methods [5], [6], [7] which can deal with two input vectors of different lengths. In general, the earth mover’s distance is faster than DP-based methods, but its retrieval accuracy is not as good [11]. Our goal is to obtain a good accuracy with a reasonable amount of computing time. Therefore, the proposed method is based on a highly efficient DP-based method for better accuracy.

3. THE SHIFTED ALIGNMENT ALGORITHM

This section presents the proposed method to QBT. The method can also be divided into two stages of IOI normalization and similarity comparison. We shall describe these two steps and explain the advantages over the state-of-art QBT methods.

Normalisation: In QBT, though the query IOI vector and its target song IOI vector are not necessarily of the same size, the ratio of their tempos should be close to a constant. In other words, the ratios of an IOI element of a query to the corresponding one of the target song should be close to a constant. To take advantage of this fact, we can shift the query IOI vector (relatively to the target song IOI vector) to construct an IOI ratio matrix in order to find the optimum mapping between IOI elements of these two sequences. An example is shown in Fig. 2(a), where the input query IOI vector is represented by \{q_i, i=1~m\}, and the reference IOI vector from the song database by \{r_j, j=1~n\}. As displayed in the figure, the reference IOI vector is shown at the top and the shifted query IOI vectors are shown below. Each element of a shifted query IOI vector is mapped to that of the reference IOI vector in the same column. Take the first shifted query IOI vector as an example, its second element \(q_2\) is mapped to \(r_1\) of the reference IOI vector, \(q_3\) is mapped to \(r_2\), etc. For each matched element pair, we divide the...
Fig. 2. Example of the normalization step of the shifted alignment algorithm: (a) Reference IOI vector and the shifted query IOI vectors. (b) IOI ratio matrix.

query IOI by its mapping reference IOI to construct an IOI ratio matrix $M$ according to the following formula:

$$M_{i,j} = \begin{cases} 
q_{i-1,j-1}/r_j & \text{if } 1 \leq i - i + j + 1 \leq \min(m,n) \\
0 & \text{otherwise}
\end{cases}$$

(3)

where the size of the matrix $M$ is $\min(m,n) \times (i_e + i_l + 1)$. $i_e$ and $i_l$ are the left- and right-shift amount of the query IOI vector, respectively. Fig. 2(b) is the IOI ratio matrix of fig. 2(a). In this example, $i_e$ and $i_l$ are 1 and 2, respectively. Since the length of the query is usually shorter, $m$ is generally much less than $n$. Besides, in practice, if the anchor position is the beginning of a song, then we can improve the computation efficiency by truncating a reference IOI vector to a length slightly longer (e.g., 5-element longer) than the length of query IOI vector.

Unlike the equation (1) which requires many different versions of normalized reference IOI vectors for similarity comparison, the proposed approach requires only one-time normalization to generate a single IOI ratio table for computing the similarity. So the proposed approach is guaranteed to more efficient.

**Similarity comparison:** In order to handle insertions and deletions in a flexible yet robust manner, we propose a dynamic programming method to compute the similarity between the query and the reference IOI vectors. The basic principle is to identify a path over the IOI ratio matrix $M$ where the elemental values along the path should be as close as possible to one another. In other words, the accumulated IOI ratio variation should be minimal along the optimal path. Fig. 3 illustrates two typical numeric examples that involve insertion and deletion in the optimal path. In fig. 3(a), query IOI vector and reference IOI vector have the same tempo, so their elements are pretty much the same except that there is an insertion in the query. That is, the fourth element of the reference IOI vector is equally split into 2 elements in the query. Fig. 3(b) is the IOI ratio matrix derived from the fig. 3(a), with the optimal path surrounded by dashed lines. The horizontal direction within the optimal path represent one-to-one sequential mapping between the two vectors without insertion or deletion. The vertical direction within the path indicates an insertion, where the 4th and 5th query IOI elements should be mapped to the 4th reference IOI element. On the other hand, Fig. 3(c) demonstrates an example of deletion where the query misses the 4th onset of the reference vector. Fig. 3(d) shows the corresponding IOI ratio matrix with the optimal path surrounded by dashed lines. The vertical shift of the path indicates a deletion where the 4th query IOI element should be mapped to the 4th and 5th reference IOI elements.

If there is no insertion or deletion in the query, each element along the optimal path should have a value close to its preceding element. With insertion or deletion, then the optimal path exhibits some specific behavior. Therefore our goal is to find the optimal path with minimal variations between neighboring elements in the path, with special consideration for specific path behavior to accommodate insertion and deletion. The variation between neighboring IOI ratio elements can be represented as the deviation between 1 and the ratio of one IOI ratio element to the preceding modified IOI ratio element, which takes into consideration the specific path behavior for accommodating insertion and deletion. The resulting recurrent equation for the optimum-value function $D_{i,j}$ for DP is shown next:
The size of the DP matrix is the least of the DP algorithms in [6], [7], [9]. In addition, our algorithm can be easily extended to the QBT system with “anywhere” anchor positions by setting the $i_e$ to the length of the reference IOI vector.

4. PERFORMANCE EVALUATION

To evaluate the proposed method, we design 3 experiments and compare the performance with that of the state-of-art algorithm. The first experiment compares the recognition rate with algorithms in MIREX QBT task. The second experiment compares their computing speeds. The third experiment demonstrates the robustness of the proposed method using a larger dataset. These experiments are described in the following sub-sections.

4.1 MIREX QBT Evaluation Task

We have submitted our algorithm to the 2012 MIREX QBT task [12], which involves two subtasks for symbolic and acoustic inputs, respectively. Because the onset detection of acoustic input is not the focus of this paper, the following experiments only consider the case of queries with symbolic input. There are 2 datasets of symbolic input, including Jang’s dataset of 890 queries (with groundtruth onsets to be used as the symbolic input) and 136 monophonic MIDIs, and Hsiao’s dataset of 410 symbolic queries and 143 monophonic MIDIs. The queries of both datasets are all tapped from the beginning of the target song. These datasets are published in 2009 and can be downloaded from the MIREX QBT webpage. The top-10 hit rate and the mean reciprocal rank (MRR) are used as the performance indices of a submitted QBT method. Fig. 5 shows the performance of 5 submitted algorithms, with (a) and (b) are respectively the results of Jang’s and Hsiao’s datasets. Out of these five submissions, “HL” do not have clear descriptions about their algorithms in the MIREX abstracts. Therefore, these 2 algorithms are not included in the experiments in section 4.2 and 4.3. “HAFR” is the implementation of [9], which claimed that its results outperformed other submissions, including the methods of [5] and [6], in MIREX 2008. The algorithm “CJ” is an improved version of [6]. The submission of “SA” is the proposed algorithm in this paper.

As shown in fig. 5(a), our algorithm outperforms almost all the other submissions except for the MRR in Jang’s dataset where our submission is ranked the second. In fact, the MRR of our algorithm is only 0.3% lower than that of “CJ”. On the other hand, the top-10 hit rate of our submission is 0.3% higher than that of “CJ”. So the performances of “CJ” and “SA” are very close in this dataset. From fig. 5(b), it is obvious that our algorithm simply outperforms all the other submission in both MRR and top-10 hit rate. As a whole, the proposed method obtains good results in MIREX QBT contest.
4.2 Evaluation of Computation Efficiency

In this experiment, we want to compare the efficiency of several QBT algorithms. We implemented three submissions (including ours) to 2012 MIREX QBT tasks in C language. The “ML” and “HL” algorithms were not included in this experiment due to the lack of clear descriptions about their algorithms in the MIREX abstracts. The experiment was conducted on a PC with an AMD Athlon 2.4GHz CPU and 1G RAM. Each algorithm was repeated 10 times over Jang’s dataset to obtain the average computing time of a single query. The results are shown in Table 1 which indicates that our algorithm is at least 3 times faster than the other two algorithms. This is due to the fact that our algorithm has an efficient way of normalization for IOI vectors (as described in section 3), leading to a smaller table for DP optimal path finding.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Top 10</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>HL</td>
<td>0.888</td>
<td>0.784</td>
</tr>
<tr>
<td>ML</td>
<td>0.876</td>
<td>0.797</td>
</tr>
<tr>
<td>HA FR</td>
<td>0.876</td>
<td>0.770</td>
</tr>
<tr>
<td>CJ</td>
<td>0.908</td>
<td>0.840</td>
</tr>
<tr>
<td>SA</td>
<td>0.821</td>
<td>0.837</td>
</tr>
</tbody>
</table>

Table 1. Speed comparison of QBT algorithms

From these two experiments, we can claim that our algorithm strike a good balance between the recognition rate and computation efficiency.

4.3 Experiment with Larger Databases

The MIREX QBT datasets are well organized for QBT research. However, both datasets contain song databases of slightly more than 100 songs. These small database sizes lead to high accuracy for all submissions in MIREX QBT task. Therefore, we designed another experiment to demonstrate how the performance varies with the dataset sizes. We collected 1000 MIDIs which are different from the MIDIs in the MIREX QBT datasets. And we enlarge the original databases by adding 250 noise MIDIs each time, and evaluate the performance in both MRR and top-10 hit rate.

Fig. 6 shows the experimental results. As the number of noise MIDIs increases, the recognition rate of each algorithm gradually decreases. In Jang’s dataset of the fig. 6(a), the top-10 hit rate of “SA” is the best among all algorithms (left subplot). However, the MRR of “SA” and “CJ” are very close and the value of one is slightly higher than the other in different number of noise MIDIs (right subplot). In fig. 6(b), our algorithm notably outperforms the others in both top-10 hit rate (left subplot) and MRR (right subplot). It is interesting to note that the decay of the top-10 hit rate of “SA” is slower than the others in both datasets, especially in Jang’s dataset. This indicates that our algorithm has better resistance to these noise MIDIs in top-10 hit rate. In both datasets, “SA” still had >85% top-10 rate and >60% MRR. Therefore we can conclude that the proposed method is more robust in dealing with a large song database.

5. CONCLUSION

In this paper, we have proposed a shifted-alignment algorithm for QBT by constructing an IOI ratio matrix, in which each element is the ratio of relative IOI elements of the query and a reference song. The similarity comparison is based on DP to deal with possible insertions and deletions of query IOI vectors. We evaluated the performance of the proposed method with two datasets. The experimental results showed that our algorithm exhibited

Fig. 5. Results of MIREX QBT evaluation task

(a) Result 1: Jang’s dataset

(b) Result 2: Hsiao’s dataset

Fig. 6. Results of the performance versus database sizes. (a) is the performance of top-10 hit rate (left subplot) and MRR (right subplot) using Jang’s dataset. (b) is the performance of top-10 hit rate (left subplot) and MRR (right subplot) using Hsiao’s dataset.
an overall better accuracy than other submissions to 2012 MIREX query-by-taping task. Moreover, the computation time is at least 3 times faster than others. We also conducted an experiment to demonstrate that our algorithm performs better and more robustly than other existing QBT algorithms in the case of large databases. In particular, our algorithm has a top-10 hit rate larger than 85% and MRR larger than 60% in both databases when the number of noise MIDIs is as high as 1000.

Although the proposed method performs well in the experiments, the recognition rate still has room for further improvement, especially in the case of “anywhere” anchor position, that is, the user is allowed to start tapping from anywhere in the middle of a song. From the experimental results, we can observe that each algorithm has its strength and weakness in dealing with different queries and database songs. Therefore, one direction of our immediate future work is to find an optimal way to combine these methods for better accuracy.

6. ACKNOWLEDGEMENT

This study is conducted under the "NSC 102-3114-Y-307-026 A Research on Social Influence and Decision Support Analytics" of the Institute for Information Industry which is subsidized by the National Science Council.

7. REFERENCES


AUTOMATIC TIMBRE CLASSIFICATION OF ETHNOMUSICOLOGICAL AUDIO RECORDINGS

Dominique Fourer, Jean-Luc Rouas, Pierre Hanna, Matthias Robine
LaBRI - CNRS UMR 5800 - University of Bordeaux
{fourer, rouas, hanna, robine}@labri.fr

ABSTRACT

Automatic timbre characterization of audio signals can help to measure similarities between sounds and is of interest for automatic or semi-automatic databases indexing. The most effective methods use machine learning approaches which require qualitative and diversified training databases to obtain accurate results. In this paper, we introduce a diversified database composed of worldwide non-western instruments audio recordings on which is evaluated an effective timbre classification method. A comparative evaluation based on the well studied Iowa musical instruments database shows results comparable with those of state-of-the-art methods. Thus, the proposed method offers a practical solution for automatic ethnomusicological indexing of a database composed of diversified sounds with various quality. The relevance of audio features for the timbre characterization is also discussed in the context of non-western instruments analysis.

1. INTRODUCTION

Characterizing musical timbre perception remains a challenging task related to the human auditory mechanism and to the physics of musical instruments [4]. This task is full of interest for many applications like automatic database indexing, measuring similarities between sounds or for automatic sound recognition. Existing psychoacoustical studies model the timbre as a multidimensional phenomenon independent from musical parameters (e.g. pitch, duration or loudness) [7, 8]. A quantitative interpretation of instrument’s timbre based on acoustic features computed from audio signals was first proposed in [9] and pursued in more recent studies [12] which aim at organizing audio timbre descriptors efficiently. Nowadays, effective automatic timbre classification methods [13] use supervised statistical learning approaches based on audio signals features computed from analyzed data. Thus, the performance obtained with such systems depends on the taxonomy, the size and the diversity of training databases. However, most of existing research databases (e.g. RWC [6], Iowa [5]) are only composed of common western instruments annotated with specific taxonomies. In this work, we revisit the automatic instrument classification problem from an ethnomusicological point of view by introducing a diversified and manually annotated research database provided by the Centre de Recherche en Ethno-Musicologie (CREM). This database is daily supplied by researchers and has the particularity of being composed of uncommon non-western musical instrument recordings from around the world. This work is motivated by practical applications to automatic indexing of online audio recordings database which have to be computationally efficient while providing accurate results. Thus, we aim at validating the efficiency and the robustness of the statistical learning approach using a constrained standard taxonomy, applied to recordings of various quality. In this study, we expect to show the database influence, the relevance of timbre audio features and the choice of taxonomy for the automatic instrument classification process. A result comparison and a cross-database evaluation is performed using the well-studied university of Iowa musical instrument database. This paper is organized as follows. The CREM database is introduced in Section 2. The timbre quantization principle based on mathematical functions describing audio features is presented in Section 3. An efficient timbre classification method is described in Section 4. Experiments and results based on the proposed method are detailed in Section 5. Conclusion and future works are finally discussed in Section 6.

2. THE CREM ETHNOMUSICOLOGICAL DATABASE

The CREM research database is composed of diversified sound samples directly recorded by ethnomusicologists in various conditions (i.e. no recording studio) and from diversified places all around the world. It contains more than 7000 hours of audio data recorded since 1932 to nowadays using different supports like magnetic tapes or vinyl discs. The vintage audio recordings of the database were carefully digitized to preserve the authenticity of the originals and contain various environment noise. The more recent audio recordings can be directly digital recorded with a high-quality. Most of the musical instruments which com-

1 CREM audio archives freely available online at: http://archives.crem-cnrs.fr/
pose this database are non-western and can be uncommon while covering a large range of musical instrument families (see Figure 1(a)). Among uncommon instruments, one can find the lute or the Ngba harp as cordophones. More uncommon instruments like Oscillating bamboo, struck machine and struck girder were classified by ethnomusicologists as idiophones. In this paper, we restricted our study to the solo excerpts (where only one monophonic or polyphonic instrument is active) to reduce the interference problems which may occur during audio analysis. A description of the selected CREM sub-database is presented in Table 1. According to this table, one can observe that this database is actually inhomogeneous. The aerophones are overrepresented while membranophones are underrepresented. Due to its diversity and the various quality of the composing sounds, the automatic ethnomusicological classification of this database may appear as challenging.

<table>
<thead>
<tr>
<th>Class name</th>
<th>Duration (s)</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>aerophones-blowed</td>
<td>1,383</td>
<td>146</td>
</tr>
<tr>
<td>cordophones-struck</td>
<td>357</td>
<td>37</td>
</tr>
<tr>
<td>cordophones-struck</td>
<td>715</td>
<td>128</td>
</tr>
<tr>
<td>cordophones-plucked</td>
<td>1,229</td>
<td>75</td>
</tr>
<tr>
<td>cordophones-bowed</td>
<td>157</td>
<td>16</td>
</tr>
<tr>
<td>idiophones-struck</td>
<td>522</td>
<td>58</td>
</tr>
<tr>
<td>idiophones-plucked</td>
<td>375</td>
<td>14</td>
</tr>
<tr>
<td>idiophones-plucked</td>
<td>94</td>
<td>82</td>
</tr>
<tr>
<td>membranophones-struck</td>
<td>170</td>
<td>19</td>
</tr>
<tr>
<td>Total</td>
<td>3,535</td>
<td>375</td>
</tr>
</tbody>
</table>

Table 1. Content of the CREM sub-database with duration and number of 10-seconds segmented excerpts.

3. TIMBRE QUANTIZATION AND CLASSIFICATION

3.1 Timbre quantization

Since preliminaries works on the timbre description of perceived sounds, Peeters et al. proposed in [12] a large set of audio features descriptors which can be computed from auditory signals. The audio descriptors define numerical functions which aim at providing cues about specific acoustic features (e.g. brightness is often associated with the spectral centroid according to [14]). Thus, the audio descriptors can be organized as follows:

- Temporal descriptors convey information about the time evolution of a signal (e.g. log attack time, temporal increase, zero-crossing rate, etc.).
- Harmonic descriptors are computed from the detected pitch events associated with a fundamental frequency ($F_0$). Thus, one can use a prior waveform model of quasi-harmonic sounds which have an equally spaced Dirac comb shape in the magnitude spectrum. The total part of sounds can be isolated from signal mixture and be described (e.g. noisiness, inharmonicity, etc.).
- Spectral descriptors are computed from signal time-frequency representation (e.g. Short-Term Fourier Transform) without prior waveform model (e.g. spectral centroid, spectral decrease, etc.).
- Perceptual descriptors are computed from auditory-filtered bandwidth versions of signals which aim at approximating the human perception of sounds. This can be efficiently computed using Equivalent Rectangular Bandwidth (ERB) scale [10] which can be combined with gammatone filter-bank [3] (e.g. loudness, ERB spectral centroid, etc.).

In this study, we focus on the sound descriptors listed in Table 2 which can be estimated using the timbre toolbox [2] and detailed in [12]. All descriptors are computed for each analyzed sound excerpt and may return null values. The harmonic descriptors of polyphonic sounds are computed using the prominent detected $F_0$ candidate (single $F_0$ estimation).

To normalize the duration of analyzed sound, we separated each excerpt in 10-seconds length segments without distinction of silence or pitch events. Thus, each segment is represented by a real vector where the corresponding time series of each descriptor is summarized by a statistic. The median and the Inter Quartile Range (IQR) statistics were chosen for their robustness to outliers.

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Descriptor name</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>Att</td>
<td>Attack duration (see ADSR model [15])</td>
<td>1</td>
</tr>
<tr>
<td>AttSp</td>
<td>Attack slope (ADSR)</td>
<td>1</td>
</tr>
<tr>
<td>Dec</td>
<td>Decay duration (ADSR)</td>
<td>1</td>
</tr>
<tr>
<td>DecSp</td>
<td>Decay slope (ADSR)</td>
<td>1</td>
</tr>
<tr>
<td>Rel</td>
<td>Release duration (ADSR)</td>
<td>1</td>
</tr>
<tr>
<td>LAT</td>
<td>Log Attack Time</td>
<td>1</td>
</tr>
<tr>
<td>Tcent</td>
<td>Temporal centroid</td>
<td>1</td>
</tr>
<tr>
<td>Edur</td>
<td>Effective duration</td>
<td>1</td>
</tr>
<tr>
<td>FreqMod, AmpMod</td>
<td>Total energy modulation (frequency,amplitude)</td>
<td>2</td>
</tr>
<tr>
<td>RMSenv</td>
<td>RMS envelope</td>
<td>2</td>
</tr>
<tr>
<td>ACor</td>
<td>Signal Auto-Correlation function (12 first coeff.)</td>
<td>24</td>
</tr>
<tr>
<td>ZCR</td>
<td>Zero-Crossing Rate</td>
<td>2</td>
</tr>
<tr>
<td>SCent</td>
<td>Harmonic spectral centroid</td>
<td>2</td>
</tr>
<tr>
<td>EScre</td>
<td>Spectral centroid of the magnitude and energy spectrum</td>
<td>2</td>
</tr>
<tr>
<td>ESSflat</td>
<td>Spectral flatness of the magnitude and energy spectrum</td>
<td>2</td>
</tr>
<tr>
<td>ESSlpe</td>
<td>Spectral slope of the magnitude and energy spectrum</td>
<td>2</td>
</tr>
<tr>
<td>ESKurt</td>
<td>Spectral Kurtosis</td>
<td>2</td>
</tr>
<tr>
<td>EShp</td>
<td>Harmonic slope</td>
<td>2</td>
</tr>
<tr>
<td>EHDec</td>
<td>Harmonic decrease</td>
<td>1</td>
</tr>
<tr>
<td>EHrOff</td>
<td>Harmonic rolloff</td>
<td>1</td>
</tr>
<tr>
<td>EHvar</td>
<td>Harmonic variation</td>
<td>1</td>
</tr>
<tr>
<td>EHfEG</td>
<td>Harmonic energy, noise energy and frame energy</td>
<td>6</td>
</tr>
<tr>
<td>EHNcss</td>
<td>Noisiness</td>
<td>2</td>
</tr>
<tr>
<td>EHf0</td>
<td>Fundamental frequency $F_0$</td>
<td>2</td>
</tr>
<tr>
<td>EHiH</td>
<td>Inharmonicity</td>
<td>2</td>
</tr>
<tr>
<td>EHTios</td>
<td>Harmonic tritimus</td>
<td>6</td>
</tr>
<tr>
<td>EHodeV</td>
<td>Harmonic odd to even partials ratio</td>
<td>2</td>
</tr>
<tr>
<td>EHdev</td>
<td>Harmonic deviation</td>
<td>2</td>
</tr>
<tr>
<td>ESGCent</td>
<td>Spectral centroid of the magnitude and energy spectrum</td>
<td>2</td>
</tr>
<tr>
<td>ESGSpd</td>
<td>Spectral spread of the magnitude and energy spectrum</td>
<td>4</td>
</tr>
<tr>
<td>ESGSkew</td>
<td>Spectral skewness of the magnitude and energy spectrum</td>
<td>4</td>
</tr>
<tr>
<td>ESKart</td>
<td>Spectral kurtosis</td>
<td>4</td>
</tr>
<tr>
<td>EShp</td>
<td>Spectral slope of the magnitude and energy spectrum</td>
<td>4</td>
</tr>
<tr>
<td>EShp</td>
<td>Spectral decrease of the magnitude and energy spectrum</td>
<td>4</td>
</tr>
<tr>
<td>EStoff, EErr</td>
<td>Spectral rolloff</td>
<td>4</td>
</tr>
<tr>
<td>EStvar</td>
<td>Spectral variation of the magnitude and energy spectrum</td>
<td>4</td>
</tr>
<tr>
<td>EStsFeg</td>
<td>Spectral frame energy of the magnitude and energy spectrum</td>
<td>4</td>
</tr>
<tr>
<td>EStsEflat</td>
<td>Spectral flatness of the magnitude and energy spectrum</td>
<td>4</td>
</tr>
<tr>
<td>ESrel, EScor</td>
<td>Spectral crest of the magnitude and energy spectrum</td>
<td>4</td>
</tr>
<tr>
<td>ESMCent, ESMSpd</td>
<td>EREB scale magnitude spectrum / gammatone centroid</td>
<td>4</td>
</tr>
<tr>
<td>ESMSpd, ESGSpd</td>
<td>EREB scale magnitude spectrum / gammatone spread</td>
<td>4</td>
</tr>
<tr>
<td>ESMSkew, ESGSkw</td>
<td>EREB scale magnitude spectrum / gammatone skewness</td>
<td>4</td>
</tr>
<tr>
<td>ESMKurt, ESGKart</td>
<td>EREB scale magnitude spectrum / gammatone kurtosis</td>
<td>4</td>
</tr>
<tr>
<td>ESMShp, ESGShp</td>
<td>EREB scale magnitude spectrum / gammatone slope</td>
<td>4</td>
</tr>
<tr>
<td>ESMDec, ESGDec</td>
<td>EREB scale magnitude spectrum / gammatone decrease</td>
<td>4</td>
</tr>
<tr>
<td>ESMrOff, ESGrOff</td>
<td>EREB scale magnitude spectrum / gammatone rolloff</td>
<td>4</td>
</tr>
<tr>
<td>ESMvar, ESGVar</td>
<td>EREB scale magnitude spectrum / gammatone variation</td>
<td>4</td>
</tr>
<tr>
<td>ESMfEG, ESGfEG</td>
<td>EREB scale magnitude spectrum / gammatone frame energy</td>
<td>4</td>
</tr>
<tr>
<td>ESMflat, ESGflat</td>
<td>EREB scale magnitude spectrum / gammatone flatness</td>
<td>4</td>
</tr>
<tr>
<td>ESMrE, ESGrE</td>
<td>EREB scale magnitude spectrum / gammatone crest</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 2. Acronym, name and number of the used timbre descriptors.

* MATLAB code available at http://www.cirmmt.org/research/tools
3.2 Classification taxonomy

In this study, we use two databases which can be annotated using different taxonomies. Due to its diversity, the CREM database was only annotated using the Hornbostel and Sachs taxonomy [16] (T1) illustrated in Figure 1(a) which is widely used in ethnomusicology. This hierarchical taxonomy is general enough to classify uncommon instruments (e.g. struck bamboo) and conveys information about sound production materials and playing styles. From an another hand, the Iowa musical instruments database [5] used in our experiments was initially annotated using a musician’s instrument taxonomy (T2) as proposed in [13] and illustrated in Figure 1(b). This database is composed of common western pitched instruments which can easily be annotated using T1 as described in Table 3. One can notice that the Iowa database is only composed of aerophones and chordophones instruments. If we consider the playing style, only 4 classes are represented if we apply T1 taxonomy to the Iowa database.

<table>
<thead>
<tr>
<th>T1 class name</th>
<th>T2 equivalence</th>
<th>Duration (s)</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>aero-blowed</td>
<td>reed/flute and brass</td>
<td>5,951</td>
<td>668</td>
</tr>
<tr>
<td>cordo-struck</td>
<td>struck strings</td>
<td>5,564</td>
<td>646</td>
</tr>
<tr>
<td>cordo-plucked</td>
<td>plucked strings</td>
<td>5,229</td>
<td>583</td>
</tr>
<tr>
<td>cordo-bowed</td>
<td>bowed strings</td>
<td>7,853</td>
<td>838</td>
</tr>
<tr>
<td>Total</td>
<td></td>
<td>24,397</td>
<td>2,735</td>
</tr>
</tbody>
</table>

Table 3. Content of the Iowa database using musician’s instrument taxonomy (T2) and equivalence with the Hornbostel and Sachs taxonomy (T1).

4. AUTOMATIC INSTRUMENT TIMBRE CLASSIFICATION METHOD

The described method aims at estimating the corresponding taxonomy class name of a given input sound.

4.1 Method overview

Here, each sound segment (cf. Section 3.1) is represented by vector of length \( p = 164 \) where each value corresponds to a descriptor (see Table 2). The training step of this method (illustrated in Figure 2) aims at modeling each timbre class using the best projection space for classification. A features selection algorithm is first applied to efficiently reduce the number of descriptors to avoid statistical over-learning. The classification space is computed using discriminant analysis which consists in estimating optimal weights over the descriptors allowing the best discrimination between timbre classes. Thus, the classification task consists in projecting an input sound into the best classification space and to select the most probable timbre class using the learned model.

4.2 Linear discriminant analysis

The goal of Linear Discriminant Analysis (LDA) [1] is to find the best projection or linear combination of all descriptors which maximizes the average distance between classes (inter-class distance) while minimizing distance between individuals from the same class (intra-class distance). This method assumes that the class affection of each individual is a priori known. Its principle can be described as follows. First consider the \( n \times p \) real matrix \( M \) where each row is a vector of descriptors associated to a sound (individual). We assume that each individual is a member of a unique class \( k \in [1, K] \). Now we define \( W \) as the intra-class variance-covariance matrix which can be estimated by:

\[
W = \frac{1}{n} \sum_{k=1}^{K} n_k W_k, \tag{1}
\]

where \( W_k \) is the variance-covariance matrix computed from the \( n_k \times p \) sub-matrix of \( M \) composed of the \( n_k \) individuals included into the class \( k \).

We also define \( B \) the inter-class variance-covariance matrix expressed as follows:

\[
B = \frac{1}{n} \sum_{k=1}^{K} n_k (\mu_k - \mu)(\mu_k - \mu)^T, \tag{2}
\]
where $\mu_k$ corresponds to the mean vector of class $k$ and $\mu$ is the mean vector of the entire dataset. According to [1], it can be shown that the eigenvectors of matrix $D = (B + W)^{-1}B$ solve this optimization problem. When the matrix $A = (B + W)$ is not invertible, a computational solution consists in using pseudoinverse of matrix $A$ which can be calculated using $A^T(AA^T)^{-1}$.

### 4.3 Features selection algorithms

Features selection aims at computing the optimal relevance of each descriptor which can be measured with a weight or a rank. The resulting descriptors subset has to be the most discriminant as possible with the minimal redundancy. In this study, we investigate the three approaches described below.

#### 4.3.1 LDA features selection

The LDA method detailed in Section 4.2 can also be used for selecting the most relevant features. In fact, the computed eigenvectors which correspond to linear combination of descriptors convey a relative weight applied to each descriptor. Thus, the significance (or weight) of descriptors can be estimated from the approximated probability density functions (pdf) using a computed histogram. According to Bayes theorem one can compute $P(c_i | f) = P(f|c_i)P(c_i)$ which can be estimated with the mutual information defined by:

$$I(C, F) = \sum_c \sum_f P(c,f) \frac{P(c,f)}{P(c)P(f)},$$

where $P(c)$ denotes the probability of $C = c$ which can be estimated from the approximated probability density functions (pdf) using a computed histogram. According to Bayes theorem one can compute $P(c,f) = P(f|c)P(c)$ where $P(f|c)$ is the pdf of the feature descriptor value $f$ into class $c$. This method can be improved using [2] by reducing simultaneously the redundancy by considering the mutual information between previously selected descriptors.

#### 4.3.3 Inertia Ratio Maximisation using features space projection (IRMFSP)

This algorithm was first proposed in [11] to reduce the number of descriptors used by timbre classification methods. It consists in maximizing the relevance of the descriptors subset for the classification task while minimizing the redundancy between the selected ones. This iterative method ($i \leq p$) is composed of two steps. The first one selects at iteration $i$ the non-previous selected descriptor which maximizes the ratio between inter-class inertia and the total inertia expressed as follow:

$$\bar{d}(i) = \arg \max_d \frac{\sum_{k=1}^{K} n_k (\mu_{d,k} - \mu_d)(\mu_{d,k} - \mu_d)^T}{\sum_{i=1}^{n} (f_{d,i} - \mu_d)(f_{d,i} - \mu_d)^T},$$

where $f_{d,i}$ denotes the value of descriptor $d \in [1, p]$ affected to the individual $i$, $\mu_{d,k}$ and $\mu_d$ respectively denote the average value of descriptor $d$ into the class $k$ and for the total dataset. The second step of this algorithm aims at orthogonalizing the remaining data for the next iteration as follows:

$$f_{d}^{(i+1)} = f_{d}^{(i)} - (f_{d}^{(i)} \cdot g_{d}) g_{d} \forall d \neq \bar{d}(i),$$

where $f_{d}^{(i)}$ is the vector of the previously selected descriptor $\bar{d}(i)$ for all the individuals of the entire dataset and $g_{d} = f_{d}^{(i)}/\|f_{d}^{(i)}\|$ is its normalized form.

### 4.4 Class modeling and automatic classification

Each instrument class is modeled into the projected classification space resulting from the application of LDA. Thus, each class can be represented by its gravity center $\hat{\mu}_k$ which corresponds to the vector of the averaged values of the projected individuals which compose the class $k$. The classification decision which affect a class $\hat{k}$ to an input sound $\hat{x}$ is simply performed by minimizing the Euclidean distance with the gravity center of each class as follows:

$$\hat{k} = \arg \min_k \|\hat{\mu}_k - \hat{x}\|_2 \forall k \in [1, K],$$

where $\|v\|_2$ denotes the $l_2$ norm of vector $v$. Despite its simplicity, this method seems to obtain good results comparable with those of the literature [12].

### 5. EXPERIMENTS AND RESULTS

In this section we present the classification results obtained using the proposed method described in Section 4.

#### 5.1 Method evaluation based on self database classification

In this experiment, we evaluate the classification of each distinct database using different taxonomies. We applied the 3-fold cross validation methodology which consists in partitioning the database in 3 distinct random subsets composed with 33% of each class (no collision between sets). Thus, the automatic classification applied on each subset is based on training applied on the remaining 66% of the
database. Figure 5.1 compares the classification accuracy obtained as a function of the number of used descriptors. The resulting confusion matrix of the CREM database using 20 audio descriptors is presented in Table 4 and shows an average classification accuracy of 80% where each instrument is well classified with a minimal accuracy of 70% for the aerophones. These results are good and seems comparable with those described in the literature [11] using the same number of descriptor. The most relevant feature descriptors (selected among the top ten) estimated by the IRMSFP and used for the classification task are detailed in Table 7. This result reveals significant differences between the two databases. As an example, harmonic descriptors are only discriminative for the CREM database but not for the Iowa database. This may be explained by the presence of membranophone in the CREM database which are not present in the Iowa database. Contrarily, spectral and perceptual descriptors seems more relevant for the Iowa database than for the CREM database. Some descriptors appear to be relevant for both database like the Spectral flatness (Sflat) and the ERB scale frame energy (ErbFErg) which describe the spectral envelope of signal.

Table 4. Confusion matrix (expressed in percent of the sounds of the original class listed on the left) of the CREM database using the 20 most relevant descriptors selected by IRMSFP.

<table>
<thead>
<tr>
<th></th>
<th>air</th>
<th>c-struc</th>
<th>c-pluc</th>
<th>c-bowed</th>
<th>i-pluc</th>
<th>i-struc</th>
<th>i-clink</th>
<th>membro</th>
</tr>
</thead>
<tbody>
<tr>
<td>aerophones</td>
<td>67</td>
<td>92</td>
<td>79</td>
<td>4</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>72</td>
</tr>
<tr>
<td>c-struc</td>
<td>11</td>
<td>100</td>
<td>79</td>
<td>76</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>72</td>
</tr>
<tr>
<td>c-pluc</td>
<td>11</td>
<td>100</td>
<td>12</td>
<td>79</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>72</td>
</tr>
<tr>
<td>c-bowed</td>
<td>11</td>
<td>100</td>
<td>3</td>
<td>79</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>72</td>
</tr>
<tr>
<td>i-pluc</td>
<td>11</td>
<td>100</td>
<td>12</td>
<td>79</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>72</td>
</tr>
<tr>
<td>i-struc</td>
<td>11</td>
<td>100</td>
<td>3</td>
<td>79</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>72</td>
</tr>
<tr>
<td>i-clink</td>
<td>11</td>
<td>100</td>
<td>12</td>
<td>79</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>72</td>
</tr>
</tbody>
</table>

5.2 Cross-database evaluation

In this experiments (see Table 5), we merged the two databases and we applied the 3-fold cross validation method based on the T1 taxonomy to evaluate the classification accuracy on both database. The resulting average accuracy is about 68% which is lower than the accuracy obtained on the distinct classification of each database. The results of cross-database evaluation applied between databases using the T1 taxonomy are presented in Table 6 and obtain a poor average accuracy of 30%. This seems to confirm our intuition that the Iowa database conveys insufficient information to distinguish the different playing styles between the non-western cordophones instruments of the CREM database.

6. CONCLUSION AND FUTURE WORKS

We applied a computationally efficient automatic timbre classification method which was successfully evaluated on an introduced diversified database using an ethnomusicalogical taxonomy. This method obtains good classification results (> 80% of accuracy) for both evaluated databases which are comparable to those of the literature. However, the cross-database evaluation shows that each database cannot be used to infer a classification to the other. This can be explained by significant differences between these databases. Interestingly, results on the merged database obtain an acceptable accuracy of about 70%. As shown in previous work [11], our experiments confirm the efficiency of IRMSFP algorithm for automatic features selection applied to timbre classification. The interpretation of the
Table 5. Confusion matrix (expressed in percent of the sounds of the original class listed on the left) of the evaluated fusion between the CREM and the Iowa database using the 20 most relevant descriptors selected by IRMSFP.

<table>
<thead>
<tr>
<th></th>
<th>Iowa T1</th>
<th>Iowa T2</th>
<th>CREM-Iowa T1</th>
</tr>
</thead>
<tbody>
<tr>
<td>gero</td>
<td>56</td>
<td>90</td>
<td>87</td>
</tr>
<tr>
<td>c-struc</td>
<td>9</td>
<td>63</td>
<td>11</td>
</tr>
<tr>
<td>c-pluc</td>
<td>20</td>
<td>26</td>
<td>2</td>
</tr>
<tr>
<td>c-bowed</td>
<td>12</td>
<td>14</td>
<td>5</td>
</tr>
<tr>
<td>c-strings</td>
<td>12</td>
<td>13</td>
<td>9</td>
</tr>
<tr>
<td>c-pluc</td>
<td>22</td>
<td>22</td>
<td>6</td>
</tr>
<tr>
<td>c-bowed</td>
<td>28</td>
<td>34</td>
<td>14</td>
</tr>
</tbody>
</table>

Table 6. Confusion matrix (expressed in percent of the sounds of the original class listed on the left) of the CREM database classification based on Iowa database training.

<table>
<thead>
<tr>
<th></th>
<th>Iowa T1</th>
<th>Iowa T2</th>
<th>CREM-Iowa T1</th>
</tr>
</thead>
<tbody>
<tr>
<td>gero</td>
<td>56</td>
<td>90</td>
<td>87</td>
</tr>
<tr>
<td>c-struc</td>
<td>9</td>
<td>63</td>
<td>11</td>
</tr>
<tr>
<td>c-pluc</td>
<td>20</td>
<td>26</td>
<td>2</td>
</tr>
<tr>
<td>c-bowed</td>
<td>12</td>
<td>14</td>
<td>5</td>
</tr>
<tr>
<td>c-strings</td>
<td>12</td>
<td>13</td>
<td>9</td>
</tr>
<tr>
<td>c-pluc</td>
<td>22</td>
<td>22</td>
<td>6</td>
</tr>
<tr>
<td>c-bowed</td>
<td>28</td>
<td>34</td>
<td>14</td>
</tr>
</tbody>
</table>

Table 7. Comparison of the most relevant descriptors estimated by IRMFSP.

<table>
<thead>
<tr>
<th></th>
<th>CREM T1</th>
<th>Iowa T1</th>
<th>Iowa T2</th>
<th>CREM-Iowa T1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sflat</td>
<td>66</td>
<td>67</td>
<td>67</td>
<td>66</td>
</tr>
<tr>
<td>Sskew</td>
<td>56</td>
<td>56</td>
<td>56</td>
<td>56</td>
</tr>
<tr>
<td>ZCR</td>
<td>77</td>
<td>77</td>
<td>82</td>
<td>82</td>
</tr>
<tr>
<td>SSKurt</td>
<td>78</td>
<td>78</td>
<td>78</td>
<td>78</td>
</tr>
</tbody>
</table>

most relevant selected features shows a significant effect of the content of database rather than on the taxonomy. However the timbre modeling interpretation applied to timbre classification remains difficult. Future works will consist in further investigating the role of descriptors by manually constraining selection before the classification process.

7. ACKNOWLEDGMENTS

This research was partly supported by the French ANR (Agence Nationale de la Recherche) DIADEMS (Description, Indexation, Acces aux Documents Ethnomusicologiques et Sonores) project (ANR-12-CORD-0022).

8. REFERENCES

MUSIC ANALYSIS AS A SMALLEST GRAMMAR PROBLEM

Kirill Sidorov Andrew Jones David Marshall
Cardiff University, UK
{K.Sidorov, Andrew.C.Jones, Dave.Marshall}@cs.cardiff.ac.uk

ABSTRACT

In this paper we present a novel approach to music analysis, in which a grammar is automatically generated explaining a musical work’s structure. The proposed method is predicated on the hypothesis that the shortest possible grammar provides a model of the musical structure which is a good representation of the composer’s intent. The effectiveness of our approach is demonstrated by comparison of the results with previously-published expert analysis; our automated approach produces results comparable to human annotation. We also illustrate the power of our approach by showing that it is able to locate errors in scores, such as introduced by OMR or human transcription. Further, our approach provides a novel mechanism for intuitive high-level editing and creative transformation of music. A wide range of other possible applications exists, including automatic summarization and simplification; estimation of musical complexity and similarity, and plagiarism detection.

1. INTRODUCTION

In his Norton Lectures [1], Bernstein argues that music can be analysed in linguistic terms, and even that there might be “a worldwide, inborn musical grammar”. Less specifically, the prevalence of musical form analyses, both large-scale (e.g. sonata form) and at the level of individual phrases, demonstrates that patterns, motifs, etc., are an important facet of a musical composition, and a grammar is certainly one way of capturing these artefacts.

In this paper we present a method for automatically deriving a compact grammar from a musical work and demonstrate its effectiveness as a tool for analysing musical structure. A key novelty of this method is that it operates automatically, yet generates insightful results. We concentrate in this paper on substantiating our claim that generating parsimonious grammars is a useful analysis tool, but also suggest a wide range of scenarios to which this approach could be applied.

Previous research into grammar-based approaches to modelling music has led to promising results. Treating harmonic phenomena as being induced by a generative grammar has been proposed in [9, 24, 27], and the explanatory power of such grammars has been demonstrated. The use of musical grammar based of the Generative Theory of Tonal Music [19] has been proposed in [11–13], for the analysis of music. Similarly, there is a number of grammar-based approaches to automatic composition, including some which automatically learn stochastic grammars or derive grammars in an evolutionary manner, although some researchers continue to craft grammars for this purpose by hand [7].

However, in these works the derivation of grammar rules themselves is performed manually [27] or semi-automatically [11–13] from heuristic musicological considerations. In some cases generative grammars (including stochastic ones) are derived or learned automatically, but they describe general patterns in a corpus of music, e.g. for synthesis [16, 22], rather than being precise analyses of individual works. In a paper describing research carried out with a different, more precise aim of visualising semantic structure of an individual work, the authors remark that they resorted to manual retrieval of musical structure data from descriptive essays “since presently there is no existing algorithm to parse the high-level structural information automatically from MIDI files or raw sound data” [5].

In this paper we present a method which addresses the above concern expressed by Chan et al. in [5], but which at the same time takes a principled, information-theoretical approach. We argue that the best model explaining a given piece of music is the most compact one. This is known as Minimum Description Length principle [23] which, in turn, is a formal manifestation of the Occam’s razor principle: the best explanation for data is the most compressive one. Hence, given a piece of music, we seek to find the shortest possible context-free grammar that generates this piece (and only this piece). The validity of our compressive modelling approach in this particular domain is corroborated by evidence from earlier research in predictive modelling of music [6] and from perception psychology [14, 25]: humans appear to find strongly compressible music (which therefore has a compact grammar) appealing.

2. COMPUTING THE SMALLEST GRAMMAR

Given a piece of music, we treat it as a sequence(s) of symbols (see Section 2.1) and we seek to find the shortest possible context-free grammar that generates this (and only this) piece. Following [21], we define the size of a grammar $G$ to be the total length of the right hand sides of all the production rules $R$, plus one for each rule (length of a separator or cost of introducing a new rule):
with a new rule. The resulting saving is, therefore [21]:

$$|G| = \sum_i (|R_i| + 1).$$  \hspace{1cm} (1)

Searching for such a grammar is known as the smallest grammar problem. It has recently received much attention due to its importance in compression and analysis of DNA sequences, see e.g. [4]. For an overview of the smallest grammar problem the reader is referred to [3].

Computing the smallest grammar is provably NP-hard (see [18] Theorem 3.1), therefore in practice we are seeking an approximation to the smallest grammar.

Various heuristics have been proposed in order to tackle the smallest grammar problem in tractable time by greedy algorithms [4, 21]. A fast on-line (linear time) algorithm called SEQUITUR has been proposed in [20]. While the focus of [20] was fast grammar inference for large sequences, rather than strong compression, [20] contains an early mention that such techniques may be applied to parsing of music. (In Section 3 we compare grammars produced by SEQUITUR with our approach.)

In [4, 21], a class of algorithms involving iterative replacement of a repeated substring is considered (termed there iterative repeat replacement (IRR)). We employ a similar procedure here, summarised in Alg. 1. First, the grammar $G$ is initialised with top level rule(s), whose right-hand sides initially are simply the input string(s). Then, in the original IRR scheme, a candidate substring $c$ is selected according to some scoring function $F$. All non-overlapping occurrences of this substring in the grammar are replaced with a new symbol $R_{n+1}$, and a new rule is added to the grammar: $R_{n+1} \rightarrow c$. Replacement of a substring of length $m$ in a string of length $n$ can be done using the Knuth-Morris-Pratt algorithm [17] in $O(m+n)$ time. The replacement procedure repeats until no further improvement is possible.

In [4, 21] the various heuristics according to which such candidate substitutions can be selected are examined; the conclusion is that the “locally most compressive” heuristic results in the shortest final grammars. Suppose a substring of length $L$ occurring $N$ times is considered for replacement with a new rule. The resulting saving is, therefore [21]:

$$F = \Delta |G| = (LN) - (L + 1 + N).$$  \hspace{1cm} (2)

Hence, we use Eq. (2) (the locally most compressive heuristic) as our scoring function when selecting candidate substrings (in line 2 of Alg. 1). We found that the greedy iterative replacement scheme of [21] does not always produce optimal grammars. We note that a small decrease in grammar size may amount to a substantial change in the grammar’s structure, therefore we seek to improve the compression performance.

To do so, instead of greedily making a choice at each iteration, we recursively evaluate (line 9) multiple ($w$) candidate substitutions (line 2) with backtracking, up to a certain depth $d_{\text{max}}$ (lines 9–15). Once the budgeted search depth has been exhausted, the remaining substitutions are done greedily (lines 4–7) as in [21]. This allows us to control the greediness of the algorithm from completely greedy ($d_{\text{max}} = 0$) to exhaustive search ($d_{\text{max}} = \infty$). We observed that using more than 2–3 levels of backtracking usually does not yield any further reduction in the size of the grammar.

Algorithm 1 \textsc{CompGram} (Compress grammar)

\begin{algorithm}
\begin{algorithmic}[1]
\State \textbf{Require}: Grammar $G$; search depth $d$.
\State (Tuning constants: max depth $d_{\text{max}}$, width $w$)
\State \textbf{loop}
\State 2: Find $w$ best candidate substitutions $C = \{c_i\}$ in $G$.
\State 3: \textbf{if} $C = \emptyset$ \textbf{then} return $G$; \textbf{end if}
\State 4: \textbf{if} recursion depth $d > d_{\text{max}}$ \textbf{then}
\State 5: Greedily choose best $c_{\text{best}}$
\State 6: $G' := \text{replace}(G, c_{\text{best}}, \text{new symbol})$
\State 7: return \text{CompGram}($G'$, $d + 1$)
\State 8: \textbf{else}
\State 9: Evaluate candidates:
\State 10: \textbf{for} $c_i \in C$ \textbf{do}
\State 11: $G' := \text{replace}(G, c_i, \text{new symbol})$
\State 12: $G'' := \text{CompGram}(G', d + 1)$
\State 13: \textbf{end for}
\State 14: $b := \arg \min \{|G''|\}$
\State 15: return $G''$
\State \textbf{end if}
\State \textbf{end loop}
\end{algorithmic}
\end{algorithm}

Selecting a candidate according to Eq. (2) in line 2 involves maximising the number of non-overlapping occurrences of a substring, which is known as the string statistics problem, the solutions to which are not cheap [2]. Therefore, as in [4] we approximate the maximal number of non-overlapping occurrences with the number of maximal repeats [10]. All $z$ maximal repeats in a string (or a set of strings) of total length $n$ can be found very fast (in $O(n + z)$ time) using suffix arrays [10]. In principle, it is possible to construct an example in which this number will be drastically different from the true number of non-overlapping occurrences (e.g. a long string consisting of a repeated symbol). However, this approximation was shown to work well in [4] and we have confirmed this in our experiments. Further, this concern is alleviated by the backtracking procedure we employ.

2.1 Representation of Music

In this paper, we focus on music that can be represented as several monophonic voices (such as voices in a fugue, or orchestral parts), that is, on the horizontal aspects of the music. We treat each voice, or orchestral part, as a string. We use (diatonic) intervals between adjacent notes, ignoring rests, as symbols in our strings. For ease of explanation of our algorithm we concentrate on the melodic information only, ignoring rhythm (note durations). Rhythmic invariance may be advantageous when melodic analysis is the prime concern. However, it is trivial to include note durations, and potentially even chord symbols and other musical elements, as symbols in additional (top level) strings.

Note that even though we take no special measures to model the relationship between the individual voices, this is happening automatically: indeed, all voices are encompassed in the same grammar and are considered for the iterative replacement procedure on equal rights as the grammar is updated.

302
3. RESULTS AND APPLICATIONS

3.1 Automatic Structural Analysis

We have applied our method to automatically detect the structure of a selection of Bach’s fugues. (Eventually we intend to analyse all of them in this way.) Figure 1 shows one example of such analysis. We show the voices of the fugue in piano roll representation, with the hierarchy of the grammar on top: rules are represented by brackets labelled by rule number. For completeness, we give the entire grammar (of size $|G| = 217$) obtained for Fugue №10 later, in Fig. 7. Figure 2 zooms in onto a fragment of the score with the rules overlaid. For comparison, we show manual analysis by a musicologist [26] in Fig. 3. Observe that all the main structural elements of the fugue have been correctly identified by our method (e.g. exp. and 1st dev. in rule $R_6$, re-exp. and 4th dev. in rule $R_9$, variant of re-exp. and 2nd dev in $R_{18}$ and $R_{19}$) and our automatic analysis is comparable to that by a human expert.

It is possible to use structures other than individual notes or intervals as symbols when constructing grammars. Figure 4 shows the simplified grammar for Fugue №10 generated using entire bars as symbols. In this experiment we first measured pairwise similarity between all bars (using Levenshtein distance [8]) and denoted each bar by a symbol, with identical or almost identical bars being denoted by the same symbol. The resulting grammar (Fig. 4) can be viewed as a coarse-grained analysis. Observe again that it closely matches human annotation (Fig. 3).

Our approach can also be used to detect prominent high-level features in music. We can compute the usage frequency for each rule and the corresponding savings in grammar size (as shown in Table 1 for Fugue №10). Most compressing rules, we argue, correspond to structurally important melodic elements. The present example illustrates our claim: rule $R_9$ corresponds to the fugue’s exposition, $R_6$ to re-exposition, and $R_9$ to the characteristic chromatic figure in the opening (cf. Figs. 1 to 3 and the score).

In addition to high-level analysis, our approach can be used to detect the smallest constituent building blocks of a piece. For example, Fig. 5 shows the lowest level rules (that use only terminals) produced in analysis of Fugue №10, and the frequency of each rule. These are the elementary “bricks” from which Bach has constructed this fugue.

In [20], SEQUITUR is applied to two Bach chorales. In Fig. 6 we replicate the experiment from [20] and comp-

<table>
<thead>
<tr>
<th>Rule</th>
<th>$R_9$</th>
<th>$R_8$</th>
<th>$R_5$</th>
<th>$R_{10}$</th>
<th>$R_{11}$</th>
<th>$R_4$</th>
<th>$R_{19}$</th>
<th>$R_7$</th>
<th>$R_{20}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Freq.</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Len.</td>
<td>16</td>
<td>12</td>
<td>6</td>
<td>7</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>E. len.</td>
<td>99</td>
<td>94</td>
<td>24</td>
<td>47</td>
<td>11</td>
<td>11</td>
<td>37</td>
<td>34</td>
<td>17</td>
</tr>
<tr>
<td>Comp.</td>
<td>96</td>
<td>91</td>
<td>67</td>
<td>44</td>
<td>38</td>
<td>38</td>
<td>34</td>
<td>31</td>
<td>30</td>
</tr>
</tbody>
</table>

Table 1. Grammar statistics for Fugue №10. The ten most compressing rules are shown. For each rule $R_i$, Freq. is the number of times a rule occurs in the grammar, Len. is its right hand side length, E. len. is the length of the rule’s expansion, and Comp. is the total saving due to this rule.

Figure 1. Automatic analysis of Bach’s Fugue №10 from WTK book I. On top: sensitivity to point errors as measured by the increase in grammar size $\Delta|G|$.
We investigated the sensitivity of the grammars generated (introducing a point error) in the 1st voice of Fugue № elsewhere, alterations in structurally dense regions and less sensitive score is plotted in Fig. 1 (experiment, we systematically altered each note in turn by our method to alterations in the original music. In one small differences between the chorales. less compressing) S EQUITUR was compromised by the structure, while the grammar of the more greedy (and hence | | pare the grammars of these two chorales generated by SEQUITUR and by our approach. The chorales are very similar except for a few subtle differences. We note that our method was able to produce a shorter grammar (|Gout| = 50 vs. |Gsequitur| = 59) and hence revealed more of the relevant structure, while the grammar of the more greedy (and hence less compressing) SEQUITUR was compromised by the small differences between the chorales.

3.2 Error Detection and Spell-checking

We investigated the sensitivity of the grammars generated by our method to alterations in the original music. In one experiment, we systematically altered each note in turn (introducing a point error) in the 1st voice of Fugue №10 and constructed a grammar for each altered score. The change in grammar size relative to that of the unaltered score is plotted in Fig. 1 (top) as a function of the alteration’s position. Observe that the grammar is more sensitive to alterations in structurally dense regions and less sensitive elsewhere, e.g. in between episodes. Remarkably, with very few exceptions (e.g. bars 10, 26) altering the piece consistently results in the grammar size increasing. We observed a similar effect in other Bach fugues and even in 19th century works (see below). We propose, only partially in jest, that this indicates that Bach’s fugues are close to structural perfection which is ruined by even the smallest alteration.

Having observed the sensitivity of the grammar size to point errors (at least in highly structured music), we propose that grammar-based modelling can be used for musical “spell-checking” to correct errors in typesetting (much like a word processor does for text), or in optical music recognition (OMR). This is analogous to compressive sensing which is often used in signal and image processing (see e.g. [15]) for denoising: noise compresses poorly. We can regard errors in music as noise and use the grammar-based model for locating such errors. We investigated this possibility with the following experiment.

As above, we introduce a point error (replacing one note) at a random location in the score to simulate a “typo” or an OMR error. We then systematically alter every note at a random location in the score to simulate a “typo” or | | | |
locations and report median performance over 100 experiments in Table 2. We have performed this experiment on Bach fugues, romantic symphonic works (Beethoven’s 5th symphony 1st mvt., Berlioz’s “Symphonie Fantastique” 1st mvt., Mendelssohn’s “Hebrides”) and Elgar’s Quartet 3rd mvt. We observed impressive performance (Table 2) on Bach’s fugues (error location narrowed down to just a few percent of the score’s size), and even in supposedly less-structured symphonic works the algorithm was able to substantially narrow down the location of potential errors. This suggests that our approach can be used to effectively locate errors in music: for example a notation editor using our method may highlight potential error locations, thus warning the user, much like word processors do for text.

A variant of the above experiment is presented in Fig. 8. We want to select between two editions of Fugue №10 in which bar 33 differs. We measured the total grammar size for the two editions and concluded that the variant in Edition B is more logical as it results in smaller grammar size $|G| = 208$ (vs. $|G| = 217$ for Edition A).

Figure 7. Automatically generated shortest grammar for Fugue №10. Here, $R_i$ are production rules ($S_i$ are the top level rules corresponding to entire voices), numbers with arrows are terminal symbols (diatonic intervals with the arrows indicating the direction).

Figure 8. Selecting between two editions of Fugue №10 using grammar size.
Hypothetically, a poor composition would remain poor even when (random) alterations are made to it and hence its grammar size would be insensitive to such alterations, while well-constructed works (like those of Bach in our examples) would suffer, in terms of grammar size, from perturbation.

4. CONCLUSIONS AND FUTURE WORK

We have posed the analysis of music as a smallest grammar problem and have demonstrated that building parsimonious context-free grammars is an appealing tool for analysis of music, as grammars give insights into the underlying structure of a piece. We have discussed how such grammars may be efficiently constructed and have illustrated the power of our model with a number of applications: automatic structural analysis, error detection and spell-checking (without prior models), high-level editing.

Future work would include augmenting the presented automatic grammatical analysis to allow inexact repetitions (variations or transformations of material) to be recognised in the grammar, and, in general, increasing the modelling power by recognising more disguised similarities in music.

5. REFERENCES


FRAME-LEVEL AUDIO SEGMENTATION FOR ABRIDGED MUSICAL WORKS

Thomas Prätzlich, Meinard Müller
International Audio Laboratories Erlangen
{thomas.praetzlich, meinard.mueller}@audiolabs-erlangen.de

ABSTRACT

Large-scale musical works such as operas may last several hours and typically involve a huge number of musicians. For such compositions, one often finds different arrangements and abridged versions (often lasting less than an hour), which can also be performed by smaller ensembles. Abridged versions still convey the flavor of the musical work containing the most important excerpts and melodies. In this paper, we consider the task of automatically segmenting an audio recording of a given version into semantically meaningful parts. Following previous work, the general strategy is to transfer a reference segmentation of the original complete work to the given version. Our main contribution is to show how this can be accomplished when dealing with strongly abridged versions. To this end, opposed to previously suggested segment-level matching procedures, we adapt a frame-level matching approach for transferring the reference segment information to the unknown version. Considering the opera “Der Freischütz” as an example scenario, we discuss how to balance out flexibility and robustness properties of our proposed frame-level segmentation procedure.

1. INTRODUCTION

Over the years, many musical works have seen a great number of reproductions, ranging from reprints of the sheet music to various audio recordings of performances. For many works this has led to a wealth of co-existing versions including arrangements, adaptations, cover versions, and so on. Establishing semantic correspondences between different versions and representations is an important step for many applications in Music Information Retrieval. For example, when comparing a musical score with an audio version, the goal is to compute an alignment between measures or notes in the score and points in time in the audio version. This task is motivated by applications such as score following [1], where the score can be used to navigate through a corresponding audio version and vice versa. The aligned score information can also be used to parameterize an audio processing algorithm such as in score-informed source separation [4, 12]. When working with two audio versions, alignments are useful for comparing different performances of the same piece of music [2,3]. In cover song identification, alignments can be used to compute the similarity between two recordings [11]. Alignment techniques can also help to transfer metadata and segmentation information between recordings. In [7], an unknown recording is queried against a database of music recordings to identify a corresponding version of the same musical work. After a successful identification, alignment techniques are used to transfer the segmentation given in the database to the unknown recording.

A similar problem was addressed in previous work, where the goal was to transfer a labeled segmentation of a reference version onto an unknown version of the same musical work [10]. The task was approached by a segment-level matching procedure, where one main assumption was that a given reference segment either appears more or less in the same form in the unknown version or is omitted completely.

In abridged versions of an opera, however, this assumption is often not valid. Such versions strongly deviate from the original by omitting a large portion of the musical material. For example, given a segment in a reference version, one may no longer find the start or ending sections of this segment in an unknown version, but only an intermediate section. Hence, alignment techniques that account for structural differences are needed. In [5], a music synchronization procedure accounting for structural differences in recordings of the same piece of music is realized with an adaption of the Needleman-Wunsch algorithm. The algorithm penalizes the skipping of frames in the alignment by adding an additional cost value for each skipped frame.
Thus, the cost for skipping a sequence of frames is dependent on the length of the sequence. In abridged versions, however, omission may occur on an arbitrary scale, ranging from several musical measures up to entire scenes of an opera. In such a scenario, a skipping of long sequences should not be more penalized as a skipping of short sequences. In this work, we will therefore use a different alignment strategy.

In this paper, we address the problem of transferring a labeled reference segmentation onto an unknown version in the case of abridged versions, see Figure 1. As our main contribution, we show how to approach this task with a frame-level matching procedure, where correspondences between frames of a reference version and frames of an unknown version are established. The labeled segment information of the reference version is then transferred to the unknown version only for frames for which a correspondence has been established. Such a frame-level procedure is more flexible than a segment-level procedure. However, on the downside, it is less robust. As a further contribution, we show how to stabilize the robustness of the frame-level matching approach while preserving most of its flexibility.

The remainder of this paper is structured as follows: In Section 2, we discuss the relevance of abridged music recordings and explain why they are problematic in a standard music alignment scenario. In Section 3, we review the segment-level matching approach from previous work (Section 3.2), and then introduce the proposed frame-level segmentation pipeline (Section 3.3). Subsequently, we present some results of a qualitative (Section 4.2) and a quantitative (Section 4.3) evaluation and conclude the paper with a short summary (Section 5).

2. MOTIVATION

For many musical works, there exists a large number of different versions such as cover songs or different performances in classical music. These versions can vary greatly in different aspects such as the instrumentation or the structure. Large-scale musical works such as operas usually need a huge number of musicians to be performed. For these works, one often finds arrangements for smaller ensembles or piano reductions. Furthermore, performances of these works are usually very long. Weber’s opera “Der Freischütz”, for example, has an average duration of about two hours. Taking it to an extreme, Wagner’s epos “Der Ring der Nibelungen”, consists of four operas having an overall duration of about 15 hours. For such large-scale musical works, one often finds abridged versions. These versions usually present the most important material of a musical work in a strongly shortened and structurally modified form. Typically, these structural modifications include omissions of repetitions and other “non-essential” musical passages. Abridged versions were very common in the early recording days due to space constraints of the sound carriers. The opera “Der Freischütz” would have filled 18 discs on a shellac record. More recently, abridged versions or excerpts of a musical work can often be found as bonus tracks on CD records. In a standard alignment scenario, abridged versions are particularly problematic as they omit material on different scales, ranging from the omission of several musical measures up to entire parts.

3. METHODS

In this section, we show how one can accomplish the task of transferring a given segmentation of a reference version, say X, onto an unknown version, say Y. The general idea is to use alignment techniques to find corresponding parts between X and Y, and then to transfer on those parts the given segmentation from X to Y.

After introducing some basic notations on alignments and segmentations (Section 3.1), we review the segment-level matching approach from our previous work (Section 3.2). Subsequently, we introduce our frame-level segmentation approach based on partial matching (Section 3.3).

3.1 Basic Notations

3.1.1 Alignments, Paths, and Matches

Let \( [1 : N] := \{1, 2, \ldots, N\} \) be an index set representing the time line of a discrete signal or feature sequence \( X = (x_1, x_2, \ldots, x_N) \). Similarly, let \( [1 : M] \) be the time line of a second sequence \( Y = (y_1, \ldots, y_M) \). An alignment between two time lines \( [1 : N] \) and \( [1 : M] \) is modeled as a set \( \mathcal{L} = (p_1, \ldots, p_L) \subseteq [1 : N] \times [1 : M] \). An element \( p_\ell = (n_\ell, m_\ell) \in \mathcal{L} \) is called a cell and encodes a correspondence between index \( n_\ell \in [1 : N] \) of the first time line and index \( m_\ell \in [1 : M] \) of the second one. In the following, we assume \( \mathcal{L} \) to be in lexicographic order. \( \mathcal{L} \) is called a match if \( (p_{\ell+1} - p_\ell) \in \mathbb{N} \times \mathbb{N} \) for \( \ell \in [1 : L - 1] \). Note that this condition implies strict monotonicity and excludes the possibility to align an index of the first time line with many indices of the other and vice versa. An alignment can also be constrained by requiring \( (p_{\ell+1} - p_\ell) \in \Sigma \) for a given set \( \Sigma \) of admissible step sizes. A typical choice for this set is \( \Sigma = \{(1, 0), (0, 1)\} \), which allows to align an index of one time line to many indices of another, and vice versa. Sometimes other sets such as \( \Sigma = \{(1,1), (1,2), (2,1)\} \) are used to align sequences which are assumed to be structurally and temporally mostly consistent. If \( \mathcal{L} \) fulfills a given step size condition, \( \mathcal{P} = \mathcal{L} \) is called a path. Note that alignments that fulfill \( \Sigma_1 \) and \( \Sigma_2 \) are both paths, but only an alignment fulfilling \( \Sigma_2 \) is also a match.
3.1.2 Segments and Segmentation

We formally define a segment to be a set \( \alpha = [s : t] \subseteq [1 : N] \) specified by its start index \( s \) and its end index \( t \). Let \( |\alpha| := t - s + 1 \) denote the length of \( \alpha \). We define a (partial) segmentation of size \( K \) to be a set \( \mathcal{A} := \{\alpha_1, \alpha_2, \ldots, \alpha_K\} \) of pairwise disjoint segments: \( \alpha_k \cap \alpha_j = \emptyset \) for \( k, j \in [1 : K], k \neq j \).

3.1.3 Labeling

Let \( [0 : K] \) be a set of labels. The label 0 plays a special role and is used to label everything that has not been labeled otherwise. A label function \( \varphi \) maps each index \( n \in [1 : N] \) to a label \( k \in [0 : K] \):

\[ \varphi : [1 : N] \rightarrow [0 : K]. \]

The pair \(([1 : N], \varphi)\) is called a labeled time line. Let \( n \in [1 : N] \) be an index, \( \alpha = [s : t] \) be a segment, and \( k \in [0 : K] \) be a label. Then the pair \((n, k)\) is called a labeled index and the pair \((\alpha, k)\) a labeled segment. A labeled segment \((\alpha, k)\) induces a labeling of all indices \( n \in \alpha \). Let \( \mathcal{A} := \{\alpha_1, \alpha_2, \ldots, \alpha_K\} \) be a segmentation of \( [1 : N] \) and \([0 : K]\) be the label set. Then the set \( \{(\alpha_k, k) | k \in [1 : K]\} \) is called a labeled segmentation of \([1 : N]\). From a labeled segmentation one obtains a labeled index function \( \varphi \) by setting \( \varphi(n) := k \) for \( n \in \alpha_k \) and \( \varphi(n) := 0 \) for \( n \in [1 : N] \setminus \bigcup_{k \in [1 : K]} \alpha_k \). Vice versa, given a label function \( \varphi \), one obtains a labeled segmentation in the following way. We call consecutive indices with the same label a run. A segmentation of \([1 : N]\) is then derived by considering runs of maximal length. We call this segmentation the segmentation induced by \( \varphi \).

3.2 Segment-Level Matching Approach

The general approach in [10] is to apply segment-level matching techniques based on dynamic time warping (DTW) to transfer a labeled reference segmentation to an unknown version. Given a labeled segmentation \( \mathcal{A} \) of \( X \), each \( \alpha_k \in \mathcal{A} \) is used as query to compute a ranked list of matching candidates in \( Y \). The matching candidates are derived by applying a subsequence variant of the DTW algorithm using the step size conditions \( \Sigma = \{(1, 1), (1, 2), (2, 1)\} \), see [8, Chapter 5]. The result of the subsequence DTW procedure is a matching score and an alignment path \( \mathcal{P} = (p_1, \ldots, p_L) \) with \( p_t = (n_k, m_k) \). \( \mathcal{P} \) encodes an alignment of the segment \( \alpha_k := [n_k : m_k] \subseteq [1 : N] \) and the corresponding segment \([m_1 : m_L] \subseteq [1 : M] \) in \( Y \). To derive a final segmentation, one segment from each matching candidate list is chosen such that the sum of the alignment scores of all chosen segments is maximized by simultaneously fulfilling the following constraints. First, the chosen segments have to respect the temporal order of the reference segmentation and second, no overlapping segments are allowed in the final segmentation. Furthermore, the procedure is adapted to be robust to tuning differences of individual segments, see [10] for further details.
3.3.2 Induced Label Function

Given a labeled time line \((1:N], \varphi_c)\) and an alignment \(\mathcal{L}\), we derive a label function \(\varphi_c\) on \([1:M]\) by setting:

\[
\varphi_c(m) := \begin{cases} 
\varphi_c(n) & \text{if } (n,m) \in \mathcal{L} \\
0 & \text{else}
\end{cases}
\]

for \(m \in [1:M]\). See Figure 4 for an illustration.

3.3.3 Local Mode Filtering

The framewise transfer of the labels may lead to very short and scattered runs. Therefore, to obtain longer runs and a more homogeneous labeling, especially at segment boundaries, we introduce a kind of smoothing step by applying a mode filter. The \textit{mode} of a sequence \(S = (s_1, s_2, \ldots, s_N)\) is the most frequently appearing value and is formally defined by \(\text{mode}(S) := \arg \max_{n \in S} \{|n \in [1:N]: s_n = s\}|\). A \textit{local mode filter} of length \(L = 2q + 1\) with \(q \in \mathbb{N}\) replaces each element \(s_n \in S, n \in [1:N]\), in a sequence by the mode of its neighborhood \((s_{n-q}, \ldots, s_{n+q})\):

\[
\text{modefilt}_q(S)(n) := \text{mode}(s_{n-q}, \ldots, s_{n+q}).
\]

Note that the mode may not be unique. In this case, we apply the following strategy in the mode filter. If the element \(s_n\) is one of the modes, \(s_n\) is left unmodified by the filter. Otherwise, one of the modes is chosen arbitrarily.

In our scenario, we apply the local mode filter on a labeled time line \((1:N], \varphi_c)\) by inputting the sequence \(\varphi_c([1:N]) := (\varphi_c(1), \varphi_c(2), \ldots, \varphi_c(N))\) into the filter, see Figure 4 for an illustration. The reason to use the mode opposed to the median to filter segment labels, is that labels are nominal data and therefore have no ordering (integer labels were only chosen for the sake of simplicity).

3.3.4 From Frames to Segments (Filling Up)

In the last step, we derive a segmentation from the label function \(\varphi_c\). As indicated in Section 3.1.3, we could simply detect maximal runs and consider them as segments. However, even after applying the mode filter, there may still be runs sharing the same label that are interrupted by non-labeled parts (labeled zero). In our scenario, we assume that all segments have a distinct label and occur in the same succession as in the reference. Therefore, in the case of a sequence of equally labeled runs that are interrupted by non-labeled parts, we can assume that the runs belong to the same segment. Formally, we assign an index in between two indices with the same label (excluding the zero label) to belong to the same segment as these indices. To construct the final segments, we iterate over each \(k \in [1:K]\) and construct the segments \(\alpha_k = [s_k : e_k]\), such that \(s_k = \min\{m \in [1:M]: \varphi(m) = k\}\), and \(e_k = \max\{m \in [1:M]: \varphi(m) = k\}\), see Figure 4 for an example.

4. EVALUATION

In this section, we compare the previous segment-level matching procedure with our novel frame-level segmenta-

```
Figure 4. Example of frame-level segmentation. The arrows indicate the match between the reference version and the unknown version. (a): Reference label function. (b): Induced label function. (c): Mode filtered version of (b) with length \(L = 3\). (d): Filling up on (c). (e): Ground truth label function.
```

4.1 Tests Set and Evaluation Measure

In the following experiments, we use the recording of Carlos Kleiber performed in 1973 with a duration of 7763 seconds as reference version. The labeled reference segmentation consists of 38 musical segments, see Figure 5. Furthermore, we consider five abridged versions that were recorded between 1933 and 1994. The segments of the opera that are performed in these versions are indicated by Figure 5. Note that the gray parts in the figure correspond to dialogue sections in the opera. In the following experiments, the dialogue sections are considered in the same way as non-labeled (non-musical) parts such as applause, noise or silence. In the partial matching algorithm, they are excluded from the reference version (by setting the similarity score in these regions to minus infinity), and in the segment-level matching procedure, the dialogue parts are not used as queries.

Throughout all experiments, we use CENS features which are a variant of chroma features. They are computed with a feature rate of 1 Hz (derived from 10 Hz pitch features with a smoothing length of 41 frames and a downsampling factor of 10), see [8]. Each feature vector covers roughly 4.1 seconds of the original audio.

In our subsequent experiments, the following segment-level matching (M4) and frame-level segmentation (F1–F4) approaches are evaluated:

- \((M4)\) – Previously introduced segment-level matching, see Section 3.2 and [10] for details.
- \((F1)\) – Frame-level segmentation using a similarity matrix computed with the cosine similarity \(s\) defined by \(s(x, y) = \langle x, y \rangle\) for features \(x\) and \(y\), see Section 3.3.
- \((F2)\) – Frame-level segmentation using a similarity matrix with enhanced path structures using the SM Toolbox [9]. For the computation of the similarity matrix, we used forward/backward smoothing with a smoothing length of 20
frames (corresponding to 20 seconds) with relative tempi between $0.5 - 2$, sampled in 15 steps. Afterwards, a thresholding technique that retained only 5% of the highest values in the similarity matrix and a scaling of the remaining values to $[0, 1]$ is applied. For details, we refer to [9] and Section 3.3.

(F3) – The same as in F2 with a subsequent mode filtering using a filter length $L = 21$ frames, see Section 3.3.3 for details.

(F4) – The segmentation derived from F3 as described in Section 3.3.4.

4.1.1 Frame Accuracy

To evaluate the performance of the different segmentation approaches, we calculate the frame accuracy, which is defined as the ratio of correctly labeled frames and the total number of frames in a version. Given a ground truth label function $\varphi_a$ and an induced label function $\varphi_e$, the frame accuracy $A_f$ is computed as following:

$$A_f := \frac{\sum_{k \in [0, K]} |\varphi^{-1}_a(k) \cap \varphi^{-1}_e(k)|}{\sum_{k \in [0, K]} |\varphi^{-1}_a(k)|}$$

We visualize the accuracy by means of an agreement sequence $\Delta(\varphi_a, \varphi_e)$ which we define as $\Delta(\varphi_a, \varphi_e)(m) := 1$ (white) if $\varphi_a(m) = \varphi_e(m)$ and $\Delta(\varphi_a, \varphi_e)(m) := 0$ (black) otherwise. The sequences $\Delta(\varphi_a, \varphi_e)$ visually correlates well with the values of the frame accuracy $A_f$, see Table 1 and the Figure 6. Note that in structural segmentation tasks, it is common to use different metrics such as the pairwise precision, recall, and f-measure [6]. These metrics disregard the absolute labeling of a frame sequence by relating equally labeled pairs of frames in an estimate to equally labeled frames in a ground truth sequence. However, in our scenario, we want to consider frames that are differently labeled in the ground truth and the induced label function as wrong. As the pairwise f-measure showed the same tendencies as the frame accuracy (which can be easily visualized), we decided to only present the frame accuracy values.

4.2 Qualitative Evaluation

In this section, we qualitatively discuss the results of our approach in more detail by considering the evaluation of the version Kna1939. For each of the five approaches, the results are visualized in a separate row of Figure 6, showing the ground truth $\varphi_a$, the induced label function $\varphi_e$ and the agreement sequence $\Delta(\varphi_a, \varphi_e)$.

For Kna1939, the segment-level matching approach M4 does not work well. Only 28% of the frames are labeled correctly. The red segment, for example, at around 1500 seconds is not matched despite the fact that it has roughly the same overall duration as the corresponding segment in the reference version, see Figure 5. Under closer inspection, it becomes clear that it is performed slower than the corresponding segment in the reference version, and that some material was omitted at the start, in the middle and the end of the segment. The frame-level matching approach F1 leads to an improvement, having a frame accuracy of $A_f = 0.520$. However, there are still many frames wrongly matched. For example, the overture of the opera is missing in Kna1939, but frames from the overture (yellow) of the reference are matched into a segment from the first act (green), see Figure 6. Considering that the opera consists of many scenes with harmonically related material and that the partial matching allows for skipping frames at any point in the alignment, it sometimes occurs that not the semantically corresponding frames are aligned, but harmonically similar ones. This problem is better addressed in approach F2, leading to an improved frame accuracy of 0.788. The enhancement of path structures in the similarity matrix in this approach leads to an increased robustness of the partial matching. Now, all high similarity values are better concentrated in path structures of the similarity matrix.

As a result, the algorithm is more likely to follow sequences of harmonically similar frames, see also Figure 3. However, to follow paths that are not perfectly diagonal, the partial matching algorithm needs to skip frames in the alignment, which leads to a more scattered label function. This is approached by F3 which applies a mode filter on the label function from F2, resulting in an improved frame accuracy.
Table 1. Frame accuracy values on abridged versions. M4: Segment-level matching, F1: Frame-level segmentation, F2: Frame-level segmentation with path-enhanced similarity matrix, F3: Mode filtering with $L = 21$ seconds on F2, F4: Derived Segmentation on F4.

<table>
<thead>
<tr>
<th></th>
<th>dur.(s)</th>
<th>M4</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
<th>F4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kn1939</td>
<td>1965</td>
<td>0.283</td>
<td>0.520</td>
<td>0.788</td>
<td>0.927</td>
<td>0.934</td>
</tr>
<tr>
<td>Kr1933</td>
<td>1417</td>
<td>0.390</td>
<td>0.753</td>
<td>0.777</td>
<td>0.846</td>
<td>0.870</td>
</tr>
<tr>
<td>Mor1939</td>
<td>1991</td>
<td>0.512</td>
<td>0.521</td>
<td>0.748</td>
<td>0.841</td>
<td>0.919</td>
</tr>
<tr>
<td>Ros1956</td>
<td>2012</td>
<td>0.887</td>
<td>0.749</td>
<td>0.817</td>
<td>0.850</td>
<td>0.908</td>
</tr>
<tr>
<td>Sch1994</td>
<td>2789</td>
<td>0.742</td>
<td>0.895</td>
<td>0.936</td>
<td>0.986</td>
<td>0.989</td>
</tr>
<tr>
<td>mean</td>
<td>2035</td>
<td>0.565</td>
<td>0.687</td>
<td>0.815</td>
<td>0.890</td>
<td>0.924</td>
</tr>
</tbody>
</table>

4.3 Quantitative Evaluation

In this section, we discuss the results of Table 1. Note that all abridged versions have less than 50% of the duration of the reference version (7763 seconds). From the mean frame accuracy values for all approaches, we can conclude that the segment-level matching (0.563) is not well suited for dealing with abridged versions, whereas the different strategies in the frame-level approaches F1 (0.687) – F4 (0.924) lead to a subsequent improvement of the frame accuracy. Using the segment-level approach, the frame accuracies for the versions Ros1956 (0.887) and Sch1994 (0.742) stand out compared to the other versions. The segments that are performed in these versions are not shortened and therefore largely coincide with the segments of the reference version. This explains why the segment-level matching still performs reasonably well on these versions.

In Figure 7, we show the frame accuracy results for the approaches M4 and F4 obtained from an experiment on a set of systematically constructed abridged versions. The frame accuracy values at 100% correspond to a subset of 10 segments (out of 38) that were taken from a complete recording of the opera “Der Freischütz” recorded by Keilberth in 1958. From this subset, we successively removed 10% of the frames from each segment by removing 5% of the frames at the start, and 5% of the frames at the end sections of the segments. In the last abridged version, only 10% of each segment remains. This experiment further supports the conclusions that the segment-level approach is not appropriate for dealing with abridged versions, whereas the frame-level segmentation approach stays robust and flexible even in the case of strong abridgments.

5. CONCLUSIONS

In this paper, we approached the problem of transferring the segmentation of a complete reference recording onto an abridged version of the same musical work. We compared the proposed frame-level segmentation approach based on partial matching with a segment-level matching strategy. In experiments with abridged recordings, we have shown that our frame-level approach is robust and flexible when enhancing the path structure of the used similarity matrix and applying a mode filter on the labeled frame sequence before deriving the final segmentation.

Acknowledgments: This work has been supported by the BMBF project Freischütz Digital (Funding Code 01UG1239A to C). The International Audio Laboratories Erlangen are a joint institution of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) and Fraunhofer IIS.

6. REFERENCES

ABSTRACT

Jingju (Beijing opera) is a Chinese traditional performing art form in which theatrical and musical elements are intimately combined. As an oral tradition, its musical dimension is the result of the application of a series of pre-defined conventions and it offers unique concepts for musicological research. Computational analyses of jingju music are still scarce, and only a few studies have dealt with it from an MIR perspective. In this paper we present the creation of a corpus of jingju music in the framework of the CompMusic project that is formed by audio, editorial metadata, lyrics and scores. We discuss the criteria followed for the acquisition of the data, describe the content of the corpus, and evaluate its suitability for computational and musicological research. We also identify several research problems that can take advantage of this corpus in the context of computational musicology, especially for melodic analysis, and suggest approaches for future work.

1. INTRODUCTION

Jingju (also known as Peking or Beijing opera) is one of the most representative genres of xiqu, the Chinese traditional form of performing arts. Just as its name suggests, it consists of a theatrical performance, xi, in which the main expressive element is the music, qu. Although it has commonalities with theatre and opera, it cannot be fully classified as any of those. In xiqu there are not equivalent figures to that of the theatre director or opera composer; instead, the actor is the main agent for creativity and performance. Each of the skills that the actor is expected to master, encompassing poetry, declamation, singing, mime, dance and martial arts, is learned and executed as pre-defined, well established conventions. It is precisely the centrality of the actor and the acting through conventions what make xiqu unique. Its musical content is also created by specific sets of such conventions.

Xiqu genres developed as adaptations of the general common principles of the art form to a specific region, especially in terms of dialect and music. The adoption of local dialects was a basic requirement for the intelligibility of the performance by local audiences. The phonetic features of these dialects, including intonation and especially linguistic tones, establish a melodic and rhythmic framework for the singing. The musical material itself derives from local tunes, which is precisely the literal meaning of qu. This implies that music in xiqu is not an original creation by the actors, but an adaptation of pre-existing material. Furthermore, each genre employs also the most representative instruments of the region for accompaniment, conveying the regional filiation also timbrally. These local features are what define each xiqu genre’s individuality. Jingju is then the regional genre of xiqu that formed in Beijing during the 19th Century, achieving one of the highest levels of refinement and complexity.

Despite the uniqueness of this tradition, the interesting aspects for musicological analysis it offers, and its international recognition, jingju music has barely been approached computationally. Most of the few studies of jingju in MIR have focused on its acoustic and timbral characteristics. Zhang and Zhou have drawn on these features for classification of jingju in comparison with other music traditions [18, 19] and other xiqu genres [20]. Sundberg et al. have analyzed acoustically the singing of two role-types [14], whilst Tian et al. have extracted timbral features for onset detection of percussion instruments [15]. More recently, Zhang and Wang have integrated domain knowledge for musically meaningful segmentation of jingju arias [21]. Related to melodic analysis, Chen [3] has implemented a computational analysis of jinju music for the characterization of pitch intonation.

The main concepts that define jingju music are shengqiang, banshi and role-type. As stated previously, the melodic material used in xiqu genres is not original, but derived from local tunes. These tunes share common features that allow them to be recognized as pertaining to that specific region, such as usual scale, characteristic timbre, melodic structure, pitch range and tessitura, ornamentation, etc. This set of features is known as shengqiang, usually translated into English as ‘mode’ or ‘modal system’ [16]. Each xiqu genre can use one or more shengqiang; and one single shengqiang can be shared by different genres. There are two main shengqiang in jingju, namely xipi and erhuang (see Table 2). Their centrality in the genre is such that jingju music as a whole has been also named by the combination of these two terms, pihuang.
The melodic features determined by the shengqiang are rhythmically rendered through a series of metrical patterns called banshi. These banshi are individually labelled and defined by a unit of metre, a tempo value and a degree of melodic density; they are associated as well to an expressive function. The system of banshi is conceived as derived from an original one, called yuanban, so that the rest of them are expansions, reductions or free realizations of the first one [8]. The banshi system in jingju consists of a core of eight patterns commonly used, plus some variants.

Each of the characters of a play is assigned to one specific, pre-defined acting class, according to their gender, age, social status, psychological profile and emotional behavior. These acting classes are known as hangdang or role-types, and each actor is specialized in the performance of one of them. Each role-type determines the specific set of conventions that must be used for the creation of the character, including those attaining to music. Consequently, shengqiang and banshi will be expressed differently by each role-type, so that these concepts cannot be studied without referencing each other. In jingju there are four general categories of role-types, with further subdivisions. We consider that the five main role-types regarding musical expression are sheng (male characters), dan (female characters), jing (painted-face), xiaosheng (young males), and laodan (old females). They are usually classified into two styles of singing, the male style, characterized for using chest voice, used by sheng, jing and laodan, and the female one, sung in falsetto and higher register, used by dan and xiaosheng.

The fullest expression of such melodic concepts occurs in the singing sections called changduan, which can be compared, but not identified, with the concept of aria in Western opera (to ease readability, we will use the term ‘aria’ throughout the paper). Consequently, we have determined the aria as our main research object, and it has been the main concern for the creation of our corpus and the analyses suggested.

In this paper we present a corpus of jingju music that we have gathered for its computational analysis. We explain the criteria followed for the collection of its different types of data, describe the main features of the corpus and discuss its suitability for research. Thereupon we explore the possibilities that the corpus offers to computational musicology, focusing in melodic analysis, specifically in the concepts of shengqiang and role-type.

2. JINGJU MUSIC RESEARCH CORPUS

In order to undertake a computational analysis of jingju music, and to exploit the unique musical concepts of this tradition from an MIR perspective, we have gathered in the CompMusic project a research corpus [12] that includes audio, editorial metadata, lyrics and scores. We introduce here the criteria for the selection of the data, describe its content and offer a general evaluation.

2.1 Criteria for data collection

For the collection of audio recordings, which is the core of the corpus, we have considered three main criteria: repertoire to be covered, sound quality and recording unit. In order to take maximum advantage of the unique features of jingju music, we have gathered recordings of mostly traditional repertoire, as well as some modern compositions based on the traditional methods. The so-called contemporary plays, since they have been created integrating compositional techniques from the Western tradition, have been disregarded for our corpus. Regarding the sound quality needed for a computational analysis, and considering the material to which we have had access, the recordings that best suited our requirements have been commercial CDs released in the last three decades in China. Finally, since our main research object is the aria, we have acquired CDs releases of single arias per track. This means that full play or full scene CDs, and video material in VCD and DVD have not been considered. These CDs have been the source from which we have extracted the editorial metadata contained in the corpus, which have been stored in MusicBrainz.¹ This platform assigns one unique ID to each entity in the corpus, so that they can be easily searchable and retrievable.

Our aim has been to include for each audio recording, whenever possible, its corresponding lyrics and music score. All releases gathered included the lyrics in their leaflets for the arias recorded. However, since they are not usable for computational purposes, we get them from specialized free repositories in the web.² As for the scores, an explanation of its function in the tradition is first needed. Since jingju music has been created traditionally by actors, no composers, drawing on pre-existing material, scores appeared only as an aide-mémoire and a tool for preserving the repertoire. Although barely used by professional actors, scores have been widely spread among amateur singers and aficionados, and are a basic resource for musicological research. In fact, in the last decades there has been a remarkable effort to publish thoroughly edited collections of scores. Although many scores are available in the web, they have not been systematically and coherently stored, what makes them not easily retrievable. Furthermore, they consist of image or pdf files, not usable computationally. Consequently, we have acquired printed publications that meet the academic standards of edition, but that will require to be converted into a machine readable format.

2.2 Description of the corpus

The audio data collection of the corpus is formed by 78 releases containing 113 CDs, consisting of collections of single arias per track. Besides these, due to the fact that

¹ http://musicbrainz.org/collection/4dd0978b-0796-4734-96d4-2b3ebe0864c
many of the consumers of these recordings are amateur singers, many releases contain extra CDs with just the instrumental accompaniment of the arias recorded in the main ones. Consequently, the corpus also contains 19 CDs with just instrumental accompaniments.

Although we do not have complete figures yet, we have computed statistics for different aspects of the corpus. The releases contain recordings by 74 singers, belonging to 7 different role-types, as indicated in Table 1.

<table>
<thead>
<tr>
<th>Role-types</th>
<th>Number of singers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laosheng</td>
<td>20</td>
</tr>
<tr>
<td>Jing</td>
<td>55</td>
</tr>
<tr>
<td>Laodan</td>
<td>66</td>
</tr>
<tr>
<td>Dan</td>
<td>257</td>
</tr>
<tr>
<td>Xiaosheng</td>
<td>43</td>
</tr>
<tr>
<td>Washeng</td>
<td>3</td>
</tr>
<tr>
<td>Chou</td>
<td>5</td>
</tr>
<tr>
<td>Xipi</td>
<td>9</td>
</tr>
<tr>
<td>Wu sheng</td>
<td>3</td>
</tr>
<tr>
<td>Chou</td>
<td>3</td>
</tr>
<tr>
<td>Dan</td>
<td>24</td>
</tr>
</tbody>
</table>

Table 1. Number of singers per role-type in the corpus.

As for the works, the corpus covers 653 arias from 215 different plays. Table 2 shows the distribution of these arias according to role-type and shengqiang. Since the number of banshi is limited and all of them frequently used, an estimation of its appearance in the corpus is not meaningful. As shown in Table 2, the corpus contains highly representative samples for the research of the two main shengqiang and the five main role-types as described in section 1. Table 3 displays more detailed numbers concerning these specific entities. The editorial metadata stored in MusicBrainz include textual information as well as cover art. For the former the original language has been maintained, that is, Chinese in simplified characters, with romanizations in the Pinyin system, stored either as pseudo-releases or aliases.

<table>
<thead>
<tr>
<th>Role-types</th>
<th>Number of arias</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laosheng</td>
<td>224</td>
</tr>
<tr>
<td>Jing</td>
<td>55</td>
</tr>
<tr>
<td>Laodan</td>
<td>66</td>
</tr>
<tr>
<td>Dan</td>
<td>257</td>
</tr>
<tr>
<td>Xiaosheng</td>
<td>43</td>
</tr>
<tr>
<td>Washeng</td>
<td>3</td>
</tr>
<tr>
<td>Chou</td>
<td>5</td>
</tr>
<tr>
<td>Xipi</td>
<td>324</td>
</tr>
<tr>
<td>Erhuang</td>
<td>200</td>
</tr>
<tr>
<td>Fan erhuang</td>
<td>31</td>
</tr>
<tr>
<td>Nanbangzi</td>
<td>25</td>
</tr>
<tr>
<td>Sipingdiao</td>
<td>23</td>
</tr>
<tr>
<td>Others</td>
<td>45</td>
</tr>
<tr>
<td>Unknown</td>
<td>5</td>
</tr>
</tbody>
</table>

Table 2. Distribution of the arias in the corpus according to role-type and shengqiang.

Regarding the scores, the corpus contains two collections of full play scores [5, 22] and an anthology of selected arias [6]. The two collections contain a total of 155 plays, 26 of which appear in both publications; the anthology contains 86 scores. This material offers scores for 317 arias of the corpus, that is 48.5% of the total.

Apart from the research corpus, but related to it, specific test corpora will be developed, consisting of collections of data used as ground truth for specific research tasks, as defined by Serra [12]. The test corpora created in the framework of theCompMusic project are accessible from the website http://compmusic.upf.edu/datasets. To date there are two test corpora related to the jingju music corpus, namely the Beijing opera percussion instrument dataset, which contains 236 audio samples of jingju percussion instruments, used by Tian et al. [15] for onset detection of these instruments, and the Beijing opera percussion pattern dataset, formed by 133 audio samples of five jingju percussion patterns, supported by transcriptions both in staff and syllable notations. Srinivasamurthy et al. [13] have used this dataset for the automatic recognition of such patterns in jingju recordings.

2.3 Evaluation of the corpus

For the evaluation of the corpus, we will draw on some of the criteria defined by Serra [12] for the creation of culture specific corpora, specifically coverage and completeness, and discuss as well the usability of the data for computational analyses.

2.3.1. Coverage

Assessing the coverage of the jingu music corpus is not an easy task, since, to the best of our knowledge, there is no reference source that estimates the number of plays in this tradition. However, compared with the number of total plays covered in our collections of full play scores, which are considered to be the most prominent publications in this matter, the number of plays represented in our corpus is considerable higher. Besides, these releases have been purchased in the specialized bookshop located in the National Academy of Chinese Theatre Arts, Beijing, the only institution of higher education in China dedicated exclusively to the training of xiqu actors, and one of the most acclaimed ones for jingju. Our corpus contains all the releases available in this bookshop at the time of writing this paper that met the criteria settled in section 2.1. Regarding the musical concepts represented in the corpus, Table 2 shows that both systems of role-type and shengqiang are equally fully covered, with unequal proportion according to their relevance in the tradition, as explained in the introduction. As for the banshi, as stated previously, they are fully covered due to their limited number and varied use. Consequently, we argue that the coverage of our corpus is highly satisfactory, in terms of variety of repertoire, availability in the market and representation of musical entities.

2.3.2. Completeness

Considering the musicological information needed for each recording according to our purposes, the editorial metadata contained in the releases are fully complete, with the exception of the 5 arias mentioned in Table 2 (0.8% of the total), which lack information about shengqiang and banshi. One important concept for aficionados of this tradition is the one of liupai, or performing schools. However, this concept is far from being well defined, and depends both on the play and on the per-
former, and usually is not specifically stated in the releases. Finally, the information related to the publication of the recordings is not consistent. Usually, the dates of recording and releasing are not available from the CDs. However, the releasing period has been restricted to the last three decades by our criteria, as stated in section 2.1, although in some rare cases some of these releases may contain recordings from earlier periods.

### 2.3.3. Usability

The data contained in the corpus are fully suitable for analysis of jingju music according to the musical concepts explained in section 1. However, not all the data are equally usable. The main difficulty is presented by the scores, to date available only in printed edition. Consequently, for their computational exploitation they need to be converted into a machine readable format. In the CompMusic project we intend to use MusicXML, maintaining the so called jianpu notation used in the originals. As for the lyrics, although most of them are freely accessible on the web, due to the fact that singers may make some changes according to their needs, some problems for the recognition of related lyrics for a specific aria might rise.

To access the corpus for research purposes, we refer to the website [http://compmusic.upf.edu/corpora](http://compmusic.upf.edu/corpora). The corpus will eventually be also available through Dunya [9], a web based browsing tool developed by the CompMusic project, which also displays content-based analyses carried out in its framework for each of the culturally specific corpora that it has gathered.

### 3. RESEARCH POSSIBILITIES FOR THE JINGJU MUSIC CORPUS

In this section we introduce research issues of relevance for each data type in our corpus with a special focus on the melodic analysis. We discuss the application of state of the art analytic approaches to our corpus, and propose specific future work.

#### 3.1 Analyses of audio, lyrics and scores

According to the research objectives in the CompMusic project, in whose framework our corpus has been gathered, audio data is the main research object, supported by information from metadata, lyrics and scores. For the analysis of the musical elements described in the first section, the vocal line of the arias is the most relevant element, since it renders the core melody of the piece. Consequently, segmentation of the vocal part and extraction of its pitch are needed steps. However, the timbral and textural characteristics of jingju music pose important challenges for these tasks. The timbre of the main accompanying instrument, the jinghu, a small, two-stringed spike fiddle, is very similar to that of the voice. Besides, the typical heterophonic texture results in the simultaneous realization of different versions of the same melody. These features make the extraction of the vocal line from the accompaniment difficult. Besides, octave errors are still frequent to state of the art algorithms for predominant melody extraction, especially for role-types of the male style of singing.

If audio is the main research object, the other types of data in our corpus offer equally interesting and challenging tasks for computational analysis. The delivery of the lyrics is the main goal of singing in jingju; therefore their analysis is essential for the understanding of the genre. Of special importance for its musical implications is the analysis of the poetic structure of the lyrics, since it determines the musical one, as well as their meaning, what would help to better define the expressive function of shengqiang and banshi. Methods from natural language processing can be applied for the identification of poetic formulae, commonly used by actors for the creation of new lyrics. As for the scores, their analysis will be beneficial for the computation of intervallic preferences, creation of cadential schemata and detection of stable pitches.

However, as stated previously, the main use of lyrics and scores according to our research purposes will be as supporting elements for audio analysis. To that aim, the main computational task is the alignment of both data types to audio. This is a challenging task, since the actors, in a tradition without the authority of a composer or playwright, have certain margins to modify lyrics and melody according to their own interpretation, as far as the main features of the aria, as sanctioned by the tradition, are maintained. In the case of lyrics, this task is even more complex due to the fact that jinju uses an art language of its own, that combines linguistic features from two dialects, the Beijing dialect and the Huguang dialect.
from the South [8, 16]. This combination is far from being systematic and consistent, what in many cases poses difficulties to the prediction of the phonetic representation required for lyrics to audio alignment. The music traditions researched in the CompMusic project present similar problems for these tasks, and specific approaches have been proposed by Şentürk et al. [11] and Dzhambazov et al. [4]. We intend to benefit from these works for the development of specific methods for jingju music.

Alignment of lyrics and scores to audio will be an important step for several analytical tasks. The information from these data types combined with the audio will allow a musically informed segmentation of the recording, either in vocal or instrumental sections, or in different structural units, from the couplet, a poetic structure which is the basic unit also for the musical one, to the syllable level. The absolute pitch value of the first degree can be computed by the combined information from the score and the pitch track. Finally, an especially interesting topic is the study of how the tonal information of the syllable is expressed in the melody. Zhang et al. [17] have applied computational methods to this issue within the context of the CompMusic project.

3.2. Characterization of shengqiang and role-type

As stated in the introduction, the two more relevant concepts for the melodic aspect of jingju are shengqiang and role-type. Chen [3] has attempted a characterization of these entities by means of pitch histograms. For the classification of audio fragments as vocal and non-vocal Chen drew on machine learning, and extracted the pitch of the vocal line with the algorithm proposed by Salamon and Gómez [10]. In order to overcome some limitations of the results in this work, we have carried out an initial experiment in which we have extracted pitch tracks for a subset of 30 arias from our corpus that have been manually pre-processed. The sample contains three arias for each of the ten combinations of the two main shengqiang and the five main role-types. We use mp3 mono files with a sampling rate of 44,100 Hz, and have annotated them with Praat [1] to the syllable level for segmentation. Pitch tracks have been obtained with the aforementioned algorithm [10] implemented in Essentia [2], whose parameters have been manually set for each aria.

The obtained pitch tracks have been used for the computation of pitch histograms. Koduri et al. have successfully characterized Carnatic ragas by histogram peak parametrization [7]. Chen has applied this methodology for the characterization of male and female styles of singing in jingju. We have expanded the same approach to our subset of 30 arias, with the aim of characterizing the ten combinations of shengqiang and role-types. Our initial observations give some evidence that pitch histograms will help describe some aspects of shengqiang and role-types as stated in the musicological literature, such as modal center, register with respect to the first degree, range and hierarchy of scale degrees, so that differences can be established between each category. Our results also show that the approach is efficient for the characterization of different role-types for the same shengqiang. However, differences are not meaningful when the two shengqiang are compared for one single role-type. Figure 1 shows how the modal center for both xipi and erhuag in a dan role-type is located around the fifth and sixth degrees, register with respect to the first degree and range are practically identical, and the differences in the hierarchy of scale degrees are not relevant enough.

![Pitch histograms for the dan role-type in the two shengqiang.](image)

In our future work we propose to expand this approach by integrating the information obtained from the lyrics and the scores. In the specific case of shengqiang, a work of melodic similarity between arias of the shengqiang according to their musical structure, specially determined by the banshi, will shed light on the melodic identity of these entities. As for the role-type, we argue that an analysis of timbre, dynamics and articulation for each category, especially at the syllable level, will offer characterizing features that complete the information obtained from the pitch histograms.

3.3. Other research tasks

Beyond the tasks described previously, jingju music offer a wide range of research possibilities. One important aspect is the rhythmic component of the arias, mainly determined by the concept of banshi. An automatic identification of banshi and segmentation of the aria in these sections is a musically meaningful, but computational challenging task, due to the different renditions of the same banshi by different role-types and even different actors, as well as to the rhythmic flexibility that characterizes jingju music. The instrumental sections of the jingju arias are also an interesting research topic, especially regarding their relationship with the melody of the vocal part and how shengqiang and banshi define their features. To this task, the CDs with only accompaniment tracks will be valuable. In the framework of the CompMusic project, Srinivasamurthy et al. [13] have presented a computational model for the automatic recognition of percussion patterns in jingju. Finally, due to the importance of the acting component of this genre, and its intimate relationship with the music, jingju is a perfect case for a com-
In this paper we have presented a corpus of jingju music, gathered with the purpose of researching its musical features from an MIR methodology. After discussing the criteria for its creation, describing its different data types and offering a general evaluation, we have suggested analytical tasks for its computational exploitation, especially focused on melodic analysis. Some state of the art approaches have been applied to a small sample of the corpus, in order to analyze their results and propose consequently further work and future tasks.

5. ACKNOWLEDGEMENTS
This research was funded by the European Research Council under the European Union’s Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583). We are thankful to G. K. Koduri for providing and helping with his code.

6. REFERENCES
MODELING TEMPORAL STRUCTURE IN MUSIC FOR EMOTION PREDICTION USING PAIRWISE COMPARISONS

Jens Madsen, Bjørn Sand Jensen, Jan Larsen
Technical University of Denmark,
Department of Applied Mathematics and Computer Science,
Richard Petersens Plads, Building 321,
2800 Kongens Lyngby, Denmark
{jenma,bjje,janla}@dtu.dk

ABSTRACT
The temporal structure of music is essential for the cognitive processes related to the emotions expressed in music. However, such temporal information is often disregarded in typical Music Information Retrieval modeling tasks of predicting higher-level cognitive or semantic aspects of music such as emotions, genre, and similarity. This paper addresses the specific hypothesis whether temporal information is essential for predicting expressed emotions in music, as a prototypical example of a cognitive aspect of music. We propose to test this hypothesis using a novel processing pipeline: 1) Extracting audio features for each track resulting in a multivariate "feature time series". 2) Using generative models to represent these time series (acquiring a complete track representation). Specifically, we explore the Gaussian Mixture model, Vector Quantization, Autoregressive model, Markov and Hidden Markov models. 3) Utilizing the generative models in a discriminative setting by selecting the Probability Product Kernel as the natural kernel for all considered track representations. We evaluate the representations using a kernel based model specifically extended to support the robust two-alternative forced choice self-report paradigm, used for eliciting expressed emotions in music. The methods are evaluated using two data sets and show increased predictive performance using temporal information, thus supporting the overall hypothesis.

1. INTRODUCTION
The ability of music to represent and evoke emotions is an attractive and yet a very complex quality. This is partly a result of the dynamic temporal structures in music, which are a key aspect in understanding and creating predictive models of more complex cognitive aspects of music such as the emotions expressed in music. So far the approach of creating predictive models of emotions expressed in music has relied on three major aspects. First, self-reported annotations (rankings, ratings, comparisons, tags, etc.) for quantifying the emotions expressed in music. Secondly, finding a suitable audio representation (using audio or lyrical features), and finally associating the two aspects using machine learning methods with the aim to create predictive models of the annotations describing the emotions expressed in music. However the audio representation has typically relied on classic audio-feature extraction, often neglecting how this audio representation is later used in the predictive models.

We propose to extend how the audio is represented by including feature representation as an additional aspect, which is illustrated on Figure 1. Specifically, we focus on including the temporal aspect of music using the added feature representation [10], which is often disregarded in the classic audio-representation approaches. In Music Information Retrieval (MIR), audio streams are often represented with frame-based features, where the signal is divided into frames of samples with various lengths depending on the musical aspect which is to be analyzed. Feature extraction based on the enframed signal results in multivariate time series of feature values (often vectors). In order to use these features in a discriminative setting (i.e. predicting tags, emotion, genre, etc.), they are often represented using the mean, a single or mixtures of Gaussians (GMM). This can reduce the time series to a single vector and make the features easy to use in traditional linear models or kernel machines such as the Support Vector Machine (SVM). The major problem here is that this approach disregards all temporal information in the extracted features. The frames could be randomized and would still have the same representation, however this randomization makes no sense musically.

In modeling the emotions expressed in music, the temporal aspect of emotion has been centered on how the labels are acquired and treated, not on how the musical content is treated. E.g. in [5] they used a Conditional Random Field (CRF) model to essentially smooth the predicted labels of an SVM, thus still not providing temporal information re-

© Jens Madsen, Bjørn Sand Jensen, Jan Larsen.
Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0). Attribution: Jens Madsen, Bjørn Sand Jensen, Jan Larsen.

This work was supported in part by the Danish Council for Strategic Research of the Danish Agency for Science Technology and Innovation under the CoSound project, case number 11-115328.
regarding the features. In [12] a step to include some temporal information regarding the audio features was made, by including some first and second order Markov properties for their CRF model, however still averaging the features for one second windows. Other approaches have ranged from simple feature stacking in [13] to actually using a generative temporal model to represent features in [17]. The latter showed that using a Dynamical Texture Mixture model to represent the feature time series of MFCCs, taking temporal dynamics into account, carried a substantial amount of information about the emotional content. In the present work, in contrast to prior work, we focus on creating a common framework by using generative models to represent the multivariate feature time series for the application of modeling aspects related to the emotions expressed in music. Since very little work has been done within this field, we make a broad comparison of a multitude of generative models of time series data. We consider how the time series are modeled on two aspects: whether the observations are continuous or discrete, and whether temporal information should be taken into account or not. This results in four different combinations, which we investigate: 1) a continuous, temporal, independent representation which includes the mean, single Gaussian and GMM models; 2) a temporal, dependent, continuous representation using Autoregressive models; 3) a discretized features representation using vector quantization in a temporally independent Vector Quantization (VQ) model; and finally 4) a representation including the temporal aspect fitting Markov and Hidden Markov Models (HMM) on the discretized data. A multitude of these models have never been used in MIR as a track-based representation in this specific setting. To use these generative models in a discriminative setting, the Product Probability Kernel (PPK) is selected as the natural kernel for all the feature representations considered. We extend a kernel-generalized linear model (kGLM) model specifically for pairwise observations for use in predicting emotions expressed in music. We specifically focus on the feature representation and the modeling pipeline and therefore use simple, well-known, frequently used MFCC features. In total, eighteen different models are investigated on two datasets of pairwise comparisons evaluated on the valence and arousal dimensions.

2. FEATURE REPRESENTATION

In order to model higher order cognitive aspects of music, we first consider standard audio feature extraction which results in a frame-based, vector space representation of the music track. Given $T$ frames, we obtain a collection of $T$ vectors with each vector at time $t$ denoted by $x_t \in \mathbb{R}^D$, where $D$ is the dimension of the feature space. The main concern here is how to obtain a track-level representation of the sequence of feature vectors for use in subsequent modelling steps. In the following, we will outline a number of different possibilities — and all these can be considered as probabilistic densities over either a single feature vector or a sequence of such (see also Table. 1).

Continuous: When considering the original feature space, i.e. the sequence of multivariate random variables, a vast number of representations have been proposed depending on whether the temporal aspects are ignored (i.e. considering each frame independently of all others) or modeling the temporal dynamics by temporal models.

In the time-independent case, we consider the feature as a bag-of-frames, and compute moments of the independent samples; namely the mean. Including higher order moments will naturally lead to the popular choice of representing the time-collapsed time series by a multivariate Gaussian distribution (or other continuous distributions). Generalizing this leads to mixtures of distributions such as the GMM (or another universal mixture of other distributions) used in an abundance of papers on music modeling and similarity (e.g. [1, 7]).

Instead of ignoring the temporal aspects, we can model the sequence of multivariate feature frames using well-known temporal models. The simplest models include AR models [10].

Discrete: In the discrete case, where features are naturally discrete or the original continuous feature space can be quantized using VQ with a finite set of codewords resulting in a dictionary (found e.g. using K-means). Given this dictionary each feature frame is subsequently assigned a specific codeword in a 1-of-$P$ encoding such that a frame at time $t$ is defined as vector $\tilde{x}_t$ with one non-zero element.

At the track level and time-independent case, each frame is encoded as a Multinomial distribution with a single draw, $\tilde{x} \sim \text{Multinomial}(\lambda, 1)$, where $\lambda$ denotes the probability of occurrence for each codeword and is computed on the basis of the histogram of codewords for the entire track. In the time-dependent case, the sequence of codewords, $\tilde{x}_0, \tilde{x}_1, \ldots, \tilde{x}_T$, can be modeled by a relatively simple (first order) Markov model, and by introducing hidden states this may be extended to the (homogeneous) Hidden Markov model with Multinomial observations ($\text{HMM}_{\text{disc}}$).

2.1 Estimating the Representation

The probabilistic representations are all defined in terms of parametric densities which in all cases are estimated using standard maximum likelihood estimation (see e.g. [2]). Model selection, i.e. the number of mixture components, AR order, and number of hidden states, is performed using
which is equivalent to formulating a PPK with a Gaussian with the given mean and a common, diagonal covariance matrix.

Table 1

<table>
<thead>
<tr>
<th>Obs. Time</th>
<th>Representation</th>
<th>Density Model</th>
<th>θ</th>
<th>Base</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuous</td>
<td>Mean</td>
<td>( p(x; \theta) \equiv \mathcal{N}(\mu, \Sigma) )</td>
<td>( \mu, \sigma )</td>
<td>Gaussian</td>
</tr>
<tr>
<td></td>
<td>Gaussian</td>
<td>( p(x; \theta) = \mathcal{N}(x</td>
<td>\mu, \Sigma) )</td>
<td>( \mu, \Sigma )</td>
</tr>
<tr>
<td></td>
<td>GMM</td>
<td>( p(x; \theta) = \sum_{i=1}^{L} \lambda_i \mathcal{N}(x</td>
<td>\mu_i, \Sigma_i) )</td>
<td>( {\lambda_i, \mu_i, \Sigma_i}_{i=1}^{L} )</td>
</tr>
<tr>
<td>Temp.</td>
<td>AR</td>
<td>( p(x_0, x_1, \ldots, x_T; \theta) = \mathcal{N}(x_0</td>
<td>\Sigma_0, \Sigma_{A,C}) )</td>
<td>( \Sigma_{A,C} )</td>
</tr>
<tr>
<td>Discrete</td>
<td>VQ</td>
<td>( p(x; \theta) = \lambda )</td>
<td>( \lambda )</td>
<td>Multinomial</td>
</tr>
<tr>
<td></td>
<td>Markov</td>
<td>( p(x_0, x_1, \ldots, x_T; \theta) = \lambda \sum_{T=1}^{M} \prod_{t=1}^{T} \mathcal{A}<em>{x_t, x</em>{t-1}} )</td>
<td>( \lambda, \Lambda )</td>
<td>Multinomial</td>
</tr>
<tr>
<td></td>
<td>HMM_disc</td>
<td>( p(x_0, x_1, \ldots, x_T; \theta) = \sum_{\gamma} \lambda_{\gamma} \prod_{t=1}^{T} \Phi_{t} )</td>
<td>( \lambda, \Lambda, \Phi )</td>
<td>Multinomial</td>
</tr>
</tbody>
</table>

Table 1. Continuous, features, \( x \in \mathbb{R}^D \), \( L \) is the number of components in the GMM, \( P \) indicates the order of the AR model, \( A \) and \( C \) are the coefficients and noise covariance in the AR model respectively and \( T \) indicates the length of the sequence. Discrete, VQ: \( x \sim \text{Multinomial}(\Lambda) \), \( \Lambda_{x_t, x_{t-1}} = p(z_t|x_{t-1}) \), \( \Lambda_{x_t, x_{t-1}} = p(x_t|x_{t-1}) \), \( \Phi_t = p(x_t|z_t) \). The basic Mean representation is often used in the MIR field in combination with a so-called squared exponential kernel [2], which is equivalent to formulating a PPK with a Gaussian with the given mean and a common, diagonal covariance matrix corresponding to the length scale which can be found by cross-validation and specifically using \( q = 1 \) in the PPK.

Bayesian Information Criterion (BIC, for GMM and HMM), or in the case of the AR model, CV was used.

2.2 Kernel Function

The various track-level representations outlined above are all described in terms of a probability density as outlined in Table 1, for which a natural kernel function is the Probability Product Kernel [6]. The PPK forms a common ground for comparison and is defined as,

\[
k(p(x;\theta), p(x';\theta')) = \int (p(x;\theta) p(x';\theta'))^q d\mathbf{x}, \quad (1)
\]

where \( q > 0 \) is a free model parameter. The parameters of the density model, \( \theta \), obviously depend on the particular representation and are outlined in Tab.1. All the densities discussed previously result in (recursive) analytical computations. [6, 11].

3. PAIRWISE KERNEL GLM

The pairwise paradigm is a robust elicitation method to the more traditional direct scaling approach and is reviewed extensively in [8]. This paradigm requires a non-traditional modeling approach for which we derive a relatively simple kernel version of the Bradley-Terry-Luce model [3] for pairwise comparisons. The non-kernel version was used for this particular task in [9].

In order to formulate the model, we will for now assume a standard vector representation for each of \( N \) audio excerpts collected in the set \( \mathcal{X} = \{x_i|i = 1, \ldots, N\} \), where \( x_i \in \mathbb{R}^D \) denotes a standard, \( D \) dimensional audio feature vector for excerpt \( i \). In the pairwise paradigm, any two distinct excerpts with index \( u \) and \( v \), where \( x_u \in \mathcal{X} \) and \( x_v \in \mathcal{X} \), can be compared in terms of a given aspect (such as arousal/valance). With \( M \) such comparisons we denote the output set as \( \gamma = \{(y_m; u_m, v_m)|m = 1, \ldots, M\} \), where \( y_m \in \{-1, +1\} \) indicates which of the two excerpts had the highest valence (or arousal). \( y_m = -1 \) means that the \( u_m \)'th excerpt is picked over the \( v_m \)'th excerpt and visa versa when \( y_m = 1 \).

The basic assumption is that the choice, \( y_m \), between the two distinct excerpts, \( u \) and \( v \), can be modeled as the difference between two function values, \( f(x_u) \) and \( f(x_v) \). The function \( f: \mathcal{X} \rightarrow \mathbb{R} \) hereby defines an internal, but latent, absolute reference of valence (or arousal) as a function of the excerpt (represented by the audio features, \( x \)).

Modeling such comparisons can be accomplished by the Bradley-Terry-Luce model [3, 16], here referred to more generally as the (logistic) pairwise GLM model. The choice model assumes logistically distributed noise [16] on the individual function value, and the likelihood of observing a particular choice, \( y_m \), for a given comparison \( m \) therefore becomes

\[
p(y_m|f_m) = \frac{1}{1 + e^{-y_m z_m}}, \quad (2)
\]

with \( z_m = f(x_{u_m}) - f(x_{v_m}) \) and \( f_m = [f(x_{u_m}), f(x_{v_m})]^T \). The main question is how the function, \( f(\cdot) \), is modeled. In the following, we derive a kernel version of this model in the framework of kernel Generalized Linear Models (kGLM).

We start by assuming a linear and parametric model of the form \( f_i = \mathbf{x}_i^\top \mathbf{w} \) and consider the likelihood defined in Eq. (2). The argument, \( z_m \), is now redefined such that \( z_m = (\mathbf{x}_{u_m} \mathbf{w}^\top - \mathbf{x}_{v_m} \mathbf{w}^\top) \). We assume that the model parameterized by \( \mathbf{w} \) is the same for the first and second input, i.e. \( x_{u_m} \) and \( x_{v_m} \). This results in a projection from the audio features \( \mathbf{x} \) into the dimensions of valence (or arousal) given by \( \mathbf{w} \), which is the same for all excerpts. Plugging this into the likelihood function we obtain:

\[
p(y_m|x_{u_m}, x_{v_m}, \mathbf{w}) = \frac{1}{1 + e^{-y_m (x_{u_m}-x_{v_m})^\top \mathbf{w}}}. \quad (3)
\]
Following a maximum likelihood approach, the effective cost function, $\psi(\cdot)$, defined as the negative log likelihood is:

$$
\psi_{GLM}(w) = -\sum_{m=1}^{M} \log p(y_m | x_{u_m}, x_{v_m}, w). \quad (4)
$$

Here we assume that the likelihood factorizes over the observations, i.e., $p(Y|\mathbf{f}) = \prod_{m=1}^{M} p(y_m | \mathbf{f}_m)$. Furthermore, a regularized version of the model is easily formulated as

$$
\psi_{GLM-L2}(w) = \psi_{GLM} + \gamma \| w \|^2, \quad (5)
$$

where the regularization parameter $\gamma$ is to be found using cross-validation, for example, as adopted here. This cost is still continuous and is solved with a L-BFGS method.

This basic pairwise GLM model has previously been used to model emotion in music [9]. In this work, the pairwise GLM model is extended to a general regularized kernel formulation allowing for both linear and non-linear models. First, consider an unknown non-linear map of an element $x \in \mathcal{X}$ into a Hilbert space, $\mathcal{H}$, i.e., $\varphi(x) : \mathcal{X} \rightarrow \mathcal{H}$. Thus, the argument $z_m$ is now given as

$$
z_m = (\varphi(x_{u_m}) - \varphi(x_{v_m})) w^T \quad (6)
$$

The representer theorem [14] states that the weights, $w$ — despite the difference between mapped instances — can be written as a linear combination of the inputs such that

$$
w = \sum_{m=1}^{M} \alpha_l (\varphi(x_{u_l}) - \varphi(x_{v_l})). \quad (7)
$$

Inserting this into Eq. (6) and applying the "kernel trick" [2], i.e. exploiting that $\langle \varphi(x), \varphi(x') \rangle_{\mathcal{H}} = k(x, x')$, we obtain

$$
z_m = (\varphi(x_{u_m}) - \varphi(x_{v_m})) \sum_{l=1}^{M} \alpha_l (\varphi(x_{u_l}) - \varphi(x_{v_l}))
= \sum_{l=1}^{M} \alpha_l (\varphi(x_{u_m}) \varphi(x_{u_l}) - \varphi(x_{u_m}) \varphi(x_{v_l}))
= \varphi(x_{v_m}) \varphi(x_{u_l}) + \varphi(x_{v_l}) \varphi(x_{u_m})
= \sum_{l=1}^{M} \alpha_l (k(x_{u_m}, x_{u_l}) - k(x_{u_m}, x_{v_l})
= k(x_{v_m}, x_{u_l}) + k(x_{v_l}, x_{u_m})
= \sum_{l=1}^{M} \alpha_l (\{x_{u_m}, x_{v_l}\}, \{x_{u_l}, x_{v_l}\}). \quad (8)
$$

Thus, the pairwise kernel GLM formulation leads exactly to standard kernel GLM like [19], where the only difference is the kernel function which is now a (valid) kernel between two sets of pairwise comparisons. If the kernel function is the linear kernel, we obtain the basic pairwise logistic regression presented in Eq. (3), but the kernel formulation easily allows for non-vectorial inputs as provided by the PPK. The general cost function for the kGLM model is defined as,

$$
\psi_{kGLM-L2}(\alpha) = -\sum_{m=1}^{M} \log p(y_m | \alpha, K) + \gamma \alpha^T K \alpha,
$$

i.e., dependent on the kernel matrix, $K$, and parameters $\alpha$. It is of the same form as for the basic model and we can apply standard optimization techniques. Predictions for unseen input pairs $\{x_r, x_s\}$ are easily calculated as

$$
\Delta f_{rs} = f(x_r) - f(x_s) \quad (9)
$$

$$
= \sum_{m=1}^{M} \alpha_m k(\{x_{u_m}, x_{v_m}\}, \{x_r, x_s\}). \quad (10)
$$

Thus, predictions exist only as delta predictions. However it is easy to obtain a “true” latent (arbitrary scale) function for a single output by aggregating all the delta predictions.

4. DATASET & EVALUATION APPROACH

To evaluate the different feature representations, two datasets are used. The first dataset (IMM) consists of $N_{IMM} = 20$ excerpts and is described in [8]. It comprises all $M_{IMM} = 190$ unique pairwise comparisons of 20 different 15-second excerpts, chosen from the USPOP2002 dataset. 13 participants (3 female, 10 male) were compared on both the dimensions of valence and arousal. The second dataset (YANG) [18] consists of $M_{YANG} = 7752$ pairwise comparisons made by multiple annotators on different parts of the $N_{YANG} = 1240$ different Chinese 30-second excerpts on the dimension of valence. 20 MFCC features have been extracted for all excerpts by the MA toolbox.

4.1 Performance Evaluation

In order to evaluate the performance of the proposed representation of the multivariate feature time series we compute learning curves. We use the so-called Leave-One-Excerpt-Out cross validation, which ensures that all comparisons with a given excerpt are left out in each fold, differing from previous work [9]. Each point on the learning curve is the result of models trained on a fraction of all available comparisons in the training set. To obtain robust learning curves, an average of 10-20 repetitions is used. Furthermore a ‘win’-based baseline (Base-flow) as suggested in [8] is used. This baseline represents a model with no information from features. We use the McNemar paired test with the Null hypothesis that two models are the same between each model and the baseline, if $p < 0.05$ then the models can be rejected as equal on a 5% significance level.

5. RESULTS

We consider the pairwise classification error on the two outlined datasets with the kGLM-L2 model, using the outlined pairwise kernel function combined with the PPK kernel (q=1/2). For the YANG dataset a single regularization parameter $\gamma$ was estimated using 20-fold cross validation used.

---

2 In the Gaussian Process setting this kernel is also known as the Pairwise Judgment kernel [4], and can easily be applied for pairwise leaning using other kernel machines such as support vector machines


4 http://www.pampalk.etma/
across all folds in the CV. The quantization of the multivariate time series, is performed using a standard online K-means algorithm [15]. Due to the inherent difficulty of estimating the number of codewords, we choose a selection specifically (8, 16, 24 and 32) for the Markov and HMM models and (256, 512 and 1024) for the VQ models. We compare results between two major categories, namely with continuous or discretized observation space and whether temporal information is included or not.

The results for the IMM dataset for valence are presented in Table 2. For continuous observations we see a clear increase in performance between the Diagonal AR (DAR) model of up to 0.018 and 0.024, compared to traditional Multivariate Gaussian and mean models respectively. With discretized observations, an improvement of performance when including temporal information is again observed of 0.025 comparing the Markov and VQ models. Increasing the complexity of the temporal representation with latent states in the HMM model, an increase of performance is again obtained of 0.016. Predicting the dimension of arousal shown on Table 3, the DAR is again the best performing model using all training data, outperforming the traditional temporal-independent models with 0.015. For discretized data the HMM is the best performing model where we again see that increasing the complexity of the temporal representation increases the predictive performance. Considering the YANG dataset, the results are shown in Table 4. Applying the Vector AR models (VAR), a performance gain is again observed compared to the standard representations like e.g. Gaussian or GMM. For discretized data, the temporal aspects again improve the performance, although we do not see a clear picture that increasing the complexity of the temporal representation increases the performance; the selection of the number of hidden states could be an issue here.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>N(μ,σ)</td>
<td>p = 400</td>
<td>0.468</td>
<td>0.385</td>
<td>0.347</td>
<td>0.297</td>
<td>0.277</td>
<td>0.260</td>
<td>0.252</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N(μ,σ)</td>
<td>0.464</td>
<td>0.394</td>
<td>0.358</td>
<td>0.328</td>
<td>0.297</td>
<td>0.279</td>
<td>0.274</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.460</td>
<td>0.369</td>
<td>0.332</td>
<td>0.299</td>
<td>0.271</td>
<td>0.257</td>
<td>0.251</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.455</td>
<td>0.371</td>
<td>0.339</td>
<td>0.297</td>
<td>0.270</td>
<td>0.254</td>
<td>0.248</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.454</td>
<td>0.372</td>
<td>0.339</td>
<td>0.297</td>
<td>0.269</td>
<td>0.254</td>
<td>0.244</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.450</td>
<td>0.369</td>
<td>0.332</td>
<td>0.299</td>
<td>0.271</td>
<td>0.257</td>
<td>0.251</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.455</td>
<td>0.371</td>
<td>0.339</td>
<td>0.297</td>
<td>0.270</td>
<td>0.254</td>
<td>0.248</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.454</td>
<td>0.372</td>
<td>0.339</td>
<td>0.297</td>
<td>0.269</td>
<td>0.254</td>
<td>0.244</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.450</td>
<td>0.369</td>
<td>0.332</td>
<td>0.299</td>
<td>0.271</td>
<td>0.257</td>
<td>0.251</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.445</td>
<td>0.366</td>
<td>0.332</td>
<td>0.299</td>
<td>0.271</td>
<td>0.257</td>
<td>0.251</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.440</td>
<td>0.366</td>
<td>0.332</td>
<td>0.299</td>
<td>0.271</td>
<td>0.257</td>
<td>0.251</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.435</td>
<td>0.366</td>
<td>0.332</td>
<td>0.299</td>
<td>0.271</td>
<td>0.257</td>
<td>0.251</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.430</td>
<td>0.366</td>
<td>0.332</td>
<td>0.299</td>
<td>0.271</td>
<td>0.257</td>
<td>0.251</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.425</td>
<td>0.366</td>
<td>0.332</td>
<td>0.299</td>
<td>0.271</td>
<td>0.257</td>
<td>0.251</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.420</td>
<td>0.366</td>
<td>0.332</td>
<td>0.299</td>
<td>0.271</td>
<td>0.257</td>
<td>0.251</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.415</td>
<td>0.366</td>
<td>0.332</td>
<td>0.299</td>
<td>0.271</td>
<td>0.257</td>
<td>0.251</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Classification error on the IMM dataset applying the pairwise kGLM-L2 model on the valence dimension. Results are averages of 20 folds, 13 subjects and 20 repetitions. McNemar paired tests between each model and baseline all result in p ≪ 0.001 except for results marked with * which has p > 0.05 with sample size of 4940.

Table 3. Classification error on the IMM dataset applying the pairwise kGLM-L2 model on the arousal dimension. Results are averages of 20 folds, 13 participants and 20 repetitions. McNemar paired tests between each model and baseline all result in p ≪ 0.001 with a sample size of 4940.

Table 4. Classification error on the YANG dataset applying the pairwise kGLM-L2 model on the valence dimension. Results are averages of 1240 folds and 10 repetitions. McNemar paired test between each model and baseline all result in p ≪ 0.001.

6. DISCUSSION

In essence we are looking for a way of representing an entire track based on the simple features extracted. That is, we are trying to find generative models that can capture meaningful information coded in the features specifically for coding aspects related to the emotions expressed in music.

Results showed that simplifying the observation space using VQ is useful when predicting the arousal data. Introducing temporal coding of VQ features by simple Markov models already provides a significant performance gain, and adding latent dimensions (i.e. complexity) a further gain is obtained. This performance gain can be attributed to the temporal changes in features and potentially hidden structures in the features not coded in each frame of the features but, by their longer term temporal structures, captured by the models.

We see the same trend with the continuous observations, i.e. including temporal information significantly increases
predictive performance. These results are specific for the features used, the complexity, and potentially the model choice might differ if other features were utilized. Future work will reveal if other structures can be found in features that describe different aspects of music; structures that are relevant for describing and predicting aspects regarding emotions expressed in music.

Another consideration when using the generative models is that the entire feature time series is replaced as such by the model, since the distances between tracks are now between the models trained on each of the tracks and not directly on the features. These models still have to be estimated, which takes time, but this can be done offline and provide a substantial compression of the features used.

7. CONCLUSION

In this work we presented a general approach for evaluating various track-level representations for music emotion prediction, focusing on the benefit of modeling temporal aspects of music. Specifically, we considered datasets based on robust, pairwise paradigms for which we extended a particular kernel-based model forming a common ground for comparing different track-level representations of music using the probability product kernel. A wide range of generative models for track-level representations was considered on two datasets, focusing on evaluating both using continuous and discretized observations. Modeling both the valence and arousal dimensions of expressed emotion showed a clear gain in applying temporal modeling on both the datasets included in this work. In conclusion, we have found evidence for the hypothesis that a statistically significant gain is obtained in predictive performance by representing the temporal aspect of music for emotion prediction using MFCC’s.

8. REFERENCES


5 We do note that using a single model across an entire musical track could potentially be over simplifying the representation, in our case only small 15-30-second excerpts were used and for entire tracks some segmentation would be appropriate.
MUSICAL STRUCTURAL ANALYSIS DATABASE BASED ON GTTM

Masatoshi Hamanaka  
Kyoto University  
masatosh@kuhp.kyoto-u.ac.jp

Keiji Hirata  
Future University Hakodate  
hirata@fun.ac.jp

Satoshi Tojo  
JAIST  
tojo@jaist.ac.jp

ABSTRACT
This paper, we present the publication of our analysis data and analyzing tool based on the generative theory of tonal music (GTTM). Musical databases such as score databases, instrument sound databases, and musical pieces with standard MIDI files and annotated data are key to advancements in the field of music information technology. We started implementing the GTTM on a computer in 2004 and ever since have collected and publicized test data by musicologists in a step-by-step manner. In our efforts to further advance the research on musical structure analysis, we are now publicizing 300 pieces of analysis data as well as the analyzer. Experiments showed that for 267 of 300 pieces the analysis results obtained by a new musicologist were almost the same as the original results in the GTTM database and that the other 33 pieces had different interpretations.

1. INTRODUCTION
For over ten years we have been constructing a musical analysis tool based on the generative theory of tonal music (GTTM) [1, 2]. The GTTM, proposed by Lerdahl and Jackendoff, is one in which the abstract structure of a musical piece is acquired from a score [3]. Of the many music analysis theories that have been proposed [4–6], we feel that the GTTM is the most promising in terms of its ability to formalize musical knowledge because it captures aspects of musical phenomena based on the Gestalt occurring in music and then presents these aspects with relatively rigid rules.

The time-span tree and prolongational trees acquired by GTTM analysis can be used for melody morphing, which generates an intermediate melody between two melodies with a systematic order [7]. It can also be used for performance rendering [8–10] and reproducing music [11] and provides a summarization of the music that can be used as a search representation in music retrieval systems [12].

In constructing a musical analyzer, test data from musical databases is very useful for evaluating and improving the performance of the analyzer. The Essen folk song collection is a database for folk-music research that contains score data on 20,000 songs along with phrase segmentation information and also provides software for processing the data [13]. The Répertoire International des Sources Musicales (RISM), an international, non-profit organization with the aim of comprehensively documenting extant musical sources around the world, provides an online catalogue containing over 850,000 records, mostly for music manuscripts [14]. The Variations3 project provides online access to streaming audio and scanned score images for the music community with a flexible access control framework [15], and the Real World Computing (RWC) Music Database is a copyright-cleared music database that contains the audio signals and corresponding standard MIDI files for 315 musical pieces [16,17]. The Digital Archive of Finnish Folk Tunes provides 8613 finish folk song midi files with annotated meta data and Matlab data matrix encoded by midi toolbox [18]. The Codaich contains 20,849 MP3 recordings, from 1941 artists, with high-quality annotations [19], and the Latin Music Database contains 3,227 MP3 files from different music genres [20].

When we first started constructing the GTTM analyzer, however, there was not much data that included both a score and the results of analysis by musicologists. This was due to the following reasons:

- There were no computer tools for GTTM analysis.
- Only a few paper-based analyses of GTTM data had been done because a data-saving format for computer analysis had not yet been defined. We therefore defined an XML-based format for analyzing GTTM results and developed a manual editor for the editing.
- Editing the tree was difficult.
- Musicologists using the manual editor to acquire analysis results need to perform a large number of manual operations. This is because the time-span and prolongational trees acquired by GTTM analysis are binary trees, and the number of combinations of tree structures in a score analysis increases exponentially with the number of notes. We therefore developed an automatic analyzer based on the GTTM.
- There was a lack of musicologists.
- Only a few hundred musicologists can analyze scores by using the GTTM. In order to encourage musicologists to co-operate with expanding the GTTM database, we publicized our analysis tool and analysis data based on the GTTM.
- The music analysis was ambiguous.
- A piece of music generally has more than one interpretation, and dealing with such ambiguity is a major problem when constructing a music analysis database. We performed experiments to compare the different analysis results obtained by different musicologists.
We started implementing our GTTM analyzer on a computer in 2004, immediately began collecting test data produced by musicologists, and in 2009 started publicizing the GTTM database and analysis system. We started the GTTM database with 100 pairs of scores and time-span trees comprising and then added the prolongational trees and chord progression data. At present, we have 300 data sets that are being used for researching music structural analysis [1]. The tool we use for analyzing has changed from its original form. We originally constructed a standalone application for the GTTM-based analysis system, but when we started having problems with bugs in the automatic analyzer, we changed the application to a client-server system.

In experiments we compared the analysis results of two different musicologists, one of whom was the one who provided the initial analysis data in the GTTM database. For 267 of 300 pieces of music the two results were the same, but the other 33 pieces had different interpretations. Calculating the coincidence of the time-spans in those 33 pieces revealed that 233 of the 2310 time-spans did not match.

This rest of this paper is organized as follows. In section 2 we describe the database design policy and data sets, in section 3 we explain our GTTM analysis tool, in section 4 we present the experimental results, and in section 5 we conclude with a brief summary.

2. GTTM DATABASE

The GTTM is composed of four modules, each of which assigns a separate structural description to a listener’s understanding of a piece of music. Their output is a grouping structure, a metrical structure, a time-span tree, and a prolongational tree (Fig. 1).

The grouping structure is intended to formalize the intuitive belief that tonal music is organized into groups comprising subgroups. The metrical structure describes the rhythmical hierarchy of the piece by identifying the position of strong beats at the levels of a quarter note, half note, one measure, two measures, four measures, and so on. The time-span tree is a binary tree and is a hierarchical structure describing the relative structural importance of notes that differentiate the essential parts of the melody from the ornamentation. The prolongational tree is a binary tree that expresses the structure of tension and relaxation in a piece of music.

2.1 Design policy of analysis database

As at this stage several rules in the theory allow only monophony, we restrict the target analysis data to monophonic music in the GTTM database.

2.1.1 Ambiguity in music analysis

We have to consider two types of ambiguity in music analysis. One involves human understanding of music and tolerates subjective interpretation, while the latter concerns the representation of music theory and is caused by the incompleteness of a formal theory like the GTTM. We therefore assume because of the former type of ambiguity that there is more than one correct result.

2.1.2 XML-based data structure

We use an XML format for all analysis data. MusicXML [22] was chosen as a primary input format because it provides a common ‘interlingua’ for music notation, analysis, retrieval, and other applications. We designed GroupingXML, MetricalXML, TimespanXML, and ProlongationalXML as the export formats for our analyzer. We also designed HarmonicXML to express the chord progression. The XML format is suitable for expressing the hierarchical grouping structures, metrical structures, time-span trees, and prolongational trees.

2.2 Data sets in GTTM database

The database should contain a variety of different musical pieces, and when constructing it we cut 8-bar-long pieces from whole pieces of music because the time required for analyzing and editing would be too long if whole pieces were analyzed.

2.2.1 Score data

We collected 300 8-bar-long monophonic classical music pieces that include notes, rests, slurs, accents, and articulations entered manually with music notation software called Finale [22]. We exported the MusicXML by using a plugin called Dolet. The 300 whole pieces and the eight bars were selected by a musicologist.

2.2.2 Analysis data

We asked a musicology expert to manually analyze the score data faithfully with regard to the GTTM, using the manual editor in the GTTM analysis tool to assist in editing the grouping structure, metrical structure, time-span tree, and prolongational tree. She also analyzed the chord progression. Three other experts crosschecked these manually produced results.

Figure 1. Grouping structure, metrical structure, time-span tree, and prolongational tree.
3. INTERACTIVE GTTM ANALYZER

Our GTTM analysis tool, called the Interactive GTTM analyzer, consists of automatic analyzers and an editor that can be used to edit the analysis results manually (Fig. 2). The graphic user interface of the tool was constructed in Java, making it usable on multiple platforms. However, some functions of the manual editor work only on MacOSX, which must use the MacOSX API.

3.1 Automatic analyzer for GTTM

We have constructed four types of GTTM analyzers: ATTA, FATTA, σGTTM, and σGTTMII [2, 23–25]. The Interactive GTTM analyzer can use either the ATTA or the σGTTMII, and there is a trade-off relationship between the automation of the analysis process and the variation of the analysis results (Fig. 3).

3.1.1 ATTA: Automatic Time-Span Tree Analyzer

We extended the original theory of GTTM with a full externalization and parameterization and proposed a machine-executable extension of the GTTM called exGTTM [2]. The externalization includes introducing an algorithm to generate a hierarchical structure of the time-span tree in a mixed top-down and bottom-up manner and the parameterization includes introducing a parameter for controlling the priorities of rules to avoid conflict among the rules as well as parameters for controlling the shape of the hierarchical time-span tree. We implemented the exGTTM on a computer called the ATTA, which can output multiple analysis results by configuring the parameters.

3.1.2 FATTA: Full Automatic Time-Span Tree Analyzer

Although the ATTA has adjustable parameters for controlling the weight or priority of each rule, these parameters have to be set manually. This takes a long time because finding the optimal values of the settings themselves takes a long time. The FATTA can automatically estimate the optimal parameters by introducing a feedback loop from higher-level structures to lower-level structures on the basis of the stability of the time-span tree [23]. The FATTA can output only one analysis result without manual configuration. However, our experimental results showed that the performance of the FATTA is not good enough for grouping structure or time-span tree analyses.
3.1.3 σGTTM

We have developed σGTTM, a system that can detect the local grouping boundaries in GTTM analysis, by combining GTTM with statistical learning [24]. The σGTTM system statistically learns the priority of the GTTM rules from 100 sets of score and grouping structure data analyzed by a musicologist and does this by using a decision tree. Its performance, however, is not good enough because it can construct only one decision tree from 100 data sets and cannot output multiple results.

3.1.4 σGTTM II

The σGTTM II system assumes that a piece of music has multiple interpretations and thus it constructs multiple decision trees (each corresponding to an interpretation) by iteratively clustering the training data and training the decision trees. Experimental results showed that the σGTTM II system outperformed both the ATTA and σGTTM systems [25].

3.2 Manual editor for the GTTM

In some cases the GTTM analyzer may produce an acceptable result that reflects the user’s interpretation, but in other cases it may not. A user who wants to change the analysis result according to his or her interpretation can use the GTTM manual editor. This editor has numerous functions that can load and save the analysis results, call the ATTA or σGTTM II analyzer, record the editing history, undo the editing, and autocorrect incorrect structures.

3.3 Implementation on client-server system

Our analyzer is updated frequently, and sometimes it is a little difficult for users to download an updated program. We therefore implement our Interactive GTTM analyzer on a client-server system. The graphic user interface on the client side runs as a Web application written in Java, while the analyzer on the server side runs as a program written in Perl. This enables us to update the analyzer frequently while allowing users to access the most recent version automatically.

4. EXPERIMENTAL RESULTS

GTTM analysis of a piece of music can produce multiple results because the interpretation of a piece of music is not unique. We compared the different analysis results obtained by different musicologists.

4.1 Condition of experiment

A new musicologist who had not been involved in the construction of the GTTM database was asked to manually analyze the 300 scores in the database faithfully with regard to the GTTM. We provided only the 8-bar-long monophonic pieces of music to the musicologist, but she could refer the original score as needed. When analyzing pieces of music, she could not see the analysis results already in GTTM database. She was told to take however much time she needed, and the time needed for analyzing one song ranged from fifteen minutes to six hours.

4.2 Analysis results

Experiments showed that the analysis results for 267 of 300 pieces were the same as the original results in the GTTM database. The remaining 33 pieces had different interpretations, so we added the 33 new analysis results to the GTTM database after they were cross-checked by three other experts.

For those 33 pieces with different interpretations, we found the grouping structure in the database to be the same as the grouping structure obtained by the new musicologist. And for all 33 pieces, in the time-span tree the root branch and branches directly connected to the root branch in the database were the same as the ones in the new musicologist’s results.

We also calculated the coincidence of time-spans in both sets of results for those 33 pieces. A time-span tree is a binary tree and each branch of a time-span tree has a time-span. In the ramification of two branches, there is a primary (salient) time-span and secondary (nonsalient) time-span in a parent time-span (Fig. 4). Two time-spans match when the start and end times of the primary and secondary time-spans are the same. We found that 233 of the 2310 time-spans in those 33 pieces of music did not match.

Figure 4. Parent and primary and secondary time-spans.

4.3 An example of analysis

"Fuga C dur" composed by Johann Pachelbel had the most unmatched time-spans when the analysis results in the GTTM database (Fig. 5a) were compared with the analysis results by the new musicologist (Fig. 5b). From another musicologist we got the following comments about different analysis results for this piece of music.

(a) Analysis result in GTTM database

In the analysis result (a), note 2 was interpreted as the start of the subject of the fuga. Note 3 is more salient than note 2 because note 2 is a non-chord tone. Note 5 is the most salient note in the time-span tree of first bar because notes 4 to 7 are a fifth chord and note 5 is a tonic of the chord. The reason that note 2 was interpreted as
the start of the subject of the fuga is uncertain, but a musicologist who is familiar with music before the Baroque era should be able to see that note 2 is the start of the subject of the fuga.

(b) Analysis result by the musicologist

The analysis result (b) was a more simple interpretation than (a) that note 1 is the start of the subject of the fuga. However, it is curious that the trees of second and third beats of the third bar are separated, because both are the fifth chord.

The musicologist who made this comment said that it is difficult to analyze a monophonic piece of music from the contrapuntal piece of music without seeing other parts. Chord information is necessary for GTTM analysis, and a musicologist who is using only a monophonic piece of music has to imagine other parts. This imagining results in multiple interpretations.

5. CONCLUSION

We described the publication of our Interactive GTTM analyzer and the GTTM database. The analyzer and database can be downloaded from the following website:

http://www.gttm.jp/

The GTTM database has the analysis data for the three hundred monophonic music pieces. Actually, the manual editor in our Interactive GTTM analyzer enables one to deal with polyphonic pieces. Although the analyzer itself works only on monophonic pieces, a user can analyze polyphonic pieces by using the analyzer’s manual editor to divide polyphonic pieces into monophonic parts. We also attempted to extend the GTTM framework to enable the analysis of polyphonic pieces [23]. We plan to publicize a hundred pairs of polyphonic score and musicologists’ analysis results.

Although the 300 pieces in the current GTTM database are only 8 bars long, we also plan to analyze whole pieces of music by using the analyzer’s slide bar for zooming piano roll scores and GTTM structures.

6. REFERENCES


Figure 5. Time-span trees of "Fuga C dur" composed by Johann Pachelbel.


THEORETICAL FRAMEWORK OF A COMPUTATIONAL MODEL OF AUDITORY MEMORY FOR MUSIC EMOTION RECOGNITION

Marcelo Caetano
Sound and Music Computing Group
INESC TEC, Porto, Portugal
mcaetano@inesctec.pt

Frans Wiering
Dep. Information and Computing Sciences
Utrecht University, The Netherlands
f.wiering@uu.nl

ABSTRACT

The bag of frames (BOF) approach commonly used in music emotion recognition (MER) has several limitations. The semantic gap is believed to be responsible for the glass ceiling on the performance of BOF MER systems. However, there are hardly any alternative proposals to address it. In this article, we introduce the theoretical framework of a computational model of auditory memory that incorporates temporal information into MER systems. We advocate that the organization of auditory memory places time at the core of the link between musical meaning and emotional responses. The main goal is to motivate MER researchers to develop an improved class of systems capable of overcoming the limitations of the BOF approach and coping with the inherent complexity of musical emotions.

1. INTRODUCTION

In the literature, the aim of music emotion recognition (MER) is commonly said to be the development of systems to automatically estimate listeners’ emotional response to music [2, 7, 8, 11, 18, 19, 33] or simply to organize or classify music in terms of emotional content [14, 17]. Applications of MER range from managing music libraries and music recommendation systems to movies, musicals, advertising, games, and even music therapy, music education, and music composition [11]. Possibly inspired by automatic music genre classification [28, 29], a typical approach to MER categorizes emotions into a number of classes and applies machine learning techniques to train a classifier and compare the results against human annotations, considered the “ground truth” [14, 19, 28, 32]. Kim et al. [14] presented a thorough state-of-the-art review, exploring a wide range of research in MER systems, focusing particularly on methods that use textual information (e.g., websites, tags, and lyrics) and content-based approaches, as well as systems combining multiple feature domains (e.g., features plus text). Commonly, music features are estimated from the audio and used to represent the music. These features are calculated independently from each other and from their temporal progression, resulting in the bag of frames (BOF) [11, 14] paradigm.

The ‘Audio Mood Classification’ (AMC) task in MIREX [5, 10] epitomizes the BOF approach to MER, presenting systems whose performance range from 25% to 70% (see Table 1). Present efforts in MER typically concentrate on the machine learning algorithm that performs the map in an attempt to break the ‘glass ceiling’ [1] thought to limit system performance. The perceived musical information that does not seem to be contained in the audio even though listeners agree about its existence, called ‘semantic gap’ [3, 31], is considered to be the cause of the ‘glass ceiling.’ However, the current approach to MER has been the subject of criticism [2, 11, 28, 31].

Knowledge about music cognition, music psychology, and musicology is seldom explored in MER. It is widely known that musical experience involves more than mere processing of music features. Music happens essentially in the brain [31], so we need to take the cognitive mechanisms involved in processing musical information into account if we want to be able to model people’s emotional response to music. Among the cognitive processes involved in listening to music, memory plays a major role [27]. Music is intrinsically temporal, and time is experienced through memory. Studies [12, 16, 25] suggest that the temporal evolution of the musical features is intrinsically linked to listeners’ emotional response to music.

In this article, we speculate that the so called ‘semantic gap’ [3] is a mere reflection of how the BOF approach misrepresents both the listener and musical experience. Our goal is not to review MER, but rather emphasize the limitations of the BOF approach and propose an alternative model that relies on the organization of auditory memory to exploit temporal information from the succession of musical sounds. For example, BOF MER systems typically encode temporal information in delta and delta-delta coefficients [1], capturing only local instantaneous temporal variations of the feature values. In a previous work [2], we discussed different MER systems that exploit temporal information differently. Here, we take a step further and propose the theoretical framework of a computational model of auditory memory for MER. Our aim is to motivate MER research to bridge the ‘semantic gap’ and break the so called ‘glass ceiling’ [1, 3, 31].
The next section discusses the complexity of musical emotions and how this relates to the glass ceiling preventing BOF MER systems to improve their performance as a motivation for proposing a paradigm change. Then, we briefly introduce the model of auditory memory adopted, followed by the proposed framework and considerations about its implementation. Finally, we present the conclusions and discuss future directions of this theoretical work.

2. MACHINE LEARNING AND MER

It is generally agreed that music conveys and evokes emotions [9, 13]. In other words, listeners might feel happy listening to a piece or simply perceive it as happy [9]. Research on music and emotions usually investigates the musical factors involved in the process as well as listeners’ response to music. There are many unanswered questions [13, 21], such as “which emotions does music express?”, “in what context do musical emotions occur?”, “how does music express emotions?”, and “which factors in music are expressive of emotions?” Researchers need to address controversial issues to investigate these questions. On the one hand, the relevant musical factors, and on the other hand, the definition and measurement of emotion.

There is evidence [13] of emotional reactions to music in terms of various subcomponents, such as subjective feeling, psychophysiology, brain activation, emotional expression, action tendency, emotion regulation and these, in turn, feature different psychological mechanisms like brain stem reflexes, evaluative conditioning, emotional contagion, visual imagery, episodic memory, rhythmic entrainment, and musical expectancy. Each mechanism is responsive to its own combination of information in the music, the listener, and the situation. Among the causal factors that potentially affect listeners’ emotional response to music are personal, situational, and musical [21]. Personal factors include age, gender, personality, musical training, music preference, and current mood; situational factors can be physical such as acoustic and visual conditions, time and place, or social such as type of audience, and occasion. Musical factors include genre, style, key, tuning, orchestration, among many others.

Most modern emotion theorists suggest that an emotion episode consists of coordinated changes in three major reaction components: physiological arousal, motor expression, and subjective feeling (the emotion triad). According to this componential approach to emotion, we would need to measure physiological changes, facial and vocal expression, as well as gestures and posture along with self-reported feelings using a rich emotion vocabulary to estimate the listeners’ emotional response. In MER, the emotional response to music is commonly collected as self-reported annotations for each music track, capturing “subjective feelings” associated or experienced by the listener. Some researchers [9] speculate that musical sounds can effectively cause emotional reactions (via brain stem reflex, for example), suggesting that certain music dimensions and qualities communicate similar affective experiences to many listeners. The literature on the emotional effects of music [9, 13] has accumulated evidence that listeners often agree about the emotions expressed (or elicited) by a particular piece, suggesting that there are aspects in music that can be associated with similar emotional responses across cultures, personal bias or preferences.

It is probably impractical to hope to develop a MER system that could account for all facets of this complex problem. There is no universally accepted model or explanation for the relationship between music and emotions. However, we point out that it is widely known and accepted that MER systems oversimplify the problem when adopting the BOF approach [11]. In this context, we propose a theoretical framework that uses the organization of auditory memory to incorporate temporal information into MER. We argue that time lies at the core of the complex relationship between music and emotions and that auditory memory mediates the processes involved. In what follows, we focus on the link between musical sounds and self-reported subjective feelings associated to them through music listening. In other words, the association between the audio features and perceived emotions.

2.1 The Glass Ceiling on System Performance

The performance of music information retrieval (MIR) systems hasn’t improved satisfactorily over the years [1, 10] due to several shortcomings. Aucouturier and Pachet [1] used the term ‘glass ceiling’ to suggest that there is a limitation on system performance at about 65% R-precision when using BOF and machine learning in music similarity. Similarly, Huq et al. [11] examined the limitations of the BOF approach to MER. They present the results of a systematic study trying to maximize the prediction performance of an automated MER system using machine learning. They report that none of the variations they considered leads to a substantial improvement in performance, which they interpret as a limit on what is achievable with machine learning and BOF.

MIREX [10] started in 2005 with the goal of systematically evaluating state-of-the-art MIR algorithms, promoting the development of the field, and increasing system performance by competition and (possibly) cooperation. MIREX included an “Audio Mood Classification” (AMC) task for the first time in 2007 ‘inspired by the growing interest in classifying music by moods, and the difficulty in the evaluation of music mood classification caused by the subjective nature of mood’ [10]. MIREX’s AMC task uses a categorical representation of emotions divided in five classes. These five ‘mood clusters’ were obtained by analyzing ‘mood labels’ (user tags) for popular music from the All Music Guide.

The MIREX wiki presents the “Raw Classification Accuracy Averaged Over Three Train/Test Folds” per system. Table 1 summarizes system performance over the years for the MIREX task AMC, showing the minimum, maximum, average, and standard deviation of these values across systems. Minimum performance has steadily

---


Table 1: MIREX AMC performance from 2007 to 2013.

<table>
<thead>
<tr>
<th></th>
<th>Minimum</th>
<th>Maximum</th>
<th>Average</th>
<th>STD</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007</td>
<td>25.67%</td>
<td>61.50%</td>
<td>52.65%</td>
<td>11.19%</td>
</tr>
<tr>
<td>2008</td>
<td>30.33%</td>
<td>63.67%</td>
<td>52.39%</td>
<td>7.72%</td>
</tr>
<tr>
<td>2009</td>
<td>34.67%</td>
<td>65.67%</td>
<td>57.67%</td>
<td>6.17%</td>
</tr>
<tr>
<td>2010</td>
<td>36.12%</td>
<td>63.78%</td>
<td>56.48%</td>
<td>6.36%</td>
</tr>
<tr>
<td>2011</td>
<td>39.81%</td>
<td>69.48%</td>
<td>57.98%</td>
<td>9.33%</td>
</tr>
<tr>
<td>2012</td>
<td>46.14%</td>
<td>67.80%</td>
<td>62.67%</td>
<td>6.17%</td>
</tr>
<tr>
<td>2013</td>
<td>28.83%</td>
<td>67.83%</td>
<td>59.81%</td>
<td>10.29%</td>
</tr>
</tbody>
</table>

improved, but maximum performance presents a less significant improvement. The standard deviation of performance across systems has a general trend towards decreasing (suggesting more homogeneity over the years). Most algorithms are also tested in different classification tasks (musical genre, for example), and the best in one task are often also very good at other tasks, maybe indicating there is more machine learning than musical knowledge involved. Sturm [28] discusses the validity of the current evaluation in MER. He argues that the current paradigm of classifying music according to emotions only allows us to conclude how well an MER system can reproduce “ground truth” labels of the test data, irrespective of whether these MER systems use factors irrelevant to emotion in music.

2.2 Bridging the Semantic Gap

In MIR, audio processing manipulates signals generated by musical performance, whereas music is an abstract and intangible cultural construct. The sounds per se do not contain the essence of music because music exists in the mind of the listener. The very notion of a ‘semantic gap’ is misleading [31]. The current BOF approach to MER views music simply as data (audio signals) and therefore misrepresents musical experience. Machine learning performs a rigid map from “music features” to “emotional labels”, as illustrated in part a) of Fig. 1, treating music as a stimulus that causes a specific emotional response irrespective of personal and contextual factors which are known to affect listeners’ emotional response [12, 16, 25] such as listeners’ previous exposure and the impact of the unfolding musical process. Memory is particularly important in the recognition of patterns that are either stored in long-term memory (LTM) from previous pieces or in short-term memory (STM) from the present piece. Music seems to be one of the most powerful cues to bring emotional experiences from memory back into awareness.

Wiggins [31] suggests to look at the literature from musicology and psychology to study the cognitive mechanisms involved in human music perception as the starting point of MIR research, particularly musical memory, for they define music. He argues that music is not just processed by the listeners, it is defined by them. Wiggins states that “music is a cognitive model”, therefore, only cognitive models are likely to succeed in processing music in a human-like way. He writes that “to treat music in a way which is not human-like is meaningless, because music is defined by humans. Finally, he concludes that the human response to memory is key to understanding the psychophysiological effect of musical stimuli, and that this domain is often missing altogether from MIR research. In this work, we view perceived musical emotions as a particular form of musical meaning [12, 16, 25], which is intimately related to musical structure by the organization of auditory memory [27], as represented in part b) of Fig. 1.

3. AUDITORY MEMORY AND MER

Conceptually, memory can be divided into three processes [27]: sensory memory (echoic memory and early processing); short-term memory (or working memory); and long-term memory. Each of these memory processes functions on a different time scale, which can be loosely related to levels of musical experience, the “level of event fusion”, the “melodic and rhythmic level”, and the “formal level”, respectively. Echoic memory corresponds to early processing, when the inner ear converts sounds into trains of nerve impulses that represent the frequency and amplitude of individual acoustic vibrations. During feature extraction, individual acoustic features (e.g., pitch, overtone structure) are extracted and then bound together into auditory events. The events then trigger those parts of long-term memory (LTM) activated by similar events in the past, establishing a context that takes the form of expectations, or memory of the recent past. Long-term memories that are a part of this ongoing context can persist as current “short-term memory” (STM). Short-term memories disappear from consciousness unless they are brought back into the focus of awareness repeatedly (e.g. by means of the rehearsal loop). When the information is particularly striking or novel, it may be passed back to LTM and cause modifications of similar memories already established, otherwise it is lost.

The three types of processing define three basic time scales on which musical events and patterns take place, which, in turn, affect our emotional response to music. The event fusion level of experience (echoic memory) is associated with pitch perception. The main characteristic of the melodic and rhythmic level is that separate events on this time scale are grouped together in the present as melodic grouping and rhythmic grouping, associated with STM. Units on the formal level of musical experience consist of entire sections of music and are associated with
4. THE PROPOSED FRAMEWORK

Fig. 2 shows the framework we propose to incorporate memory processes in MER systems to illustrate how auditory memory affects musical experience. The blocks associated with the system have a white background, while memory processes have a dark background. The arrows represent the flow of information, while the dashed line represents the relationship between memory processes and system blocks. The proposed framework can be interpreted as an extension of the traditional approach (shaded area) by including two blocks, previous exposure and unfolding musical process. In the BOF approach, the music features are associated with echoic memory, related to very short temporal scales and uncorrelated with the past or predictions of future events. The framework we propose includes the “Unfolding Musical Process” and “Previous Exposure” to account for LTM and STM. The “Unfolding Musical Process” represents the listeners’ perception of time (related to musical context and expectations), while “Previous Exposure” represents the personal and cultural factors that makes listeners unique.

4.1 Unfolding Musical Process

The unfolding music process uses temporal information from the current music stream to account for repetitions and expectations. As Fig. 2 suggests, the unfolding musical process acts as feedback loop that affects the map between the music features and the listener response. The dynamic aspect of musical emotion relates to the cognition of musical structure [12, 16, 25]. Musical emotions change over time in intensity and quality, and these emotional changes covary with changes in psycho-physiological measures [16, 25]. The human cognitive system regulates our expectations to make predictions [12]. Music (among other stimuli) influences this principle, modulating our emotions. As the music unfolds, the model is used to generate expectations, which are implicated in the experience of listening to music. Musical meaning and emotion depend on how the actual events in the music play against this background of expectations.

4.2 Previous Exposure

The framework in Fig. 2 illustrates that previous exposure accounts for musical events stored in LTM that affect the listeners’ emotional response to music. Musical emotions may change according to musical genre [6], cultural background, musical training and exposure, mood, physiological state, personal disposition and taste [9, 12]. This information is user specific and depends on context thus it cannot be retrieved from the current music stream, rather, it has to be supplied by the listener.

5. IMPLEMENTATION ISSUES

Here we address how to treat individual components of the model, which parts need human input and which are automatic, and how the different system components communicate and what information they share. The proposed framework urges for a paradigm change in MER research rather than simply a different kind of MER systems, including representing the music stream, collecting time-stamped annotations, and system validation and evaluation [28]. Thus we propose a class of dynamic MER systems that continuously estimate how the listener’s perceived emotions unfold in time from a time-varying input stream of audio features calculated from different musically related temporal levels.

5.1 Music Stream as System Input

The proposed system input is a music stream unfolding in time rather than a static (BOF) representation. To incorporate time into MER, the system should monitor the temporal evolution of the music features [25] at different time scales, the “level of event fusion”, the “melodic and rhythmic level”, and the “formal level”. The feature vector should be calculated for every frame of the audio signal and kept as a time series (i.e., a time-varying vector of features). Time-series analysis techniques such as linear prediction and correlations (among many others) might be used to extract trends and model information at later stages.
5.2 Music Features

Eerola [6, 7] proposes to select musically relevant features that have been shown to relate to musical emotions. He presents a list of candidate features for a computational model of emotions that can be automatically estimated from the audio and that would allow meaningful annotations of the music, dividing the features into musically relevant levels related to three temporal scales. Snyder [27] describes three different temporal scales for musical events based on the limits of human perception and auditory memory. Coutinho et al. [4] sustain that the structure of affect elicited by music is largely dependent on dynamic temporal patterns in low-level music structural parameters. In their experiments, a significant part of the listeners’ reported emotions can be predicted from a set of six psychoacoustic features, namely, loudness, pitch level, pitch contour, tempo, texture, and sharpness. Schubert [26] used loudness, tempo, melodic contour, texture, and spectral centroid as predictors in linear regression models of valence and arousal.

Fig. 1 suggests that MER systems should use the musical structure to estimate musical meaning such as emotions. Musical structure emerges from temporal patterns of music features. In other words, MER systems should include information about the rate of temporal change of music features, such as how changes in loudness correlate with the expression of emotions rather than loudness values only. These loudness variations, in turn, form patterns of repetition on a larger temporal scale related to the structure of the piece that should also be exploited. Thus the features should be hierarchically organized in a musically meaningful way according to auditory memory [27].

5.3 Listener Response and System Output

Recently, some authors started investigating how the emotional response evolves in time as the music unfolds. Krumhansl [16] proposes to collect listener’s responses continuously while the music is played, recognizing that retrospective judgements are not sensitive to unfolding processes. Recording listener’s emotional ratings over time as time-stamped annotations requires listeners to write down the emotional label and a time stamp as the music unfolds, a task that has received attention [20]. Emotions are dynamic and have distinctive temporal profiles that are not captured by traditional models (boredom is very different from astonishment in this respect, for example). In this case, the temporal profiles would be matched against prototypes stored in memory. Some musical websites allow listeners to ‘tag’ specific points of the waveform (for instance, SoundCloud 3 ), a valuable source of temporal annotations for popular music.

5.4 Unfolding Musical Process

The unfolding musical process acts as feedback loop that exploits the temporal evolution of music features at the three different time scales. The temporal correlation of each feature must be exploited and fed back to the mapping mechanism (see ‘unfolding musical process’) to estimate listeners’ response to the repetitions and the degree of “surprise” that certain elements might have [26]. Schubert [25] studied the relationship between music features and perceived emotion using continuous response methodology and time-series analysis. Recently, MER systems started tracking temporal changes [4, 22–24, 30]. However, modeling the unfolding musical process describes how the time-varying emotional trajectory varies as a function of music features. Korhonen et al. [15] use auto-regressive models to predict current musical emotions from present and past feature values, including information about the rate of change or dynamics of the features.

5.5 Previous Exposure

Previous exposure is responsible for system customization and could use reinforcement learning to alter system response to the unfolding musical process. Here, the user input tunes the long-term system behavior according to external factors (independent from temporal evolution of features) such as context, mood, genre, cultural background, etc. Eerola [6] investigated the influence of musical genre on emotional expression and reported that there is a set of music features that seem to be independent of musical genre. Yang et al. [33] studied the role of individuality in MER by evaluating the prediction accuracy of group-wise and personalized MER systems by simply using annotations from a single user as “ground truth” to train the MER system.

7. CONCLUSIONS

Research on music emotion recognition (MER) commonly relies on the bag of frames (BOF) approach, which uses machine learning to train a system to map music features to a region of the emotion space. In this article, we discussed why the BOF approach misrepresents musical experience, underplays the role of memory in listeners’ emotional response to music, and neglects the temporal nature of music. The organization of auditory memory plays a major role in the experience of listening to music. We proposed a framework that uses the organization of auditory memory to bring time to the foreground of MER. We prompted MER researchers to represent music as a time-varying vector of features and to investigate how the emotions evolve in time as the music develops, representing the listener’s emotional response as an emotional trajectory. Finally, we discussed how to exploit the unfolding music process and previous exposure to incorporate the current musical context and personal factors into MER systems.

The incorporation of time might not be enough to account for the subjective nature of musical emotions. Culture, individual differences and the present state of the listener are factors in understanding aesthetic responses to music. Thus a probabilistic or fuzzy approach could also represent a significant step forward in understanding aesthetic responses to music. We prompt MER researchers to

---

3 http://soundcloud.com/
adopt a paradigm change to cope with the complexity of human emotions in one of its canonical means of expression, music.

7. ACKNOWLEDGEMENTS

This work was partially supported by the Media Arts and Technologies project (MAT), NORTE-07-0124-FEDER-000061, which is financed by the North Portugal Regional Operational Programme (ON.2 O Novo Norte), under the National Strategic Reference Framework (NSRF), through the European Regional Development Fund (ERDF), and by national funds, through the Portuguese funding agency, Fundao para a Ciˆencia e a Tecnologia (FCT). Frans Wiering is supported by the FES project COMMIT/.

8. REFERENCES


Methods for music structure discovery usually process a music track by first detecting segments and then labeling them. Depending on the assumptions made on the signal content (repetition, homogeneity or novelty), different methods are used for these two steps. In this paper, we deal with the segmentation in the case of repetitive content. In this field, segments are usually identified by looking for sub-diagonals in a Self-Similarity-Matrix (SSM). In order to make this identification more robust, Goto proposed in 2003 to cumulate the values of the SSM over constant-lag and search only for segments in the SSM when this sum is large. Since the various repetitions of a segment start simultaneously in a self-similarity-matrix, Serra et al. proposed in 2012 to cumulate these simultaneous values (using a so-called structure feature) to enhance the novelty of the starting and ending time of a segment. In this work, we propose to combine both approaches by using Goto method locally as a prior to the lag-dimensions of Serra et al. structure features used to compute the novelty curve. Through a large experiment on RWC and Isophonics test-sets and using MIREX segmentation evaluation measure, we show that this simple combination allows a large improvement of the segmentation results.

1. INTRODUCTION

Music structure segmentation aims at estimating the large-scale temporal entities that compose a music track (for example the verse, chorus or bridge in popular music). This segmentation has many applications such as browsing a track by parts, a first step for music structure labeling or audio summary generation [15], music analysis, help for advanced DJ-ing.

The method used to estimate the music structure segments (and/or labels) depends on the assumptions made on the signal content. Two assumptions are commonly used [13] [14]. The first assumption considers that the audio signal can be represented as a succession of segments with homogeneous content inside each segment. This assumption is named “homogeneity assumption” and the estimation approach named “state approach”. It is closely related to another assumption, named “novelty”, that considers that the transition between two distinct homogeneous segments creates a large “novelty”. The second assumption considers that some segments in the audio signal are repetitions of other ones. This assumption is named “repetition assumption”. In this case the “repeated” segments can be homogeneous or not. When they are not, the approach is named “sequence approach”.

In this paper, we deal with the problem of estimating the segments (starting and ending times) in the case of repeated/ non-homogeneous segments (“sequence” approach).

1.1 Related works

Works related to music structure segmentation are numerous. We refer the reader to [13] or [3] for a complete overview on the topic. We only review here the most important works or the ones closely related our proposal.

Methods relying on the homogeneity or novelty assumption. Because homogeneous segments form “blocks” in a time-time-Self-Similarity-Matrix (SSM) and because transitions from one homogeneous segment to the next looks like a checkerboard kernel, Foote [5] proposes in 2000 to convolve the matrix with a 2D-checkerboard-kernel. The result of the convolution along the main diagonal leads to large value at the transition times. Since, an assumption on the segment duration has to be made for the kernel of Foote, Kaiser and Peeters [9] propose in 2013 to use multiple-temporal-scale kernels. They also introduce two new kernels to represent transitions from homogeneous to non-homogeneous segments (and vice versa). Other approaches rely on information criteria (such as BIC, Akaike or GLR) applied to the sequence of audio features. Finally, labeling methods (such as k-means, hierarchical agglomerative clustering of hidden-Markov-model) also inherently allow performing time-segmentation.

Methods relying on the repetition assumption. Because repeated segments (when non-homogeneous) form sub-diagonals in a Self-Similarity Matrix (SSM), most methods perform the segmentation by detecting these sub-diagonals in the SSM.

If we denote by $S(i,j) = S(t_i,t_j)$ $i,j \in [1,N]$ the time-time-SSM between the pairs of times $t_i$ and $t_j$, the time-lag-SSM [1] is defined as $L(i,l) = L(t_i,t_l = t_j - t_i)$, since $t_j - t_i \geq 0$ the matrix is upper-diagonal. The lag-matrix can be computed using $L(i,l) = S(i,j = i + l)$ with $i + l \leq N$. 

In [6], Goto proposes to detect the repetitions in a time-lag-SSM using a two-step approach. He first detects the various lags \( l_k \) at which potential repetitions may occur. This is done by observing that when a repetition (at the same-speed) occurs, a vertical line (at constant lag) exists in the time-lag-SSM (see Figure 1). Therefore, the sum over the times of the time-lag-SSM for this specific lag will be large. He proposes to compute the function 
\[
 f(l) = \sum_{t\in[0,N-l]} \frac{1}{N} L(t, l)
\]
A peak in \( f(l) \) indicates that repetitions exist at this specific lag. Then, for each detected peaks \( l_k \), the corresponding column of \( L(t, l_k) \) is analyzed in order to find the starting and ending times of the segments.

Serra et al. method [16] for music structure segmentation also relies in the time-lag-SSM but works in the opposite way. In order to compute the lower-diagonal part of the matrix \( (t_j - t_i < 0) \), They propose to apply circular permutation. The resulting matrix is named circular-time-lag-matrix (CTLM) and is computed using \( L^*(i, l) = S(i, k + l) \), for \( i, l \in [1, N] \) and \( k = i + l - 2 \mod N \). They then use the fact that the various repetitions of a same segment start and end at the same times in the CTLM. They use the CENS (Chroma Energy distribution Normalized Statistics) features [12] extracted using the Chroma Toolbox [11]. The CENS feature is a sort of quantized version of the chroma feature smoothed over time by convolution with a long duration Hann window. The CENS features \( x_c(t_i) \) for \( i \in [1, N] \) are 12-dimensional vector with a sampling rate of 2 Hz. \( x_c(t_i) \) is in the range [0, 1]. It should be noted that these features are \( l^2 \)-normed.

\[ \text{Figure 1. Illustration of Goto method [6] on a time-lag Self-Similarity-Matrix (SSM) and Serra et al. method [16] on a circular-time-lag-matrix (CTLM).} \]

1.2 Paper objective and organization

In this paper, we deal with the problem of estimating the segments (starting and ending times) in the case of repeated/ non-homogeneous segments (“sequence” approach). We propose a simple, but very efficient, method that allows using Goto method as a prior lag-probability of segments in Serra et al. method. Indeed, Serra et al. method works efficiently when the “structure” feature \( g(i) \) is clean, i.e. contains large values when a segment crosses \( g(i) \) and is null otherwise. Since, this is rarely the case, we propose to create a prior assumption \( f(l) \) on the dimensions of \( g(i) \) that may contain segments. To create this prior assumption, we use a modified version of Goto method applied locally in time to the CTLM (instead of to the time-lag-SSM).

Our proposed method for music structure segmentation is presented in part 2. We then evaluate it and compare its performance to state-of-the-art algorithms in part 3 using the RWC-Popular-Music and Isophonics/Beatles test-sets. Discussions of the results and potential extensions are discussed in part 4.

2. PROPOSED METHOD

2.1 Feature extraction

In order to represent the content of an audio signal, we use the CENS (Chroma Energy distribution Normalized Statistics) features [12] extracted using the Chroma Toolbox [11]. The CENS feature is a sort of quantized version of the chroma feature smoothed over time by convolution with a long duration Hann window. The CENS features \( x_c(t_i) \) for \( i \in [1, N] \) are 12-dimensional vector with a sampling rate of 2 Hz. \( x_c(t_i) \) is in the range [0, 1]. It should be noted that these features are \( l^2 \)-normed.

2.2 Self-Similarity-Matrix

From the sequence of CENS features we compute a time-time Self-Similarity-Matrix (SSM) [4] \( S(i, j) \) using as similarity measure the scalar-product between the feature vector at time \( t_i \) and \( t_j \): \( S(i, j) = < x_c(t_i), x_c(t_j) > \). In order to highlight the diagonal-repetitions in the SSM while reducing the influence of noise values, we then apply the following process.

1. We apply a low-pass filter in the direction of the diagonals and high-pass filter in the orthogonal direction. For this, we use the kernel \([-0.3, 1, -0.3]\) replicated 12 times to lead to a low-pass filter of duration 6 s.

2. We apply a threshold \( \tau \in [0, 1] \) to the resulting SSM. \( \tau \) is chosen such as to keep only \( \beta \% \) of the values of the SSM. Values below \( \tau \) are set to a negative penalty-value \( \alpha \). The interval \([\tau, 1]\) is then mapped to the interval \([0, 1]\).

3. Finally, we apply a median filter over the diagonals of the matrix. For each value \( S(i, j) \), we look in the backward and forward diagonals of \( \delta \)-points duration each \([i - \delta, j - \delta], \ldots, (i, j) \ldots (i + \delta, j + \delta)] \). If more than \( 50\% \) of these points have a value of \( \alpha \), \( S(i, j) \) is also set to \( \alpha \).

By experiment, we found \( \beta = 6\% \) (percentage of values kept), \( \alpha = -2 \) (lower values) and \( \delta = 10 \) frames (interval duration) to be good values.

1 \[ \sum_{n=1,12}\frac{x^2_n(t_i)}{2} = 1 \]
2 Since the vectors are \( l^2 \)-normed, this is equivalent to the use of a cosine-distance.
3 Since the sampling rate of \( x_c(t_i) \) is 2 Hz, this corresponds to a duration of 5 s. The median filter is then applied on a window of 10 s total duration.
2.3 Proposed method: introducing lag-prior

As mentioned before, Serra et al. method works efficiently when the “structure” feature \( g(i) \) is clean, i.e. contains large values when a segment crosses \( g(i) \) and is null otherwise. Unfortunately, this is rarely the case in practice.

If we model the structure feature \( g(i) \) as the true contribution of the segments \( \hat{g}(i) \) and a background noise (modeled as a centered Gaussian noise) \( N_{\mu=0,\sigma} : g(i) = \hat{g}(i) + N_{\mu=0,\sigma} \), one can easily shows that the expectation of \( c(i) = ||g(i + 1) - g(i)||^2 \) is equal to

- \( K + 2\sigma^2 \) for the starting/ending of \( K \) segments at \( t_i \)
- \( 2\sigma^2 \) otherwise.

If \( \sigma \) (the amount of background noise in the CTLM) is large, then it may be difficult to discriminate between both case for small \( K \). In the opposite, the expectation of the values of Goto function \( f(t) = \sum_{i} L^*(t_i, l) \) remains independent of \( \sigma \) hence on the presence of background noise (in the case of a centered Gaussian noise).

We therefore propose to use \( f(t) \) as a prior on the lags, i.e. the dimensions of \( g(i) \). This will favor the discrimination provided by \( c(i) \) (in Serra et al. approach, all the lags/dimensions of \( g(i) \) are considered equally).

For this, we consider, the circular time-lag (CMLT) \( L^*(t, l) \) as a joint probability distribution \( p(t, l) \).

Serra et al. novelty curve \( c(i) \) can be expressed as

\[
c_1(t) = \int_l \left| \frac{\partial}{\partial t} p(t, l) \right|^2 dl
\]

In our approach, we favor the lags at which segments are more likely. This is done using a prior \( p(l) \):

\[
c_2(t) = \int_l p(l) \left| \frac{\partial}{\partial t} p(t, l) \right|^2 dl
\]

In order to compute the prior \( p(l) \) we compute \( f(t) \) as proposed by Goto but applied to the CMLT. In other words, we compute, the marginal of \( p(t, l) \) over \( t \):

\[
p(t) = \int_{t=0}^{t=N} p(t, l) dt.
\]

As a variation of this method, we also propose to compute the prior \( p(l) \) locally on \( t \): \( p_t(t) = \int_{t-\Delta}^{t+\Delta} p(t, l) dt \). This leads to the novelty curve

\[
c_3(t) = \int_l p_t(l) \left| \frac{\partial}{\partial t} p(t, l) \right|^2 dl
\]

By experiment, we found \( \Delta = 20 \) (corresponding to 10 s), to be a good value.

2.4 Illustrations

In Figure 2, we illustrate the computation of \( c_1(t) \), \( c_2(t) \) and \( c_3(t) \) on a real signal (the track 19 from RWC Popular Music).

In Figure 2 (A) we represent Serra et al. [16] method. On the right of the time-lag-circular-matrix (CTLM), we represent the novelty curve \( c_1(t) \) (red-curve) and super-imposed to it, the ground-truth segments (black dashed lines).

In Figure 2 (B) we represent the computation of \( c_2(t) \) (using a global lag-prior). Below the CTLM we represent the global prior \( p(l) \) (blue curve) obtained using Goto method applied to the CMLT. On the right of the CTLM

![Diagram](image)
we represent \( c_2(t) \) using this global lag-prior. Compared to the above \( c_1(t) \), we see that \( c_2(t) \) allows a larger discrimination between times that correspond to ground-truth starts and ends of segments and that do not.

In Figure 2 (C) we represent the computation of \( c_3(t) \) (using a local lag-prior). Below the CTLM we represent the local prior \( p_l(t) \) in matrix form obtained using Goto method applied locally in time to the CMLT. On the right of the CTLM we represent \( c_3(t) \) using this local lag-prior. Compared to the above \( c_1(t) \) and \( c_2(t) \), we see that \( c_3(t) \) allows an even larger discrimination.

2.5 Estimation of segments start and end times

Finally, we estimate the starting and ending time of the repetitions from the novelty curves \( c_1(t) \), \( c_2(t) \) or \( c_3(t) \). This is done using a peak picking process. \( c_3(t) \) is first normalized by min-max to the interval \([0,1]\). Only the values above \( 0.1 \) are considered. \( t_i \) is considered as a peak if \( i = \arg \max_j c_3(t_j) \) with \( j \in [i-10, i+10] \), i.e. if \( t_i \) is the maximum peak within a \( \pm 5 \) s duration interval.

The flowchart of our Music Structure Segmentation method is represented in the left part of Figure 3.

![Flowchart of the proposed Music Structure Segmentation method](image)

Figure 3. Flowchart of the proposed Music Structure Segmentation method.

3. EVALUATION

In this part, we evaluate the performances of our proposed method for estimating the start and end times of music structure segments. We evaluate our algorithm using the three methods described in part 2.3: – without lag-prior \( c_1(t) \) (this is equivalent to the original Serra et al. algorithm although our features and the pre-processing of the CTLM differ from the ones of Serra et al.), – with global lag-prior \( c_2(t) \), – with local lag-prior \( c_3(t) \).

3.1 Test-Sets

In order to allow comparison with previously published results, we evaluate our algorithm on the following test-sets:

- **RWC-Pop-A**: is the RWC-Popular-Music test-set [8], which is a collection of 100 music tracks. The annotations into structures are provided by the AIST [7].
- **RWC-Pop-B** is the same test-set but with annotations provided by IRISA [2].
- **Beatles-B** is the Beatles test-set as part of the Isophonics test-set, which is a collection of 180 music tracks from the Beatles. The annotations into structure are provided by Isophonics [10].

3.2 Evaluation measures

To evaluate the quality of our segmentation we use, as it is the case in the MIREX (Music Information Retrieval Evaluation xChange) Structure Segmentation evaluation task, the Recall (R), Precision (P) and F-Measure (F). We compute those with a tolerance window of 3 and 0.5 s.

3.3 Results obtained applying our lag-prior method to the SSM as computed in part 2.2.

In Table 1 we indicate the results obtained for the various configurations and test-sets. We compare our results with the ones published in Serra et al. [16] and to the best score obtained during the two last MIREX evaluation campaign: MIREX-2012 and MIREX-2013 on the same test-sets 5 6.

For the three test-sets, and a 3 s tolerance window, the use of our lag-prior allows a large increase of the F-measure:

- RWC-Pop-A: \( c_1(t) : 66.0\% \), \( c_2(t) : 72.9\% \), \( c_3(t) : 76.9\% \).
- RWC-Pop-B: \( c_1(t) : 67.3\% \), \( c_2(t) : 72.0\% \), \( c_3(t) : 78.2\% \).
- Beatles-B: \( c_1(t) : 65.7\% \), \( c_2(t) : 69.8\% \), \( c_3(t) : 76.1\% \).

For the 0.5 s tolerance window, the F-measure also increase but in smaller proportion.

The F-measure obtained by our algorithm is just below the one of [16], but our features and pre-processing of the SSM much simpler. This means that applying our lag-priors to compute \( c_{2,3}(t) \) on Serra et al. pre-processed matrix could even lead to larger results. We discuss this in the next part 3.4. We see that for the two RWC test-sets and a 3 s tolerance window, our algorithm achieve better results than the best results obtained in MIREX (even the ones obtained by Serra et al. – SMGA1). It should be noted that the comparison for the Beatles-B test-set cannot be made since MIREX use the whole Isophonics test-set and not only the Beatles sub-part.

Statistical tests: For a @3s tolerance window, the differences of results obtained with \( c_2(t) \) and \( c_3(t) \) are statistically significant (at 5%) for all three test-sets. They are not for a @0.5s tolerance window.

Discussion: For the RWC-Pop-B test-set, using \( c_3(t) \) instead of \( c_1(t) \) increases the F@3s for 88/100 tracks, for the Beatles-B for 144/180 tracks. In Figure 4,

---

5 These annotations are available at [http://musicdata.gforge.inria.fr/structureAnnotation.html](http://musicdata.gforge.inria.fr/structureAnnotation.html).
6 The MIREX test-set named "M-2010 test-set Original" corresponds to RWC-Pop-A, "M-2010 test-set Quaero" to RWC-Pop-B.

---

340
we illustrate one of the examples for which the use of $c_3(t)$ decreases the results over $c_1(t)$. As before the discrimination obtained using $c_3(t)$ (right sub-figure) is higher than the ones obtained using $c_1(t)$ (left sub-figure). However, because of the use of the prior $p_l(t)$ which is computed on a long duration window $[t-\Delta, t+\Delta]$ represents 20 s), $c_3(t)$ favors the detection of long-duration segments. In the example of Figure 4, parts of the annotated segments (black dashed lines) are very short segments which therefore cannot be detected with the chosen duration $\Delta$ for $p_l(t)$.

### Table 1. Results of music structure segmentation using our lag-prior method applied to the SSM as computed in part 2.2.

<table>
<thead>
<tr>
<th>Method</th>
<th>F @3s</th>
<th>P @3s</th>
<th>R @3s</th>
<th>F @0.5s</th>
<th>P @0.5s</th>
<th>R @0.5s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serra et al. [16]</td>
<td>0.791</td>
<td>0.817</td>
<td>0.783</td>
<td>0.2359</td>
<td>0.2469</td>
<td>0.2319</td>
</tr>
<tr>
<td>MIREX-2012 (SMGA1 on M-2010 test-set Original)</td>
<td>0.710</td>
<td>0.741</td>
<td>0.7007</td>
<td>0.3009</td>
<td>0.3745</td>
<td>0.2562</td>
</tr>
<tr>
<td>MIREX-2013 (FK2 on M-2010 test-set Original)</td>
<td>0.6574</td>
<td>0.8160</td>
<td>0.5599</td>
<td>0.313</td>
<td>0.355</td>
<td>0.308</td>
</tr>
</tbody>
</table>

### Table 2. Results obtained applying our lag-prior method to the SSM as computed by Serra et al. [16]

<table>
<thead>
<tr>
<th>Method</th>
<th>0.8</th>
<th>0.81</th>
<th>0.805</th>
<th>0.2678</th>
<th>0.2867</th>
<th>0.2558</th>
</tr>
</thead>
</table>

### Figure 4. Illustration of a case for which $c_3(t)$ (right sub-figure) decrease the results over $c_1(t)$ (left sub-figure). $F@3s(c_1(t)) = 0.93$ and $F@3s(c_3(t)) = 0.67$ [Track 20 form RWC-Pop-B].

### 3.4 Results obtained applying our lag-prior method to the SSM as computed by Serra et al. [16]

In order to assess the use of $c_2,3(t)$ as a generic process to improve the estimation of the segments on a SSM; we applied $c_2(t)$ to the SSM computed as proposed in [16] instead of the SSM proposed in part 2.2. The SSM will be computed using the CENS features instead of the HPCP used in [16]. For recall, in [16] the recent past of the features is taken into account by stacking the feature vectors of past frames (we used a value $m$ corresponding to 3 s). The SSM is then computed using a K nearest neighbor algorithm (we used a value of $k = 0.04$). Finally the SSM matrix is convolved with a long bivariate rectangular Gaussian kernel $G = g_1g_2^T$ (we used $s_1 = 0.5$ s $s_2 = 30$ s and $\sigma^2 = 0.16$). $c_4(t)$ is then computed from the resulting SSM. The flowchart of this method is represented in the right part of Figure 3.

Results are given in Table 2 for the various configurations and test-sets. $c_1(t)$ represents Serra et al. method [16]. As one can see, the use of a global prior ($c_2(t)$) allows to increase the results over $c_1(t)$ for the three test-sets and the two tolerance window (@3s and @0.5s). Surprisingly, this time, results obtained with a local prior ($c_3(t)$) are lower than the ones obtained with a global prior ($c_2(t)$). This can be explained by the fact that Serra et al. method applies a long duration low-pass filter ($s_1 = 30$ s) to the SSM. It significantly delays in time the maximum value of a segment in the SSM, hence delays $p_l(t)$, hence delays $c_3(t)$. In the opposite, because $c_2(t)$ is global, it is not sensitive to Serra et al. delay.

**Statistical tests:** For a @3s tolerance window, the difference of results obtained with $c_2(t)$ (0.805) and $c_1(t)$ (0.772) is only statistically significant (at 5%) for the Beatles-B test-set. For a @0.5s tolerance window, the differences are statistically significant (at 5%) for all three test-sets.
Table 2. Results of music structure segmentation using our lag-prior method applied to the SSM as computed by [16].

| Method | RWC-Pop-A |  |  |  |  |  |  |  |
|--------|-----------|---|---|---|---|---|---|
|        | F @0.5s  | P @0.5s | R @0.5s | F @0.5s  | P @0.5s | R @0.5s |
| $c_1(t)$ (without lag-prior) with Serra front-end | 0.780 | 0.846 | 0.742 | 0.254 | 0.277 | 0.246 |
| $c_2(t)$ (with global lag-prior) with Serra front-end | 0.784 | 0.843 | 0.750 | 0.289 | 0.316 | 0.275 |
| $c_3(t)$ (with local lag-prior) with Serra front-end | 0.735 | 0.827 | 0.682 | 0.245 | 0.300 | 0.215 |
|        | RWC-Pop-B |  |  |  |  |  |  |  |
|        | F @0.5s  | P @0.5s | R @0.5s | F @0.5s  | P @0.5s | R @0.5s |
| $c_1(t)$ (without lag-prior) with Serra front-end | 0.797 | 0.795 | 0.818 | 0.338 | 0.326 | 0.359 |
| $c_2(t)$ (with global lag-prior) with Serra front-end | 0.823 | 0.846 | 0.820 | 0.389 | 0.408 | 0.381 |
| $c_3(t)$ (with local lag-prior) with Serra front-end | 0.797 | 0.856 | 0.765 | 0.336 | 0.369 | 0.318 |
|        | Beatles-B |  |  |  |  |  |  |  |
|        | F @0.5s  | P @0.5s | R @0.5s | F @0.5s  | P @0.5s | R @0.5s |
| $c_1(t)$ (without lag-prior) with Serra front-end | 0.772 | 0.792 | 0.773 | 0.371 | 0.365 | 0.394 |
| $c_2(t)$ (with global lag-prior) with Serra front-end | 0.805 | 0.813 | 0.817 | 0.439 | 0.430 | 0.450 |
| $c_3(t)$ (with local lag-prior) with Serra front-end | 0.799 | 0.790 | 0.827 | 0.422 | 0.416 | 0.442 |

4. CONCLUSION AND FUTURE WORKS

In this paper, we have proposed a simple, but very efficient, method that allows using Goto 2003 method as a prior lag-probability on Serra et al. structure feature method. We provided the rational for such a proposal, and proposed two versions of the method: one using a global lag prior, one using a local lag prior. We performed a large-scale experiment of our proposal in comparison to state-of-the-art algorithms using three test-sets: RWC-Popular-Music with two sets of annotations and Isophonics/Beatles. We showed that the introduction of the lag-prior allows a large improvement of the F-Measure results (with a tolerance window of 3 s) over the three sets. Also, our method improves over the best results obtained by Serra et al. or during MIREX-2012 and MIREX-2013.

Future works will concentrate on integrating this prior lag probability on an EM (Expectation-Maximization) algorithm to estimate the true $p(t, l)$. Also, we would like to use this segmentation as a first step to a segment labeling algorithm.

Acknowledgements This work was partly founded by the French government Programme Investissements d’Avenir (PIA) through the Bee Music Project.

5. REFERENCES


ABSTRACT

Features of linguistic tone contours are important factors that shape the distinct melodic characteristics of different genres of Chinese opera. In Beijing opera, the presence of a two-dialectal tone system makes the tone-melody relationship more complex. In this paper, we propose a novel data-driven approach to analyze syllable-sized tone-pitch contour similarity in a corpus of Beijing Opera (381 arias) with statistical modeling and machine learning methods. A total number of 1,993 pitch contour units and attributes were extracted from a selection of 20 arias. We then build Smoothing Spline ANOVA models to compute matrices of average melodic contour curves by tone category and other attributes. A set of machine learning and statistical analysis methods are applied to 30-point pitch contour vectors as well as dimensionality-reduced representations using Symbolic Aggregate approXimation (SAX). The results indicate an even mixture of shapes within all tone categories, with the absence of evidence for a predominant dialectal tone system in Beijing opera. We discuss the key methodological issues in melody-tone analysis and future work on pair-wise contour unit analysis.

1. INTRODUCTION

Recent development in signal processing and cognitive neuroscience, among other fields, has revived the research on the relationship between speech and musical melody [10]. Singing in tone languages offers a particularly convenient entry point to compare musical and speech melodies, allowing us to gain insight into the ways the prosody of a particular language shapes its music. In a tone language, as opposed to an intonation language, the pitch contour of a speech sound (often a syllable) can be used to distinguish lexical meaning. In singing, however, such pitch contour can be overridden by the melody of the music, making the lyrics difficult to decode by listeners.

In such consideration, musicologists have observed that features of the prosody of the local dialect often play an important role in shaping the melodic characteristics of the regional operas in China [9, 15]. On the other hand, it is generally assumed that Beijing opera had incorporated linguistic tone systems from both the Hu-Guang (HG) dialect and Beijing (BJ) dialect [22]. Xu [19] reviewed 90 years of research on the dialect tone system in Beijing opera, and concluded that there is no agreement as to which system is predominant in shaping the melodic characteristics of the genre.

In sum, previous work indicates that the overall degree and manner of the melody-tone relationship is not entirely clear, partly due to the limitation that music scholars typically were not able to go beyond analyzing a few arias by hand [19]. In this paper, we propose a novel approach to melody-tone similarity by applying statistical modeling and machine learning methods to a set of 20 arias selected from a corpus of 381 arias of Beijing opera audio recording. The research questions are defined as follows: (1) How similar are syllable-sized melodic contours within a given tone category? (2) How similar is the “average” melodic contour to its corresponding prototype contour in speech in the same tone category? (3) Which tone system (BJ or HG) better predicts the shape of melodic contours?

Following preprocessing, we apply clustering algorithms and statistical analysis to 30-point feature vectors of pitch contours, as well as dimensionality-reduced feature vectors represented symbolically using the Symbolic Aggregate approXimation (SAX) algorithm [8]. Special considerations are given to existing hypotheses regarding the distribution of the tone systems in Beijing opera. Lastly, we build Smoothing Spline ANOVA Models to compute matrices of average melodic contour curves by tone category and other attributes.

2. KEY ISSUES IN STUDYING MELODY-TONE SIMILARITY

2.1 Beijing Opera: Performance Practice

Several features of Beijing opera may explain why the melody tone relationship remains challenging. First, the composition process of Beijing opera assumes no designated composer for any given opera. Rather, each opera is composed by re-arranging prototype arias from a inventory of arias according to the rhythmic type, role type, tempo, tempo,
Second, we define the hypotheses and specific goals in this work. We observe that in the review of tone systems in Beijing opera [19], one key assumption is that one of the two underlying dialectal systems must dominate. However, we also find evidence in the literature [22] that one may expect to find an even mixture of contours from both dialects. In this work, we consider both hypotheses and find our data to be more consistent with the second hypothesis.

3. DATA COLLECTION

3.1 Beijing Opera Audio Data Collection

The music in Beijing opera is mainly structured according to two basic principles, shengqiang and banshi, which in a broad sense define respectively its melodic and rhythmic components [17]. On top of these two structural principles, the system of role-types impose particular constraints to the execution of shengqiang and banshi. The interaction of these three components, hence, offers a substantial account of Beijing opera music. Our current collection includes 48 albums, which contain 510 recordings (tracks) featuring 381 arias and over 46 hours of audio [14].

The current study focuses on a small selection of 20 arias from the corpus to serve as a manageable starting point of the melody-tone relationship analysis. This set is selected according to a number of criteria: (1) we selected only yuanban, a rhythm type in which the duration of a syllable sized unit bears the most similarity to that of pitch; (2) we selected both types of shengqiang, namely xipi and erhuang; (3) we selected five role types: D(dan), 3(jing), LD(laodan), LS(laosheng), and XS(xiaosheng). For each combination of shengqiang and role types, we selected two arias, yielding a total of 20 arias for analysis.

3.2 Data Preprocessing

The vocal frames of the audio recordings of the 20 arias are partially-automatically segmented into syllable sized unit with boundary alignment correction by hand. The segmentation is implemented as timestamps of a TextGrid file in the speech processing software Praat [2]. The textgrid is later integrated with the metadata labels from the annotation process.

Following segmentation, we annotate the audio with lyrics extracted from the online Beijing opera libretto database jingju.net. The Chinese-character lyrics files are converted into romanized pinyin form with tone marks in the end (1,2,3, or 4) using an implementation of Java library pinyin4j. A Praat Script is implemented to compute similarity between melodic and linguistic tone F0 contours should be ruled out.

2 We must bear in mind also that speech tones are generated under a different mechanism than pitch contours in singing. For one thing, the latter has a more planned mechanism of design - the composition of the music. In speech, as the qTA model has demonstrated [12], speakers may have a pitch target (defined by a linear equation) in mind during articulation, but the actual F0 realization is subject to a set of much complex physiological and contextual linguistic factors, which may be modeled by a third-order critically damped system [12]. This complication does not exist in music: in singing, a singer can realize the exact F0 target as planned. Therefore, we propose that approaches that directly compute similarity between melodic and linguistic tone F0 contours should be ruled out.

3 Some cite three dialects [22]. HuGuang, Beijing, and ZhongZhou YinYun.

4 Automatic segmentation using forced-alignment with machine-readable form of the score is currently being developed. For the current study, we used the result of a trained spectral-based classifier [3] that is able to separate the pure instrumental frames of the audio signal from those frames that contain both vocal and instrumental parts. The result of this segmentation is in many cases the voiced segment (vowel) of a syllable, which is precisely the unit of our analysis.
to automatically parse the romanized lyrics files and to annotate the segmented audio files. The metadata attributes (shengqiang, role type, artist, duration, tone category, and word) are also automatically annotated for each segmented unit.

3.3 Pitch Contour Extraction

We then proceed to the extraction of F0 values for each annotated pitch contours of interest. The F0 is computed using the MELODIA salience function [13] within the Essentia audio signal processing library in Python [1], in order to minimize the interference of background instrumental ensemble to the computation of F0 of the primary vocal signal. For the sake of analysis, we produce down-sampled 30-point F0 vectors by using equidistant sampling across each pitch contour. All F0 values are normalized so that each contour has a mean F0 of 0 and sd of 1. A 5-point weighted averaging sliding window is applied to smooth the signal.

4. PROPOSED APPROACH

In this section we overview the methodology employed in the analysis of the extracted pitch contour dataset. As discussed above in 2.2, all methodology are boiled down to addressing the research question (1), which attempts to analyze and describe the variances and clusters found in melodic contours of each tone category and across categories. Research question (2) and (3), both of which involve comparing music with speech melody, can only be addressed indirectly by the average curves computed by the SSANOVA model for each tone category.

4.1 Time Series Representation

In a standard melodic similarity task, such as query-by-humming (QBH), the goal of the task is usually to match the melody as precisely as possible. However, in the current task, our goal is in a way to model the human perception of tone. An important capacity of human cognition is its capacity to abstract away the commonalities from groups of pitch contours with much different fine detail variations. In this study, we experiment with the Symbolic Aggregate approXimation (SAX) [8] representation of pitch contour vectors.

SAX offers a lower dimension coarse representation, whose distance lower-bounds true distance of time series. It transforms the pitch contour into a symbolic representation with length (nseg=desired length of the feature vector) and alphabet size (m) parameters, the latter being used to divide the pitch space of the contour into m equiprobable segments assuming a Gaussian distribution of F0 values.

In this work, we rely on the SAX representation (1) as a effective and economic way to represent the shapes of time series in statistical analysis; and (2) as a coarse symbolic representation for clustering. To ensure the validity of SAX to reflect the true shape of the original 30-point vector, we experiment with different parameters and use four different ways to evaluate the effectiveness of the SAX representation (discussed below).

4.2 Methodology

As discussed in 2.2, we consider two different analytical approaches in this work based on the two hypotheses regarding the distribution of tone systems in Beijing opera. In the first hypothesis (H1), we assume that there is one predominant tone system (BJ or HG) in Beijing opera. We define a time-series clustering task with the goal of clustering all tone contours into four big clusters, corresponding to four tone categories. Using dynamic time warping (DTW) as the distance measure, we perform K-means Clustering and Agglomerative Clustering (hierarchical) on the 30-point pitch vectors. Using the lower bounding mindist distance measure defined for SAX-based symbolic representation, we also perform K-means Clustering on the SAX string vectors of length 5 (alphabet size is 3).

In the second hypothesis (H2), we expect an even mixture of tone systems and tone shapes in all tone categories. In this scenario, our goal is to perform exploratory cluster analysis on the distribution of contours shapes within each tone categories. More specifically, we perform statistical and clustering analysis on the SAX-based shapes within and across tone categories. In addition, we investigate distribution of attributes associated with each sub-cluster of shape.

We infer from literature [22] that regardless of the distribution of tone systems, the first tone is expected to have the most consistent flat shape if a reasonably strong correlation is assumed between linguistic tone and melodic contour (Notice that tone 1 has the same flat shape across dialects in Figure 1). More specifically, a musico logical analysis by hand reveals that the most predominant shape in tone 1 is flat or flat with a final fall (henthforce referred to as Hypothesis 3, or H3, also inferred from rules described in [22]).

Lastly, we build a Smoothing Spline ANOVA model with the goal of (1) computing average pitch contours for each tone category, and (2) quantifying the variances accounted for by each predictor variable in different tone categories. Smoothing splines are essentially a piecewise polynomial function that connects discrete data points called knots. It includes a smoothing parameter to find the

---

5 The unvoiced part in the beginning of the syllable is skipped in the down-sampling. In addition, the downsampling strategy is also fine tuned in order to filter out the spurious pitch values computed by MELODIA in the beginning portion of the voiced segments.

6 Also known as categorical perception in cognitive sciences.

7 SAX is the first symbolic representation for time series that allows for dimensionality reduction and indexing with a lower-bounding distance measure. In classic data mining tasks such as clustering, classification, index, etc., SAX is as good as well-known representations such as DWT and DFT, while requiring less storage space. Even though SAX representation is mostly used outside of MIR, it has been applied to the QBH task [16].

8 Strictly speaking, the Gaussian assumption is not met in the pitch space musical notes. However, due to the nature of the task that does not require precise mapping, we use the original SAX implementation without revising the Gaussian assumption.
best fit when the data tend to be noisy, estimated by minimizing the following function:

$$G(x) = \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \int_{a}^{b} (f''(u))^2 \, du \tag{1}$$

where $n$ is the number of data points, $\lambda$ is the smoothing parameter, and $a$ and $b$ are the x coordinates of the endpoint of the spline.

The Smoothing Spline ANOVA (SSANOVA) is of the following form, each component of $f$ is estimated with a smoothing spline:

$$f = \mu + \beta x + \text{main group effect} + \text{smooth}(x) + \text{smooth}(x; \text{group}) \tag{2}$$

where the main group effects correspond to the smoothing splines for each dataset, smooth($x$) is the single smoothing spline that would be the best fit for all of the data put together, and the interaction term smooth($x; \text{group}$) is the smoothing spline representing the difference between a main effect spline and the smooth($x$) spline \cite{4}.

5. RESULTS AND DISCUSSION

Evaluation of SAX representation. Experimentation with different values of nseg and alphabet size shows that, in order to capture the abstract nature of tone perception and to minimize the effect of large amount of noise in pitch movements, a limit of nseg $<= 3$ must be placed. This is a reasonable limit considering that linguists use only two or three segments to represent tone contours in any tone language \cite{10}. In this work, we use nseg=2 and alphabet size of 3. This choice of parameterization is evaluated as a sufficient representation for the perception of pitch contour shapes in four different ways.

First, a perceptual evaluation is carried out by having a human listener judge the shape of the contours as flat, rising, or falling (n=50). The result shows that the SAX representation achieves a 88% accuracy. Second, hierarchical clustering is performed on all contours in a given tone category. The result is then compared with the SAX labels. Figure 2 shows that in addition to meaningful groupings of tone labels across four tones, F=Falling, R=Rising, Z=Flat.

Clustering of 4 tones (H1). Unsupervised K-means Clustering with 30-point vectors cannot learn any meaningful grouping of tone categories regardless of the number of desired clusters (performed in data mining tool Weka\cite{7} with Euclidean distance, and in R with DTW distance, number of desired clusters varied within [4,10], otherwise default setting). Likewise, hierarchical clustering with DTW distance cannot find any meaningful groupings of DTW distance at any level. This shows that we cannot find a distinct, predominant shape for a given tone category, and failure to cluster melodic contours into meaningful groups that correspond to four tones.

Exploratory within-category shape analysis (H2 and H3). First, we use the validated SAX representations to compute the distribution of three shapes rising(R), falling(F), flat(Z) within each tone category. Figure 3 shows that consistent with H2, each tone category consists of a even mixture of all shapes, with the absence of a dominant shape \cite{12}. To get a more fine-grained analysis of the distributions of shapes, a two-sample test on hypothesis of population proportion is performed across tones and shapes. Results show that the proportion of rising is significantly different across four tones from the proportion from a tone category are classified into SAX class labels with a mean accuracy of 85.2%.

---

\cite{9} The SSANOVA does not return an F value. Instead, the smoothing parameters of the components smooth ($x$) and smooth ($x$; group) are compared to determine their relative contributions to the equation \cite{4}. In this paper, we use the implementation of gss package in statistical computing language R.

\cite{10} In linguistics convention, high tone=H, low tone=L, rising=LH, falling=HL, falling rising=HLH, etc.

\cite{11} Tone1="bb", tone2="ac", tone3="bc", tone4="ca", a< b < c in pitch space.

\cite{12} Tone 5 is a neutral tone whose contour shape depends on the tone the precedes it. It exists in our dataset but is not under consideration in the current analysis.
of flat ($\chi^2 = 17.4065, \text{df} = 4, \ p = 0.002$) or falling ($\chi^2 = 18.238, \text{df} = 4, \ p = 0.001$). The proportion of flat and falling are not significantly different ($p = 0.96$). Furthermore, a one-sample test show that, only the proportion of rising shape is significantly different across four tones ($\chi^2 = 21.9853, \text{df} = 4, \ p = 0.0005$), but not between tone 2, tone 3, tone 4 (except with the difference between tone 2 and tone 4 that reached significance at $p = 0.04$). Therefore, with the exception of tone 1 and tone 2 ($p=0.22$, tone 2 seem to behave more similarly to tone 1), the proportion of rising is highly significantly different between in tone 1 and other tones, whereas no strong significant differences are found among other tones. This result supports the H3 discussed above in asserting that tone 1 is mostly consisted of a mixture of flat and falling shapes (to be more specific, flat and flat-falling in H3).

**Table 1. SSANOVA Model comparison**

<table>
<thead>
<tr>
<th>model parameter</th>
<th>levels (nominal)</th>
<th>R-squared (T1)</th>
<th>R-squared (T2)</th>
<th>R-squared (T3)</th>
<th>R-squared (T4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>word</td>
<td>468</td>
<td>0.178</td>
<td>0.0772</td>
<td>0.0566</td>
<td>0.0667</td>
</tr>
<tr>
<td>artist</td>
<td>15</td>
<td>0.0885</td>
<td>0.0606</td>
<td>0.0465</td>
<td>0.0420</td>
</tr>
<tr>
<td>shengqiang</td>
<td>2</td>
<td>0.027</td>
<td>0.0235</td>
<td>0.0154</td>
<td>0.0123</td>
</tr>
<tr>
<td>position</td>
<td>4</td>
<td>0.028</td>
<td>0.0211</td>
<td>0.0189</td>
<td>0.0103</td>
</tr>
<tr>
<td>role type</td>
<td>5</td>
<td>0.029</td>
<td>0.0273</td>
<td>0.0242</td>
<td>0.0180</td>
</tr>
<tr>
<td>all</td>
<td>na</td>
<td>0.032</td>
<td>0.028</td>
<td>0.0249</td>
<td>0.2000</td>
</tr>
</tbody>
</table>

and artist are the best predictors of all the predictor variables (as well as all combinations of predictor variables not shown here). However, it is noticeable that the even the best model only explains less than 20% of the variance among all pitch curves in a given tone category. This indicates a large amount of variation in the shape of the contours. On the other hand, the consistently larger value of R-squared for tone 1 indicates positive evidence for a more consistent shape in tone 1, as stated in the H3 discussed above.

**Figure 5. Average curves computed by the time+word SSANOVA model.**

Average curves of four tones are computed based on this model (Figure 5), with confidence intervals shown in dashed lines. The interpretation of these average curves should be done with caution, because of the low R squared value and large standard error in the model. In particular, tone 1 and tone 2 has average contours that differ from both HG and BJ system; tone 3 and tone 4 show resemblance to BJ and HG system, respectively.

6. CONCLUSION AND FUTURE WORK

This work constitutes a preliminary step in the computational approaches to the linguistic tone-melodic contour similarity in Beijing opera singing. In this work, we focused on the single-syllable sized contours by adopting different methodologies based on competing hypothesis of tone systems. We have demonstrated the effective-

---

15 And also notice that the R-squared value is highly correlated with the number of levels in the nominal attributes.
ness of SAX-based representations in tasks of shape analysis and time-series mining. The results indicate a even mixture of shapes within each tone category, with the absence of a dominant tone system in Beijing opera. In addition, we found evidence supporting the hypothesis that tone 1 is sung with more consistent shape than other tones. Overall, our results point to low degree of similarity in single-syllable pitch contours. Given the discussion and methodology proposed here, we expect future research on pair-wise syllable contour similarity analysis to yield more promising results.

7. ACKNOWLEDGEMENT

This research was funded by the European Research Council under the European Union Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583).

8. REFERENCES


A PROXIMITY GRID OPTIMIZATION METHOD TO IMPROVE AUDIO SEARCH FOR SOUND DESIGN

Christian Frisson, Stéphane Dupont, Willy Yvart, Nicolas Riche, Xavier Siebert, Thierry Dutoit

ABSTRACT

Sound designers organize their sound libraries either with dedicated applications (often featuring spreadsheet views), or with default file browsers. Content-based research applications have been favoring cloud-like similarity layouts. We propose a solution combining the advantages of these: after feature extraction and dimension reduction (Student-Stochastic Neighbor Embedding), we apply a proximity grid, optimized to preserve nearest neighborhoods between the adjacent cells. By counting direct vertical / horizontal / diagonal neighbors, we compare this solution over a standard layout: a grid ordered by filename. Our evaluation is performed on subsets of the One Laptop Per Child sound library, either selected by thematic folders, or filtered by tag. We also compare 3 layouts (grid by filename without visual icons, with visual icons, and proximity grid) by a user evaluation through known-item search tasks. This optimization method can serve as a human-readable metric for the comparison of dimension reduction techniques.

1. INTRODUCTION

Sound designers source sounds in massive collections, heavily tagged by themselves and sound librarians. If a set of sounds to compose the desired sound effect is not available, a Foley artist records the missing sound and tags these recordings as accurately as possible, identifying many physical (object, source, action, material, location) and digital (effects, processing) properties. When it comes to looking for sounds in such collections, successive keywords can help the user to filter down the results. But at the end of this process, hundreds of sounds can still remain for further review. This creates an opportunity for content-based information retrieval approaches and other means for presenting the available content. From these observations, we elicited the following research question: can content-based organization complement or outperform context-based organization once a limit is reached when filtering by tag?

This work partly addresses this question and presents a solution to interactively browse collections of textural sounds after these have been filtered by tags.

We organize sounds in a two-dimensional map using content-based features extracted from their signal. These features are mapped to two visual variables. First, the position of the sample on the screen is obtained after applying dimension reduction over the features followed by a proximity grid that structures items on a grid which facilitates navigation and visualization, in particular by reducing the cluttering. The organization of the samples on the grid is optimized using a novel approach that preserves the proximity on the grid of a maximum of nearest neighbors in the original high-dimensional feature space. Second, the shape of the sample is designed to cue one important content-based feature, the perceptual sharpness (a measure of the “brightness” of the sound).

This approach is evaluated through a known-item search task. Our experiments provide one of the first positive result quantitatively showing the interest of MIR-based visualization approaches for sound search, when then proper acoustic feature extraction, dimension reduction, and visualization approaches are being used.

The paper is organized as follows. First, in section 2, we examine the landscape of existing systems dedicated to browsing files in sound design. We then describe how we designed our system in Section 3. In section 4, we describe our evaluation approach, experiments and obtained results. We finish by summarizing our contributions and provide an glimpse of future research directions.

2. BACKGROUND

This section provides a review of the literature and empirical findings on systems for sound design, and outlines some results and gaps that motivated this work.

Systems for mining sounds, particularly for sound design, are actually rather scarce. These may however share some similarities with systems targeted to the management of music collections, in particular in the content-based processing workflow that allows to organize the audio files. A comprehensive survey on these aspects has been proposed by Casey et al. [4]. We nevertheless believe that the design of the user interface of each system class might benefit from different cues from information visualization and human-computer interaction, and that major progress is still possible in all these areas.
2.1 Research-grade systems

The work presented in [18] underlines that few published research provide accurate usability evaluations on such systems, beyond informal and heuristic ones. The author justifies that this may have occurred because complementary research communities have actually been evolving essentially in separate silos. These include the music information retrieval and the human-computer interaction communities. In that work, 20 systems with auditory display are nevertheless reviewed and compared, including 2 audio browsers that are presented hereafter.

Sonic Browser focused on information visualization [7], and later approached content-based organization through the Marsyas framework [3]. A 2D starfield display allows to map the metadata of audio files to visual variables. Its HyperTree view consists in a spring-layout hierarchical graph visualization for browsing the file tree of sound collections. They qualitatively evaluated these views with 15 students through timed tasks and a questionnaire [2]; and their system against the Microsoft Windows 2000 explorer through a think-aloud protocol with 6 students [7].

SoundTorch, the most recent content-based audio browser, has been designed by people aware of audio engineering practices [11]. It relies on Mel-Frequency Cepstral Coefficients (MFCCs) as features, clustered with a Self-Organizing Map (SOM) but initialized with smooth gradients rather than randomly, so that the horizontal axis corresponds to a tonal-to-noisy continuum and the vertical axis to pitch increase / dull-to-bright. In addition to cueing in the variety of content through the position of the nodes corresponding to sounds, SoundTorch makes use of the node shape to convey additional information: the temporal evolution of the power of the signal is mapped to a circle.

It is the only related work to provide a quantitative user evaluation. They positively evaluated known- and described-item search tasks comparatively to a list-based application. A dozen of users were involved. However, it is not clear from this comparison whether the approach outperforms the list-based application because of its content-based capabilities, or else because of its interactive abilities (particularly its instant playback of closely-located nodes in the map), or both. Moreover, it has been chosen to randomize the sound list. Sound designers either buy commercial sound libraries that are tagged properly and named accordingly, or else record their own. They also usually spend a significant amount of time to tag these libraries. Therefore, to our opinion, a more realistic baseline for comparison should be a basic ordering by filename.

CataRT is an application developed in the Max/MSP modular dataflow framework, that “mosaics” sounds into small fragments for concatenative synthesis. A 2D scatter plot allows to browse the sound fragments, assigning features to the axes. The authors recently applied a distribution algorithm that optimizes the spreading of the plotted sounds by means of iterative Delaunay triangulation and a mass-spring model, so as to solve the non-uniform density inherent to a scatter plot, and open new perspectives for non-rectangular interfaces such as the circular reacTable and complex geometries of physical spaces to sonify. To our knowledge, no user study has yet been published for this tool. It is however claimed as future work [12].

In summary, it appears that no evaluation have been proposed previously on the specific contribution of content-based analysis to the efficiency of sound search. This is a gap we started to address in this work.

2.2 Commercial systems

It is worth mentioning here that some commercial systems, some making use of content-based approaches, have also been proposed, although no quantitative evaluation of those can be found in the literature. A pioneering application is SoundFisher by company Muscle Fish [21], start-up of scientists that graduated in the field of audio retrieval. Their application allowed to categorize sounds along several acoustic features (pitch, loudness, brightness, bandwidth, harmonicity) whose variations over time are estimated by average, variance and autocorrelation. Sounds are compared from the Euclidean distance over these features. The browser offers several views: a detail of sound attributes (filename, samplerate, file size...) in a spreadsheet, a tree of categories resulting from classification by example (the user providing a set of sounds), and a scatter plot to sort sounds along one feature per axis.

A second product, AudioFinder by Iced Audio 1 mimics personal music managers such as Apple iTunes: on top a textual search input widget allows to perform a query, a top pane proposes a hierarchical view similar to the “column” view of the Finder to browse the file tree of the collection, a central view features a spreadsheet to order the results along audio and basic file metadata, a left pane lists saved results like playlists. A bottom row offers waveform visualizations and the possibility to apply audio effect processing to quickly proof the potential variability of the sounds before dropping these into other creative applications.

A major product, Soundminer HD 2, provides a similar interface, plus an alternative layout named 3D LaunchPad that allows, similarly to the Apple Finder CoverFlow view, to browse sounds (songs) by collection (album) cover, with the difference that the former is a 2D grid and the latter a 1D rapid serial visualization technique.

Other companies facilitating creativity such as Adobe with Bridge 3 provide more general digital asset management solutions that are accessible through their entire application suite. These focus on production-required capabilities and seem to avoid content-based functionalities.

From our contextual inquiry we noticed that sound designers also make use of simple browsers, such as the default provided by the operating system, optionally associated to a spreadsheet to centralize tags.

\[1\]http://www.icedaudio.com
\[2\]http://www.soundminer.com
\[3\]http://www.adobe.com/products/bridge.html
3. OUR SOLUTION

Our system blends knowledge gained from the fields of multimedia information retrieval (content-based organization), human-computer interaction (usability evaluation) and information visualization (visual variables).

3.1 A multimedia information retrieval pipeline

One first step is feature extraction. For sound and music, a large variety of temporal and/or spectral features have been proposed in the literature [4, 15]. We based our features set from [6] since their evaluation considered textural sounds. In short, we used a combination of derivatives of and statistics (standard deviation, skewness and/or kurtosis) over MFCCs and Spectral Flatness (SF). We did not perform segmentation as our test collections contain textural sounds of short length and steady homogeneity.

Another important step is dimension reduction. From our perspective, one of the most promising approach is Stochastic Neighborhood Embedding (SNE) using Student-t distributions (t-SNE) [13]. It has been previously qualitatively evaluated on sound collection visualization [6, 9]. The method has an interesting information retrieval perspective, as it actually aims at probabilistically preserving high-dimensional neighbors in a lower-dimensional projection (2D in our work), and actually maximizes continuity (a measure that can intuitively be related to recall in information retrieval) in the projected space. One emergent result is that recordings from the same source seem to have only slight variations that are almost always neighbors in the 2D representation, as the recall is high. Another popular but older approach for dimensionality reduction are SOMs. In [14], it has been compared with most recent techniques, and in particular the Neighbor Retrieval Visualizer (NeRV, a generalization of SNE). SOMs produced the most trustworthy (a measure that can intuitively be related to precision in information retrieval) projection but the NeRV was superior in terms of continuity and smoothed recall. As SNE is a special case of NeRV where a tradeoff is set so that only recall is maximized, we infer from those results that SNE is a better approach for our purposes than SOM. Qualitative evaluations of different approaches applied to music retrieval have been undertaken [19]: Multidimensional Scaling (MDS), NeRV and Growing SOMs (GSOM). Users described MDS to result in less positional changes, NeRV to better preserve cluster structures and GSOM to have less overlappings. NeRV and presumably t-SNE seem beneficial in handling cluster structures.

Besides, we propose in this paper an approach to reduce the possible overlappings in t-SNE. An undesirable artifact of the original t-SNE approach however comes from the optimization procedure, which relies on gradient descent with a randomly initialized low-dimensional representation. It creates a stability issue, where several runs of the algorithm may end up in different representations after convergence. This works against the human memory. We thus initialized the low-dimensional representation using the two first axes of a Principal Component Analysis (PCA) of the whole feature set.

3.2 Mapping audio features to visual variables

Displaying such a representation results in a scatter plot or starfield display. We address two shortcomings: 1) clusters of similar sounds might not be salient, and 2) this visualization technique may cause overlap in some areas. SonicBrowser [7], that we analyzed in the previous section, and the work of Thomas Grill [9], dedicated to textural sounds, approached the first issue by mapping audio features to visual variables. Ware’s book [20] offer great explanations and recommendations to use visual variables to support information visualization tailored for human perception. Thomas Grill’s approach was to map many perceptual audio features to many visual variables (position, color, texture, shape), in one-to-one mappings.

3.2.1 Content-based glyphs as sound icons

Grill et al. designed a feature-fledged visualization technique mapping perceptual qualities in textural sounds to visual variables [9]. They chose to fully exploit the visual space by tiling textures: items are not represented by a distinct glyph, rather by a textured region. In a first attempt to discriminate the contribution of information visualization versus media information retrieval in sound browsing, we opted here for a simpler mapping. We mapped the mean over time of perceptual sharpness to the value in the Hue Saturation Value (HSV) space of the node color for each sound, normalized against the values for all sounds in each collection. A sense of brightness is thus conveyed in both the audio and visual channels through perceptual sharpness and value. We also used the temporal evolution of perceptual sharpness to define a clockwise contour of the nodes, so that sounds of similar average brightness but different temporal evolution could be better discriminated. To compute positions, perceptual sharpness was also added to the feature selection, intuited it would gather items that are similar visually. The choice of perceptual sharpness was motivated by another work of Grill et al. [10]: they aimed at defining features correlated to perceived characteristics of sounds that can be named or verbalized through personal constructs. High-low, or brightness of the sound, was the construct the most correlated to an existing feature: perceptual sharpness.

3.2.2 A proximity grid optimizing nearest neighbors

For the removal of clutter in 2D plots, two major approaches exist: reducing the number of items to display, or readjusting the position of items. In our context, we want to display all the items resulting of search queries by tag filtering. For this purpose, we borrow a method initially designed to solve the problem of overlap for content-based image browsing [16]: a proximity grid [1]. Their work is heavily cited respectively for the evaluation of multidimensional scaling techniques [1] and as a pioneering application of usability evaluation for multimedia information retrieval [16], but almost never regarding the proximity grid. To our knowledge, no audio or music browser approached this solution.
A proximity grid consists in adapting the coordinates of each item of a 2D plot to magnetize these items on an evenly-distributed grid. Basalaj proposed several variants to compute a proximity grid: greedy methods with spiral search to find empty cells and empty/swap/bump strategies to assign items to cells; an improved greedy method replacing spiral search by shortest distance estimation; a “squeaky wheel” optimization using simulated annealing, and a genetic algorithm [1]. We implemented the simplest greedy method with all strategies. To determine the order of the items to assign to cells, we used the fast minimum spanning tree algorithm implementation from the machine learning library mlpack of Boruvka’s dual-tree based on k-dimensional trees [5]. Applied in high dimension of the audio features, the empty strategy starts with shortest edges while it is the opposite for swap and bump strategies, according to Basalaj. We opted for a simplification: a spiral search always turning clockwise and starting above the desired cell, while it is recommended to choose the rotation and first next cell from exact distance computation between the actual coordinates of the node and the desired cell.

The minimal side of a square grid is the ceil of the square root of the collection size, providing the most space efficient density. To approximate a least distorted grid, the collection size can be taken as grid side. To come up with a tradeoff between density and neighborhood preservation, we estimate the number of high-dimensional nearest neighbors (k=1) preserved in 2D at a given grid resolution simply by counting the number of pairs in adjacent cells. We distinguish the amounts of horizontal, vertical and diagonal neighbors since different search patterns may be opted by users: mostly horizontal or vertical for people accustomed respectively to western and non-western reading order, diagonal may be relevant for grids of light density.

For our experiments described in the next section, we prepared the collections by qualitative selection of the optimal grid resolution based on the amounts of horizontal, vertical and diagonal adjacent neighbors computed for each resolution between the minimal side and the least distorted approximate, comparing such amounts between a proximity grid applied after dimension reduction and a grid ordered by filename. Not all collections presented a proximity grid resolution that outperformed a simple grid by filename in terms of neighbor preservation.

4. EXPERIMENTS

4.1 Open dataset

The One Laptop Per Child (OLPC) sound library⁴ was chosen so as to make the following tests easily reproducible, for validation and comparison perspectives, and because it is not a dataset artificially generated to fit with expected results when testing machine learning algorithms. It is licensed under a Creative Commons BY license (requiring attribution). It contains 8458 sound samples, 90 sub-libraries combine diverse types of content or specialize into one type, among which: musical instruments riffs or single notes, field recordings, Foley recording, synthesized sounds, vocals, animal sounds. It is to be noted, especially for subset libraries curated by Berklee containing Foley sound design material, that within a given subset most samples seem to have been recorded, if not named, by a same author per subset. It is thus frequent to find similar sounds named incrementally, for instance Metal on the ground $n$ with $n$ varying from 1 to 4. These are likely to be different takes of a recording session on a same setting of sound-ing object and related action performed on it. Ordering search results by tag filtering in a list by path and filename, similarly to a standard file browser, will thus imprint local neighborhoods to the list.

4.2 Evaluation method

We chose to perform a qualitative and quantitative evaluation: qualitative through a feedback questionnaire, quantitative through known-item search tasks as popularized recently for video browsers by the Video Browser Showdown [17]. In the context of audio browsers, for each task the target sound is heard, the user has to find it back as fast as possible using a given layout. Font’s thesis compared layouts for sound browsing: automatic (PCA), direct mapping (scatter plot) and random map [8]. Time and speeds were deliberately not investigated, claiming that people employ different search behaviors.

⁴http://wiki.laptop.org/go/Free_sound_samples
4.3 Design

We undertook four experiments: the first comparing grid and glyph-less cloud layouts motivated us to add glyph representations (cloud was outperformed), the second and third confirmed that a proximity grid was to be investigated (cloud still outperformed), the last validated these choices.

We recorded several metrics (success times, pointer distances and speeds, audio hovers) and ratings from feedback questionnaires. Here we only report the last experiment and only analyze times taken to successfully find targets.

The fourth experiment was designed as a within-subject summative evaluation. Figure 2 shows the exact sequence of tasks presented to the users. An additional collection was used for training tasks with each layout.

Each layout was given a nickname: grid for the simple grid ordered by filename, album for its upgrade with glyphs, metro for the proximity grid of optimal resolution for neighbors preservation. These short nicknames brought two advantages: facilitating their instant recognition when announced by the test observer at the beginning of each task, and suggesting search patterns: horizontal land mowing for grid and album, adjacent cell browsing for metro. The metro layout was described to users using the metaphor of metro maps: items (stations) can form (connect) local neighborhoods and remote “friends” (through metro lines usually identified by color).

4.4 Participants and apparatus

16 participants (5 female) of mean age 28 (+/- 6.3) each performed 9 tasks on 3 different collections. Besides 2 subjects, all the participants have studied or taught audiovisual communication practices (sound design, film edition).

They were asked which human sense they favored in their work (if not, daily) on a 5-point Likert scale, 1 for audition to 5 for vision: on average 3.56 (+/- 0.60). All self-rated themselves with normal audition, 10 with corrected vision.

We used an Apple Macbook Pro Late 2013 laptop with 15-inch Retina display, with a RME FireFace UCX sound card, and a pair of Genelec active loudspeakers. A 3Dconnexion Space Navigator 3D mouse was repurposed into a buzzer to submit targets hovered by the touchpad, with audio feedback of the closest node to the pointer.

4.5 Results

A one-way ANOVA shows that there is a quite significant difference between views within subjects on success times (p=.02), more on self-reported ratings of efficiency (p<.001) and pleasurability (p<.001). Mean and standard deviations are compared in table 1. A Tukey multiple comparisons of success times means at a 95% family-wise confidence level on layouts shows that metro outperformed grid (p=.01), but album was not significantly better than grid (p=.34) or worse than metro (p=.26).

4.6 Discussion

Feature extraction is a one-shot offline process at indexing time. Dimension reduction for layout computation is a process that should be close to real-time so as not to slow down search tasks and that is likely to be performed at least once per query. Decent results can be achieved by combining only content-based icons and simple ordering by filename. A content-based layout comes at a greater computational cost but brings significant improvements.

5. CONCLUSION, FUTURE WORKS

We proposed a method to assist sound designers in reviewing results of queries by browsing a sound map optimized for nearest neighbors preservation in adjacent cells of a proximity grid, with content-based features cued through glyph-based representations. Through a usability evaluation of known-item search tasks, we showed that this solution was more efficient and pleasurable than a grid of sounds ordered by filenames.

An improvement to this method would require to investigate all blocks from the multimedia information retrieval data flow. First, other features tailored for sound effects should be tried. Second, we have noticed that some of the first high-dimensional nearest neighbors are positioned quite far away in 2D, already past dimension reduction. Reducing pairwise distance preservation errors may be an investigation track.
6. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their careful recommendations. We thank all the testers for their time and patience in performing tasks that were sometimes too difficult. This work has been partly funded by the Walloon Region of Belgium through GreenTIC grant SONIXTRIP.

7. REFERENCES


INTRODUCING A DATASET OF EMOTIONAL AND COLOR RESPONSES TO MUSIC

Matevž Pesek¹, Primož Godec¹, Mojca Poredoš¹, Gregor Strle², Jože Guna³,
Emilija Stojmenova³, Matevž Pogačnik³, Matija Marolt¹

¹University of Ljubljana, Faculty of Computer and Information Science
matevz.pesek,primoz.godec,mojca.poredos,matija.marolt@fri.uni-lj.si
²Scientific Research Centre of the Slovenian Academy of Sciences and Arts,
gregor.strle@zrc-sazu.si
³University of Ljubljana, Faculty of Electrotechnics
joze.guna,emilija.stojmenova,matevz.pogacnik@fe.uni-lj.si

ABSTRACT

The paper presents a new dataset of mood-dependent and color responses to music. The methodology of gathering user responses is described along with two new interfaces for capturing emotional states: the MoodGraph and MoodStripe. An evaluation study showed both interfaces have significant advantage over more traditional methods in terms of intuitiveness, usability and time complexity. The preliminary analysis of current data (over 6,000 responses) gives an interesting insight into participants’ emotional states and color associations, as well as relationships between musically perceived and induced emotions. We believe the size of the dataset, interfaces and multi-modal approach (connecting emotional, visual and auditory aspects of human perception) give a valuable contribution to current research.

1. INTRODUCTION

There is no denial that strong relationship exists between music and emotions. On one hand, music can express and induce a variety of emotional responses in listeners and can change our mood (e.g. make us happy – we consider mood to be a longer lasting state). On the other hand, our current mood strongly influences our choice of music - we listen to different music when we’re sad than when we’re happy.

It is therefore not surprising that this relationship has been studied within a variety of fields, such as philosophy, psychology, musicology, anthropology or sociology [1]. Within Music Information Retrieval, the focus has been on mood estimation from audio (a MIREX task since 2007), lyrics or tags and its use for music recommendation and playlist generation, e.g. [2-5].

To estimate and analyze the relationship between mood and music, several datasets were made available in the past years. The soundtracks dataset for music and emotion contains single mean ratings of perceived emotions (labels and values in a three-dimensional model are given) for over 400 film music excerpts [6]. The MoodSwings Turk Dataset contains on average 17 valence-arousal ratings for 240 clips of popular music [7]. The Cal500 contains a set of mood labels for 500 popular songs [8], at around three annotations per song, and the MTV Music Data Set [9] a set of 5 bipolar valence-arousal ratings for 192 popular songs.

In this paper, we introduce a new dataset that captures users’ mood states, their perceived and induced emotions to music and their association of colors with music. Our goals when gathering the dataset were to capture data about the user (emotional state, genre preferences, their perception of emotions) together with ratings of perceived and induced emotions on a set of unknown music excerpts representing a variety of genres. We aimed for a large number of annotations per song, to capture the variability, inherent in user ratings.

In addition, we wished to capture the relation between color and emotions, as well as color and music, as we believe that color is an important factor in music visualizations. A notable effort has been put into visualizing the music data on multiple levels: audio signal, symbolic representations and meta-data [10]. Color tone mappings can be applied onto the frequency, pitch or other spectral components [11], in order to describe the audio features of the music [12], or may represent music segments. The color set used for most visualizations is picked instinctively by the creator. To be able to provide a more informed color set based on emotional qualities of music, our goal thus was to find out whether certain uniformity exist in the perception of relations between colors, emotions and music.

The paper is structured as follows: section 2 describes the survey and its design, section 3 provides preliminary analyses of the gathered data and survey evaluation and section 4 concludes the paper and describes our future work.
2. ONLINE SURVEY

We gathered the dataset with an online survey, with the intention to reach a wide audience and gather a large number of responses. We started our survey design with a preliminary questionnaire, which provided some basic guidelines for the overall design. We formed several research questions to drive the design and finally implemented the survey which captures the user’s current emotional state, their perception of colors and corresponding emotions, as well as emotions perceived and induced from music, along with the corresponding color. After the first round of response gathering was completed, we performed a new survey designed to evaluate different aspects of user experience with our original survey.

2.1 Preliminary study

Although there exists some consent that a common set of basic emotions can be defined [13], in general there is no standard set of emotion labels that would be used in music and mood researches. Some authors choose labeled sets intuitively, with no further explanation [14]. In contrast, we performed an initial study in order to establish the relevant set of labels. For the purpose of eliminating the cultural and lingual bias on the labelling, we performed our survey in Slovenian language for Slovene-speaking participants.

The preliminary questionnaire asked the user to describe their current emotional state through a set of 48 emotion labels selected from literature [15-17], each with an intensity-scale from 1 (inactive) to 7 (active). The questionnaire was solved by 63 participants. Principal component analysis of the data revealed that first three components explain 64% of the variance in the dataset. These three components strongly correlate to 17 emotion labels chosen as emotional descriptors for our survey.

We also evaluated the effectiveness of the continuous color wheel to capture relationships between colors and emotions. Responses indicated the continuous color scale to be too complex and misleading for some users. Thus, a modified discrete-scale version with 49 colors displayed on larger tiles was chosen for the survey instead. The 49 colors have been chosen to provide a good balance between the complexity of the full continuous color wheel and the limitations of choosing a smaller subset of colors.

2.2 The survey

The survey is structured into three parts, and contains questions that were formulated according to our hypotheses and research goals:

- user’s mood impacts their emotional and color perception of music;
- relations between colors and emotions are uniform in groups of users with similar mood and personal characteristics;
- correlation between sets of perceived and induced emotions depends both on the personal musical preferences, as well as on the user’s current mood;
- identify a subset of emotionally ambiguous music excerpts and study their characteristics;
- mappings between colors and music depend on the music genre;
- perceived emotions in a music excerpt are expected to be similar across listeners, while induced emotions are expected to be correlated across groups of songs and users with similar characteristics.

We outline all parts of the survey in the following subsections, a more detailed overview can be found in [18].

2.2.1 Part one – personal characteristics

The first part of the survey contains nine questions that capture personal characteristics of users. Basic demographics were captured: age, gender, area of living, native language. We also included questions regarding their music education, music listening and genre preferences. We decided not to introduce a larger set of personal questions, as the focus of our research lies in investigating the interplay of colors, music and emotions and we did not want to irritate the users with a lengthy first part. Our goal was to keep the amount of time spent for filling in the survey to under 10 minutes.

2.2.2 Part two - mood, emotions and colors

The second part of our survey was designed to capture information about the user’s current mood, their perception of relation between colors and emotions and their perception of emotions in terms of pleasantness and activeness.

The user’s emotional state was captured in several ways. First, users had to place a point in the valence-arousal space. This is a standard mood estimation approach, also frequently used for estimation of perceived emotions in music. Users also indicated the preferred color of their current emotional state, as well as marked the presence of a set of emotion labels by using the MoodStripe interface (see Figure 1).

Figure 1: The MoodStripe allows users to express their emotional state by dragging emotions onto a canvas, thereby denoting their activity.
To match colors with emotions, users had to pick a color in the color wheel that best matches a given emotion label (10 labels were presented to each user). Finally, users had to assess how they perceive the pleasantness and activeness of emotions by placing a set of emotion labels into the valence-arousal space using the MoodGraph (see Figure 2). This enables us to evaluate the variability of placement of emotion labels in terms of their activeness and pleasantness and compare it to data gathered in part three, where users described musical excerpts in a similar manner.

2.2.3 Part three - music in relation to colors and emotions

In the third part of our survey users were asked to complete two tasks on a set of ten 15-second long music excerpts. These were randomly selected from a database of 200 music excerpts. When compiling the database, we strived for a diverse, yet unknown set of music pieces, to avoid judgments based on familiarity with the content. The database contains 80 songs from the royalty free online music service Jamendo, representing a diverse variety of “standard” genres, with songs unknown to the wider audience. 80 songs were included from a dataset of film music excerpts [6], 20 from a database of folk music and 20 from a contemporary electro-acoustic music collection.

After listening to an excerpt, users were first asked to choose the color best representing the music from the color wheel. Next, users were asked to describe the music by dragging emotion labels onto the valence-arousal space using the MoodGraph interface (Figure 2). Two different sets of labels were used for describing induced and perceived emotions, as different emotions correspond with respective category [19], and at least one label from each category had to be placed onto the space. Shown and

![Figure 2: The MoodGraph: users drag emotion labels onto the valence-arousal space. Induced emotions are marked with a person icon, perceived emotions with a note icon.](image)

2.3 Evaluation survey

After responses were gathered, we performed an additional evaluation survey, where we asked participants to evaluate the original survey. Although the survey was anonymous, users had the opportunity to leave their email at the end, which we used to invite them to fill in the evaluation questionnaire. Participants were presented a set of twelve questions about different aspects of the survey: user experience, complexity of the questionnaire, and aspects of our new MoodGraph and MoodStripe interfaces. Some of the questions were drawn from the existing evaluation standard NASA load task index [20], while others were intended to evaluate different aspects of our interfaces.

3. RESULTS

The survey was taken by 952 users, providing 6609 mood/color-perception responses for the 200 music excerpts used. We thus obtained a large number of responses per music excerpt (each has 33 responses on average), including sets of induced and perceived emotion labels, their placement in the valence-arousal space, as well as the color describing the excerpt. To our knowledge, no currently available mood-music dataset has such a high ratio of user annotations per music excerpt. The data, as well as music excerpts will be made public as soon as the second round of response gathering, currently underway, will be finished.

In the following subsections, we provide some preliminary analyses of our data.

3.1 Demographic analysis

The basic demographic characteristics of the 952 participants are as follows. The average age of participants was 26.5 years, the youngest had 15, the oldest 64 years. 65% of participants are women, 66% are from urban areas. 50% have no music education, 47% do not play instruments or sing. The amount of music listening per day is evenly spread from less than 1 hour to over 4 hours. 3% claimed they were under the influence of drugs when taking the survey.

3.2 Colors and emotions

In the second part of the survey, participants indicated their emotional state within the valence-arousal space, as well as by choosing a color. Relations between the color hue and location in the valence-arousal space are not very consistent, but overall less active emotional states correspond more with darker blue-violet hues, while the more active ones to red-yellow-green hues. There is also a statistically significant positive correlation between color saturation and value (in a HSV color model) and activeness, as well as pleasantness of emotions: the more positive and active the user’s emotional state is, the more vivid the colors are.

Colors attributed to individual emotion labels, as well as the placement of labels in the valence-arousal space are visible in Figure 3. Associations between colors and emotions are quite consistent and in line with previous research [21-24]. Fear (A) and anger (F) are basic negative emotions and have dark blue/violet or black hues. Sadness (I)
and relaxation (J), interestingly are also very similar, although different in valence. Energetic (C) as a very active mood is mostly red, joy (B) and liveliness (G) somewhat less (more yellowish, even green). Another interesting outcome is that similar red-yellow-green hues are also prevalent for disappointment (E) and discontent (H). Happiness (D) is very distinct, in pastels of green and blue (similar to [21-24]). As these hues are often related to inner balance (peace), their choice for happiness, by some definitions a state where ones needs are satisfied, reflects the participants’ notion that happiness and inner balance are related[21, 24].

Figure 3: position of emotions in the valence-arousal space, and their colors. A: fear, B: joy, C: energy, D: happiness, E: disappointment, F: anger, G: liveliness, H: discontent, I: relaxation, J: sadness

3.3 Relationships between induced and perceived emotions

In part three of the survey participants were asked to mark induced and perceived emotions for individual music excerpt by dragging emotion labels from the respective categories onto the valence-arousal space (see Figure 2). Here, we focus on the relationship between induced and perceived emotions.

Figure 4 shows the centroids (averages) for induced-perceived emotion pairs of participants’ ratings for each music excerpt: ‘anger’, ‘relaxed’, ‘happiness’, ‘joy’, ‘sadness’, ‘calmness’, ‘anticipation’ and ‘fear’. Positions of induced-perceived emotion pairs (Figure 4) loosely correspond to the positions of participant’s emotional states in the valence-arousal space from Figure 3, with some obvious differences. For example (with respect to B, D and I on Figure 3), positive induced-perceived emotion pairs, such as relaxed, happiness and joy (B, C and D in Figure 4) occupy a more central space in the ‘pleasant/active’ quadrant of valence-arousal space. Similarly, negative emotion pairs (A, E and H in Figure 4) are also more central on the ‘unpleasant’ quadrants than corresponding emotions on Figure 3, but have significantly larger variance and spread on valence-arousal space compared to positive emotions (apart from relaxed (B)), especially along arousal dimension.

Let us compare the relationships in Figure 4. There is a noticeable variance between induced and perceived emotions for negative emotions, such as fear (H), anger (A) and sadness (E), as they spread over both arousal and valence axes. The central position of sadness (E) along the arousal dimension is especially interesting, as it is typically associated with low arousal (compare to J in Figure 3). Furthermore, all three negative emotions (A, E and H) are in certain musical contexts experienced or perceived as pleasant. On the other hand, positive induced-perceived emotion pairs, such as joy (D) and happiness (C), tend to be more similar on both valence (positive) and arousal (relatively high) dimension and consequently have less variance. More neutral emotions, such as calmness (F) and anticipation (G), occupy the center, with relaxed (B) untypically potent on the arousal dimension.

Figure 4: Representation of relationships between induced-perceived emotion pairs of all music excerpts (induced centroid: green star, perceived centroid: red circle). A: anger, B. relaxation, C. happiness, D: joy, E: sadness, F: calmness, G: anticipation, H: fear

Discriminating between induced and perceived emotions in music is a complex task and to date there is no universally agreed upon theory, or emotional model, that would best capture emotional experiences of listeners (see e.g. [19, 25-29]). Many argue (e.g. [6, 19, 28, 30, 31]) that simple valence-arousal dimensional model (one that MoodGraph is based on) might be too reductionist, as it ignores the variance of emotions and results in inherently different emotions occupying similar regions of valence-arousal space (e.g., compare regions of fear (H), anger (A) and sadness (E) in Figure 4). Our preliminary results nevertheless show some interesting aspects of induction and perception of musical emotions. For example, the representations of relationships among and within induced-perceived emotion pairs shown in Figure 4 support Gabrielsson’s theory of four basic types of relationship between induced and perceived emotions in relation to music: positive/in agreement, negative/opposite, non-systematic/neutral and absent/no relationship [25]. Positive relationship is the most common (e.g., when music perceived to express sad emotions also evokes such emotions in the listener), resulting in the overlap (in some cases above 60%; see e.g. [19, 26, 29]) of induced-perceived emotion pairs. In one study [32], researchers found extremely strong positive correlation for induced and perceived emotions on both valence and arousal dimensions, and concluded that results show “listeners will typically feel the emotions ex-
pressed by the song” [p. 93]. However, our preliminary results do not support this claim. There is a significant variance among induced-perceived emotion pairs, particularly among negative emotions. Furthermore, while effects of positive correlation between induced and perceived emotions are evident (especially in positive emotions), other types of relationships are equally significant: from negative/opposite, non-matching, to complex and neutral. The preliminary results clearly show differential variance across induced and perceived emotions (in line with recent findings [33]).

When analyzing the induced-perceived emotion pairs in MoodGraph, we’ve found that: a) they do not necessarily positively correlate, b) they occupy different regions and c) even when they fall into the same region of valence-arousal space, both rotation and standard deviation within each induced-perceived emotion pair are significantly larger than reported in some of the previous studies (e.g., [32]). This shows that participants understood both concepts (i.e. induced vs. perceived emotion) and were able to differentiate emotions from both categories on the valence-arousal space.

One reason for large amount of variance in representations of induced/perceived pairs is probably due to the model itself, as participants can rate both induced and perceived emotions together and directly onto MoodGraph after listening to the music excerpt. Another advantage, we argue, is the construction of the MoodGraph itself. While bearing similarity with traditional approach to dimensional modeling (a classic example being Russell’s circumplex model of affect [15]), the MoodGraph has no pre-defined and categorically segmented/discrete regions of valence-arousal space, hence avoiding initial bias, while still offering an intuitive interface – the participant is free to drag emotion labels onto MoodGraph according to her preferences and interpretation of the valence-arousal space.

### 3.4 Evaluation Survey

The online evaluation questionnaire was filled-in by 125 users, who all took part in our survey. Results were positive and indicate that the survey was properly balanced and the new interfaces were appropriate. Detailed results can be found in [34]. To summarize, responses show appropriate mental difficulty of the questionnaire, while the physical difficulty seems to be more uniformly distributed across participants. Thus, it can be speculated that the listening part of the questionnaire represents a physical challenge to a significant number of participants. The presented MoodGraph interface was quite intuitive; however, it was also time demanding. Considering the task load of the interface (combining three distinctive tasks), this was expected. The number of emotions in MoodGraph categories was slightly unbalanced and should be extended in our future work. The MoodStripe interface represents a significant improvement over a group of radio buttons, both in intuitiveness and time complexity. Participants also indicated that the set of 49 colors available for labeling emotions may not be large enough, so we will consider enlarging the set of color tones in our future work.

### 4. Conclusions

We intend to make the gathered dataset available to the public, including the musical excerpts, data on users’ personal characteristics and emotional state, their placement of emotions within the valence/arousal space, their perceived and induced emotional responses to music and their perception of color in relation to emotions and music. This will open new possibilities for evaluating and re-evaluating mood estimation and music recommendation approaches on a well annotated dataset, where the ground truth lies in the statistically significant amount of responses per song, rather than relying on annotations of a small number of users.

Shortly, we will start with the second round of response gathering with an English version of the survey. We also intend to enlarge the number of music excerpts in the music dataset and provide it to the users who have already participated in this study. Thus, we hope to further extend and diversify the dataset.

Preliminary analyses already show new and interesting contributions, and next to answering the questions already posed in section 2.2, the dataset will provide grounds for our future work (and work of others), including:

- previously introduced mood estimation algorithms will be evaluated by weighting the correctness of their predictions of perceived emotion responses for music excerpts. New mood estimation algorithms will be developed, building upon the newly obtained data;
- we will explore modelling of relations between music and colors chosen by users in the survey. Results may be useful for music visualization, provided that correlations between audio and visual perception will be consistent enough;
- music recommendation interfaces will be explored, presenting recommendations in a visual manner with the intent to raise user satisfaction by reducing the textual burden placed on the user. The interface will include personal characteristics and their variability in the decision model;
- the dataset can also be used in other domains, as responses that relate colors to emotions based on the user’s emotional state can be used independently.

### 5. References


IN-DEPTH MOTIVIC ANALYSIS BASED ON MULTIPARAMETRIC CLOSED PATTERN AND CYCLIC SEQUENCE MINING

Olivier Lartillot
Aalborg University, Department of Architecture, Design and Media Technology, Denmark
olartillot@gmail.com

ABSTRACT

The paper describes a computational system for exhaustive but compact description of repeated motivic patterns in symbolic representations of music. The approach follows a method based on closed heterogeneous pattern mining in multiparametrical space with control of pattern cyclicity. This paper presents a much simpler description and justification of this general strategy, as well as significant simplifications of the model, in particular concerning the management of pattern cyclicity. A new method for automated bundling of patterns belonging to same motivic or thematic classes is also presented.

The good performance of the method is shown through the analysis of a piece from the JKUPDD database. Ground-truth motives are detected, while additional relevant information completes the ground-truth musicological analysis. The system, implemented in Matlab, is made publicly available as part of MiningSuite, a new open-source framework for audio and music analysis.

1. INTRODUCTION

The detection of repetitions of sequential representations in symbolic music is a problem of high importance in music analysis. It enables the detection of repeated motifs and themes, and of structural repetition of musical passages.

1.1 Limitation of previous approaches

Finding these patterns without knowing in advance their actual description is a difficult problem. Previous approaches have shown the difficulty of the problem related to the combinatorial explosion of possible candidate patterns [2]. Some approaches tackle this issue by generating a large set of candidate patterns and applying simple global heuristics, such as finding longest or most frequent patterns [3,8]. Similarly, other approaches base the search for patterns on general statistical characteristics [5]. The problem is that there is no guarantee that this global filtering leads to a selection of patterns corresponding to those selected by musicologists and perceived by listeners.

1.2 Exhaustive mining of closed and cyclic patterns

In our research, we endeavour to reveal the factors underlying this structural explosion of possible patterns and to formalise heuristics describing how listeners are able to consensually perceive clear pattern structures out of this apparent maze. We found that pattern redundancy is based on two core issues [6]:

- closed pattern mining: When a pattern is repeated, all underlying pattern representations it encompasses are repeated as well. In simple string representation, studied in section 2, these more general patterns correspond to prefixes, suffixes and prefixes of suffixes. The proliferation of general patterns, as shown in Figure 1, leads to combinatorial explosion. Restricting the search to the most specific (or "maximal") patterns is excessively selective as it filters out potentially interesting patterns (such as CDE in Figure 1), and would solely focus on large sequence repetitions. By restricting the search to closed patterns – i.e., patterns that have more occurrences than their more specific patterns –, all pattern redundancy is filtered out without loss of information. [6] introduces a method for exhaustive closed pattern mining.

- pattern cyclicity: When repetitions of a pattern are immediately successive, another combinatorial set of possible sequential repetitions can be logically inferred [2], as shown in Figure 2. This redundancy can be avoided by explicitly modelling the cyclic loop in the pattern representation, and by generalising the notion of closed pattern accordingly.

By carefully controlling these factors of combinatorial redundancy without damaging the non-redundant pattern information, the proposed approach in [6] enables to output an exhaustive description of pattern repetitions. Previous approaches did not consider those issues and performed instead global filtering techniques that broadly miss the rich pattern structure.
1.3 New approach

In this paper, we propose a simplified description and modelling of this exhaustive pattern mining approach. In section 2, we present the problem of closed pattern mining on the simple case of monoparametric string analysis, introduce a simplified algorithmic implementation, and present a new way to simply justify the interest of the approach. In section 3, the approach is generalised to the multidimensionality of the musical parametric space. Section 4 discusses pattern cyclicity and presents a new simple model that solves this issue. In section 5, the interest of the method is shown through the analysis of a piece of music from the JKUPDD database.

2. CORE PRINCIPLES OF THE MODEL

2.1 Advantages of incremental one-pass approach

As explained in the previous section, testing the closedness of a pattern requires comparing its number of occurrences with those of all the more specific patterns. Previous computer science researches in closed pattern mining (one recent being [9]) incrementally construct the closed patterns dictionary while considering the whole document to be analysed (in our case, the piece of music). This requires the design of complex algorithms to estimate the number of occurrences of each possible pattern candidate.

We introduced in [6] a simpler approach based on an incremental single pass throughout the document (i.e., from the beginning to the end of the piece of music), during which the closed pattern dictionary is incrementally constructed: for each successive note $n$ in the sequence, all patterns in the subsequence ending to that note $n$ are exhaustively searched for. The main advantage of the incremental approach is based on the following property.

**Lemma 2.1** (Closed pattern characterisation). *When following the incremental approach, for any closed pattern $P$, there exists a particular moment in the piece of music where an occurrence $O$ of $P$ can be inferred while no occurrence of any more specific pattern can be inferred.*

**Proof.** There are three alternative conditions concerning the patterns more specific than $P$:

- There is no pattern more specific than $P$. In this case, the observation is evident.
- There is only one pattern $S$ more specific than $P$. For instance, in Figure 3, $S = ABCD$ is more specific than $P = CD$. Since $P$ is closed, it has more occurrences than $S$, so there exists an occurrence of $P$ that is not occurrence of $S$.
- There are several patterns $S_1, \ldots, S_n$ more specific than $P$. For instance, in Figure 1, $S_1 = ABCDE$ and $S_2 = ABCDE$ are both more specific than $P = CDE$. As soon as two different more specific patterns $S_1$ (one or several time) and $S_2$ (first time) have appeared in the sequence, pattern $P$ can be detected, since it is repeated in $S_1$ and $S_2$, but $S_2$ is not detected yet, since it has not been repeated yet.

As soon as we detect a new pattern repetition, such that for that particular occurrence where the repetition is detected, there is no more specific pattern repetition, we can be sure that the discovered pattern is closed.

When considering a given pattern candidate at a given point in the piece of music, we need to be already informed about the eventual existence of more specific pattern occurrences at the same place. Hence, for a given note, patterns need to be extended in decreasing order of specificity.

To details further the approach, let’s consider in a first simple case the monoparametric contiguous string case, where the main document is a sequence of symbols, and where pattern occurrences are made of contiguous sub-strings. In this case, ‘more general than’ simple means ‘is a subsequence of’. In other words, a more general pattern is a prefix or a suffix of a more specific pattern. Let’s consider these two aspects separately:

- Since the approach is incremental, patterns are constructed by incrementally extending their prefixes (in grey in Figure 1). Patterns are therefore represented as chains of prefixes, and the pattern dictionary is represented as a prefix tree. In this paradigm, if a given pattern $P$ is a prefix of a closed pattern $S$, and if both have same number of occurrences, the prefix $P$ can still be considered as a closed pattern, in the sense that it is an intermediary state to the constitution of the closed pattern $S$.
The closedness of a pattern depends hence solely on the patterns to which it is a suffix. Thanks to the incremental one-pass approach, these more specific patterns are already inferred. The only constraint to be added is that when a given note is considered, the candidate patterns should be considered in decreasing order of specificity, i.e. from the longest to the shortest (which are suffixes of the longer ones). For instance, in Figure 3, when analysing the last note, E, there are two candidate patterns for extension, ABCD and CD. Since we first extend the most specific pattern ABCDE, when considering then the more general pattern CD, extension CDE is found as non-closed and thus not inferred.

2.2 Algorithmic details

Following these principles, the main routine of the algorithm simply scans the musical sequence chronologically, from the first to the last note. Integrating a new note consists in checking:

- whether pattern occurrence(s) ending at the previous note can be extended with the new note,
- whether the new note initiates the start of a new pattern occurrence.

The extension of a pattern occurrence results from two alternative mechanisms:

**Recognition** the new note is recognised as a known extension of the pattern.

**Discovery** the new note continues the occurrence in the same way that a previous note continued an older occurrence of the pattern: the pattern is extended with this new common description, and the two occurrences are extended as well.

Concerning the discovery mechanism, the identification of new notes continuing older contexts can be implemented using a simple associative array, storing the note following each occurrence according to its description. This will be called a *continuation memory*. Before actually extending the pattern, we should make sure that the extended pattern is closed.

2.3 Specific Pattern Class

Searching for all closed patterns in a sequence, instead of all possible patterns, enables an exhaustive pattern analysis without combinatorial explosion: all non-closed patterns can be deduced from the closed pattern analysis. Yet, the set of closed patterns can remain quite large and the exhaustive collection of their occurrences can become cumbersome. [6] proposes to limit the analysis, without any loss of information, to closed patterns' specific classes, which correspond to pattern occurrences that are not included in occurrences of more specific patterns. For instance, in Figure 3, the specific class of CD contains only its first occurrence, because the two other ones are superposed to occurrences of the more specific pattern ABCDE.

We propose a simpler model for the determination of specific class of closed patterns. Non-specific occurrences are regenerated whenever necessary. Because occurrences of a given pattern are not all represented, the notes following these occurrences are not memorised, although they could generate new pattern extensions. To circumvent this issue, the extension memory related to any given pattern contains the extensions not only of that pattern but also of any more specific pattern.

3. MULTIPARAMETRIC PATTERN MINING

The model presented in the previous section searches for sequential patterns on monoparametric sequences, composed of a succession of symbols taken from a given alphabet. Music cannot be reduced to unidimensional parametric description.

3.1 Parametric space

The problem needs to be generalised by taking into account three main aspects:

- Notes are defined by a hierarchically structured combination of parameters (diatonic and chromatic pitch and pitch class, metrical position, etc.).
- Notes are defined not only in terms of their absolute position on fixed scales, but also relatively to a given local context, and in particular with respect to the previous notes (defining pitch interval, gross contour, rhythmic values, etc.). These interval representations are also hierarchically structured. Gross contour, for instance, is a simple description of the inter-pitch interval between successive notes as “increasing”, “decreasing” or “unison”. Matching along gross contour enables to track intervallic augmentation and diminution. For instance, in the example in section 5, the first interval of the fugue subject is either a decreasing third or a decreasing second. The actual diatonic pitch interval representation differs, but the gross contour remains constantly “decreasing”.
- A large part of melodic transformations can be understood as repetitions of sequential patterns that do not follow strictly all the parametric descriptions, but only a subset. For instance, a rhythmical variation of a melodic motif consists in repeating the pitch sequence, while developing the rhythmical part more freely.
to their corresponding more specific fields. Methods have been implemented that enable to compare two parametric descriptions, in order to see if they are equal, or if one is subsumed into the other, and if not, to compute the intersection of the two descriptions.

The multiparametric description is integrated in the two core mechanisms of the incremental pattern mining model as follows:

**Recognition** As before, the observed parametric description of the new note is compared to the descriptions of the patterns’ extensions. If the pattern extension’s description fits only partially, a new more general pattern extension is created (if not existing yet) related to the common description.

**Discovery** The continuation memory is structured in the same way as the parametric space: for each possible parametric field, an associative memory stores pattern continuations according to their values along that particular parametric field. As soon as a stored pattern continuation is identified with the current note along a particular parametric field, the complete parametric description common to these two contexts is computed, and the pattern extension is attempted along that common parametric description. As before, a pattern is extended only if the extended pattern is closed.

### 4. PATTERN CYCLICITY

A solution to the problem of cyclicity introduced in section 1.2 was proposed in [6] through the formalisation of cyclic patterns, where the last state of the chain representing the pattern is connected back to its first state, formalising this compelling expectation of the return of the periodic pattern. One limitation of the approach is that it required the explicit construction of cyclic pattern, which demanded contrived algorithmic formalisations. The problem gets even more difficult when dealing with multiparametric space, in particular when the pattern is only partially extended, i.e., when the expected parametric description is replaced by a less specific parametric matching, such as in the musical example shown in Figure 4. In this case, a more general pattern cyclic needs to be constructed, leading to the inference of a complex network of pattern cycles particularly difficult to conceptualise and implement.

We propose a simpler approach: instead of formalising cyclic patterns, pattern cyclicity is represented on the pattern occurrences directly. Once a successive repetition of a pattern has been detected, such as the 3-note pattern starting the musical example in Figure 4, the two occurrences are fused into one single chain of notes, and all the subsequent notes in the cyclic sequence are progressively added to that chain. This cyclic chain is first used to track the development of the new cycle (i.e., the third cycle, since there were already two cycles). The tracking of each new cycle is guided by a model describing the expected sequence of musical parameters. Initially, for the third cycle, this
The analysis offered by the computational model offers much richer information than simply listing the occurrences of the subjects and countersubjects. It shows what musical descriptions characterise them, and details particular commonalities shared by occurrences of these subjects and countersubjects. For instance entries M1 and U1 belong to a same more specific pattern that describes their particular development. L1, U1 and U3 start all with a decreasing third interval, and so on.

The model presented in this paper does not yet integrate mechanisms for the reduction of ornamentation, as discussed in the next section. The only melodic ornamentation appearing in pattern #1 is the addition of a passing note after the first note of occurrences L2 and L3. This leads to a small error in the model’s results, where the first actual note is not detected.

The thematic class related to ground-truth pattern #2, which is the first countersubject, is extracted in the same way, forming a paradigmatic sheaf. The pattern class given by the model corresponds mostly to the ground truth. Here again, some occurrences present similar extensions that are inventoried by the model, although they are ignored in the ground truth. The last occurrence, which is a suffix of the pattern, is also detected accordingly. On the other hand, the second last occurrence is not properly detected, once again due to the addition of passing notes.

Pattern #3, which is the second countersubject, is more problematic, because it is only 7 notes long. Several other longer patterns are found by the model, and the specificity of pattern #3 is not grounded on characteristics purely re-
lated to pattern repetition. As aforementioned, the ground-truth selection of these three patterns are based on principles related to fugue rules, namely the synchronisation of the three patterns along the separate voices. It seems questionable to expect a general pattern mining algorithm non-specialised to a particular type of music to be able to infer this type of configuration.

6. CONCLUSION
The approach is incremental, progressively analysing the musical sequence through one single pass. This enables control of the structural complexity in a way similar to the way listeners perceive music.

Gross contour needs to be constrained by factors related to local saliency and short-term memory. The integration of more complex melodic transformation such as ornamentation and reduction is currently under investigation. Motivic repetition with local ornamentation is detected by reconstructing, on top of “surface-level” monodic voices, longer-term relations between non-adjacent notes related to deeper structures, and by tracking motives on the resulting syntagmatic network. More generally, the analysis of polyphony is under study, as well as the application of the pattern mining approach to metrical analysis. The system, implemented in Matlab, is made publicly available as part of MiningSuite, a new open-source framework for audio and music analysis.

7. ACKNOWLEDGMENTS
This work was funded by an Academy of Finland research fellowship at the Finnish Centre of Excellence in Interdisciplinary Music Research at the University of Jyväskylä. The research is continued in the context of the European project Learning to Create (Lrn2Cre8), which acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET grant number 610859.

8. REFERENCES

---

Available at http://code.google.com/p/miningsuite/.
mir_eval:  
A TRANSPARENT IMPLEMENTATION OF COMMON MIR METRICS

Colin Raffel1,*, Brian McFee1,2, Eric J. Humphrey3, Justin Salamon1,4, Oriol Nieto3, Dawen Liang1, and Daniel P. W. Ellis1

1LabROSA, Dept. of Electrical Engineering, Columbia University, New York  
2Center for Jazz Studies, Columbia University, New York  
3Music and Audio Research Lab, New York University, New York  
4Center for Urban Science and Progress, New York University, New York

ABSTRACT

Central to the field of MIR research is the evaluation of algorithms used to extract information from music data. We present mir_eval, an open source software library which provides a transparent and easy-to-use implementation of the most common metrics used to measure the performance of MIR algorithms. In this paper, we enumerate the metrics implemented by mir_eval and quantitatively compare each to existing implementations. When the scores reported by mir_eval differ substantially from the reference, we detail the differences in implementation. We also provide a brief overview of mir_eval’s architecture, design, and intended use.

1. EVALUATING MIR ALGORITHMS

Much of the research in Music Information Retrieval (MIR) involves the development of systems that process raw music data to produce semantic information. The goal of these systems is frequently defined as attempting to duplicate the performance of a human listener given the same task [5]. A natural way to determine a system’s effectiveness might be for a human to study the output produced by the system and judge its correctness. However, this would yield only subjective ratings, and would also be extremely time-consuming when evaluating a system’s output over a large corpus of music.

Instead, objective metrics are developed to provide a well-defined way of computing a score which indicates each system’s output’s correctness. These metrics typically involve a heuristically-motivated comparison of the system’s output to a reference which is known to be correct. Over time, certain metrics have become standard for each task, so that the performance of systems created by different researchers can be compared when they are evaluated over the same dataset [5]. Unfortunately, this comparison can be confounded by small details of the implementations or procedures that can have disproportionate impacts on the resulting scores.

For the past 10 years, the yearly Music Information Retrieval Evaluation eXchange (MIREX) has been a forum for comparing MIR algorithms over common datasets [6]. By providing a standardized shared-task setting, MIREX has become critically useful for tracking progress in MIR research. MIREX is built upon the Networked Environment for Music Analysis (NEMA) [22], a large-scale system which includes exhaustive functionality for evaluating, summarizing, and displaying evaluation results. The NEMA codebase includes multiple programming languages and dependencies (some of which, e.g. Matlab, are proprietary) so compiling and running it at individual sites is nontrivial. In consequence, the NEMA system is rarely used for evaluating MIR algorithms outside of the setting of MIREX [6]. Instead, researchers often create their own implementations of common metrics for evaluating their algorithms. These implementations are thus not standardized, and may contain differences in details, or even bugs, that confound comparisons.

These factors motivate the development of a standardized software package which implements the most common metrics used to evaluate MIR systems. Such a package should be straightforward to use and well-documented so that it can be easily adopted by MIR researchers. In addition, it should be community-developed and transparently implemented so that all design decisions are easily understood and open to discussion and improvement.

Following these criteria, we present mir_eval, a software package which intends to provide an easy and standardized way to evaluate MIR systems. This paper first discusses the architecture and design of mir_eval in Section 2, then, in Section 3, describes all of the tasks covered by mir_eval and the metrics included. In order to validate our implementation decisions, we compare mir_eval to existing software in Section 4. Finally, we discuss and summarize our contributions in Section 5.
2. **mir_eval’s Architecture**

mir_eval is a Python library which currently includes metrics for the following tasks: Beat detection, chord estimation, pattern discovery, structural segmentation, melody extraction, and onset detection. Each task is given its own submodule, and each metric is defined as a separate function in each submodule. Each task submodule also includes common data pre-processing steps for the task. Every metric function includes detailed documentation, example usage, input validation, and references to the original paper which defined the metric. mir_eval also includes a submodule io which provides convenience functions for loading in task-specific data from common file formats (e.g. comma/tab separated values, .lab files [7], etc.). For readability, all code follows the PEP8 style guide [21]. mir_eval’s only dependencies outside of the Python standard library are the free and open-source SciPy/Numpy [9] and scikit-learn [15] libraries.

In order to simplify the usage of mir_eval, it is packaged with a set of “evaluator” scripts, one for each task. These scripts include all code necessary to load in data, pre-process it, and compute all metrics for a given task. The evaluators allow for mir_eval to be called directly from the command line so that no knowledge of Python is necessary. They are also distributed as executables for Windows and Mac OS X, so that mir_eval may be used with no dependencies installed.

3. **Tasks Included in mir_eval**

In this section, we enumerate the tasks and metrics implemented in mir_eval. Due to space constraints, we only give high-level descriptions for each metric; for exact definitions see the references provided.

3.1 **Beat Detection**

The aim of a beat detection algorithm is to report the times at which a typical human listener might tap their foot to a piece of music. As a result, most metrics for evaluating the performance of beat tracking systems involve computing the error between the estimated beat times and some reference list of beat locations. Many metrics additionally compare the beat sequences at different metric levels in order to deal with the ambiguity of tempo [4].

mir_eval includes the following metrics for beat tracking, which are defined in detail in [4]: The F-measure of the beat sequence, where an estimated beat is considered correct if it is sufficiently close to a reference beat; Cemgil’s score, which computes the sum of Gaussian errors for each beat; Goto’s score, a binary score which is 1 when at least 25% of the estimated beat sequence closely matches the reference beat sequence; McKinney’s P-score, which computes the cross-correlation of the estimated and reference beat sequences represented as impulse trains; continuity-based scores which compute the proportion of the beat sequence which is continuously correct; and finally the Information Gain of a normalized beat error histogram over a uniform distribution.

3.2 **Chord Estimation**

Despite being one of the oldest MIREX tasks, evaluation methodology and metrics for automatic chord estimation is an ongoing topic of discussion, due to issues with vocabularies, comparison semantics, and other lexicographical challenges unique to the task [14]. One source of difficulty stems from an inherent subjectivity in “spelling” a chord name and the level of detail a human observer can provide in a reference annotation [12]. As a result, a consensus has yet to be reached regarding the single best approach to comparing two sequences of chord labels, and instead are often compared over a set of rules, i.e. Root, Major-Minor, and Sevenths, with or without inversions.

To efficiently compare chords, we first separate a given chord label into its constituent parts, based on the syntax of [7]. For example, the chord label G:maj(6)/5 is mapped to three pieces of information: the root (“G”), the root-invariant active semitones as determined by the quality shorthand (“maj”) and scale degrees (“6”), and the bass interval (“5”).

Based on this representation, we can compare an estimated chord label with a reference by the following rules as used in MIREX 2013 [2]: Root requires only that the roots are equivalent; Major-Minor includes Root, and further requires that the active semitones are equivalent subject to the reference chord quality being Maj or min; Sevenths follows Major-minor, but is instead subject to the reference chord quality being one of Maj, min, Maj7, min7, 7, or minmaj7; and finally, Major-Minor-Inv and Sevenths-Inv include Major-Minor and Sevenths respectively, but further require that the bass intervals are equivalent subject to the reference bass interval being an active semitone. The “subject to...” conditions above indicate that a comparison is ignored during evaluation if the given criteria is not satisfied.

Track-wise scores are computed by weighting each comparison by the duration of its interval, over all intervals in an audio file. This is achieved by forming the union of the boundaries in each sequence, sampling the labels, and summing the time intervals of the “correct” ranges. The cumulative score, referred to as weighted chord symbol recall, is tallied over a set audio files by discrete summation, where the importance of each score is weighted by the duration of each annotation [2].

3.3 **Pattern Discovery**

Pattern discovery involves the identification of musical patterns (i.e. short fragments or melodic ideas that repeat at least twice) both from audio and symbolic representations. The metrics used to evaluation pattern discovery systems attempt to quantify the ability of the algorithm to not only determine the present patterns in a piece, but also to find all of their occurrences.

Collins compiled all previously existent metrics and proposed novel ones [3] which resulted in 19 different scores, each one implemented in mir_eval: Standard F-measure, Precision, and Recall, where an estimated prototype pattern is considered correct only if it matches...
(up to translation) a reference prototype pattern; Establishment F-measure, Precision, and Recall, which compute the number of reference patterns that were successfully found, no matter how many occurrences were found; Occurrence F-measure, Precision, and Recall, which measure whether an algorithm is able to retrieve all occurrences of a pattern; Three-layer F-measure, Precision, and Recall, which capture both the establishment of the patterns and the occurrence retrieval in a single set of scores; and the First N patterns metrics, which compute the target proportion establishment recall and three-layer precision for the first N patterns only in order to measure the ability of the algorithm to sort the identified patterns based on their relevance.

3.4 Structural Segmentation

Evaluation criteria for structural segmentation fall into two categories: boundary annotation and structural annotation. Boundary annotation is the task of predicting the times at which structural changes occur, such as when a verse transitions to a refrain. Structural annotation is the task of assigning labels to detected segments. The estimated labels may be arbitrary strings — such as A, B, C, etc. — and they need not describe functional concepts. In both tasks, we assume that annotations express a partitioning of the track into intervals.

mir_eval implements the following boundary detection metrics: Boundary Detection Precision, Recall, and F-measure Scores where an estimated boundary is considered correct if it falls within a window around a reference boundary [20]; and Boundary Deviation which computes median absolute time difference from a reference boundary to its nearest estimated boundary, and vice versa [20]. The following structure annotation metrics are also included: Pairwise Classification Precision, Recall, and F-measure Scores for classifying pairs of sampled time instants as belonging to the same structural component [10]; Rand Index 1 which clusters reference and estimated annotations and compares them by the Rand Index [17]; and the Normalized Conditional Entropy where sampled reference and estimated labels are interpreted as samples of random variables $Y_R, Y_E$ from which the conditional entropy of $Y_R$ given $Y_E$ (Under-Segmentation) and $Y_E$ given $Y_R$ (Over-Segmentation) are estimated [11].

3.5 Melody Extraction

Melody extraction algorithms aim to produce a sequence of frequency values corresponding to the pitch of the dominant melody from a musical recording [19]. An estimated pitch series is evaluated against a reference by computing the following five measures defined in [19], first used in MIREX 2005 [16]: Voicing Recall Rate which computes the proportion of frames labeled as melody frames in the reference that are considered correct (i.e., within half a semitone of the reference frequency); Raw Chroma Accuracy where the estimated and reference $f_0$ sequences are mapped onto a single octave before computing the raw pitch accuracy; and the Overall Accuracy, which computes the proportion of all frames correctly estimated by the algorithm, including whether non-melody frames where labeled by the algorithm as non-melody. Prior to computing these metrics, both the estimate and reference sequences must be sampled onto the same time base.

3.6 Onset Detection

The goal of an onset detection algorithm is to automatically determine when notes are played in a piece of music. As is also done in beat tracking and segment boundary detection, the primary method used to evaluate onset detectors is to first determine which estimated onsets are “correct”, where correctness is defined as being within a small window of a reference onset [1]. From this, Precision, Recall, and F-measure scores are computed.

4. COMPARISON TO EXISTING IMPLEMENTATIONS

In order to validate the design choices made in mir_eval, it is useful to compare the scores it reports to those reported by an existing evaluation system. Beyond pinpointing intentional differences in implementation, this process can also help find and fix bugs in either mir_eval or the system it is being compared to.

For each task covered by mir_eval, we obtained a collection of reference and estimated annotations and computed or obtained a score for each metric using mir_eval and the evaluation system being compared to. In order to facilitate comparison, we ensured that all parameters and pre-processing used by mir_eval were equivalent to the reference system unless otherwise explicitly noted. Then, for each reported score, we computed the relative change between the scores as their absolute difference divided by their mean, or

$$\frac{|s_m - s_e|}{(s_m + s_e)/2}$$

where $s_m$ is the score reported by mir_eval and $s_e$ is the score being compared to. Finally, we computed the average relative change across all examples in the obtained dataset for each score.

For the beat detection, chord estimation, structural segmentation, and onset detection tasks, MIREX releases the output of submitted algorithms, the ground truth annotations, and the reported score for each example in each data set. We therefore can directly compare mir_eval to MIREX for these tasks by collecting all reference and estimated annotations, computing each metric for each example using identical pre-processing and parameters as appropriate, and comparing the result to the score reported by

---

1 The MIREX results page refers to Rand Index as “random clustering index”.

---

15th International Society for Music Information Retrieval Conference (ISMIR 2014)
MIREX. We chose to compare against the results reported in MIREX 2013 for all tasks.

In contrast to the tasks listed above, MIREX does not release ground truth annotations or algorithm output for the melody extraction and pattern discovery tasks. As a result, we compared mir_eval’s output on smaller development datasets for these tasks. For melody extraction, the ADC2004 dataset used by MIREX is publicly available. We performed melody extraction using the “SG2” algorithm evaluated in 2011 [18] and compared mir_eval’s reported scores to those of MIREX. For pattern discovery, we used the development dataset released by Collins [3] and used the algorithms submitted by Nieto and Farbood [13] for MIREX 2013 to produce estimated patterns. We evaluated the estimated patterns using the MATLAB code released by Collins [3]. The number of algorithms, examples, and total number of scores for all tasks are summarized in Table 1.

<table>
<thead>
<tr>
<th>Task</th>
<th>Algorithms</th>
<th>Examples</th>
<th>Scores</th>
</tr>
</thead>
<tbody>
<tr>
<td>Beat Detection</td>
<td>20</td>
<td>679</td>
<td>13580</td>
</tr>
<tr>
<td>Segmentation</td>
<td>8</td>
<td>1397</td>
<td>11176</td>
</tr>
<tr>
<td>Onset Detection</td>
<td>11</td>
<td>85</td>
<td>935</td>
</tr>
<tr>
<td>Chord Estimation</td>
<td>12</td>
<td>217</td>
<td>2604</td>
</tr>
<tr>
<td>Melody</td>
<td>1</td>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>Pattern Discovery</td>
<td>4</td>
<td>5</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 1. Number of scores collected for each task for comparison against mir_eval.

The resulting average relative change for each metric is presented in Table 2. The average relative change for all of the pattern discovery metrics was 0, so they are not included in this table. For many of the other metrics, the average relative change was less than a few tenths of a percent, indicating that mir_eval is equivalent up to rounding/precision errors. In the following sections, we enumerate the known implementation differences which account for the larger average relative changes.

4.1 Non-greedy matching of events

In the computation of the F-measure, Precision and Recall metrics for the beat tracking, boundary detection, and onset detection tasks, an estimated event is considered correct (a “hit”) if it falls within a small window of a reference event. No estimated event is counted as a hit for more than one reference event, and vice versa. In MIREX, this assignment is done in a greedy fashion, however in mir_eval we use an optimal matching strategy. This is accomplished by computing a maximum bipartite matching between the estimated events and the reference events (subject to the window constraint) using the Hopcroft-Karp algorithm [8]. This explains the observed discrepancy between mir_eval and MIREX for each of these metrics. In all cases where the metric differs, mir_eval reports a higher score, indicating that the greedy matching strategy was suboptimal.

4.2 McKinney’s P-score

When computing McKinney’s P-score [4], the beat sequences are first converted to impulse trains sampled at a 10 millisecond resolution. Because this sampling involves quantizing the beat times, shifting both beat sequences by a constant offset can result in substantial changes in the P-score. As a result, in mir_eval, we normalize the beat sequences by subtracting from each reference and estimated beat location the minimum beat location in either series. In this way, the smallest beat in the estimated and reference beat sequences is always 0 and the metric remains the same even when both beat sequences have a constant offset applied. This is not done in MIREX (which uses the Beat Evaluation Toolbox [4]), and as a result, we observe a considerable average relative change for the P-score metric.

4.3 Information Gain

The Information Gain metric [4] involves the computation of a histogram of the per-beat errors. The Beat Evaluation Toolbox (and therefore MIREX) uses a non-uniform histogram binning where the first, second and last bins are smaller than the rest of the bins while mir_eval uses a standard uniformly-binned histogram. As a result, the Information Gain score reported by mir_eval differs substantially from that reported by MIREX.

4.4 Segment Boundary Deviation

When computing the median of the absolute time differences for the boundary deviation metrics, there are often an even number of reference or estimated segment boundaries, resulting in an even number of differences to compute the median over. In this case, there is no “middle” element to choose as the median. mir_eval follows the typical convention of computing the mean of the two middle elements in lieu of the median for even-length sequences, while MIREX chooses the larger of the two middle elements. This accounts for the discrepancy in the reference-to-estimated and estimated-to-reference boundary deviation metrics.

4.5 Interval Sampling for Structure Metrics

When computing the structure annotation metrics (Pairwise Precision, Recall, and F-measure, Rand Index, and Normalized Conditional Entropy Over- and Under-Segmentation Scores), the reference and estimated labels must be sampled to a common time base. In MIREX, a fixed sampling grid is used for the Rand Index and pairwise classification metrics, but a different sampling rate is used for each, while a fixed number of samples is used for the conditional entropy scores. In mir_eval, the same fixed sampling rate of 100 milliseconds is used for all structure annotation metrics, as specified in [23].

Furthermore, in MIREX the start and end time over which the intervals are sampled depends on both the reference and estimated intervals while mir_eval always samples with respect to the reference to ensure fair comparison across multiple estimates. This additionally requires
that estimated intervals are adjusted to span the exact duration specified by the reference intervals. This is done by adding synthetic intervals when the estimated intervals do not span the reference intervals or otherwise trimming estimated intervals. These differences account for the average relative changes for the structure annotation metrics.

4.6 Segment Normalized Conditional Entropy

When adding intervals to the estimated annotation as described above, mir_eval ensures that the labels do not conflict with existing labels. This has the effect of changing the normalization constant in the Normalized Conditional Entropy scores. Furthermore, when there’s only one label, the Normalized Conditional Entropy scores are not well defined. MIREX assigns a score of 1 in this case; mir_eval assigns a score of 0. This results in a larger average change for these two metrics.

4.7 Melody Voicing False Alarm Rate

When a reference melody annotation contains no unvoiced frames, the Voicing False Alarm Rate is not well defined. MIREX assigns a score of 1 in this case, while mir_eval assigns a score of 0 because, intuitively, no reference unvoiced frames could be estimated, so no false alarms should be reported. In the data set over which the average relative change for the melody metrics was computed, one reference annotation contained no unvoiced frames. This discrepancy caused a large inflation of the average relative change reported for the Voicing False Alarm Rate due to the small number of examples in our dataset.

4.8 Weighted Chord Symbol Recall

The non-negligible average relative changes seen in the chord metrics are caused by two main sources of ambiguity. First, we find some chord labels in the MIREX reference annotations lack well-defined, i.e. singular, mappings into a comparison space. One such example is D:maj(*1)/#1.

While the quality shorthand indicates “major”, the asterisk implies the root is omitted and thus it is unclear whether D:maj(*1)/#1 is equivalent to D:maj1. Second, and more importantly, such chords are likely ignored during evaluation, and we are unable to replicate the exact exclusion logic used by MIREX. This has proven to be particularly difficult in the two inversion rules, and manifests in Table 2. For example, Bb:maj(9)/9 was not excluded from the MIREX evaluation, contradicting the description provided by the task specification [2]. This chord alone causes an observable difference between mir_eval and MIREX’s results.

5. TOWARDS TRANSPARENCY AND COMMUNITY INVOLVEMENT

The results in Section 4 clearly demonstrate that differences in implementation can lead to substantial differences in reported scores. This corroborates the need for transparency and community involvement in comparative evaluation. The primary motivation behind developing mir_eval is to establish an open-source, publicly refined implementation of the most common MIR metrics. By encouraging MIR researchers to use the same easily understandable evaluation codebase, we can ensure that different systems are being compared fairly.

While we have given thorough consideration to the design choices made in mir_eval, we recognize that standards change over time, new metrics are proposed each year, and that only a subset of MIR tasks are currently implemented in mir_eval. Towards this end, mir_eval is hosted on Github,2 which provides a straightforward way of proposing changes and additions to the codebase using the Pull Request feature. With active community participation, we believe that mir_eval can ensure that MIR research converges on a standard methodology for evaluation.

2http://github.com/craffel/mir_eval
6. ACKNOWLEDGEMENTS
The authors would like to thank Matthew McVicar for helpful advice on comparing chord labels and Tom Collins for sharing his MATLAB implementation to evaluate musical patterns. Support provided in part by The Andrew W. Mellon Foundation and the National Science Foundation, under grants IIS-0844654 and IIS-1117015.

7. REFERENCES


COMPUTATIONAL MODELING OF INDUCED EMOTION USING GEMS

Anna Aljanaki  
Utrecht University  
A.Aljanaki@uu.nl

Frans Wiering  
Utrecht University  
F.Wiering@uu.nl

Remco C. Veltkamp  
Utrecht University  
R.C.Veltkamp@uu.nl

ABSTRACT

Most researchers in the automatic music emotion recognition field focus on the two-dimensional valence and arousal model. This model though does not account for the whole diversity of emotions expressible through music. Moreover, in many cases it might be important to model induced (felt) emotion, rather than perceived emotion. In this paper we explore a multidimensional emotional space, the Geneva Emotional Music Scales (GEMS), which addresses these two issues. We collected the data for our study using a game with a purpose. We exploit a comprehensive set of features from several state-of-the-art toolboxes and propose a new set of harmonically motivated features. The performance of these feature sets is compared. Additionally, we use expert human annotations to explore the dependency between musicologically meaningful characteristics of music and emotional categories of GEMS, demonstrating the need for algorithms that can better approximate human perception.

1. INTRODUCTION

Most of the effort in automatic music emotion recognition (MER) is invested into modeling two dimensions of musical emotion: valence (positive vs. negative) and arousal (quiet vs. energetic) (V-A) [16]. Regardless of the popularity of V-A, the question of which model of musical emotion is best has not yet been solved. The difficulty is, on one hand, in creating a model that reflects the complexity and subtlety of the emotions that music can demonstrate, while on the other hand providing a linguistically unambiguous framework that is convenient to use to refer to such a complex non-verbal concept as musical emotion. Categorical models, possessing few (usually 4–6, but sometimes as many as 18) [16] classes are oversimplifying the problem, while V-A has been criticized for a lack of discerning capability, for instance in the case of fear and anger. Other pitfalls of V-A model are that it was not created specifically for music, and is especially unsuited to describe induced (felt) emotion, which might be important for some MER tasks, e.g. composing a playlist using emotional query and in any other cases when the music should create a certain emotion in listener. The relationship between induced and perceived emotion is not yet fully understood, but they are surely not equivalent — one may listen to angry music without feeling angry, but instead feel energetic and happy. It was demonstrated that some types of emotions (especially negative ones) are less likely to be induced by music, though music can express them [17].

In this paper we address the problem of modeling induced emotion by using GEMS. GEMS is a domain-specific categorical emotional model, developed by Zentner et al. [17] specifically for music. The model was derived via a three-stage collection and filtering of terms which are relevant to musical emotion, after which the model was verified in a music listening-context. Being based on emotional ontology which comes from listeners, it must be a more convenient tool to retrieve music than, for instance, points on a V-A plane. The full GEMS scale consists of 45 terms, with shorter versions of 25 and 9 terms. We used the 9-term version of GEMS (see Table 1) to collect data using a game with a purpose.

Emotion induced by music depends on many factors, some of which are external to music itself, such as cultural and personal associations, social listening context, the mood of the listener. Naturally, induced emotion is also highly subjective and varies a lot across listeners, depending on their musical taste and personality. In this paper we do not consider all these factors and will only deal with the question to which extent induced emotion can be modeled using acoustic features only. Such a scenario, when no input from the end-user (except for, maybe, genre preferences) is available, is plausible for a real-world application of a MER task. We employ four different feature sets: low-level features related to timbre and energy, extracted using OpenSmile,¹ and a more musically motivated feature set, containing high-level features, related to mode, rhythm, and harmony, from the MIRToolbox,² PsySound³ and SonicAnnotator.⁴ We also enhance the performance of the latter by designing new features that describe the harmonic content of music. As induced emotion is a highly subjective phenomenon, the performance of the model will be con?rmed by the amount of agreement between listeners which provide the ground-truth. As far as audio-based features are not perfect yet, we try to estimate this upper bound for our data by employing human experts, who an-

1. opensmile.sourceforge.net
2. jyu.fi/hum/laitokset/musiikkien/research/coe/materials/mirtoolbox
3. psysound.wikidot.com
4. isophonics.net/SonicAnnotator

© Anna Aljanaki, Frans Wiering, Remco C. Veltkamp.  
notate a subset of the data with ten musicological features.

**Contribution.** This paper explores computational approaches to modeling induced musical emotion and estimates the upper boundary for such a task, in case when no personal or contextual factors can be taken into account. It is also suggested that more than two dimensions are necessary to represent emotional adequately. New features for harmonic description of music are proposed.

2. RELATED WORK

Music emotion recognition is a young, but fast-developing field. Reviewing it in its entirety is out of scope of this paper. For such a review we are referring to [16]. In this section we will briefly summarize the commonly used methods and approaches that are relevant to this paper.

Automatic MER can be formulated both as a regression and classification problem, depending on the underlying emotional model. As such, the whole entirety of machine learning algorithms can be used for MER. In this paper we are employing Support Vector Regression (SVR), as it demonstrated good performance [7, 15] and can learn complex non-linear dependencies from the feature space. Below we describe several MER systems.

In [15], V-A is modeled with acoustic features (spectral contrast, DWCH and other low-level features from Marsyas and PsySound) using SVR, achieving performance of 0.76 for arousal and 0.53 for valence (in terms of Pearson’s r here and further). In [7], five dimensions (basic emotions) were modeled with a set of timbral, rhythmic and tonal features, using SVR. The performance varied from 0.59 to 0.69. In [5], pleasure, arousal and dominance were modeled with AdaBoost.RM using features extracted from audio, MIDI and lyrics. An approach based on audio features only performed worse than multimodal features approach (0.4 for valence, 0.72 for arousal and 0.62 for dominance).

Various chord-based statistical measures have already been employed for different MIR tasks, such as music similarity or genre detection. In [3], chordal features (longest common chord sequence and histogram statistics on chords) were used to find similar songs and to estimate their emotion (in terms of valence) based on chord similarity. In [9], chordal statistics is used for MER, but the duration of chords is not taken into account, which we account for in this paper. Interval-based features, described here, to our knowledge have not been used before.

A computational approach to modeling musical emotion using GEMS has not been adopted before. In [11], GEMS was used to collect data dynamically on 36 musical excerpts. Listener agreement was very good (Cronbach’s alpha ranging from 0.84 to 0.98). In [12], GEMS is compared to a three-dimensional (valence-arousal-tension) and categorical (anger, fear, happiness, sadness, tenderness) models. The consistency of responses is compared, and it is found that GEMS categories have both some of the highest (joyful activation, tension) and some of the lowest (wonder, transcendence) agreement. It was also found that GEMS categories are redundant, and valence and arousal dimensions account for 89% of variance. That experiment, though, was performed on 16 musical excerpts only, and the excerpts were selected using criteria based on V-A model, which might have resulted in bias.

3. DATA DESCRIPTION

The dataset that we analyze consists of 400 musical excerpts (44100 Hz, 128 kbps). Each excerpt is 1 minute long (except for 4 classical pieces which were shorter than 1 minute). It is evenly split (100 pieces per genre) by four genres (classical, rock, pop and electronic music). In many studies, musical excerpts are specially selected for their strong emotional content that best fits the chosen emotional model, and only the excerpts that all the annotators agree upon, are left. In our dataset we maintain a good ecological validity by selecting music randomly from a Creative Commons recording label Magnatune, only making sure that the recordings are of good quality.

Based on conclusions from [11, 12], we renamed two GEMS categories by replacing them with one of their subcategories (wonder was replaced with amazement, and transcendence with solemnity). Participants were asked to select no more than three emotional terms from a list of nine. They were instructed to describe how music made them feel, and not what it expressed, and were encouraged to do so in a game context [1]. All the songs were annotated by at least 10 players (mean = 20.8, SD = 14).

The game with a purpose was launched and advertised through social networks. The game, as well as annotations and audio, are accessible online. More than 1700 players have contributed. The game was streaming music for 138 hours in total. A detailed description and analysis of the data can be found in [1] or in a technical report. [2]

We are not interested in modeling irritation from non-preferred music, but rather differences in emotional perception across listeners that come from other factors. We introduce a question to report disliking the music and discard such answers. We also clean the data by computing Fleiss’s kappa on all the annotations for every musical excerpt, and discarding the songs with negative kappa (this indicates that the answers are extremely inconsistent (33 songs)). Fleiss’s kappa is designed to estimate agreement, when the answers are binary or categorical. We use this very loose criteria, as it is expected to find a lot of disagreement. We retain the remaining 367 songs for analysis.

The game participants were asked to choose several categories from a list, but for the purposes of modeling we translate the annotations into a continuous space by using the following equation:

\[
\text{score}_{ij} = \frac{1}{n} \sum_{k=1}^{n} a_{ik},
\]

where \(\text{score}_{ij}\) is an estimated value of emotion \(i\) for song \(j\). \(a_{ik}\) is the answer of the \(k\)-th participant on a question whether emotion \(i\) is present in song \(j\) or not (answer is

\[5\] www.emotify.org
\[6\] www.projects.science.uu.nl/memotion/emotifydata/
either 0 or 1), and \( n \) is the total number of participants, who listened to song \( j \).

The dimensions that we obtain are not orthogonal: most of them are somewhat correlated. To determine the underlying structure, we perform Principal Components Analysis. According to a Scree test, three underlying dimensions were found in the data, which together explain 69% of variance. Table 1 shows the three-component solution rotated with varimax. The first component, which accounts for 32% of variance, is mostly correlated with calmness vs. power, the second (accounts for 23% of variance) with joyful activation vs. sadness, and the third (accounts for 14% of variance) with solemnity vs. nostalgia. This suggests that the underlying dimensional space of GEMS is three-dimensional. We might suggest that it resembles valence-arousal-triviality model [13].

4. HARMONIC FEATURES

It has been repeatedly shown that valence is more difficult to model than arousal. In this section we describe features, that we added to our dataset to improve prediction of modality in music.

Musical chords, as well as intervals are known to be important for affective perception of music [10], as well as other MIR tasks. Chord and melody based features have been successfully applied to genre recognition of symbolically represented music [8]. We compute statistics on the intervals and chords occurring in the piece.

4.1 Interval Features

We segment audio, using local peaks in the harmonic change detection function (HCDF) [6]. HCDF describes tonal centroid fluctuations. The segments that we obtain are mostly smaller than 1 second and reflect single notes, chords or intervals. Based on the wrapped chromagrams computed from the spectrum of this segments, we select two highest (energy-wise) peaks and compute the interval between them. For each interval, we compute its combined duration, weighted by its loudness (expressed by energy of the bins). Then, we sum up this statistics for intervals and their inversions. Figure 1 illustrates the concept (each bar corresponds to the musical representation of a feature that we obtain). As there are 6 distinct intervals with inversions, we obtain 6 features. We expect that augmented fourths and fifths (tritone) could reflect tension, contrary to perfect fourths and fifths. The proportion of minor thirds and major sixths, as opposed to proportion of major thirds and minor sixths, could reflect the modality. The interval-inversion pairs containing seconds are rather unrestful.

4.2 Chord Features

To extract chord statistics, we used 2 chord extraction tools, HPA\(^7\) (Harmonic Progression Analyzer) and Chordino\(^8\) plugins for Sonic Annotator. The first plugin provides 8 types of chords: major, minor, seventh, major and minor seventh, diminished, sixth and augmented. The second plugin, in addition to these eight types, also provides minor sixth and slash chords (chords for which bass note is different from the tonic, and might as well not belong to the chord). The chords are annotated with their onsets and offsets. After experimentation, only the chords from Chordino were left, because those demonstrated more correlation with the data. We computed the proportion of each type of chord in the dataset, obtaining nine new features. The slash chords were discarded by merging them with their base chord (e.g., Am/F chord is counted as a minor chord). The distribution of chords was uneven, with major chords being in majority (for details see Figure 2). Examining the accuracy of these chord extraction tools was not our goal, but the amount of disagreement between the two tools could give an idea about that (see Figure 2). From our experiments we concluded that weighting the chords by their duration is an important step, which improves the performance of chord histograms.

---

\(^{7}\) patterns.enm.bris.ac.uk/hpa-software-package

\(^{8}\) isophonics.net/nulis-chroma
5. MANUALLY ASSESSED FEATURES

In this section we describe an additional feature set that we composed using human experts, and explain the properties of GEMS categories through perceptual musically motivated factors. Because of huge time load that manual annotation creates we only could annotate part of the data (60 pieces out of 367).

5.1 Procedure

Three musicians (26–61 years, over 10 years of formal musical training) annotated 60 pieces (15 pieces from each genre) from the dataset with 10 factors, on a scale from 1 to 10. The meaning of points on the scale was different for each factor (for instance, for tempo 1 would mean ‘very slow’ and 10 would mean ‘very fast’). The list of factors was taken from the study of Wedin [13]: tempo (slow—fast), articulation (staccato—legato), mode (minor—major), intensity (pp—ff), tonalness (atonal—tonal), pitch (bass—treble), melody (unmelodious—melodious), rhythmic clarity (vague—firm). We added rhythmic complexity (simple—complex) to this list, and eliminated style (date of composition) and type (serious—popular) from it.

5.2 Analysis

After examining correlations with the data, one of the factors was discarded as non-informative (simple or complex harmony). This factor lacked consistency between annotators as well. Table 2 shows the correlations (Spearman’s ρ) between manually assessed factors and emotional categories. We used a non-parametric test, because distribution of emotional categories is not normal, skewed towards smaller values (emotion was more often not present than present). All the correlations are significant with p-value < 0.01, except for the ones marked with asterisk, which are significant with p-value < 0.05. The values that are absent or marked with double asterisks failed to reach statistical significance, but some of them are still listed, because they illustrate important trends which are very probable to reach significance should we have more data.

Many GEMS categories were quite correlated (tenderness and nostalgia: r = 0.5, tenderness and calmness: r = 0.52, power and joyful activation: r = 0.4). All of these have, however, musical characteristics that allow listeners to differentiate them, as we will see below.

Both nostalgia and tenderness correlate with slow tempo and legato articulation, but tenderness is also correlated with higher pitch, major mode, and legato articulation (as opposed to staccato for nostalgia). Calmness is characterized by slow tempo, legato articulation and smaller intensity, similarly to tenderness. But tenderness features a correlation with melodiousness and major mode as well. Both power and joyful activation are correlated with fast tempo, and intensity, but power is correlated with minor mode and joyful activation with major mode.

As we would expect, tension is strongly correlated with non-melodiousness and atonality, lower pitch and minor mode. Sadness, strangely, is much less correlated with mode, but it more characterized by legato articulation, slow tempo and smaller rhythmic complexity.

6. EVALUATION

6.1 Features

We use four toolboxes for MIR to extract features from audio: MIRToolbox, OpenSmile, PsySound and two VAMP plugins for SonicAnnotator. We also extract harmonic features, described in Section 4. These particular tools are chosen because the features they provide were specially designed for MER. MIRToolbox was conceived as a tool for investigating a relationship between emotion and features in music. OpenSmile combines features from Speech Processing and MIR and demonstrated good performance on cross-domain emotion recognition [14]. We evaluate three following computational and one human-assessed feature sets:

1. MIRToolbox + PsySound: 40 features from MIRToolbox (spectral features, HCDF, mode, inharmonicity etc.) and 4 features related to loudness from PsySound (using the loudness model of Chalupper and Fastl).

2. OpenSmile: 6552 low-level supra-segmental features (chroma features, MFCCs or energy, and statistical...
3. **MP+Harm**: to evaluate performance of harmonic features, we add them to the first feature set. It doesn’t make sense to evaluate them alone, because they only cover one aspect of music.

4. **Musical feature set**: these are 9 factors of music described in section 5.

6.2 Learning Algorithm

After trying SVR, Gaussian Processes Regression and linear regression, we chose SVR (the LIBSVM implementation⁹) as a learning algorithm. The best performance was achieved using the RBF kernel, which is defined as follows:

\[ k(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right), \]

where \( \gamma \) is a parameter given to SVR. All the parameters, \( \mathcal{C} \) (error cost), epsilon (slack of the loss function) and \( \gamma \), are optimized with grid-search for each feature set (but not for each emotion). To select an optimal set of features, we use recursive feature elimination (RFE). RFE assigns weights to features based on output from a model, and removes attributes until performance is no longer improved.

6.3 Evaluation

We evaluate the performances of the four systems using 10-fold cross-validation, splitting the dataset by artist (there are 140 distinct artists per 400 songs). If a song from artist A appears in the training set, there will be no songs from this artist in the test set. Table 3 shows evaluation results. The accuracy of the models differs greatly per category, while all the feature sets demonstrate the same pattern of success and failure (for instance, perform badly on amazement and well on joyful activation). This reflects the fact that these two categories are very different in their subjectiveness. Figure 3 illustrates the performance of the systems (\( r \)) for each of the categories and Cronbach’s alpha (which measures agreement) computed on listener’s answers (see [1] for more details), and shows that they are highly correlated. The low agreement between listeners results in conflicting cues, which limit model performance.

In general, the accuracy is comparable to accuracy achieved for perceived emotion by others [5,7,15], though it is somewhat lower. This might be explained by the fact that all the categories contain both arousal and valence components, and induced emotion annotations are less consistent. In [7], tenderness was predicted with \( R = 0.67 \), as compared to \( R = 0.57 \) for MP+Harm system in our case. For power and joyful activation, the predictions from the best systems (MP+Harm and OpenSmile) demonstrated 0.56 and 0.68 correlation with the ground truth, while in [5,15] it was 0.72 and 0.76 for arousal.

The performance of all the three computational models is comparable, though MP+Harm model performs slightly better in general. Adding harmonic features improves average performance from 0.43 to 0.47, and performance of the best system (MP+Harm) decreases to 0.35 when answers from people who disliked the music are not discarded. As we were interested in evaluating the new features, we checked which features were considered important by RFE. For power, the tritone proportion was important (positively correlated with power), for sadness, the proportion of minor chords, for tenderness, the proportion of seventh chords (negatively correlates), for tension, the proportion of tritones, for joyful activation, the proportion of seconds and inversions (positive correlation).

The musical feature set demonstrates the best performance as compared to all the features derived from signal-processing, demonstrating that our ability to model human perception is not yet perfect.

7. CONCLUSION

We analyze the performance of audio features on prediction of induced musical emotion. The performance of the best system is somewhat lower than can be achieved for perceived emotion recognition. We conduct PCA and find

<table>
<thead>
<tr>
<th>Feature set</th>
<th>MIRT Toolbox + PsySound</th>
<th>OpenSmile</th>
<th>MP + Harm</th>
<th>Musicological</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( r )</td>
<td>RMSE</td>
<td>( r )</td>
<td>RMSE</td>
</tr>
<tr>
<td>Amazement</td>
<td>( 0.70 \pm 1.8 )</td>
<td>( 0.99 \pm 1.6 )</td>
<td>( 0.19 \pm 1.5 )</td>
<td>( 0.95 \pm 1.3 )</td>
</tr>
<tr>
<td>Solemnity</td>
<td>( 0.35 \pm 1.4 )</td>
<td>( 0.80 \pm 0.9 )</td>
<td>( 0.42 \pm 1.6 )</td>
<td>( 0.95 \pm 1.3 )</td>
</tr>
<tr>
<td>Tenderness</td>
<td>( 0.50 \pm 1.0 )</td>
<td>( 0.84 \pm 1.0 )</td>
<td>( 0.52 \pm 1.2 )</td>
<td>( 0.95 \pm 0.7 )</td>
</tr>
<tr>
<td>Nostalgia</td>
<td>( 0.53 \pm 1.6 )</td>
<td>( 0.82 \pm 1.2 )</td>
<td>( 0.53 \pm 1.8 )</td>
<td>( 0.89 \pm 0.7 )</td>
</tr>
<tr>
<td>Calmness</td>
<td>( 0.55 \pm 1.4 )</td>
<td>( 0.83 \pm 0.9 )</td>
<td>( 0.55 \pm 1.6 )</td>
<td>( 0.89 \pm 0.7 )</td>
</tr>
<tr>
<td>Power</td>
<td>( 0.48 \pm 1.8 )</td>
<td>( 0.82 \pm 1.3 )</td>
<td>( 0.56 \pm 0.9 )</td>
<td>( 0.84 \pm 0.9 )</td>
</tr>
<tr>
<td>Joyful activation</td>
<td>( 0.63 \pm 0.8 )</td>
<td>( 0.77 \pm 0.1 )</td>
<td>( 0.68 \pm 0.8 )</td>
<td>( 0.80 \pm 0.8 )</td>
</tr>
<tr>
<td>Tension</td>
<td>( 0.38 \pm 1.4 )</td>
<td>( 0.87 \pm 0.2 )</td>
<td>( 0.41 \pm 1.9 )</td>
<td>( 0.94 \pm 1.9 )</td>
</tr>
<tr>
<td>Sadness</td>
<td>( 0.41 \pm 1.3 )</td>
<td>( 0.87 \pm 1.1 )</td>
<td>( 0.40 \pm 1.8 )</td>
<td>( 0.96 \pm 1.8 )</td>
</tr>
</tbody>
</table>

Table 3. Evaluation of 4 feature sets on the data. Pearson’s \( r \) and RMSE with their standard deviations (across cross-validation rounds) are shown.

---

⁹ www.csie.ntu.edu.tw/~cjlin/libsvm/
three dimensions in the GEMS model, which are best explained by axes spanning calmness—power, joyful activation—sadness and solemnity—nostalgia). This finding is supported by other studies in the field [4, 13].

We conclude that it is possible to predict induced musical emotion for some emotional categories, such as tenderness and joyful activation, but for many others it might not be possible without contextual information. We also show that despite this limitation, there is still room for improvement by developing features that can better approximate human perception of music, which can be pursued in future work on emotion recognition. 10

8. REFERENCES


COGNITION-INSPIRED DESCRIPTORS FOR SCALABLE COVER SONG RETRIEVAL

Jan Van Balen, Dimitrios Bountouridis, Frans Wiering, Remco Veltkamp
Utrecht University, Department of Information and Computing Sciences
j.m.h.vanbalen@uu.nl, d.bountouridis@uu.nl

ABSTRACT

Inspired by representations used in music cognition studies and computational musicology, we propose three simple and interpretable descriptors for use in mid- to high-level computational analysis of musical audio and applications in content-based retrieval. We also argue that the task of scalable cover song retrieval is very suitable for the development of descriptors that effectively capture musical structures at the song level. The performance of the proposed descriptions in a cover song problem is presented. We further demonstrate that, due to the musically-informed nature of the proposed descriptors, an independently established model of stability and variation in covers songs can be integrated to improve performance.

1. INTRODUCTION

This paper demonstrates the use of three new cognition-inspired music descriptors for content-based retrieval.

1.1 Audio Descriptors

There is a growing consensus that some of the most widely used features in Music Information Research, while very effective for engineering applications, do not serve the dialog with other branches of music research [1]. As a classic example, MFCC features can be shown to predict human ratings of various perceptual qualities of a sound. Yet, from the perspective of neuropsychology, claims that they mathematically approximate parts of auditory perception have become difficult to justify as more parts of the auditory pathway are understood.

Meanwhile, a recent analysis of evaluation practices by Sturm [18] suggests that MIR systems designed to classify songs into high-level attributes like genre, mood or instrumentation may rely on confounded factors unrelated to any high-level property of the music, even if their performance numbers approach 100%. Researchers have focused too much on the same datasets and as a result, today, top performing genre and mood recognition systems rely on the same low-level features that are used to classify bird sounds. ¹

We also observe that, despite the increasing availability of truly big audio data and the promising achievements of MIR over the last decade, studies that turn big audio data into findings about music itself seem hard to find. Notable exceptions include studies on scales and intonation, and [16]. In the latter, pitch, timbre and loudness data were analyzed for the Million Song Dataset, focusing on the distribution and transitions of discretized code words. Yet, we have also observed that this analysis sparks debate among music researchers outside the MIR field, in part because of the descriptors used. The study uses the Echo Nest audio features provided with the dataset, which are computed using undisclosed, proprietary methods and therefore objectively difficult in interpretation.

1.2 Towards Cognitive Audio Descriptors

On the long term we would like to model cognition-level qualities of music such as its complexity, expectedness and repetitiveness from raw audio data. Therefore we aim to design and evaluate features that describe harmony, melody and rhythm on a level that has not gained the attention it deserves in MIR's audio community, perhaps due to the 'success' of low-level features discussed above. In the long run, we believe, this will provide insights into the building blocks of music: riffs, motives, choruses, and so on.

1.3 Cover Song Detection

In this section, we argue that the task of scalable cover song retrieval is very suitable for developing descriptors that effectively capture mid- to high-level musical structures, such as chords, riffs and hooks.

Cover detection systems take query song and a database and aim to find other versions of the query song. Since many real-world cover versions drastically modulate multiple aspects of the original: systems must allow for deviations in key, tempo, structure, lyrics, harmonisation and phrasing, to name just a few. Most successful cover detection algorithms are built around a two-stage architecture. In the first stage, the system computes a time series representation of the harmony or pitch for each of the songs in a database. In the second stage, the time series representing

¹ largely MFCC and spectral moments, see [6, 18] for examples
the query is compared to each of these representations, typically by means of some kind of alignment, i.e. computing the locations of maximum local correspondence between the two documents being compared. See [15] for more on this task and an overview of cover detection strategies.

2. SCALABLE COVER SONG RETRIEVAL

Generally, alignment methods are computationally expensive but effective. Results achieved this way have reached mean average precision (MAP) figures of around 0.75 at the MIREX evaluation exchange. ²

When it comes to large-scale cover detection (hundreds of queries and thousands of songs), however, alignment-based methods can become impractical. Imagine a musicologist whose aim is not to retrieve matches to a single query, but to study all the relations in a large, representative corpus. Alignment-based techniques are no longer an option: a full pair-wise comparison of 10,000 documents would take weeks, if not months. ³

This is why some researchers have been developing scalable techniques for cover song detection. Scalable strategies are often inspired by audio fingerprinting and involve the computation of an indexable digest of (a set of) potentially stable landmarks in the time series, which can be stored and matched through a single inexpensive look-up. Examples include the ‘jumpcodes’ approach by [2], the first system to be tested using the Million Song Dataset. This study reports a recall of 9.6% on the top 1 percent of retrieved candidates. Another relevant example is the interval-gram approach by Walters [19], which computes fingerprinting-inspired histograms of local pitch intervals, designed for hashing using wavelet decomposition.

Reality shows that stable landmarks are relatively easy to find when looking for exact matches (as in fingerprinting), but hard to find in real-world cover songs. A more promising approach was presented by Bertin-Mahieux in [3], where the 2D Fourier transform of beat-synchronized chroma features is used as the primary representation. The accuracy reported is several times better than for the system based on jumpcodes. Unfortunately, exactly what the Fourier transformed features capture is difficult to explain.

The challenges laid out in the above paragraph make cover song detection an ideal test case to evaluate a special class of descriptors: harmony, melody and rhythm descriptors, global or local, which have a fixed dimensionality and some tolerance to deviations in key, tempo and global structure. If a collection of descriptors can be designed that accurately describes a song’s melody, harmony and rhythm in a way that is robust to the song’s precise structure, tempo and key, we should have a way to determine similarity between the ‘musical material’ of two songs and assess if the underlying composition is likely to be the same.

3. PITCH AND HARMONY DESCRIPTORS

There is an increasing amount of evidence that the primary mechanism governing musical expectations is statistical learning [7, 12]. On a general level, this implies that the conditional probabilities of musical events play a large role in their cognitive processing. Regarding features and descriptors, it justifies opportunities of analyzing songs and corpora in terms of probably distributions. Expectations resulting from the exposure to statistical patterns have in turn been shown to affect the perception of melodic complexity and familiarity. See [7] for more on the role of expectation in preference, familiarity and recall.

We propose three new descriptors: the pitch bihistogram, the chroma correlation coefficients and the harmonization feature. The pitch bihistogram describes melody and approximates a histogram of pitch bigrams. The chroma correlation coefficients relate to harmony. They approximate the co-occurrence of chord notes in a song. The third representation, the harmonization feature, combines harmony and melody information. These three descriptors will now be presented in more detail.

3.1 The Pitch Bihistogram

Pitch bigrams are ordered pairs of pitches, similar to word or letter bigrams used in computational linguistics. Several authors have proposed music descriptions based on pitch bigrams, most of them from the domain of cognitive science [10, 11, 13]. Distributions of bigrams effectively encode first-degree expectations. More precisely: if the distribution of bigrams in a piece is conditioned on the first pitch in the bigram, we obtain the conditional frequency of a pitch given the one preceding it.

The first new feature we introduce will follow the bigram paradigm. Essentially, it captures how often two pitches \( p_1 \) and \( p_2 \) occur less than a distance \( d \) apart.

Assume that a melody time series \( P(t) \), quantized to semitones and folded to one octave, can be obtained. If a pitch histogram is defined as:

\[
h(p) = \frac{1}{P(t)=p},
\]

with \( n \) the length of the time series and \( p \in \{1, 2, \ldots, 12\} \), the proposed feature is then defined:

\[
B(p_1, p_2) = \sum_{P(t_1)=p_1, P(t_2)=p_2} w(t_2 - t_1)
\]

where

\[
w(x) = \begin{cases} 
\frac{1}{2}, & \text{if } 0 < x < d, \\
0, & \text{otherwise}.
\end{cases}
\]

This will be referred to as the \textbf{pitch bihistogram}, a bigram representation that can be computed from continuous melodic pitch. Note that the use of pitch classes rather than pitch creates an inherent robustness to octave errors in the melody estimation step, making the feature insensitive to one of the most common errors encountered in pitch extraction.
Alternatively, scale degrees can be used instead of absolute pitch class. In this scenario, the pitch contour \( P(t) \) must first be aligned to an estimate of the piece’s overall tonal center. As a tonal center, the tonic can be used. However, for extra robustness to misestimating the tonic, we suggest to use the tonic for major keys and the minor third for minor keys.

### 3.2 Chroma Correlation Coefficients

The second feature representation we propose focuses on vertical rather than horizontal pitch relation. It encodes which pitches appear simultaneously in a signal.

\[
C(p_1, p_2) = \text{corr}(c(t, p_1), c(t, p_2)),
\]

where \( c(t, p) \) is a 12-dimensional chroma time series (also known as pitch class profile) computed from the song audio. From this chroma representation of the song \( c(t, p) \) we compute the correlation coefficients between each pair of chroma dimensions to obtain a \( 12 \times 12 \) matrix of chroma correlation coefficients \( C(p_1, p_2) \). Like the pitch bihistogram, the chroma features can be transposed to the same tonal center (tonic or third) based on an estimate of the overall or local key.

### 3.3 Harmonisation Feature

Finally, the harmonisation feature is a set of histograms of the harmonic pitches \( p_h \in \{1, \ldots, 12\} \) as they accompany each melodic pitch \( p_m \in \{1, \ldots, 12\} \). It is computed from the pitch contour \( P(t) \) and a chroma time series \( c(t, p_h) \), which should be adjusted to have the same sampling rate and aligned to a common tonal center.

\[
H(p_m, p_h) = \sum_{P(t) = p_m} c(t, p_h).
\]

From a memory and statistical learning perspective, the chroma correlation coefficients and harmonisation feature may be used to approximate expectations that include: the expected consonant pitches given a chord note, the expected harmony given a melodic pitch, and the expected melodic pitch given a chord note. Apart from [8], where a feature resembling the chroma correlation coefficients is proposed, information of this kind has yet to be exploited in a functioning (audio) MIR system. Like the pitch bihistogram and the chroma correlation coefficients, the harmonisation feature has a dimensionality of \( 12 \times 12 \).

### 4. EXPERIMENTS

To evaluate the performance of the above features for cover song retrieval, we set up a number of experiments around the covers80 dataset by Ellis [5]. This dataset is a collection of 80 cover song pairs, divided into a fixed list of 80 queries and 80 candidates. Though covers80 is not actually ‘large-scale’, it is often used for benchmarking4 and its associated audio data are freely available. In contrast, the much larger Second Hand Songs dataset is distributed only in the form of standard Echo Nest features. These features do not include any melody description, which is the basis for the descriptors proposed in this study.

Regarding scalability, we chose to follow the approach taken in [19], in which the scalability of the algorithm follows from the simplicity of the matching step. The proposed procedure is computationally scalable in the sense that, with the appropriate hashing strategy, matching can be performed in constant time with respect to the size of the database. Nevertheless, we acknowledge that the distinguishing power of the algorithm must be assessed in the context of much more data. A large scale evaluation of our algorithm, adapted to an appropriate dataset and extended to include hashing solutions and indexing, is planned as future work.

#### 4.1 Experiment 1: Global Fingerprints

In the first experiment, the three descriptors from section 3 were extracted for all 160 complete songs. Pitch contours were computed using Melodia and chroma features using HPCP, using default settings [14].5 For efficiency in computing the pitch bihistogram, the pitch contour was median-filtered and downsampled to \( \frac{1}{4} \) of the default frame rate. The bihistogram was also slightly compressed by taking its square root.

The resulting representations \( (B, C \text{ and } H) \) were then scaled to the same range by whitening them for each song individually (subtracting the mean of their \( n \) dimensions, and dividing by the standard deviation; \( n = 144 \)). To avoid relying on key estimation, features in this experiment were not aligned to any tonal center, but transposed to all 12 possible keys. In a last step of the extraction stage, the features were scaled with a set of dedicated weights \( w = (w_1, w_2, w_3) \) and concatenated to \( 12 \times 432 \)-dimensional vectors, one for each key. We refer to these vectors as the global fingerprints.

In the matching stage of the experiment, the distances between all queries and candidates were computed using a cosine distance. For each query, all candidates were ranked by distance. Two evaluation metrics were computed: recall at 1 (the proportion of covers retrieved among the top 1 result for each query; \( R_1 \)) and recall at 5 (proportion of cover retrieved ‘top 5’; \( R_5 \)).

#### 4.2 Experiment 2: Thumbnail Fingerprints

In a second experiment, the songs in the database were first segmented into structural sections using structure features as described by Serra [17]. This algorithm performed best at the 2012 MIREX evaluation exchange in the task of ‘music structure segmentation’, both for boundary recovery and for frame pair clustering. (A slight simplification was made in the stage where sections are compared: no dynamic time warping was applied in our model.) From this segmentation, two non-overlapping thumbnails are selected as follows:

---

4 results for this dataset have been reported by at least four authors [15] 5 mtg.upf.edu/technologies
1. Simplify the sequence of section labels (e.g., ababCabCC): merge groups of section labels that consistently appear together (e.g., AACACC for the example above).

2. Compute the total number of seconds covered by each of the labels A, B, C... and find the two section labels covering most of the song.

3. Return the boundaries of the first appearance of the selected labels.

The fingerprint as described above was computed for the full song as well as for the resulting thumbnails, yielding three different fingerprints: one global and two thumbnail fingerprints, stored separately. As in experiment 1, we transposed these thumbnails to all keys, resulting in a total of 36 fingerprints extracted per song: 12 for the full song, 12 for the first thumbnail and 12 for the second thumbnail.

### 4.3 Experiment 3: Stability Model

In the last experiment, we introduced a model of stability in cover song melodies. This model was derived independently, through analysis of a dataset of annotated melodies of cover songs variations. Given the melody contour for a song section, the model estimates the stability at each point in the melody. Here, stability is defined as the probability of the same pitch appearing in the same place in a performed variation of that melody.

The stability estimates produced by the model are based on three components that are found to correlate with stability: the duration of notes, the position of a note inside a section, and the pitch interval. The details of the model and its implementation are described in the following section.

### 5. STABILITY MODEL

The model we apply is a quantitative model of melody stability in cover songs. As it has been established for applications broader than the current study, it is based on a unique, manually assembled collection of annotated cover songs melodies. The dataset contains four transcribed melodic variations for 45 so-called ‘cliques’ of cover songs, a subset of the Second Hand Songs dataset. Some songs have one section transcribed, some have more, resulting in a total of 240 transcriptions.

For the case study presented here, transcriptions were analysed using multiple sequence alignment (MSA) and a probabilistic definition of stability.

#### 5.1 Multiple Sequence Alignment

Multiple sequence alignment is a bioinformatics method that extends pairwise alignment of symbolic arrays to a higher number of sequences [4]. Many approaches to MSA exist, some employing hidden markov models or genetic algorithms. The most popular is progressive alignment.

This technique creates an MSA by combining several pairwise alignments (PWA) starting from the most similar sequences, constructing a tree usually denoted as the ‘guide tree’. Unlike MSA, pairwise alignment has been researched extensively in the (symbolic) MIR community, see [9] for an overview.

Whenever two sequences are aligned, a consensus can be computed, which can be used for the alignment connecting the two sequences to the rest of the three. The consensus is a new compromise sequence formed using heuristics to resolve the ambiguity at non-matching elements. These heuristics govern how gaps propagate through the tree, or whether ‘leaf’ or ‘branch’ elements are favored. The current model favors gaps and branch elements.

When the root consensus of the tree is reached, a last iteration of PWAs aligns each sequence to the root consensus to obtain the final MSA. Figure 1 shows two sets of melodic sequences (mapped to a one-octave alphabet \{A \ldots L\}) before and after MSA. Note that the MSA is based on a PWA strategy which maximizes an optimality criterion based on not just pitch but also duration and onset times.

#### 5.2 Stability

The stability of a note in a melody is now defined as the probability of the same note being found in the same position in an optimally aligned variation of that melody.

Empirically, given a set of \(N\) aligned sequences

\[
\{s_k(i)\} \quad i = 1 \ldots n, \quad k = 1 \ldots N
\]

we compute the stability of event \(s_k(i)\) as:

\[
\text{stab}(s_k(i)) = \frac{1}{N-1} \sum_{j=1}^{N} s_j(i) \quad s_k(i) = = s_k(i)
\]

As an example, in a position \(i\) with events \(s_1(i) = A, s_2(i) = A, s_3(i) = A\) and \(s_4(i) = B\), the stability of \(A\) is 0.66. The stability of \(B\) is 0.

#### 5.3 Findings

As described in the previous section, we drew a random sample of notes from the dataset in order to observe how stability behaves as a function of the event’s pitch, duration and position inside the song section.

The first relationship has ‘position’ as the independent variable and describes the stability as it evolves throughout
the section. Figure 2 shows how stability changes with position. The mean and 95% CI for the mean are shown for two different binnings of the position variable. The 4-bin curve illustrates how stability generally decreases with position. The more detailed 64-bin curve shows how the first two thirds of a melody are more stable than the last, though an increased stability can be seen at the end of the section.

Figure 3 shows the stability of notes as a function of their duration. The distribution of note durations is centered around 1% of the segment length. Below and above this value, the stability goes slightly up. This suggests that notes with less common durations are more stable. However, the trend is weak compared with the effect of position. Note duration information will therefore not be used in the experiments in this study.

Figure 4 shows the stability (mean and 95% CI for the mean) of a note given the pitch interval that follows. Note how the relative stability of one-semitone jumps stands out compared to repetitions and two-semitone jumps, even though two-semitone jumps are far more frequent. This suggests again that less-frequent events are more stable. More analysis as to this hypothesis will be performed in a later study.

6. DISCUSSION

Table 1 summarizes the results of the experiments.

In the experiments where each descriptor was tested individually, the harmony descriptors (chroma correlation coefficients) performed best: we obtained an accuracy of over 30%. When looking at the top 5, there was a recall of 53.8%. The recall at 5 evaluation measure is included to give an impression of the performance that could be gained if the current system were complemented with an alignment-based approach to sort the top-ranking candidates, as proposed by [19].

The next results show that, for the three features together, the global fingerprints outperform the thumbnail fingerprints (42.5% vs. 37.5%), and combining both types does not increase performance further. In other configurations, thumbnail fingerprints were observed to outperform the global fingerprints. This is possibly the result of segmentation choices: short segments produce sparse fingerprints, which are in turn farther apart in the feature space than ‘dense’ fingerprints.

In experiment 3, two components of the stability model were integrated in the cover detection system. The 4-bin stability vs. position curve (scaled to the [0, 1] range) was used as a weighting to emphasize parts of the melody before computing the thumbnails’ pitch bihistogram. The stability per interval (compressed by taking its square root) was used to weigh the pitch bihistogram directly.

With the stability information added to the model, the top 1 precision reaches 45.0%. The top 5 recall is 56.3%. This result is situated between the accuracy of the first alignment-based strategies (42.5%), and the accuracy of a recent scalable system (53.8%; [19]). We conclude that the descriptors capture enough information to discriminate between individual compositions, which we set out to show.

7. CONCLUSIONS

In this study, three new audio descriptors are presented. Their interpretation is discussed, and results are presented for an application in cover song retrieval. To illustrate the benefit of feature interpretability, an independent model of cover song stability is integrated into the system.

We conclude that current performance figures, though not state-of-the-art, are a strong indication that scalable cover detection can indeed be achieved using interpretable, cognition-inspired features. Second, we observe that the pitch bihistogram feature, the chroma correlation coefficients and the harmonisation feature capture enough information to discriminate between individual compositions, proving that they are at the same time meaningful and informative, a scarce resource in the MIR feature toolkit. Finally, we have demonstrated that cognition-level audio description and scalable cover detection can be successfully addressed together.
Table 1. Summary of experiment results. \( w \) are the feature weights. Performance measures are recall at 1 (proportion of covers retrieved ‘top 1’; \( R_1 \)) and recall at 5 (proportion of cover retrieved among ‘top 5’; \( R_5 \)).

<table>
<thead>
<tr>
<th>Descriptor</th>
<th>( R_1 )</th>
<th>( R_5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global fingerprints</td>
<td>( B )</td>
<td>0.288</td>
</tr>
<tr>
<td></td>
<td>( C )</td>
<td>0.313</td>
</tr>
<tr>
<td></td>
<td>( H )</td>
<td>0.200</td>
</tr>
<tr>
<td>( w = (2, 3, 1) )</td>
<td></td>
<td>0.425</td>
</tr>
<tr>
<td>Thumbnail fingerprints</td>
<td>( w = (2, 3, 1) )</td>
<td>0.388</td>
</tr>
<tr>
<td>Global + thumbnail fingerprints</td>
<td>( w = (2, 3, 1) )</td>
<td>0.425</td>
</tr>
<tr>
<td>Both fingerprints + stability model</td>
<td>( w = (2, 3, 1) )</td>
<td>0.450</td>
</tr>
</tbody>
</table>

As future work, tests will be carried out to assess the discriminatory power of the features when applied to a larger cover song problem.

8. ACKNOWLEDGEMENTS
This research is supported by the NWO CATCH project COGITCH (640.005.004), and the FES project COMMIT/.

9. REFERENCES
A CROSS-CULTURAL STUDY OF MOOD IN K-POP SONGS

Xiao Hu  
University of Hong Kong  
xiaohu@hku.hk

Jin Ha Lee  
University of Washington  
jinhale@uw.edu

Kahyun Choi  
University of Illinois  
{kahyu2,jdownie}@illinois.edu

ABSTRACT

Prior research suggests that music mood is one of the most important criteria when people look for music—but the perception of mood may be subjective and can be influenced by many factors including the listeners’ cultural background. In recent years, the number of studies of music mood perceptions by various cultural groups and of automated mood classification of music from different cultures has been increasing. However, there has yet to be a well-established testbed for evaluating cross-cultural tasks in Music Information Retrieval (MIR). Moreover, most existing datasets in MIR consist mainly of Western music and the cultural backgrounds of the annotators were mostly not taken into consideration or were limited to one cultural group. In this study, we built a collection of 1,892 K-pop (Korean Pop) songs with mood annotations collected from both Korean and American listeners, based on three different mood models. We analyze the differences and similarities between the mood judgments of the two listener groups, and propose potential MIR tasks that can be evaluated on this dataset.

1. INTRODUCTION

The mood of music is arguably one of the strongest factors behind people’s motivation of listening to music [4]. Recognizing the importance of music mood, an increasing number of studies have been exploring the use of mood data to improve users’ access to music. Recent studies in Music Information Retrieval (MIR) have indicated that people from different cultural backgrounds may perceive music mood differently ([2], [9]). In an effort toward establishing a global MIR system that can serve users from different parts of the world, researchers have developed and evaluated algorithms that can work on classifying music from different cultures and/or labeled by listeners from different countries ([17], [14]). Despite the growing interests on cultural influences on MIR ([7], [14]), we still do not have a well-established testbed for cross-cultural MIR tasks where methods proposed by interested researchers can be properly evaluated and compared. Music Information Retrieval Evaluation eXchange (MIREX), which is the primary MIR evaluation venue, has yet to add a cross-cultural evaluation task. This study aims to work toward filling this gap by 1) building a dataset consisting of 1,892 songs from a non-Western culture (i.e., K-pop or Korean Pop) and labels based on three music mood models annotated by listeners from two distinct cultural groups (i.e., American and Korean); 2) analyzing the differences and similarities between mood labels provided by American and Korean listeners on the same set of 1,892 K-pop songs; and 3) proposing cross-cultural MIR tasks that can be evaluated using this dataset.

2. RELATED WORK

Music is a medium beyond the boundary of languages, countries and cultures. As many MIR systems need to be designed to serve global users, researchers have been paying more attention to cross-cultural issues in MIR. Lee and Hu [9] compared mood labels on a set of 30 Western songs provided by American, Chinese, and Korean listeners and found that cultural background indeed influenced people’s perception of music mood. Yang and Hu [17] compared mood labels on U.S. pop songs provided by Western listeners to labels on Chinese pop songs provided by Chinese listeners. The datasets were larger (nearly 500 songs) in their study, although the labels were applied to two separate datasets and thus may not be directly comparable. In this paper, we compare music mood perceptions of the same set of K-pop songs from American and Korean listeners, making the mood annotations directly comparable.

K-pop is increasingly becoming popular with international audiences, as evidenced by the launch of Billboard K-pop Hot 100 chart in 2012, and is actively sought by people from different cultural backgrounds. K-pop has unique characteristics due to its history; Korean culture has been heavily influenced by American pop culture since the 1950s; yet is deeply rooted in the long history of East Asia. A recent study by Lee et al. [7] discussed the differences in the perception of K-pop genres by American and Korean listeners based on how they applied genre labels to K-pop music. In this study, we focus on the mood aspect of K-pop, aiming to improve the understanding of how mood can be used as a descriptor for organizing and accessing music by users from different cultures.

Currently there exist several influential datasets in music mood recognition ([3], [5]). However, most of them contain primarily Western music and the cultural background of annotators was either not specified [3] or not controlled [5]. To the best of our knowledge, the dataset built in this study is the first of its kind that is composed


1 The tag data will be incorporated into MIREX for use in a variety of MIR evaluations and released incrementally over time when not needed for MIREX.

2 http://www.billboard.com/articles/news/467764/billboard-k-pop-hot-100-launches-sistar-is-no-1-on-new-korea-chart
of a significant amount of non-Western music, annotated by listeners from two distinct cultures, and labeled based on three music mood models. In MIREX, there have been two mood-related (sub)-tasks: Audio Mood Classification (AMC) starting from 2007 and the mood tag subtask in Audio Tag Classification (ATC) starting from 2009. Both tasks consist of Western songs labeled by listeners from unspecified cultural backgrounds [3]. This new dataset will enable evaluation tasks that explore the cross-cultural generalizability of automated music mood recognition systems [17].

3. STUDY DESIGN

3.1 The K-Pop Music Dataset

The dataset consists of 1,892 K-pop songs across seven dominant music genres in K-pop, namely Ballad, Dance/Electronic, Folk, Hip-hop/Rap, Rock, R&B/Soul, and Trot [7]. 30 second music clips were extracted from each song and presented to the listeners for mood annotation. This was to mitigate the cognitive load of annotators and to minimize the effect of possible mood changes during the entire duration of some songs (which can happen for some songs but is beyond the scope of this study).

3.2 Music Mood Models

In representing music mood, there are primarily two kinds of models: categorical and dimensional [5]. In categorical models, music mood is represented as a set of discrete mood categories (e.g., happy, sad, calm, angry, etc.) and each song is assigned to one or more categories. This study adopted two categorical models used in MIREX: 1) the five mood clusters (Table 1) used in the Audio Mood Classification task [3] where each song is labeled with one mood cluster exclusively; and 2) the 18 mood groups (Figure 2) used in the mood tag subtask in Audio Tag Classification where each song is labeled with up to six groups. Besides being used in MIREX, these two models were chosen due to the fact that they were developed from empirical data of user judgments and in a way that is completely independent from any dimensional models, and thus they can provide a contrast to the latter.

Unlike categorical models, dimensional models represent a “mood space” using a number of dimensions with continuous values. The most influential dimensional model in MIR is Russell’s 2-dimensional model [11], where the mood of each song is represented as a pair of numerical values indicating its degree in the Valence (i.e., level of pleasure) and Arousal (i.e., level of energy) dimensions. Both categorical and dimensional models have their advantages and disadvantages. The former uses natural language terms and thus is considered more intuitive for human users, whereas the latter can represent the degree of mood(s) a song may have (e.g., a little sad). Therefore, we used both kinds of models when annotating the mood of our K-pop song set. In addition to the 5 mood clusters and 18 mood groups, the K-pop songs were also annotated with the Valence-Arousal 2-dimensional model.

1 http://www.music-ir.org/mirex/wiki/MIREX_HOME

| Cluster1 (C_1) | passionate, rousing, confident, boisterous, rowdy |
| Cluster2 (C_2) | rollicking, cheerful, fun, sweet, amiable/good natured |
| Cluster3 (C_3) | literate, poignant, wistful, bittersweet, autumnal, brooding |
| Cluster4 (C_4) | humorous, silly, campy, quirky, whimsical, witty, wry |
| Cluster5 (C_5) | aggressive, fiery, tense/anxious, intense, volatile, visceral |

Table 1. Five mood clusters in the MIREX AMC task.

3.3 Annotation Process

For a cross-cultural comparison, a number of American and Korean listeners were recruited to annotate the mood of the songs. The American listeners were recruited via a well-known crowdsourcing platform, Amazon Mechanical Turk (MTurk), where workers complete tasks requiring human intelligence for a small fee. MTurk has been recognized as a quick and cost-effective way of collecting human opinions and has been used successfully in previous MIR studies (e.g., [6], [8]). In total, 134 listeners who identified themselves as American participated in the annotations based on the three mood models.

For the five-mood cluster model, each “HIT” (Human Intelligence Task, the name for a task in MTurk) contained 22 clips with two duplicates for a consistency check. Answers were only accepted if the annotations on the duplicate clips were the same. Participants were paid $2.00 for successfully completing each HIT. For the 18-group model, we paid $1.00 for each HIT, which contained 11 clips with one duplicate song for consistency check. There were fewer clips in each HIT of this model as the cognitive load was heavier: it asked for multiple (up to six) mood labels out of 18. For the Valence-Arousal (V-A) dimensional model we designed an interface with two slide scales in the range of [-10.0, 10.0] (Figure 1). We paid $1.00 for each HIT, which contained 11 clips with one duplicate song for a consistency check. Consistency was defined such that the difference between the two annotations of the duplicate clips in either dimension should be smaller than 2.0. The threshold was based on the findings in [16] where a number of listeners gave V-A values to the same songs in two different occasions and the differences never exceeded 10% of the entire range. For each of the three mood representation models, three annotations were collected for each music clip. The total cost was approximately $1800.

As there was no known crowdsourcing platform for Korean people, the nine Korean listeners who participated in the annotation were recruited through professional and personal networks of the authors. The annotation was done with our in-house annotation systems, which are similar to those in MTurk. All instructions and mood labels/dimensions were translated into Korean to minimize possible misunderstanding of the terminology. Similarly, each song received three annotations in each mood model. The payments to annotators were also comparable to those in MTurk. Although the total number of annotators in the two cultural groups differs, each song had exactly
six independent annotations on which the following analysis and comparisons are based.

Figure 1. Annotation interface of the VA model (horizontal dimension is Valence, vertical is Arousal).

4. RESULTS

The annotations by American and Korean listeners are compared in terms of judgment distribution, agreement levels, and confusion between the two cultural groups. The Chi-square independence test is applied to estimate whether certain distributions were independent with listeners’ cultural background.

4.1 Distribution of Mood Judgment

Table 2 shows the distribution of mood judgment of listeners from both cultural groups across five mood clusters. A Chi-square independence test indicates that the distribution does depend on cultural group \( (p < 0.001, df = 4, \chi^2 = 396.90) \). American listeners chose \( \text{C}_1 \) (passionate) and \( \text{C}_5 \) (aggressive) more often while Korean listeners chose \( \text{C}_2 \) (cheerful), \( \text{C}_3 \) (bittersweet) and \( \text{C}_4 \) (silly/quirky) more often. It is noteworthy that both groups chose \( \text{C}_3 \) (bittersweet) most often among all five clusters. This is different from [9] where both American and Korean listeners chose \( \text{C}_2 \) (cheerful) most often for American Pop songs. This difference may indicate that K-pop songs are generally more likely to express \( \text{C}_3 \) moods than American Pop songs.

<table>
<thead>
<tr>
<th></th>
<th>( \text{C}_1 )</th>
<th>( \text{C}_2 )</th>
<th>( \text{C}_3 )</th>
<th>( \text{C}_4 )</th>
<th>( \text{C}_5 )</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>American</td>
<td>1768</td>
<td>897</td>
<td>2225</td>
<td>311</td>
<td>475</td>
<td>5676</td>
</tr>
<tr>
<td>Korean</td>
<td>959</td>
<td>1321</td>
<td>2598</td>
<td>453</td>
<td>345</td>
<td>5676</td>
</tr>
</tbody>
</table>

Table 2. Judgment distributions across 5 mood clusters.

With the 18-mood group model, a listener may label a song with up to six mood groups. The American listeners chose 13,521 groups in total, resulting in an average of 2.38 groups per song. The Korean listeners chose 7,465 groups in total, which resulted in 1.32 groups per song. The fact that American listeners assigned almost twice as many groups to each song as Korean listeners did may be related to the individualism/collectivism dichotomy found in psychology and cultural studies [13]; Americans tend to be individualistic and are more flexible in accepting a range of ideas (mood groups in this case) than people from collectivistic cultures (often represented by East Asian cultures). Future studies employing more qualitative approaches are warranted to verify this speculation.

Figure 2 shows the distribution of judgments across the 18 mood groups. A chi-square test verified that the distribution is statistically significantly dependent on cultural backgrounds \( (p < 0.001, df = 17, \chi^2 = 1664.49) \). Americans used “gleeful”, “romantic”, “brooding”, “earnest”, “hopeful”, and “dreamy” more often than Koreans, while Koreans applied “sad” more frequently than Americans. Both groups used “angry” and “anxious” very rarely, probably due to the nature of K-pop songs. Similar observations were made in [17], where mood labels applied to Chinese and Western Pop songs were compared and radical moods such as “aggressive” and “anxious” were applied much more infrequently to Chinese songs than to Western songs. This may indicate a cultural difference in music: Chinese and Korean cultures tend to restrain and/or censor the expression of radical or destructive feelings whereas in Western cultures people are willing and free to express all kinds of feelings [10].

Figure 2. Judgment distributions across 18 mood groups (each group is represented by one representative term).

Figure 3 shows the boxplot of the annotations based on the VA dimensional space given by the two groups of listeners. The V-A scores given by Americans are more scattered than those by Koreans, suggesting that Americans were more willing to choose extreme values. In addition, the means and medians indicate that Americans rated the songs with lower arousal values but higher valence values than Koreans \( (p < 0.001 \text{ in non-paired } t \text{-test for both cases}) \). In other words, Americans tended to consider the songs to be less intense and more positive than did Koreans. This may also reflect the cultural difference.
that individuals from Western cultures tend to experience and/or express more positive emotions than those from Eastern cultures [12], and Asians present themselves as less aroused compared to Americans and Europeans [1].

4.2 Agreements Within and Across Cultural Groups
In order to find out whether listeners from the same cultural background agree more with each other than with those from another cultural group, we examined the agreement among annotations provided by listeners in each cultural group as well as across cultural groups. The agreement measures used are the Sokal-Michener coefficient and intra-class correlation (ICC). The former is appropriate for categorical data while the latter is used for numerical data in the V-A space.

4.2.1 Sokal-Michener coefficient
The Sokal-Michener (S-M) coefficient is the ratio of the number of pairs with the same values and the total number of variables [2][9], and therefore a higher value indicates a higher agreement. For instance, if two listeners \(i\) and \(j\) had the same mood judgments on 189 of the 1892 songs, the S-M coefficient between them is approximately 0.1. Table 3 shows the average S-M coefficient aggregated across all pairs of annotators within and across cultural groups on the five-cluster annotations. It is not surprising that Koreans reached a higher agreement than Americans since they are annotating songs originating from their own culture. This is consistent with the findings in [2] and [9], where American listeners reached a higher agreement on the mood of American Pop songs than did Korean and Chinese listeners. The agreement level was the lowest when annotations from American and Korean listeners (cross-cultural) were paired up. The distribution of agreed vs. disagreed judgments is significantly dependent on whether the listeners are from the same cultural group or not, evidenced by the Chi-square test results (Table 3). Listeners from the same cultural group tend to agree more with each other than with those from a different culture.

The analysis is more complex for the 18 group annotation, as each judgment can associate multiple labels with a song. To measure the agreement, we paired up labels applied to a song by any two annotators, and then calculated the S-M coefficient as the proportion of matched pairs among all pairs. For example, if annotator_1 labelled a song S with g1, g2, g3 and annotator_2 labelled it with g1, g4, then there were six annotation pairs and only one of them matched (i.e., g1 matched g1). The S-M coefficient in this case is \(1/6 = 0.17\). Although the denominator increases when more labels are chosen, the chances they get matched also increase. All annotations from all listeners within each cultural group and across cultural groups were paired up in this way, and the resultant S-M coefficients are shown in Table 4. Again, the agreement level within Koreans was higher than that within Americans and also across cultural groups. However, the agreement within Americans was at the same level as the cross-cultural agreement, which is further evidenced by the statistically insignificant result of the Chi-square test.

4.2.2 Intra-Class Correlation
The intra-class correlation (ICC) is a measure of agreement when ratings are given based on a continuous scale [15]. In the case of V-A annotation in this study, there is a different set of raters (listeners) for each item (song), and thus the one-way random model is used to calculate ICC within each group (3 raters) and across both groups (6 raters), for the valence and arousal dimensions. As shown in Table 5, cross-cultural agreement on valence is lower than within-cultural ones. Unlike five mood cluster annotation, both groups showed similar level of agreement on both dimensions. It is also noteworthy that the agreement on arousal annotation is much higher than valence annotation within- and cross-culturally. This is consistent with earlier MIR literature where valence has been recognized as more subjective than arousal [5].

### Table 3. S-M coefficients of the five-cluster annotation within and across cultural groups

<table>
<thead>
<tr>
<th></th>
<th>American</th>
<th>Korean</th>
<th>(\chi^2)</th>
<th>df</th>
<th>(p)</th>
</tr>
</thead>
<tbody>
<tr>
<td>American</td>
<td>0.47</td>
<td>0.43</td>
<td>25.35</td>
<td>1</td>
<td>&lt;0.001</td>
</tr>
<tr>
<td>Korean</td>
<td>0.45</td>
<td>0.56</td>
<td>249.71</td>
<td>1</td>
<td>&lt;0.001</td>
</tr>
</tbody>
</table>

### Table 4. S-M coefficient of the 18-group annotation within and across cultural groups

<table>
<thead>
<tr>
<th></th>
<th>American</th>
<th>Korean</th>
<th>(\chi^2)</th>
<th>df</th>
<th>(p)</th>
</tr>
</thead>
<tbody>
<tr>
<td>American</td>
<td>0.11</td>
<td>0.11</td>
<td>1.72</td>
<td>1</td>
<td>0.054</td>
</tr>
<tr>
<td>Korean</td>
<td>0.11</td>
<td>0.15</td>
<td>156.88</td>
<td>1</td>
<td>&lt;0.001</td>
</tr>
</tbody>
</table>

### Table 5. ICC of Valence Arousal annotations within and across cultural groups

<table>
<thead>
<tr>
<th></th>
<th>American</th>
<th>Korean</th>
<th>Cross-Cultural</th>
</tr>
</thead>
<tbody>
<tr>
<td>Valence</td>
<td>0.27</td>
<td>0.28</td>
<td>0.23</td>
</tr>
<tr>
<td>Arousal</td>
<td>0.55</td>
<td>0.54</td>
<td>0.54</td>
</tr>
</tbody>
</table>

4.3 Confusion Between Cultural Groups
To further our understanding on the difference and similarity of mood perceptions between the two cultural groups, we also examined the disagreement between listeners in the two groups in each of the three types of annotations. For the 5-cluster annotation, Table 6 shows the confusion matrix of the 1,438 songs with agreed labels by at least two listeners in each cultural group. Each cell shows the number of songs labeled as one mood cluster by Koreans (column) and another by Americans (row). The cells on the (highlighted) diagonal are numbers of songs agreed by the two groups, while other cells represent the disagreement between the two groups. The matrix shows that both groups agreed more on C_3 (bittersweet) within themselves (661 and 842 songs respectively as shown by the “Total” cells). The bold numbers indicate major disagreements between the two groups. There are 268 songs Korean listeners judged as C_3 (bittersweet) that Americans judged as C_1 (passionate). The two groups only agreed on C_5 (aggressive) on 18 songs, whereas 49 songs judged as C_5 (aggressive) by Americans were judged by the Koreans as C_1 (passionate).
Table 7 shows the confusion matrix of the seven mood groups (due to space limit) with the most agreed songs by majority vote among the Korean listeners. The biggest confusion/discrepancy is between “exciting” and “gleeful”: 135 songs perceived as “gleeful” by Americans were perceived as “exciting” by Koreans. Other major confusions are between “exciting” and “cheerful”, and “sad” and “mournful.” These moods have similar semantics in terms of valence (both “sad” and “mournful” have low valence values) and arousal (both “exciting” and “gleeful” have high arousal values), which may explain the confusion between these terms. Similarly, there are few songs with disagreement between mood labels with very distinct semantics, such as “exciting” vs. “sad/calm/mournful”; “calm” vs. “cheerful/gleeful”; and “gleeful” vs. “mournful”.

Table 6. Cross-tabulation between 5-cluster annotations across cultural groups

It is interesting to see that a number of songs perceived as “romantic” by Americans were seen as “sad” (31 songs) and “calm” (30 songs) by Koreans. On the other hand, 18 songs perceived as “romantic” by Koreans were viewed as “calm” by Americans. “Romantic” was seldom confused with other high arousal moods such as “exciting” or “cheerful” by either Koreans or Americans, suggesting that both cultures tend to associate “romantic” with low arousal music.

Table 7. Cross-tabulation between 18-annotation groups across cultural groups

For the 2-D annotation, we show the disagreement between the two groups in the four quadrants of the 2-D space (Table 8). Both groups agreed more with listeners from their own cultural group on the first quadrant (+A+V) and the third quadrant (-A-V) (as shown by the “Total” cells). The largest discrepancy was observed between –A+V and –A-V: 116 songs were perceived as having negative arousal and positive valence (-A+V) by Americans but negative valence (-A-V) by Koreans. Similarly, for the songs perceived as having positive arousal by both groups, 118 of them were again perceived as having positive valence (+A+V) by Americans but negative valence (+A-V) by Koreans. This is consistent with our finding that Korean listeners are more likely to label negative moods than Americans (Section 4.1).

Table 8. Cross-tabulation among the four quadrants in 2-D annotations across cultural groups

5. DISCUSSIONS

5.1 Differences and Similarities Between Groups

The results show that mood judgments and the level of agreement are dependent on the cultural background of the listeners. A number of differences were found between the annotations of the two groups. First, Americans assigned a larger number of labels to each song, and applied more extreme valence and arousal values than Koreans (Figure 3). We speculate that perhaps this is related to the fact that the Western culture tends to encourage individualism and divergent thinking more than the Eastern culture [13]. The difference in the number of annotators is another possible explanation. Both of these factors will be further explored in future work. Second, compared to Americans, Koreans were more likely to label songs with negative moods such as “bittersweet”, “sad,” and “mournful” (Table 2, Figure 2), give lower valence values (Figure 3), and agree with each other more often on songs with negative valence (Table 9). These observations were consistent with and supported by findings in previous cultural studies that people from Western cultures tend to experience and/or express more positive emotions than those from Eastern cultures [12]. The fact that Americans in this study could not understand the lyrics of the songs may also have contributed to these results. Sometimes song lyrics and melody may express different moods to invoke complex emotions (e.g., dark humor). In particular, a recent trend among K-pop artists to use faster tempo in Ballad songs may make the melody sound positive or neutral, although the lyrics are sad or melancholy as is the convention for Ballad songs.

It is also found that agreements of within-cultural groups are higher than that of cross-cultural groups based on the comparison of S-M coefficient, and ICC values (on valence only). For within-cultural group agreement, Koreans reached a higher agreement than Americans on 5-cluster annotation, which may be explained by the fact that Koreans were more familiar with the K-pop songs used in this study than Americans. Prior familiarity with
songs was also identified as a factor affecting the agreement level of mood perception in previous studies [2]. Some similarities were also found between the annotations of the two groups: (i) both groups applied and agreed on C.3 (bittersweet) more often than other mood clusters (Tables 2 and 8); (ii) both groups seldom applied radical mood labels such as “aggressive”, “angry”, “anxious” (Table 2 and Figure 2); and (iii) both groups agreed more on songs with +A+V and –A–V values (Table 9). These similarities can potentially be attributed to the nature of the K-pop songs. A previous study comparing mood labels on Western and Chinese Pop songs also found that there were significantly fewer radical mood labels assigned to Chinese Pop songs than to Western songs [17]. This may reflect Eastern Asian preferences for non-aggressive music, perhaps due to their tradition of being more conservative and limiting the expression of feelings [10]. Another likely explanation would be the censorship and regulation1 that still heavily affects the popular music culture in countries like South Korea and China.

5.2 Proposed MIR Evaluation Tasks

One of the main contributions of this study is to build a large cross-cultural dataset for MIR research. The unique characteristics of the dataset built for this study make it suitable for various evaluation tasks involving cross-cultural components. Specifically, for each of the three annotation sets (i.e., 5-clusters, 18-groups, and 2-dimensions), both within- and cross-cultural evaluations can be performed. For the former, both training and test data can be extracted from the datasets with annotations by listeners from the same cultural group (by cross-validation, for example); for the latter, models can be trained by the dataset annotated by listeners in one culture and applied to the dataset annotated by listeners in another culture. These tasks will be able to evaluate whether mood recognition models often used in Western music can be equally applied to 1) non-Western music, specifically K-Pop songs; 2) K-Pop songs annotated by American and/or Korean listeners; and 3) cross-cultural music mood recognition, for both categorical mood classification [17] and dimensional mood regression [5].

6. CONCLUSIONS AND FUTURE WORK

This study analyzed music mood annotations on a large set of K-Pop songs provided by listeners from two distinct cultural groups, Americans and Koreans, using three mood annotation models. By comparing annotations from the two cultural groups, differences and similarities were identified and discussed. The unique characteristics of the dataset built in this study will allow it to be used in future MIR evaluation tasks with an emphasis on cross-cultural applicability of mood recognition algorithms and systems. Future work will include detailed and qualitative investigation on the reasons behind the differences between mood judgments of these two user groups as well as listeners from other cultural groups.

7. ACKNOWLEDGMENT

Funding support: Korean Ministry of Trade, Industry and Energy (Grant #100372) & the A.W. Mellon Foundation.

8. REFERENCES


1 http://freemuse.org/archives/7294
CADENCE DETECTION IN WESTERN TRADITIONAL STANZAIC SONGS USING MELODIC AND TEXTUAL FEATURES

Peter van Kranenburg, Folgert Karsdorp
Meertens Institute, Amsterdam, Netherlands
{peter.van.kranenburg,folgert.karsdorp}@meertens.knaw.nl

ABSTRACT

Many Western songs are hierarchically structured in stanzas and phrases. The melody of the song is repeated for each stanza, while the lyrics vary. Each stanza is subdivided into phrases. It is to be expected that melodic and textual formulas at the end of the phrases offer intrinsic clues of closure to a listener or singer. In the current paper we aim at a method to detect such cadences in symbolically encoded folk songs. We take a trigram approach in which we classify trigrams of notes and pitches as cadential or as non-cadential. We use pitch, contour, rhythmic, textual, and contextual features, and a group of features based on the conditions of closure as stated by Narmour [11]. We employ a random forest classification algorithm. The precision of the classifier is considerably improved by taking the class labels of adjacent trigrams into account. An ablation study shows that none of the kinds of features is sufficient to account for good classification, while some of the groups perform moderately well on their own.

1. INTRODUCTION

This paper presents both a method to detect cadences in Western folk-songs, particularly in folk songs from Dutch oral tradition, and a study to the importance of various musical parameters for cadence detection.

There are various reasons to focus specifically on cadence patterns. The concept of cadence has played a major role in the study of Western folk songs. In several of the most important folksong classification systems, cadence tones are among the primary features that are used to put the melodies into a linear ordering. In one of the earliest classification systems, devised by Ilmari Krohn [10], melodies are firstly ordered according to the number of phrases, and secondly according to the sequence of cadence tones. This method was adapted for Hungarian melodies by Bártok and Kodály [16], and later on for German folk songs by Suppan and Stief [17] in their monumental Melodietypen des Deutschen Volksanges. Bronson [3] introduced a number of features for the study of Anglo-American folk song melodies, of which final cadence and mid-cadence are the most prominent ones. One of the underlying assumptions is that the sequence of cadence tones is relatively stable in the process of oral transmission. Thus, variants of the same melody are expected to end up near to each other in the resulting ordering.

From a cognitive point of view, the perception of closure is of fundamental importance for a listener or singer to understand a melody. In terms of expectation [8, 11], a final cadence implies no continuation at all. It is to be expected that specific features of the songs that are related to closure show different values for cadential patterns as compared to non-cadential patterns. We include a subset of features that are based on the conditions of closure as stated by Narmour [11, p.11].

Cadence detection is related to the problem of segmentation, which is relevant for Music Information Retrieval [21]. Most segmentation methods for symbolically represented melodies are either based on pre-defined rules [4, 18] or on statistical learning [1, 9, 12]. In the current paper, we focus on the musical properties of cadence formulas rather than on the task of segmentation as such.

Taking Dutch folk songs as case study, we investigate whether it is possible to derive a general model of the melodic patterns or formulas that specifically indicate melodic cadences using both melodic and textual features. To address this question, we take a computational approach by employing a random forest classifier (Sections 5 and 6).

To investigate which musical parameters are of importance for cadence detection, we perform an ablation study in which we subsequently remove certain types of features in order to evaluate the importance of the various kinds of features (Section 7).

2. DATA

We perform all our experiments on the folk song collection from the Meertens Tune Collections (MTC-FS, version 1.0), which is a set of 4,120 symbolically encoded Dutch folk songs.1 Roughly half of it consists of transcriptions from field recordings that were made in the Netherlands during the 20th century. The other half is taken from song books that contain repertoire that is directly related to the recordings. Thus, we have a coherent collection of songs that reflects Dutch everyday song culture in the early 20th century. Virtually all of these songs have a stanzaic structure. Each stanza repeats the melody, and each stanza

1 Available from: http://www.liederenbank.nl/mtc.
consists of a number of phrases. Both in the transcriptions and in the song books, phrase endings are indicated. Figure 1 shows a typical song from the collection. The language of the songs is standard Dutch with occasionally some dialect words or nonsense syllables. All songs were digitally encoded by hand at the Meertens Institute (Amsterdam) and are available in Humdrum **kern format. The phrase endings were encoded as well and are available for computational analysis and modeling.

3. OUR APPROACH

Our general approach is to isolate trigrams from the melodies and to label those as either cadential or non-cadential. A cadential trigram is the last trigram in a phrase. We compare two kinds of trigrams: trigrams of successive notes (note-trigrams), and trigrams of successive pitches (pitch-trigrams), considering repeated pitches as one event. In the case of pitch-trigrams, a cadence pattern always consists of the three last unique pitches of the phrase. There are two reasons for including pitch-trigrams. First, pitch repetition is often caused by the need to place the right number of syllables to the melody. It occurs that a quarter note in one stanza corresponds to two eighth notes in another stanza because there is an extra syllable at that spot in the song text. Second, in models of closure in melody [11,15] successions of pitches are of primary importance.

Figure 1 depicts all pitch-trigrams in the presented melody. The trigram that ends on the final note of a phrase is a cadential trigram. These are indicated in bold. Some cadential trigrams cross a phrase boundary when the next phrase starts with the same pitch.

From each trigram we extract a number of feature values that reflect both melodic and textual properties. We then perform a classification experiment using a Random Forest Classifier [2]. This approach can be regarded a ‘bag-of-trigrams’ approach, where each prediction is done independently of the others, i.e. all sequential information is lost. Therefore, as a next step we take the labels of the direct neighboring trigrams into account as well. The final classification is then based on a majority vote of the predicted labels of adjacent trigrams. These steps will be explained in detail in the next sections.

![Figure 1. Examples of pitch-trigrams. The cadential trigrams are indicated in bold.](image)

4. FEATURES

We represent each trigram as a vector of feature values. We measure several basic properties of the individual pitches and of the pattern as a whole. The code to automatically extract the feature values was written in Python, using the music21 toolbox [5]. The features are divided into groups that are related to distinct properties of the songs. Some features occur in more than one group. The following overview shows all features and in parentheses the value for the first trigram in Figure 1. Detailed explanations are provided in sections 4.1 and 4.2.

Pitch Features
- Scale degree Scale degrees of the first, second, and third item (5, 1, 3).
- Range Difference between highest and lowest pitch (4).
- Has contrast third Whether there are both even and odd scale degrees in the trigram (False).

Contour Features
- Contains leap Whether there is a leap in the trigram (True).
- Is ascending Whether the first and second intervals, and both are ascending (False, True, False).
- Is descending Whether the first and second intervals, and both are descending (True, False, False).
- Large-small Whether the first interval is large and the second is small (True).
- Registral change Whether there is a change in direction between the first and the second interval (True).

Rhythmic Features
- Beat strength The metric weights of the first, second and third item (0.25, 1.0, 0.25).
- Min beat strength The smallest metric weight (0.25).
- Next is rest Whether a rest follows the first, second and third item (False, False, False).
- Short-long Whether the second item is longer than the first, and the third is longer than the second (False, False).
- Meter The meter at the beginning of the trigram (“6/8”).

Textual Features
- Rhymer Whether a rhyme word ends at the first, second and third item (False, False, False).
- Word stress Whether a stressed syllable is at the first, second and third item (True, True, True).
- Distance to last rhyme Number of notes between the last the first, second and third item and the last rhyme word or beginning of the melody (0, 1, 2).

Narmour Closure Features
- Beat strength The metric weights of the first, second and third item (0.25, 1.0, 0.25).
- Next is rest Whether a rest follows the first, second and third item (False, False, False).
- Short-long Whether the second item is longer than the first, and the third is longer than the second (False, False).
- Large-small Whether the first interval is large (≥ fifth) and the second is small (≤ third) (True).
- Registral change Whether there is a change in direction between the first and the second interval (True).

Contextual Features
- Next is rest third Whether a rest or end of melody follows the third item (False).
- Distance to last rhyme Number of notes between the last the first, second and third item and the last rhyme word or beginning of the melody (0, 1, 2).

4.1 Melodic Features

Several of the features need some explanation. In this section we describe the melodic features, while in the next section, we explain how we extracted the textual features.

HasContrastThird is based on the theory of Jos Smits-Van Waesberghe [15], the core idea of which is that a melody gets its tension and interest by alternating between
pitches with even and uneven scale degrees, which are two contrasting series of thirds.

The metric weight in the Rhythmic features is the beat-strength as implemented in music21’s meter model.

The Narmour features are based on the six (preliminary) conditions of closure that Narmour states at the beginning of his first book on the Implication-Realisation theory [11, p.11]: “[...] melodic closure on some level occurs when 1. a rest, an onset of another structure, or a repetition interrupts an implied pattern; 2. metric emphasis is strong; 3. consonance resolves dissonance; 4. duration moves cumulatively (short note to long note); 5. intervallic motion moves from large interval to small interval; 6. registral direction changes (up to down, down to up, lateral to down, up to lateral, or down to lateral). Of course, these six may appear in any combination.” Because the melodies are monophonic, condition 3 has no counterpart in our feature set.

The contextual features are not features of the trigram in isolation, but are related to the position in the melody. In an initial experiment we found that the distance between the first note of the trigram and the last cadence is an important predictor for the next cadence. Since this is based on the ground-truth label, we cannot include it directly into our feature set. Since we expect rhyme in the text to have a strong relation with cadence in the melody, we include the distance to the last rhyme word in number of notes.

4.2 Textual Features

In many poetical texts, phrase boundaries are determined by sequences of rhyme. These establish a structure in a text, both for aesthetics pleasure and memory aid [14]. In folk music, phrasal boundaries established by sequences of rhyme are likely to relate to phrases in the melody.

We developed a rhyme detection system which allows us to extract these sequences of rhyming lyrics. Because of orthographical ambiguities (e.g. cruise, where /kru:/ is represented by ui whereas in muse it is represented by u), it is not as straightforward to perform rhyme detection on orthographical representations of words. Therefore, we transform each word into its phonological representation (e.g. cruise becomes /kru:/ and bike /baik/).

We approach the problem of phonemicization as a supervised classification task, where we try to predict for each character in a given word its corresponding phoneme. We take a sliding window-based approach where for each focus character (i.e. the character for which we want to predict its phonemic representation) we extract as features \( n \) characters to the left of the focus character, \( n \) characters to the right, and the focus character itself. Figure 3 provides a graphical representation of the feature vectors extracted for the word cruise. The fourth column represents the focus character with a context of three characters before and three after the focus character. The last column represents the target phonemes which we would like to predict. Note that the first target phoneme in Figure 3 is preceded by an apostrophe (’k), which represents the stress position on the first (and only) syllable in cruise. This symbolic notation of stress in combination with phonology allows us to simultaneously extract a phonological representation of the input words as well as their stress patterns. For all words in the lyrics in the dataset we apply our sliding window approach with \( n = 5 \), which serves as input for the supervised classifier. In this paper we make use of a \( k = 1 \) Nearest Neighbor Classifier as implemented by [6] using default settings, which was trained on the data of the e-Lex database\(^2\). In the running text of our lyrics, 89.5\% of the words has a direct hit in the instance base, and for the remaining words in many cases suitable nearest neighbors were found. Therefore, we consider the phonemicization sufficiently reliable.

We assume that only content words (nouns, adjectives, verbs and adverbials) are possible candidate rhyme words. This assumption follows linguistic knowledge as phrases typically do not end with function words such as determiners, prepositions, etcetera. Function words are part of a closed category in Dutch. We extract all function words from the lexical database e-Lex and mark for each word in each lyric whether it is a function word.

We implemented rhyme detection according to the rules for Dutch rhyme as stated in [19]. The algorithm is straightforward. We compare the phoneme-representations of two words backwards, starting at the last phoneme, until we reach the first vowel, excluding schwas. If all phonemes

\[
\begin{array}{cccc}
\_ & \_ & c & r & u & i & k \\
\_ & \_ & c & r & u & i & s & r \\
c & r & u & i & s & e & w:
\end{array}
\]

\[
\begin{array}{cccc}
r & u & i & s & e & \_ & \_ & 0 \\
u & i & s & e & \_ & \_ & 0
\end{array}
\]

Figure 3. Example sliding window for phoneme classification.

\( \text{http://tst-centrale.org/en/producten/lexica/e-lex/7-25} \)

Figure 2. Rhyme as detected by our method. The first line shows the original text after removing non-content words. The second line shows the phonological representations of the words (in SAMPA notation). The third line shows whether rhyme is detected (’True’ if a rhyme word \textit{ends} at the corresponding note).
and the vowel are exactly the same, the two words rhyme.

As an example we take *kinderen* (‘children’) and *hin-
deren* (‘to hinder’). The phoneme representations as pro-
duced by our method are /kɪndərən/ and /hmɪndərən/. The
first vowel starting from the back of the word, exclud-
ing the schwas (/ə/), is /ɪ/. Starting from this vowel,
the phoneme representations of both words are identical (/ɪndərən/). Therefore these words rhyme.

We also consider literal repetition of a word as ‘rhyme’,
but not if a sequence of words is repeated literally, such
as in the example in Figure 1. Such repetition of entire
phrases occurs in many songs. Labeling all words as rhyme
words would weaken the relation with cadence or ‘end-of-
sentence’. We only label the last word of repeated phrases
as a rhyme word. Figure 2 shows an example.

5. CLASSIFICATION WITH SINGLE LABELS

As a first approach we consider the trigrams independently.
A melody is represented as ‘bag-of-trigrams’. Each tri-
gram has a ground-truth label that is either ‘cadence’ or
‘no cadence’, as depicted in Figure 1 for pitch-trigrams.

We employ a Random Forest classifier [2] as imple-
mented in the Python library scikit-learn [13]. This classi-
fier combines *n* decision trees (*predictors*) that are trained
on random samples extracted from the data (with replace-
ment). The final classification is a majority vote of the pre-
dictions of the individual trees. This procedure has proven
to perform more robustly than a single decision tree and
is less prone to over-fitting the data. Given the relatively
large size of our data set, we set the number of predictors
to 50 instead of the default 10. For the other parameters,
we keep the default values.

The evaluation is performed by 10-fold cross-validation.
One non-trivial aspect of our procedure is that we construct
the folds at the level of the songs, rather than at that of indi-
vidual trigrams. Since it is quite common for folk songs to
have phrases that are literally repeated, folding at the level
of the songs, rather than at that of individual trigrams. This
procedure is applied throughout this paper.

The results are shown in Table 1. For both classes aver-
ages of the values for the precision, the recall and the $F_1$
measure over the folds are included, as well as the standard
deviation of the $F_1$ measure, which indicates the variation
over the folds. The number of items in both classes (*sup-
port*) shows that cadences are clearly a minority class.

We observe that the note-trigrams lead to slightly better
cadence-detection as compared to pitch-trigrams. Appar-
tently, the repetition of pitches does not harm the discrim-
inability. Furthermore, there is an unbalance between the
precision and the recall of the cadence-trigrams. The pre-
cision is rather high, while the recall is moderate.

6. CLASSIFICATION WITH LABEL TRIGRAMS

When our cadence detection system predicts the class of a
new trigram, it is oblivious of the decisions made for earlier
predictions. One particularly negative effect of this near-
sightedness is that the classifier frequently predicts two (or
even more) cadences in a row, which, given our our train-
ing material, is extremely unlikely. We attempt to circum-
vent this ‘defect’ using a method, developed by [20] that
predicts trigrams of class labels instead of single, binary
labels. Figure 4 depicts the standard single class classi-
fication setting, where each trigram is predicted indepen-
dent of all other predictions. In the label trigram setting
(see Figure 5), the original class labels are replaced with
the class label of the previous trigram, the class label of
the current trigram and the label of the next trigram. The
learning problem is transformed into a sequential learn-
ing problem with two stages. In the first stage we predict
for each trigram a label trigram $y^{(3)} = (y_1, y_2, y_3)$ where
$y \in \{0, 1\}$. To arrive at the final single class predictions
(i.e. is it a cadence or not), in the second stage we take
the majority vote over the predictions of the focus trigram
and those of its immediate left and right neighboring tri-
grams. Take $t_4$ in Figure 5 as an example. It predicts that
the current trigram is a cadence. The next trigram and the
previous trigram also predict it to be a cadence and based
on this majority vote, the final prediction is that $t_4$ is a
cadence. Should $t_5$ and $t_3$ both have predicted the zero
class (e.g. $y^{(5)} = (0, 0, 0)$ and $y^{(3)} = (0, 1, 0)$), the ma-
majority vote would be 0. The advantage of this method is
that given the negligible number of neighboring cadences
in our training data, we can virtually rule out the possibility
to erroneously predict two or more cadences in a row.

Table 2 shows the performance of the label-trigram clas-
sifier for both classes and both for pitch and note trigrams.
The values show an important improvement for the precision
of cadence-detection and a slight improvement of the
recall. The lower number of false positives is what we ex-
erct by observing the classification of adjacent trigrams
as ‘cadence’ in the case of the single-label classifier.

<table>
<thead>
<tr>
<th>Class</th>
<th>pr</th>
<th>rec</th>
<th>$F_1$</th>
<th>$\sigma_{F_1}$</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>note-trigrams</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cadence</td>
<td>0.84</td>
<td>0.72</td>
<td>0.78</td>
<td>0.01</td>
<td>23,925</td>
</tr>
<tr>
<td>nocadence</td>
<td>0.96</td>
<td>0.98</td>
<td>0.97</td>
<td>0.01</td>
<td>183,780</td>
</tr>
<tr>
<td>pitch-trigrams</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cadence</td>
<td>0.85</td>
<td>0.69</td>
<td>0.76</td>
<td>0.01</td>
<td>23,838</td>
</tr>
<tr>
<td>nocadence</td>
<td>0.95</td>
<td>0.98</td>
<td>0.96</td>
<td>0.00</td>
<td>130,992</td>
</tr>
</tbody>
</table>

Table 1. Results for single labels.

<table>
<thead>
<tr>
<th>Class</th>
<th>pr</th>
<th>rec</th>
<th>$F_1$</th>
<th>$\sigma_{F_1}$</th>
<th>support</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>note-trigrams</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cadence</td>
<td>0.89</td>
<td>0.72</td>
<td>0.80</td>
<td>0.01</td>
<td>23,925</td>
</tr>
<tr>
<td>nocadence</td>
<td>0.96</td>
<td>0.99</td>
<td>0.98</td>
<td>0.00</td>
<td>183,780</td>
</tr>
<tr>
<td>pitch-trigrams</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cadence</td>
<td>0.89</td>
<td>0.71</td>
<td>0.79</td>
<td>0.01</td>
<td>23,838</td>
</tr>
<tr>
<td>nocadence</td>
<td>0.95</td>
<td>0.98</td>
<td>0.97</td>
<td>0.01</td>
<td>130,992</td>
</tr>
</tbody>
</table>

Table 2. Results for classification with label trigrams.
7. ABLATION STUDY

To study the importance of the various kinds of features, we perform an ablation study. We successively remove each of the groups of features as defined in section 4 from the full set and do a classification experiment with the remaining features. Subsequently, we perform a similar series of classification experiments, but now with each single group of features. The first series shows the importance of the individual groups of features, and the second series shows the predictive power for each of the groups. Because the groups are assembled according to distinct properties of music and text, this will give insight in the importance of various musical and textual parameters for cadence detection. We use the label-trigram classifier with the note-trigrams, which performed best on the full set.

We expect occurrence of rests to be a very strong predictor, because according to our definition a ‘rest’ always follows after the final cadence, and we know that in our corpus rests almost exclusively occur between phrases. Therefore, we also take the three features that indicate whether a rest occurs in the trigram or directly after it, as a separate group. The performance when leaving these three features out will show whether they are crucial for cadence detection.

Table 3 shows the evaluation measures for each of the feature subsets. Precision, recall and $F_1$ for class ‘cadence’ are reported. Again, the values are averaged over 10 folds.

We see that none of the single groups of features is crucial for the performance that was achieved with the complete set of features. The basic melodic features ($F_{pitch}$, $F_{contour}$, and $F_{rhythmic}$) all perform very bad on their own, showing low to extremely low recall values. The contour features even do not contribute at all. Only the rhythmic features yield some performance. The features on rest are included in the set of rhythmic features. The classification with just the features on rest, $F_{rest}$ shows very high precision and low recall. Still, the recall with all rhythmic features is higher than only using the rest-features. Since rests are so tightly related to cadences in our corpus, the high precision for $F_{rest}$ is what we expected. If we exclude the rest-features, the precision stays at the same level as for the entire feature set and the recall drops with 0.06, which shows that only a minority of the cadences exclusively rely on rest-features to be detected.

The set of features that is based on the conditions of closure as formulated by Narmour shows high precision and low recall. Especially the high precision is interesting, because this confirms Narmour’s conditions of closure. Apparently, most patterns that are classified as cadence based on this subset of features, are cadences indeed. Still, the low recall indicates that there are many cadences that are left undetected. One cause could be that the set of conditions as stated by Narmour is not complete, another cause could be the discrepancy between our features and Narmour’s conditions. Further investigation would be necessary to shed light on this. Removing the Narmour-based features from the full feature set does not have a big impact. The other features have enough predictive power.

The textual features on their own show moderate precision and very moderate recall. They are able to discern certain kinds of cadences to a certain extent, while missing most of the other cadences. The drop of 0.14 in recall for $F_{all} \setminus F_{textual}$ as compared to the full set shows that text features are crucial for a considerable number of cadences to be detected. The same applies to a somewhat lesser extent to contextual features. Removing the contextual features from the full set causes a drop of 0.05 in the recall, which is considerable but not extreme. It appears that the group of cadence trigrams for which the contextual features are crucial is not very big.
8. CONCLUSION AND FUTURE WORK
In this paper we developed a system to detect cadences in Western folk songs. The system makes use of a Random Forest Classifier that on the basis of a number of handcrafted features (both musical and textual) is able to accurately locate cadences in running melodies. In a follow-up experiment we employ a method, originally developed for textual sequences, that predicts label-trigrams instead of the binary labels ‘cadence’ or ‘non-cadence’. We show that incorporating the predictions of neighboring instances into the final prediction, has a strong positive effect on precision without a loss in recall.

In the ablation study we found that all groups of features, except for the contour features, contribute to the overall classification, while none of the groups is crucial for the majority of the cadences to be detected. This indicates that cadence detection is a multi-dimensional problem for which various properties of melody and text are necessary.

The current results give rise to various follow-up studies. A deeper study to the kinds of errors of our system will lead to improved features and increased knowledge about cadences. Those that were detected exclusively by textual features form a particular interesting case, possibly giving rise to new melodic features. Next, n-grams other than trigrams as well as skip-grams [7] could be used, we will compare the performance of our method with existing symbolic segmentation algorithms, and we want to make use of other features of the text such as correspondence between syntactic units in the text and melodic units in the melody.

9. REFERENCES
DISCOVERING TYPICAL MOTIFS OF A RĀGA FROM ONE-LINERS OF SONGS IN CARNATIC MUSIC

Shrey Dutta  
Dept. of Computer Sci. & Engg.  
Indian Institute of Technology Madras  
shrey@cse.iitm.ac.in

Hema A. Murthy  
Dept. of Computer Sci. & Engg.  
Indian Institute of Technology Madras  
hema@cse.iitm.ac.in

ABSTRACT

Typical motifs of a rāga can be found in the various songs that are composed in the same rāga by different composers. The compositions in Carnatic music have a definite structure, the one commonly seen being pallavi, anupallavi and charanam. The tala is also fixed for every song.

Taking lines corresponding to one or more cycles of the pallavi, anupallavi and charanam as one-liners, one-liners across different songs are compared using a dynamic programming based algorithm. The density of match between the one-liners and normalized cost along-with a new measure, which uses the stationary points in the pitch contour to reduce the false alarms, are used to determine and locate the matched pattern. The typical motifs of a rāga are then filtered using compositions of various rāgas. Motifs are considered typical if they are present in the compositions of the given rāga and are not found in compositions of other rāgas.

1. INTRODUCTION

Melody in Carnatic music is based on a concept called rāga. A rāga in Carnatic music is characterised by typical phrases or motifs. The phrases are not necessarily scale-based. They are primarily pitch trajectories in the time-frequency plane. Although for annotation purposes, rāgas in Carnatic are based on 12 srutis (or semitones), the gamakas associated with the same semitone can vary significantly across rāgas. Nevertheless, although the phrases do not occupy fixed positions in the time-frequency (t-f) plane, a listener can determine the identity of a rāga within few seconds of an alāpana. An example, is a concert during the “music season” in Chennai, where more than 90% of the audience can figure out the rāga. This despite the fact that more than 80% of the audience are nonprofessionals. The objective of the presented work is to determine typical motifs of a rāga automatically. This is obtained by analyzing various compositions that are composed in a particular rāga. Unlike Hindustani music, there is a huge repository of compositions that have been composed by a number of composers in different rāgas. It is often stated by musicians that the famous composers have composed such that a single line of a composition is replete with the motifs of the rāga. In this paper, we therefore take one-liners of different compositions and determine the typical motifs of the rāga.

Earlier work, [9, 10], on identifying typical motifs depended on a professional musician who sung the typical motifs for that rāga. These typical motifs were then spotted in alāpanas which are improvisational segments. It was observed that the number of false alarms were high. High ranking false alarms were primarily due to partial matches with the given query. Many of these were considered as an instance of the queried motif by some musicians. Alapana is an improvisational segment, the rendition of the same motif could be different across alapanas especially among different schools. On the other hand, compositions in Carnatic music are rendered more or less in a similar manner. Although the music evolved through the oral tradition and fairly significant changes have crept into the music, compositions renditions do not vary very significantly across different schools. The number of variants for each line of the song can vary quite a lot though. Nevertheless, the meter of motifs and the typical motifs will generally be preserved.

It is discussed in [15] that not all repeating patterns are interesting and relevant. In fact, the vast majority of exact repetitions within a music piece are not musically interesting. The algorithm proposed in [15] mostly generates interesting repeating patterns along with some non-interesting ones which are later filtered during post-processing. The work presented in this paper is an attempt from a similar perspective. The only difference is that typical motifs of rāgas need not be interesting to a listener. The primary objective for discovering typical motifs, is that these typical motifs can be used to index the audio of a rendition. Typical motifs could also be used for rāga classification. The proposed approach in this work generates similar patterns across one-liners of a rāga. From these similar patterns, the typical motifs are filtered by using compositions of various rāgas. Motifs are considered typical of a rāga if they are present in the compositions of a particular rāga and are NOT found in other rāgas. This filtering approach is similar to anti-corpus approach of Conklin [6, 7] for the discovery of distinctive patterns.
Most of the previous work, regarding discovery of repeated patterns of interest in music, is on western music. In [11], B. Jansen et al. discusses the current approaches on repeated pattern discovery. It discusses string based methods and geometric methods for pattern discovery. In [14], Lie Lu et al. used constant Q transforms and proposed a similarity measure between musical features for doing repeated pattern discovery. In [15], Meredith et al. presented Structure Induction Algorithms (SIA) using a geometric approach for discovering repeated patterns that are musically interesting to the listener. In [4, 5], Collins et al. introduced improvements in Meredith’s Structure Induction Algorithms. There has also been some significant work on detecting melodic motifs in Hindustani music by Joe Cheri Ross et. al. [16]. In this approach, the melody is converted to a sequence of symbols and a variant of dynamic programming is used to discover the motif.

In a Carnatic music concert, many listeners from the audience are able to identify the rāga at the very beginning of the composition, usually during the first line itself — a line corresponds to one or more tala cycles. Thus, first lines of the compositions could contain typical motifs of a rāga. A pattern which is repeated within a first line could still be not specific to a rāga. Whereas, a pattern which is present in most of the first lines could be a typical motif of that rāga. Instead of just using first lines, we have also used other one-liners from compositions, namely, lines from the pallavi, anupallavi and charanam. In this work, an attempt is made to find repeating patterns across one-liners and not within a one-liner. Typical motifs are filtered from the generated repeating patterns during post processing. These typical motifs are available online 1

The length of the typical motif to be discovered is not known a priori. Therefore there is a need for a technique which can itself determine the length of the motif at the time of discovering it. Dynamic Time Warping (DTW) based algorithms can only find a pattern of a specific length since it performs end-to-end matching of the query and test sequence. There is another version of DTW known as Unconstrained End Point-DTW (UE-DTW) that can match the whole query with a partial test but still the query is not partially matched. Longest Common Subsequence (LCS) algorithm on the other hand can match the partial query with partial test sequence since it looks for a longest common subsequence which need not be end-to-end. LCS by itself is not appropriate as it requires discrete symbols and does not account for local similarity. A modified version of LCS known as Rough Longest Common Subsequence takes continuous symbols and takes into account the local similarity of the longest common subsequence. The algorithm proposed in [13] to find rough longest common sequence between two sequences fits the bill for our task of motif discovery. An example of RLCS algorithm matching two partial phrases is shown in Figure 1. The two music segments are represented by their tonic normalized smoothed pitch contours [9, 10]. The stationary points, where the first derivative is zero, of the tonic normalized pitch contour are first determined. The points are then interpolated using cubic Hermite interpolation to smooth the contour.

In previous uses of RLCS for motif spotting task [9,10], a number of false alarms were observed. One of the most prevalent false alarms is the test phrase with a sustained note which comes in between the notes of the query. The slope of the linear trend in stationary points along with its standard deviation is used to address this issue.

The rest of the paper is organized as follows. In Section 2 the use of one-liners of compositions to find motifs is discussed. Section 3 discusses the optimization criteria to find the rough longest common subsequence. Section 4 describes the proposed approach for discovering typical motifs of rāgas. Section 5 describe the dataset used in this work. Experiments and results are presented in Section 6.

2. ONE-LINERS OF SONGS

As previously mentioned, first line of the composition contains the characteristic traits of a rāga. The importance of the first lines and the rāga information it holds is illustrated in great detail in the T. M. Krishna’s book on Carnatic music [12]. T. M. Krishna states that opening section called “pallavi” directs the melodic flow of the rāga. Through its rendition, the texture of the rāga can be felt. Motivated by this observation, an attempt is made to verify the conjecture that typical motifs of a rāga can be obtained from the first lines of compositions.

Along with the lines from pallavi, we have also selected few lines from other sections, namely, ‘anupallavi’ and ‘charanam’. Anupallavi comes after pallavi and the melodic movements in this section tend to explore the rāga in the higher octave [12]. These lines are referred to as one-liners for a rāga.

3. OPTIMIZATION CRITERIA TO FIND ROUGH LONGEST COMMON SUBSEQUENCE

The rough longest common subsequence (rlcs) between two sequences, \( X = \langle x_1, x_2, \cdots, x_n \rangle \) and \( Y = \langle y_1, y_2, \cdots \rangle \)

\[ \text{rlcs}(X, Y) = \langle x_{k_1}, x_{k_2}, \cdots, x_{k_m} \rangle \]

where \( k_1 < k_2 < \cdots < k_m \) and \( x_{k_i} \) is an element in \( X \) and \( y_{k_i} \) is an element in \( Y \) for all \( i \).
...length of $S$ this work, a few quantities need to be defined. Cussing the optimization measures used to find the rlcs in more constraint is used to reduce false alarms. Before discussing the optimization measures used to find the rlcs in this work, a few quantities need to be defined.

\[ l_{XY}^w = \sum_{k=1}^{s} \text{sim}(x_{i_k}, y_{j_k}) \] (1)

\[ g_X = i_s - i_1 + 1 - s \] (2)

\[ g_Y = j_s - j_1 + 1 - s \] (3)

Let $S_{XY} = \{(x_{i_1}, y_{j_1}), (x_{i_2}, y_{j_2}), \ldots, (x_{i_s}, y_{j_s})\}, 1 \leq i_1 < i_2 < \cdots < i_s \leq n, 1 \leq j_1 < j_2 < \cdots < j_s \leq m$; be a rough common subsequence (rcs) of length $s$ and $\text{sim}(x_{i_k}, y_{j_k}) \in [0, 1]$ be the similarity between $x_{i_k}$ and $y_{j_k}$ for $k = 1, \cdots, s$. Equation (1) defines the weighted length of $S_{XY}$ as sum of similarities, $\text{sim}(x_{i_k}, y_{j_k}), k = 1, \cdots, s$. Thus, weighted length is less than or equal to $s$.

The number of points in the shortest substring of sequence $X$, containing the rcs $S_{XY}$, that are not the part of the rcs $S_{XY}$ are termed as gaps in $S_{XY}$ with respect to sequence $X$ as defined by Equation (2). Similarly, Equation (3) defines the gaps in $S_{XY}$ with respect to sequence $Y$. Small gaps indicate that the distribution of rcs is dense in that sequence.

The optimization measures to find the rlcs are described as follows.

### 3.1 Density of the match

Equation (4) represents the distribution of the rcs $S_{XY}$ in the sequences $X$ and $Y$. This is called density of match, $\delta_{S_{XY}}$. This quantity needs to be maximized to make sure the subsequence, $S_{XY}$, is locally similar. $\beta \in [0, 1]$ weighs the individual densities in sequences $X$ and $Y$.

\[ \delta_{S_{XY}} = \beta \frac{l_{w}^{S_{XY}}}{l_{w}^{S_{XY}} + g_X} + (1 - \beta) \frac{l_{w}^{S_{XY}}}{l_{w}^{S_{XY}} + g_Y} \] (4)

### 3.2 Normalized weighted length

The weighted length of rcs is normalized as shown in Equation (5) to restrict its range to $[0, 1]$, $n$ and $m$ are the lengths of sequences $X$ and $Y$, respectively.

\[ \hat{l}_{S_{XY}}^w = \frac{l_{w}^{S_{XY}}}{\min(m, n)} \] (5)

### 3.3 Linear trend in stationary points

As observed in [9, 10], the rlcs obtained using only the above two optimization measures suffered from a large number of false alarms for the motif spotting task. The false alarms generally constituted of long and sustained notes.

---

**Figure 2.** (a) Pitch contour of the five phrases which are considered similar. Stationary points are marked in green and red for the true positives and false alarms respectively. (b) Pitch values only at the stationary points. Slope of the linear trend in stationary points along-with its standard deviation helps in reducing the false alarms.
This resulted in good normalised weighted lengths and density. To address this issue, the slope and standard deviation of the slope of the linear trend in stationary points of a phrase are estimated. Figure 2 shows a set of phrases. This set has five phrases which are termed as similar phrases based on their density of match and normalized weighted length. The first two phrases, shown in green, are true positives while the remaining, shown in red, are false alarms. Figure 2 also shows the linear trend in stationary points for the corresponding phrases. It is observed that the trends are similar for true positives when compared to that of the false alarms. The slope of the linear trend for the fifth phrase is more. Two slopes are of different sign and thus, the penalization is maximized. This similarity has negative value when the positive, the similarity in the linear trend should be high.

The standard deviation of the linear trend is used to reduce the deviation is less. Therefore, a combination of the slope and standard deviation is used to filter alarms. The slope of the linear trend for the fifth phrase is similar for true positives when compared to that of the false positives while the remaining, shown in red, are false alarms. It is observed that the trends are similar for true positives when compared to that of the false alarms. The slope of the linear trend for the fifth phrase is more. Two slopes are of different sign and thus, the penalization is maximized. This similarity has negative value when the positive, the similarity in the linear trend should be high.

Let the stationary points in the shortest substring of sequences X and Y containing the rcs $S_{XY}$ be $(x_{q_1}, x_{q_2}, \cdots, x_{q_{t_x}})$ and $(y_{t_y}, y_{t_y + 1}, \cdots, y_{t_y + t_y})$ respectively, where $t_x$ and $t_y$ are the number of stationary points in the respective substrings. Equation (6) estimates the slope of the linear trend, of stationary points in the substring of sequence X, as the mean of the first difference of stationary points, which is same as $\frac{x_{q_{k+1}} - x_{q_k}}{t_x - 1}$ [8]. Its standard deviation is estimated using Equation (7). Similarly, $\mu_{XY}$ and $\sigma_{XY}$ are also estimated for substring of sequence Y.

$$\mu_{XY} = \frac{1}{t_x - 1} \sum_{k=1}^{t_x-1} (x_{q_{k+1}} - x_{q_k})$$ (6)

$$\sigma_{XY}^2 = \frac{1}{t_x - 1} \sum_{k=1}^{t_x-1} ((x_{q_{k+1}} - x_{q_k}) - \mu_{XY})^2$$ (7)

Let $z_1 = \mu_{XY}$, $\sigma_{XY}$, and $z_2 = \mu_{XY}$, $\sigma_{XY}$. For a true positive, the similarity in the linear trend should be high. Equation (8) calculates this similarity which needs to be maximized. This similarity has negative value when the two slopes are of different sign and thus, the penalization is more.

$$\rho_{XY} = \begin{cases} \max(z_1, z_2) - \min(z_1, z_2) & \text{if } z_1 < 0; z_2 < 0 \\ \min(z_1, z_2) / \max(z_1, z_2) & \text{otherwise} \end{cases}$$ (8)

Finally, Equation (9) combines these three optimization measures to get a score value which is maximized. Then the rcs, $R_{XY}$, between the sequences X and Y is defined, as an rcs with a maximum score, in Equation (10). The rcs $R_{XY}$ can be obtained using dynamic programming based approach discussed in [9, 13].

$$\text{Score}_{XY} = \alpha \delta_{XY} \frac{\mu_{XY}}{\sigma_{XY}} + (1 - \alpha) \rho_{XY}$$ (9)

$$R_{XY} = \text{argmax} \left( \frac{\text{Score}_{XY}}{S_{XY}} \right)$$ (10)

### Table 1. Database of one-liners

<table>
<thead>
<tr>
<th>Rāga</th>
<th>Number of one-liners</th>
<th>Average duration (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bhairavi</td>
<td>17</td>
<td>16.87</td>
</tr>
<tr>
<td>Kamboji</td>
<td>12</td>
<td>13</td>
</tr>
<tr>
<td>Kalyani</td>
<td>9</td>
<td>12.76</td>
</tr>
<tr>
<td>Shankarabharanam</td>
<td>12</td>
<td>12.55</td>
</tr>
<tr>
<td>Vārali</td>
<td>9</td>
<td>9.40</td>
</tr>
<tr>
<td><strong>Overall</strong></td>
<td><strong>59</strong></td>
<td><strong>12.91</strong></td>
</tr>
</tbody>
</table>

4. **DISCOVERING TYPICAL MOTIFS OF RĀGAS**

Typical motifs of a rāga are discovered using one-liners of songs in that rāga. For each voiced part in an oneliner of a rāga, rlc is found with the overlapping windows in voiced parts of other one-liners of that rāga. Only those rlc is selected whose score values and lengths (in seconds) are greater than thresholds $\tau_{scr}$ and $\tau_{len}$ respectively. The voiced parts which generated no rlc is interpreted to have no motifs. The rlc generated for a voiced part are grouped and this group is interpreted as a motif found in that voiced part. This results in a number of groups (motifs) for a rāga. Further, filtering is performed to isolate typical motifs of that rāga.

4.1 **Filtering to get typical motifs of a rāga**

The generated motifs are filtered to get typical motifs of a rāga using compositions of various rāgas. The most representative candidate of a motif, a candidate with highest score value, is selected to represent that motif or group. The instances of a motif are spotted in the compositions of various rāgas as explained in [9, 10]. Each motif is considered as a query to be searched for in a composition. The rlc is found between the query and overlapped windows in a composition. From the many generated rlc from many compositions of a rāga, top $\tau_n$ rlc with highest score values are selected. The average of these score values defines the presence of this motif in that rāga. A motif of a rāga is isolated as its typical motif if the presence of this motif is more in the given rāga than in other rāgas. The value of $\tau_n$ is selected empirically.

5. **DATASET**

The one-liners are selected from five rāgas as shown in Table 1. The lines are sung by a musician in isolation. This is done to ensure that the pitch estimation does not get affected due to the accompanying instruments. The average duration of the one-liners is 12.91 seconds. As mentioned earlier, these one-liners come from the various sections of the composition, primarily from the pallavi.

The compositions used for filtering also comes from the same five rāgas as shown in Table 2. These compositions are taken from the Charsur collection [1]. These are segments from live concerts with clean recording.
The pitch of the music segment is used as a basic feature in this work. This pitch is estimated from Justin Solomons algorithm [17] which is efficiently implemented in the essentia open-source C++ library [2]. This pitch is further normalized using tonic and then smoothed by interpolating the stationary points of the pitch contour using cubic spline interpolation.

The similarity, \( \text{sim}(x_{ik}, y_{jk}) \), between two symbols \( x_{ik} \) and \( y_{jk} \) is defined in the Equation (11), where \( s_t \) is the number of cent values that represent one semitone. For this work, the value of \( s_t \) is 10. The penalty is low when the two symbols are within one semitone while the penalty is significant for larger deviations. This is performed to ensure that although significant variations are possible in Carnatic music, variations larger than a semitone might result in a different rāga.

\[
\text{sim}(x_{ik}, y_{jk}) = \begin{cases} 
1 - \frac{|x_{ik} - y_{jk}|^3}{(3s_t)^3} & \text{if } |x_{ik} - y_{jk}| < 3s_t \\
0 & \text{otherwise}
\end{cases}
\]

(11)

The similarity threshold, \( \tau_{\text{sim}} \), is empirically set to 0.45 which accepts similarities when two symbols are less than 2.5 semitones (approx.) apart, although penalty is high after a semitone. The threshold on the score of rcls, \( \tau_{\text{scr}} \), is empirically set to 0.6 to accept rcls with higher score values. The threshold on the length of the rcls, \( \tau_{\text{len}} \), is set to 2 seconds to get longer motifs. The value of \( \beta \) is set to 0.5 to give equal importance to the individual densities in both the sequences and \( \alpha \) value is set to 0.6 which gives more importance to density of match and normalized weighted length as compared to linear trend in stationary points. \( \tau_\alpha \) is empirically set to 3.

The similar patterns found across one-liners of a rāga are summarized in Table 3. Some of these similar patterns are not typical of the rāga. These are therefore filtered out by checking for their presence in various compositions. The summary of the resulting typical motifs is given in Table 4. The average length of all the typical motifs is sufficiently longer than what were used in [10]. The shorter motifs used in [10] also resulted in great deal of false alarms. The importance of longer motifs was discussed in [9] where the longer motifs were inspired from the rāga test conducted by Rama Verma [3]. Rama Verma

<table>
<thead>
<tr>
<th>Rāga Name</th>
<th>Number of compositions</th>
<th>Average duration (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bhairavi</td>
<td>20</td>
<td>1133</td>
</tr>
<tr>
<td>Kamboji</td>
<td>10</td>
<td>1310.3</td>
</tr>
<tr>
<td>Kalyani</td>
<td>16</td>
<td>1204.3</td>
</tr>
<tr>
<td>Shankarabharanam</td>
<td>10</td>
<td>1300.6</td>
</tr>
<tr>
<td>Varali</td>
<td>18</td>
<td>1022</td>
</tr>
<tr>
<td>Overall</td>
<td>74</td>
<td>1194</td>
</tr>
</tbody>
</table>

Table 2. Database of compositions

6. EXPERIMENTS AND RESULTS

The pitch of the music segment is used as a basic feature in this work. This pitch is estimated from Justin Solomons algorithm [17] which is efficiently implemented in the essentia open-source C++ library [2]. This pitch is further normalized using tonic and then smoothed by interpolating the stationary points of the pitch contour using cubic spline interpolation.

The similarity, \( \text{sim}(x_{ik}, y_{jk}) \), between two symbols \( x_{ik} \) and \( y_{jk} \) is defined in the Equation (11), where \( s_t \) is the number of cent values that represent one semitone. For this work, the value of \( s_t \) is 10. The penalty is low when the two symbols are within one semitone while the penalty is significant for larger deviations. This is performed to ensure that although significant variations are possible in Carnatic music, variations larger than a semitone might result in a different rāga.

\[
\text{sim}(x_{ik}, y_{jk}) = \begin{cases} 
1 - \frac{|x_{ik} - y_{jk}|^3}{(3s_t)^3} & \text{if } |x_{ik} - y_{jk}| < 3s_t \\
0 & \text{otherwise}
\end{cases}
\]

(11)

The similarity threshold, \( \tau_{\text{sim}} \), is empirically set to 0.45 which accepts similarities when two symbols are less than 2.5 semitones (approx.) apart, although penalty is high after a semitone. The threshold on the score of rcls, \( \tau_{\text{scr}} \), is empirically set to 0.6 to accept rcls with higher score values. The threshold on the length of the rcls, \( \tau_{\text{len}} \), is set to 2 seconds to get longer motifs. The value of \( \beta \) is set to 0.5 to give equal importance to the individual densities in both the sequences and \( \alpha \) value is set to 0.6 which gives more importance to density of match and normalized weighted length as compared to linear trend in stationary points. \( \tau_\alpha \) is empirically set to 3.

The similar patterns found across one-liners of a rāga are summarized in Table 3. Some of these similar patterns are not typical of the rāga. These are therefore filtered out by checking for their presence in various compositions. The summary of the resulting typical motifs is given in Table 4. The average length of all the typical motifs is sufficiently longer than what were used in [10]. The shorter motifs used in [10] also resulted in great deal of false alarms. The importance of longer motifs was discussed in [9] where the longer motifs were inspired from the rāga test conducted by Rama Verma [3]. Rama Verma

<table>
<thead>
<tr>
<th>Rāga Name</th>
<th>Number of typical motifs</th>
<th>Average duration (secs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bhairavi</td>
<td>10</td>
<td>4.52</td>
</tr>
<tr>
<td>Kamboji</td>
<td>5</td>
<td>3.40</td>
</tr>
<tr>
<td>Kalyani</td>
<td>6</td>
<td>4.48</td>
</tr>
<tr>
<td>Shankarabharanam</td>
<td>5</td>
<td>3.64</td>
</tr>
<tr>
<td>Varali</td>
<td>2</td>
<td>4.79</td>
</tr>
<tr>
<td>Overall</td>
<td>12</td>
<td>4.32</td>
</tr>
</tbody>
</table>

Table 3. Summary of discovered similar patterns across one-liners

Table 4. Summary of typical motifs isolated after filtering used motifs of approximately 3 seconds duration. The typical motifs discovered in our work are also of similar duration. All the patterns of Kamboji and Kalyani are filtered out resulting in no typical motifs for these rāgas. We have earlier discussed that the compositions in Carnatic music are composed in a way that the rāga information is present at the very beginning. Therefore, without a doubt we are sure that the typical motifs are present in the one-liners we have used for Kalyani and Kamboji. But, it is possible that these typical motifs are not repeating sufficient number of times across one-liners (two times in our approach) or their lengths are shorter than the threshold we have used. These could be the reasons we are not able to pick them up. All the typical patterns are verified by a musician. According to his judgment, all the filtered patterns were indeed typical motifs of the corresponding rāgas. Although, he noted that one typical motif in Varali is a smaller portion of the other discovered typical motif of Varali. This repetition of smaller portion is observed in Shankarabharanam as well.

7. CONCLUSION AND FUTURE WORK

This paper presents an approach to discover typical motifs of a rāga from the one-liners of the compositions in that rāga. The importance of one-liners is discussed in detail. A new measure is introduced, to reduce the false alarms, in the optimization criteria for finding rough longest common subsequence between two given sequences. Using the RLCS algorithm, similar patterns across one-liners of a rāga are found. Further, the typical motifs are isolated by a filtering technique, introduced in this paper, which uses compositions of various rāgas. These typical motifs
are validated by a musician. All the generated typical motifs are found to be significantly typical of their respective rāgas.

In this work, only one musician’s viewpoint is considered on validating the characteristic nature of the discovered typical motifs. In future, we would like to conduct a MOS test, asking other experts and active listeners to determine the rāga from the typical motifs. We would also like to perform rāga classification of the compositions and alapanas using the typical motifs. In future, we would also like to do a thorough comparison of our approach with other methods. In this paper, we have only addressed one prevalent type of false alarms. Other types of false alarms also need to be identified and addressed. It should be considered that approaches taken to reduce the false alarms do not affect the true positives significantly. Further, these experiments need to be repeated for a much larger number of one-liners from many rāgas such that the typical motifs repeat significantly across one-liners and thus get captured. It will also be interesting to automatically detect and extract the one-liners from the available compositions. This will enable the presented approach to scale to a large number of rāgas.

8. ACKNOWLEDGMENTS

This research was partly funded by the European Research Council under the European Unions Seventh Framework Program, as part of the CompMusic project (ERC grant agreement 267583). We are grateful to Vignesh Ishwar for recording the one-liners. We would also like to thank Sridharan Sankaran, Nauman Dawalatabad and Manish Jain for their invaluable and unconditional help in completing this paper.

9. REFERENCES


Oral Session 5

Structure
ANALYZING SONG STRUCTURE WITH SPECTRAL CLUSTERING

Brian McFee  
Center for Jazz Studies  
Columbia University  
brm2132@columbia.edu

Daniel P.W. Ellis  
LabROSA  
Columbia University  
dpwe@ee.columbia.edu

ABSTRACT

Many approaches to analyzing the structure of a musical recording involve detecting sequential patterns within a self-similarity matrix derived from time-series features. Such patterns ideally capture repeated sequences, which then form the building blocks of large-scale structure.

In this work, techniques from spectral graph theory are applied to analyze repeated patterns in musical recordings. The proposed method produces a low-dimensional encoding of repetition structure, and exposes the hierarchical relationships among structural components at differing levels of granularity. Finally, we demonstrate how to apply the proposed method to the task of music segmentation.

1. INTRODUCTION

Detecting repeated forms in audio is fundamental to the analysis of structure in many forms of music. While small-scale repetitions — such as instances of an individual chord — are simple to detect, accurately combining multiple small-scale repetitions into larger structures is a challenging algorithmic task. Much of the current research on this topic begins by calculating local, frame-wise similarities over acoustic features (usually harmonic), and then searching for patterns in the all-pairs self-similarity matrix.

In the majority of existing work on structural segmentation, the analysis is flat, in the sense that the representation does not explicitly encode nesting or hierarchical structure in the repeated forms. Instead, novelty curves are commonly used to detect transitions between sections.

1.1 Our contributions

In this paper, we formulate the structure analysis problem in the context of spectral graph theory. By combining local consistency cues with long-term repetition encodings and analyzing the eigenvectors of the resulting graph Laplacian, we produce a compact representation that effectively encodes repetition structure at multiple levels of granularity. To effectively link repeating sequences, we formulate an optimally weighted combination of local timbre consistency and long-term repetition descriptors.

To motivate the analysis technique, we demonstrate its use for the standard task of flat structural annotation. However, we emphasize that the approach itself can be applied more generally to analyze structure at multiple resolutions.

1.2 Related work

The structural repetition features used in this work are inspired by those of Serra et al. [11], wherein structure is detected by applying filtering operators to a lag-skewed self-similarity matrix. The primary deviation in this work is the graphical interpretation and subsequent analysis of the filtered self-similarity matrix.

Recently, Kaiser et al. demonstrated a method to combine tonal and timbral features for structural boundary detection [6]. Whereas their method forms a novelty curve from the combination of multiple features, our feature combination differs by using local timbre consistency to build internal connections among sequences of long-range tonal repetitions.

Our general approach is similar in spirit to that of Grohgan et al. [4], in which diagonal bands of a self-similarity matrix are expanded into block structures by spectral analysis. Their method analyzed the spectral decomposition of the self-similarity matrix directly, whereas the method proposed here operates on the graph Laplacian. Similarly, Kaiser and Sikora applied non-negative matrix factorization directly to a self-similarity matrix in order to detect blocks of repeating elements [7]. As we will demonstrate, the Laplacian provides a more direct means to expose block structure at multiple levels of detail.

2. GRAPHICAL REPETITION ENCODING

Our general structural analysis strategy is to construct and partition a graph over time points (samples) in the song. Let \( X = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{d \times n} \) denote a \( d \)-dimensional time series feature matrix, e.g., a chromagram or sequence of Mel-frequency cepstral coefficients. As a first step toward detecting and representing repetition structure, we form a binary recurrence matrix \( R \in \{0, 1\}^{n \times n} \), where

\[
R_{ij}(X) := \begin{cases} 
1 & x_i, x_j \text{ are mutual } k\text{-nearest neighbors} \\
0 & \text{otherwise}, 
\end{cases}
\]

and \( k > 0 \) parameterizes the degree of connectivity.

Ideally, repeated structures should appear as diagonal stripes in \( R \). In practice, it is beneficial to apply a smooth-
ing filter to suppress erroneous links and fill in gaps. We apply a windowed majority vote to each diagonal of \( R \), resulting in the filtered matrix \( R' \):

\[
R'_{ij} := \max \{ R_{i+i,j+t} | t \in -w, -w+1, \ldots, w \},
\]

(2)

where \( w \) is a discrete parameter that defines the minimum length of a valid repetition sequence.

### 2.1 Internal connectivity

The filtered recurrence matrix \( R' \) can be interpreted as an unweighted, undirected graph, whose vertices correspond to samples (columns of \( X \)), and edges correspond to equivalent position within a repeated sequence. Note, however, that successive positions \( (i, i+1) \) will not generally be connected in \( R' \), so the constituent samples of a particular sequence may not be connected.

To facilitate discovery of repeated sections, edges between adjacent samples \( (i, i+1) \) and \( (i, i-1) \) are introduced, resulting in the sequence-augmented graph \( R^+ \):

\[
\Delta_{ij} := \begin{cases} 
1 & |i-j| = 1 \\
0 & \text{otherwise} 
\end{cases},
\]

(3)

\[
R^+_{ij} := \max(\Delta_{ij}, R'_{ij}).
\]

(4)

With appropriate normalization, \( R^+ \) characterizes a Markov process over samples, where at each step \( i \), the process either moves to an adjacent sample \( i+1 \), or a random repetition of \( i \); a process exemplified by the Infinite Jukebox [8].

Equation (4) combines local temporal connectivity with long-term recurrence information. Ideally, edges would exist only between pairs \( (i, j) \) belonging to the same structural component, but of course, this information is hidden. The added edges along the first diagonals create a fully connected graph, but due to recurrence links, repeated sections will exhibit additional internal connectivity. Let \( i \) and \( j \) denote two repetitions of the same sample at different times; then \( R^+ \) should contain sequential edges \( \{i,i+1\} \), \( \{j,j+1\} \) and repetition edges \( \{i,j\} \), \( \{i+1,j+1\} \). On the other hand, unrelated sections with no repetition edges can only connect via sequence edges.

### 2.2 Balancing local and global linkage

The construction of eq. (4) describes the intuition behind combining local sequential connections with global repetition structure, but it does not balance the two competing goals. Long tracks with many repetitions can produce recurrence links which vastly outnumber local connectivity connections. In this regime, partitioning into contiguous sections becomes difficult, and subsequent analysis of the graph may fail to detect sequential linkage.

If we allow (non-negative) weights on the edges, then the combination can be parameterized by a weighting parameter \( \mu \in [0,1] \):

\[
R''_{ij} := \mu R^+_{ij} + (1 - \mu) \Delta_{ij}.
\]

(5)

This raises the question: how should \( \mu \) be set? Returning to the motivating example of the random walk, we opt for a process that on average, tends to move either in sequence or across (all) repetitions with equal probability. In terms of \( \mu \), this indicates that the combination should assign equal weight to the local and repetition edges. This suggests a balancing objective for all frames \( i \):

\[
\mu \sum_j R''_{ij} \approx (1 - \mu) \sum_j \Delta_{ij}.
\]

Minimizing the average squared error between the two terms above leads to the following quadratic optimization:

\[
\min_{\mu \in [0,1]} \frac{1}{2} \sum_i (\mu d_i(R') - (1 - \mu) d_i(\Delta))^2,
\]

(6)

where \( d_i(G) := \sum_j G_{ij} \) denotes the degree (sum of incident edge-weights) of \( i \) in \( G \). Treating \( d(\cdot) := [d_i(\cdot)]_{i=1}^n \) as a vector in \( \mathbb{R}_+^n \) yields the optimal solution to eq. (6):

\[
\mu^* = \frac{\|d(\Delta), d(R') + d(\Delta)\|_2}{\|d(\Delta), d(R') + d(\Delta)\|_2^2}.
\]

(7)

Note that because \( \Delta \) is non-empty (contains at least one edge), it follows that \( \|d(\Delta)\|^2 > 0 \), which implies \( \mu^* < 1 \). Similarly, if \( R' \) is non-empty, then \( \mu^* < 1 \), and the resulting combination retains the full connectivity structure of the unweighted \( R^+ \) (eq. (4)).

### 2.3 Edge weighting and feature fusion

The construction above relies upon a single feature representation to determine the self-similarity structure, and uses constant edge weights for the repetition and local edges. This can be generalized to support feature-weighted edges by replacing \( R' \) with a masked similarity matrix:

\[
R''_{ij} \mapsto R''_{ij} S_{ij},
\]

(8)

where \( S_{ij} \) denotes a non-negative affinity between frames \( i \) and \( j \), e.g., a Gaussian kernel over feature vectors \( x_i, x_j \):

\[
S_{ij} \text{rep} := \exp \left(-\frac{1}{2 \sigma^2} \|x_i - x_j\|^2 \right)
\]

Similarly, \( \Delta \) can be replaced with a weighted sequence graph. However, in doing so, care must be taken when selecting the affinity function. The same features used to detect repetition (typically harmonic in nature) may not capture local consistency, since successive frames do not generally retain harmonic similarity.

Recent work has demonstrated that local timbre differences can provide an effective cue for structural boundary detection [6]. This motivates the use of two contrasting feature descriptors: harmonic features for detecting long-range repeating forms, and timbral features for detecting local consistency. We assume that these features are respectively supplied in the form of affinity matrices \( S_{ij} \text{rep} \) and \( S_{ij} \text{loc} \). Combining these affinities with the detected repetition structure and optimal weighting yields the sequence-augmented affinity matrix \( A \):

\[
A_{ij} := \mu R''_{ij} S_{ij} \text{rep} + (1 - \mu) \Delta_{ij} S_{ij} \text{loc},
\]

(9)

where \( R'' \) is understood to be constructed solely from the repetition affinities \( S_{ij} \text{rep} \), and \( \mu \) is optimized by solving (7) with the weighted affinity matrices.
3. GRAPH PARTITIONING AND STRUCTURAL ANALYSIS

The Laplacian is a fundamental tool in the field of spectral graph theory, as it can be interpreted as a discrete analog of a diffusion operator over the vertices of the graph, and its spectrum can be used to characterize vertex connectivity [2]. This section describes in detail how spectral clustering can be used to analyze and partition the repetition graph constructed in the previous section, and reveal musical structure.

3.1 Background: spectral clustering

Let $D$ denote the diagonal degree matrix of $A$:

$$D := \text{diag}(d(A)).$$

The symmetric normalized Laplacian $L$ is defined as:

$$L := I - D^{-1/2} A D^{-1/2}. \quad (10)$$

The Laplacian forms the basis of spectral clustering, in which vertices are represented in terms of the eigenvectors of $L$ [15]. More specifically, to partition a graph into $m$ components, each point $i$ is represented as the vector of the $i$th coordinates of the first $m$ eigenvectors of $L$, corresponding to the $m$ smallest eigenvalues.\footnote{An additional length-normalization is applied to each vector, to correct for scaling introduced by the symmetric normalized Laplacian [15].} The motivation for this method stems from the observation that the multiplicity of the bottom eigenvalue $\lambda_0 = 0$ corresponds to the number of connected components in a graph, and the corresponding eigenvectors encode component memberships amongst vertices.

In the non-ideal case, the graph is fully connected, so $\lambda_0$ has multiplicity 1, and the bottom eigenvector trivially encodes membership in the graph. However, in the case of $A$, we expect there to be many components with high intra-connectivity and relatively small inter-connectivity at the transition points between sections. Spectral clustering can be viewed as an approximation method for finding normalized graph cuts [15], and it is well-suited to detecting and pruning these weak links.

Figure 1 illustrates an example of the encoding produced by spectral decomposition of $L$. Although the first eigenvector (column) is uninformative, the remaining bases clearly encode membership in the diagonal regions depicted in the affinity matrix. The resulting pair-wise frame similarities for this example are shown in Figure 2, which clearly demonstrates the ability of this representation to iteratively reveal nested repeating structure.

To apply spectral clustering, we will use $k$-means clustering with the (normalized) eigenvectors $Y \in \mathbb{R}^{n \times M}$ as features, where $M > 0$ is a specified maximum number of structural component types. Varying $M$ — equivalently, the dimension of the representation — directly controls the granularity of the resulting segmentation.

### Algorithm 1 Boundary detection

**Input:** Laplacian eigenvectors $Y \in \mathbb{R}^{n \times m}$.

**Output:** Boundaries $b$, segment labels $c \in [m]^n$.

1. \textbf{function} \texttt{BOUNDARY-DETECT}($Y$)
2. \quad $\bar{y}_i \leftarrow Y_{i,:}/\|Y_{i,:}\|$ \quad // Normalize each row $Y_{i,:}$.
3. \quad Run $k$-means on $\{\bar{y}_i\}_{i=1}^n$ with $k = m$
4. \quad Let $c_i$ denote the cluster containing $\bar{y}_i$
5. \quad \textbf{return} $(b, c)$

3.2 Boundary detection

For a fixed number of segment types $m$, segment boundaries can estimated by clustering the rows of $Y$ after truncating to the first $m$ dimensions. After clustering, segment boundaries are detected by searching for change-points in the cluster assignments. This method is formalized in Algorithm 1. Note that the number of segment types is distinct from the number of segments because a single type (e.g., verse) may repeat multiple times throughout the track.

3.3 Laplacian structural decomposition

To decompose an input song into its structural components, we propose a method, listed as Algorithm 2, to find boundaries and structural annotations at multiple levels of structural complexity. Algorithm 2 first computes the Laplacian as described above, and then iteratively increases the set of eigenvectors for use in Algorithm 1. For $m = 2$, the first two eigenvectors — corresponding to the two smallest eigenvalues of $L$ — are taken. In general, for $m$ types of repeating component, the bottom $m$ eigenvectors are used to label frames and detect boundaries. The result is a sequence of boundaries $B^m$ and frame labels $C^m$, for values $m \in 2, 3, \ldots, M$.

Note that unlike most structural analysis algorithms, Algorithm 2 does not produce a single decomposition of the song, but rather a sequence of decompositions ordered by increasing complexity. This property can be beneficial in visualization applications, where a user may be interested in the relationship between structural components at multiple levels. Similarly, in interactive display applications, a user may request more or less detailed analyses for a track. Since complexity is controlled by a single, discrete parameter $M$, this application is readily supported with a minimal set of interface controls (e.g., a slider).

However, for standardized evaluation, the method must produce a single, flat segmentation. Adaptively estimating the appropriate level of analysis in this context is somewhat ill-posed, as different use-cases require differing levels of detail. We apply a simple selection criterion based on the level of detail commonly observed in standard datasets [5, 12]. First, the set of candidates is reduced to those in which the mean segment duration is at least 10 seconds. Subject to this constraint, the segmentation level $\tilde{m}$ is selected to maximize frame-level annotation entropy. This strategy tends to produce solutions with approximately balanced distributions over the set of segment types.
4. EXPERIMENTS

To evaluate the proposed method quantitatively, we compare boundary detection and structural annotation performance on two standard datasets. We evaluate the performance of the method using the automatic complexity estimation described above, as well as performance achieved for each fixed value of $m$ across the dataset.

Finally, to evaluate the impact of the complexity estimation method, we compare to an oracle model. For each track, a different $m^*$ is selected to maximize the evaluation metric of interest. This can be viewed as a simulation of interactive visualization, in which the user has the freedom to dynamically adapt the level of detail until she is satisfied. Results in this setting may be interpreted as measuring the best possible decomposition within the set produced by Algorithm 2.

4.1 Data and evaluation

Our evaluation data is comprised of two sources:

**Beatles-TUT:** 174 structurally annotated tracks from the Beatles corpus [10]. A single annotation is provided for each track, and annotations generally correspond to functional components (e.g., verse, refrain, or solo).

**SALAMI:** 735 tracks from the SALAMI corpus [12]. This corpus spans a wide range of genres and instrumentation, and provides multiple annotation levels for each track. We report results on functional and small-scale annotations.

In each evaluation, we report the $F$-measure of boundary detection at 0.5-second and 3-second windows. To evaluate structural annotation accuracy, we report pairwise frame classification $F$-measure [9]. For comparison purposes, we report scores achieved by the method of Serrà et
Algorithm 2 Laplacian structural decomposition

**Input:** Affinities: \( S^{\text{rep}}, S^{\text{loc}} \in \mathbb{R}^{n \times n} \), maximum number of segment types \( 0 < M \leq n \)

**Output:** Boundaries \( B^m \) and frame labels \( C^m \) for \( m = 2 \ldots M 

1: function LSD(\( S^{\text{rep}}, S^{\text{loc}}, M \))
2: \( R \leftarrow \text{eq. (1) on } S^{\text{rep}} \) \( / \) Recurrence detection
3: \( R' \leftarrow \text{eq. (2) on } R \) \( / \) Recurrence filtering
4: \( A \leftarrow \text{eq. (9)} \) \( / \) Sequence augmentation
5: \( L \leftarrow I - D^{-1/2} A D^{-1/2} \)
6: for \( m = 2, 3, \ldots, M \) do
7: \( Y \leftarrow \text{bottom } m \text{ eigenvectors of } L \)
8: \( (B^m, C^m) \leftarrow \text{BOUNDARY-DETECT}(Y) \)
9: return \((B^m, C^m)\) for \( m = 2 \ldots M \)

\( \text{al.} \), denoted here as SMGA [11].

4.2 Implementation details

All input signals are sampled at 22050Hz (mono), and analyzed with a 2048-sample FFT window and 512-sample hop. Repetition similarity matrices \( S^{\text{rep}} \) were computed by first extracting log-power constant-Q spectrograms over 72 bins, ranging from \( C2 (32.7 \text{ Hz}) \) to \( C8 (2093.0 \text{ Hz}) \).

Constant-Q frames were mean-aggregated between detected beat events, and stacked using time-delay embedding with one step of history as in [11]. Similarity matrices were then computed by applying a Gaussian kernel to each pair of beat-synchronous frames \( i \) and \( j \). The bandwidth parameter \( \sigma^2 \) was estimated by computing the average squared distance between each \( x_i \) and its \( k \)th nearest neighbor, with \( k = 1 + \left[ 2 \log_2 n \right] \) (where \( n \) denotes the number of detected beats). The same \( k \) was used to connect nearest neighbors when building the recurrence matrix \( R \), with the additional constraint that frames cannot link to neighbors within 3 beats of each other, which prevents self-similar connections within the same measure. The majority vote window was set to \( w = 17 \).

Local timbre similarity \( S^{\text{loc}} \) was computed by extracting the first 13 Mel frequency cepstral coefficients (MFCC), mean-aggregating between detected beats, and then applying a Gaussian kernel as done for \( S^{\text{rep}} \).

All methods were implemented in Python with NumPy and librosa [1, 14].

4.3 Results

The results of the evaluation are listed in Tables 1 to 3. For each fixed \( m \), the scores are indicated as \( L_m \). \( L \) indicates the automatic maximum-entropy selector, and \( L^* \) indicates the best possible \( m \) for each metric independently.

As a common trend across all data sets, the automatic \( m \)-selector often achieves results comparable to the best fixed \( m \). However, it is consistently outperformed by the oracle model \( L^* \), indicating that the output of Algorithm 2 often contains accurate solutions, the automatic selector does not always choose them.

In the case of SALAMI (small), the automatic selector performs dramatically worse than many of the fixed-\( m \) methods, which may be explained by the relatively different statistics of segment durations and numbers of unique segment types in the small-scale annotations as compared to Beatles and SALAMI (functional).

To investigate whether a single \( m \) could simultaneously optimize multiple evaluation metrics for a given track, we plot the confusion matrices for the oracle selections on SALAMI (functional) in Figure 3. We observe that the \( m \) which optimizes \( F_2 \) is frequently larger than those for \( F_{0.5} \) — as indicated by the mass in the lower triangle of the left plot — or \( F_{\text{par}} \) — as indicated by the upper triangle of the right plot. Although this observation depends upon our particular boundary-detection strategy, it is corroborated by previous observations that the 0.5-second and 3.0-second metrics evaluate qualitatively different objectives [13]. Consequently, it may be beneficial in practice to provide segmentations at multiple resolutions when the specific choice of evaluation criterion is unknown.

5. CONCLUSIONS

The experimental results demonstrate that the proposed structural decomposition technique often generates solutions which achieve high scores on segmentation evaluation metrics. However, automatically selecting a single “best” segmentation without a priori knowledge of the evaluation criteria...
Figure 3. Confusion matrices illustrating the oracle selection of the number of segment types \( m \in [2, 10] \) for different pairs of metrics on SALAMI (functional). While \( m = 2 \) is most frequently selected for all metrics, the large mass off-diagonal indicates that for a given track, a single fixed \( m \) does not generally optimize all evaluation metrics.

<table>
<thead>
<tr>
<th>Method</th>
<th>( F_{0.5} )</th>
<th>( F_3 )</th>
<th>( F_{\text{pair}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( L_2 )</td>
<td>0.151 ± 0.11</td>
<td>0.195 ± 0.13</td>
<td>0.451 ± 0.19</td>
</tr>
<tr>
<td>( L_3 )</td>
<td>0.171 ± 0.12</td>
<td>0.259 ± 0.16</td>
<td>0.459 ± 0.17</td>
</tr>
<tr>
<td>( L_4 )</td>
<td>0.186 ± 0.12</td>
<td>0.315 ± 0.17</td>
<td>0.461 ± 0.15</td>
</tr>
<tr>
<td>( L_5 )</td>
<td>0.195 ± 0.12</td>
<td>0.354 ± 0.17</td>
<td>0.455 ± 0.14</td>
</tr>
<tr>
<td>( L_6 )</td>
<td>0.207 ± 0.12</td>
<td>0.391 ± 0.18</td>
<td>0.452 ± 0.13</td>
</tr>
<tr>
<td>( L_7 )</td>
<td>0.214 ± 0.12</td>
<td>0.420 ± 0.18</td>
<td>0.445 ± 0.13</td>
</tr>
<tr>
<td>( L_8 )</td>
<td>0.224 ± 0.12</td>
<td>0.448 ± 0.18</td>
<td>0.435 ± 0.13</td>
</tr>
<tr>
<td>( L_9 )</td>
<td>0.229 ± 0.12</td>
<td>0.467 ± 0.18</td>
<td>0.425 ± 0.13</td>
</tr>
<tr>
<td>( L_{10} )</td>
<td>0.234 ± 0.12</td>
<td>0.486 ± 0.18</td>
<td>0.414 ± 0.13</td>
</tr>
<tr>
<td>( L )</td>
<td>0.192 ± 0.11</td>
<td>0.344 ± 0.15</td>
<td>0.448 ± 0.16</td>
</tr>
<tr>
<td>( L^* )</td>
<td>0.292 ± 0.15</td>
<td>0.525 ± 0.19</td>
<td>0.561 ± 0.16</td>
</tr>
</tbody>
</table>

SMGA 0.173 ± 0.08 0.518 ± 0.12 0.493 ± 0.16

remains a challenging practical issue.

6. ACKNOWLEDGMENTS

The authors acknowledge support from The Andrew W. Mellon Foundation, and NSF grant IIS-1117015.

7. REFERENCES


IDENTIFYING POLYPHONIC PATTERNS FROM AUDIO RECORDINGS USING MUSIC SEGMENTATION TECHNIQUES

Oriol Nieto and Morwaread M. Farbood
Music and Audio Research Lab
New York University
{oriol, mfarbood}@nyu.edu

ABSTRACT

This paper presents a method for discovering patterns of note collections that repeatedly occur in a piece of music. We assume occurrences of these patterns must appear at least twice across a musical work and that they may contain slight differences in harmony, timbre, or rhythm. We describe an algorithm that makes use of techniques from the music information retrieval task of music segmentation, which exploits repetitive features in order to automatically identify polyphonic musical patterns from audio recordings. The novel algorithm is assessed using the recently published JKU Patterns Development Dataset, and we show how it obtains state-of-the-art results employing the standard evaluation metrics.

1. INTRODUCTION

The task of discovering repetitive musical patterns (of which motives, themes, and repeated sections are all examples) consists of retrieving the most relevant musical ideas that repeat at least once within a specific piece [1, 8]. Besides the relevant role this task plays in musicological studies, especially with regard to intra-opus analysis, it can also yield a better understanding of how composers write and how listeners interpret the underlying structure of music. Computational approaches to this task can dramatically simplify not only the analysis of a specific piece, but of an entire corpus, potentially offering interesting explorations and relations of patterns across works. Other potential applications include the improved navigation across both large music collections and stand-alone pieces, or the development of computer-aided composition tools.

Typically the task of automatically discovering musical patterns uses symbolic representations of music [3]. Methods that assume a monophonic representation have been proposed, and operate on various musical dimensions such as chromatic/diatonic pitch, rhythm, or contour [4, 9, 10]. Other methods focusing on polyphonic music as input have also been presented, mostly using geometric representations in Euclidean space, with a different axis assigned to each musical dimension [6, 11]. Similar techniques have also been explored [7, 11, 12] that attempt to arrive at a compressed representation of an input, multidimensional point set. Other methods using cognitively inspired rules with symbolic representations of music have also been proposed [6, 16]. Working with the score of a musical piece instead of its audio representation can indeed reduce the complexity of the problem, however this also significantly narrows the applicability of the algorithm, since it is not necessarily common to have access to symbolic representations of music, particularly when working with genres such as jazz, rock, or Western popular music.

Methods using audio recordings as input have also been explored. A good recent example is [3], where the authors first estimate the fundamental frequency (F0) from the audio in order to obtain the patterns using a symbolic-based approach. Another one uses a probabilistic approach to matrix factorization in order to learn the different parts of a western popular track in an unsupervised manner [20]. Yet another method uses a compression criterion where the most informative (i.e., repeated) parts of a piece are identified in order to automatically produce a musical “summary” [17].

In this paper, we propose a method using audio recordings as input in an attempt to broaden the applicability of pattern discovery algorithms. We make use of tools that are commonly employed in the music information retrieval task of music segmentation combined with a novel score-based greedy algorithm in order to identify the most repeated parts of a given audio signal. Finally, we evaluate the results using the JKU Patterns Development Dataset and the metrics proposed in the Music Information Retrieval Evaluation eXchange (MIREX) [1].

The outline of this paper is as follows: In Section 2 we review a set of music segmentation techniques that will be used in our algorithm; in Section 3 we detail our method to extract musical patterns, including the score-based greedy algorithm; in Section 4 we present the evaluation and the results; and in Section 5 we draw various conclusions and identify areas for future work.
2. MUSIC SEGMENTATION TECHNIQUES

The task of music segmentation is well-established in the music informatics literature (see [18] for a review). Its goal is to automatically identify all the non-overlapping musical segments (or sections) of a given track, such that the concatenation of all of them reconstructs the entire piece. Once these segments are identified, they are labeled based on their similarity (e.g., verse, chorus, coda). Therefore, this task can be divided into two different subproblems: the discovery of the boundaries that define all the segments, and the grouping of the segments into different labels. In this work we will use tools that focus mainly on the former subproblem.

There is general agreement among researchers that any given boundary is defined by at least one of these three characteristics: repetition, homogeneity, and/or novelty [18]. In our case, we center the discussion on the repetition boundaries, since, as we will see in Section 3, repetition is the defining feature of the musical patterns.

2.1 Extracting Repetitive Boundaries

In this subsection we review a standard technique to extract boundaries characterized by repetition (also known as a sequence approach), from an input audio signal \( x \). For a more detailed explanation, we refer the reader to [13]. The process can be summarized in three different steps:

i The signal \( x \) is transformed into a series of feature vectors \( C = (c_1, ..., c_N) \) that divide \( x \) into \( N \) frames and capture specific frame-level characteristics of the given signal. In our case, we will only focus on harmonic features, more specifically on chromagrams (or pitch class profiles).

ii \( C \) is used in order to obtain a self-similarity matrix (SSM) \( S \), a symmetric matrix such that \( S(n, m) = d(c_n, c_m), \forall n, m \in [1 : N] \), where \( d \) is a distance function (e.g., Euclidean, cosine, Manhattan).

iii The resulting matrix \( S \) will contain diagonal paths (or semi-diagonal in case of slight tempo variations) or stripes that will indicate the repetition of a specific part of the audio signal \( x \). These paths can be extracted using greedy algorithms (e.g., as described in [13]). The final boundaries are given by the endpoints of these paths.

An example of an SSM using the Euclidean distance with the identified boundaries is shown in Figure 1. As can be seen, the annotated boundaries are visually associated with the paths of the matrix. The identification of patterns, as opposed to the task of segmentation, allows overlapping patterns and occurrences, so we base our algorithm on greedy methods to extract paths from an SSM.

2.2 Transposition-Invariant Self-Similarity Matrix

It is common to analyze pieces that contain key-transposed repetitions. It is therefore important for an algorithm to be invariant to these transpositions when identifying repetitions. One effective method for solving this problem [14] involves a technique that can be described in two steps: (1) compute 12 different SSMs from harmonic representations (e.g., chromagrams), each corresponding to a transposition of the 12 pitches of the Western chromatic scale, and 2) obtain the transposition-invariant SSM by keeping the minimum distance across the 12 matrices for all the \( N \times N \) distances in the output matrix. Formally:

\[
S(n, m) = \min_{k \in [0:11]} \{S_k(n, m)\}, \forall n, m \in [1 : N] \quad (1)
\]

where \( S \) is the transposition-invariant SSM, and \( S_k \) is the \( k \)-th transposition of the matrix \( S \).

3. IDENTIFYING MUSICAL PATTERNS

The discovery of patterns and their various occurrences involves retrieving actual note collections (which may nest and/or overlap), and so this task can be seen as more complex than structural segmentation, which involves labeling a single, temporal partition of an audio signal. We define a repeating musical pattern to be a short idea that is repeated at least once across the entire piece, even though this repetition may be transposed or contain various time shifts. Therefore, each pattern is associated with a set of occurrences that will not necessarily be exact. The patterns and their occurrences may overlap with each other, and this is perfectly acceptable in the context of pattern discovery. An optimal algorithm for this task would (1) find all the patterns contained in a piece and (2) identify all the occurrences across the piece for each pattern found. In this section we describe our algorithm, which uses audio recordings as input and finds polyphonic patterns as well as a list of all the discovered occurrences for each of the patterns. A block-diagram of the entire process is depicted in Figure 2.
3.1 Rhythmic-Synchronous Harmonic Feature Extraction

Given a one-channel audio signal $x$ sampled at 11025 Hz representing a piece of music, we compute the spectrogram using a Blackman window of $N_w = 290$ milliseconds, with a hop size of $N_w/2$. We then compute a constant-Q transform from the spectrogram starting at 55 Hz (corresponding to the note A1 in standard tuning) comprising four octaves. Finally, we collapse each of the 12 pitches of the western scale into a single octave to obtain a chromagram, a matrix of $12 \times N$, which is commonly used to represent harmonic features [18]. We normalize the chromagram such that the maximum energy for a given time frame is 1. In this harmonic representation we can no longer differentiate between octaves, but its compactness and the energy of each pitch class will become convenient when identifying harmonic repetitions within a piece.

We then use a beat tracker [5] in order to average the time frames into rhythmic frames. Instead of using the traditional beat-synchronous approach, which is typically employed in a segmentation task, we divide each beat duration by 4 and aggregate accordingly, thus having $N = 4B$ time frames, where $B$ is the number of beats detected in the piece. The motivation behind this is that patterns may not start at the beat level, as opposed to the case for long sections. Furthermore, adding a finer level of granularity (i.e., analyzing the piece at a sixteenth-note level instead of every fourth note or at the beat level) should yield better results.

3.2 Finding Repeated Segments

We make use of the transposition-invariant SSM $S$ described in Section 2.2, computed from the chromagram of a given audio signal using the Euclidean distance, in order to identify repeated segments. As opposed to the task of segmentation, the goal here is to find all possible repeated segments in $S$, independent of how short they are or the amount of overlap present. The other major difference is that we do not aim to find all of the segments of the piece, but rather identify all of the repeated ones. Repeated segments appear in $S$ as diagonal “stripes”, also known as paths. If the beat-tracker results in no errors (or if the piece contains no tempo variations), these stripes will be perfectly diagonal.

3.2.1 Quantifying Paths with a Score

We propose a score-based greedy algorithm to efficiently identify the most prominent paths in $S$. Starting from $S \in \mathbb{R}^{N \times N}$, we set half of its diagonals to zero, including the main one, due to its symmetrical properties, resulting in $\hat{S}$, s.t. $\hat{S}(n,m) = 0$ if $n \leq m$ and $\hat{S}(n,m) = S(n,m)$ if $n > m$, $\forall n,m \in [1:N]$ . We then compute a score function $\sigma$ for each possible path in all the non-zero diagonals of $\hat{S}$, resulting in a search space of $N(N-1)/2$ possible positions in which paths can start.

Before introducing the score function $\sigma$, we define a trace function given a square matrix $X \in \mathbb{R}^{N \times N}$ with an offset parameter $\omega$:

$$\text{tr}(X,\omega) = \sum_{i=1}^{N-\omega} X(i,i+\omega), \omega \in \mathbb{Z} \quad (2)$$

As can be seen from this equation, when $\omega = 0$ we have the standard trace function definition.

The score function $\sigma$ uses various traces of the matrix that comprises a possible path in order to quantify the degree of repetition of the path. If a possible path starts at indices $n,m$ and has a duration of $M$ time frames, then the matrix that the path defines is $P \in \mathbb{R}^{M \times M}$, s.t. $P(i,j) = \hat{S}(n+i-1,m+j-1), \forall i,j \in [1:M]$. We now can define the score $\sigma$ as the sum of the closest traces to the diagonal of $P$ (i.e., those with a small $\omega$) and subtract the traces that are farther apart from the diagonal (i.e., where $\omega$ is greater). We then normalize in order to obtain a score independent from the duration $M$ of the possible path:

$$\sigma(\rho) = \frac{\left(\sum_{\omega=-\rho}^{\rho} \text{tr}(P,\omega) - \text{tr}(P,\pm\rho)\right)}{M + \sum_{i=1}^{\rho-1} 2(M-i)} \quad (3)$$

where $\rho \in \mathbb{N}$ is the maximum offset to be taken into account when computing the traces of $P$. The greater the $\rho$, the greater the $\sigma$ for segments that contain substantial energy around their main diagonal (e.g., paths that contain significant rhythmic variations), even though the precision decreases as $\rho$ increases.

Examples for various $\sigma(\rho)$ can be seen in Figure 3. For a perfectly clean path (left), we see that $\rho = 1$ gives the maximum score of 1. However, the score decreases as $\rho$ increases, since there is zero energy in the diagonals right next to the main diagonal. On the other hand, for matrices extracted from audio signals (middle and right), we can see that the scores $\sigma(1)$ are low, indicating that the diagonals next to the main diagonal contain amounts of energy similar to the main diagonal. However, when $\rho > 1$, the score is substantially different from a matrix with a path (middle) and a matrix without one (right).
Figure 3. Three examples showing the behavior of the path score $\sigma(\rho)$. The one on the left shows a synthetic example of a perfect path. The one in the middle contains a real example of a path in which there is some noise around the diagonal of the matrix. In the example on the right, a matrix with no path is shown.

3.2.2 Applying the Score

For all $N(N - 1)/2$ positions in which paths can potentially start in $\hat{\mathcal{S}}$, we want to extract the most prominent ones (i.e., the ones that have a high $\sigma$). At the same time, we want to extract the paths from beginning to end in the most accurate way possible. The algorithm that we propose assigns a certain $\sigma$ to an initial possible path $\hat{x}$ of a minimum length of $\nu$ time frames, which reduces the search space to $(N - \nu + 1)(N - \nu)/2$. If the score $\sigma$ is greater than a certain threshold $\theta$, we increase the possible path by one time frame, and recompute $\sigma$ until $\sigma \leq \theta$. By then, we can store the path $\hat{x}$ as a segment in the set of segments $\mathcal{Z}$. In order to avoid incorrectly identifying possible paths that are too close to the found path, we zero out the found path from $\hat{\mathcal{S}}$, including all the closest $\rho$ diagonals, and keep looking, starting from the end of the recently found path.

The pseudocode for this process can be seen in Algorithm 1, where $|x|$ returns the length of the path $x$, $\{x\}$ returns the path in which all elements equal $x$, the function ComputeScore computes the $\sigma(\rho)$ as described in Section 3.2.1, OutOfBounds($x, X$) checks whether the path $x$ is out of bounds with respect to $X$, IncreasePath($x$) increases the path $x$ by one (analogously as DecreasePath), and ZeroOutPath($X, x, \rho$) assigns zeros to the path $x$ found in $X$, including all the closest $\rho$ diagonals.

Algorithm 1 Find Repeated Segments

Require: $\hat{\mathcal{S}}, \rho, \theta, \nu$
Ensure: $\mathcal{Z} = \{z_1, \ldots, z_k\}$

for $\hat{z} \in \hat{\mathcal{S}} \land |\hat{z}| = \nu \land \hat{z} \neq \emptyset$ do
  $b \leftarrow$ False
  $\sigma \leftarrow$ ComputeScore($\hat{z}, \rho$)
  while $\sigma > \theta \land \neg$OutOfBounds($\hat{z}, \hat{\mathcal{S}}$) do
    $b \leftarrow$ True
    $\hat{z} \leftarrow$ IncreasePath($\hat{z}$)
    $\sigma \leftarrow$ ComputeScore($\hat{z}, \rho$)
  end while
  if $b$ then
    $\mathcal{Z}$:add(DecreasePath($\hat{z}$))
    ZeroOutPath($\hat{\mathcal{S}}, \hat{z}, \rho$)
  end if
end for
return $\mathcal{Z}$

An example of the paths found by the algorithm is shown in Figure 4. Parts of some segments are repeated as standalone segments (i.e., segments within segments), therefore allowing overlap across patterns as expected in this task. Observe how some of the segments repeat almost exactly across the piece—there is a set of patterns at the top of the matrix that appears to repeat at least three times. The next step of the algorithm is to cluster these segments together so that they represent a single pattern with various occurrences.

Figure 4. Paths found (marked in white) using the proposed algorithm for Chopin’s Op. 24 No. 4., with $\theta = 0.33, \rho = 2$.

3.3 Clustering the Segments

Each segment $z \in \mathcal{Z}$, defined by the two indices in which it starts $(s_i, s_j)$ and ends $(e_i, e_j)$ in $\hat{\mathcal{S}}$, contains two occurrences of a pattern: the one that starts in $s_i$ and ends in $e_i$ and the one that occurs between the time indices $s_j$ and $e_j$. In order to cluster the repeated occurrences of a single pattern, we find an occurrence for each segment $z \in \mathcal{Z}$ if one of the other segments in $\mathcal{Z}$ starts and ends in similar locations with respect to the second dimension of $\hat{\mathcal{S}}$. Note that we set to zero the bottom left triangle of the matrix as explained in Section 3.2.1, so we cannot use the first dimension to cluster the occurrences. Formally, a segment $\hat{z}$ is an occurrence of $z$ if

$$s^z_j - \Theta \leq s^z_j \leq s^z_j + \Theta \land (e^z_j - \Theta \leq e^z_j \leq e^z_j + \Theta)$$

where $s^z_j$ represents the starting point of the segment $z$ in the second dimension of $\hat{\mathcal{S}}$ and analogously $e^z_j$ is the ending point, and $\Theta$ is a tolerance parameter.

3.4 Final Output

At this point, we have a set of patterns with their respective occurrences represented by their starting and ending timeframe indices. Even though the algorithm is not able to distinguish the different musical lines within the patterns, we can use the annotated score to output the exact notes that occur during the identified time indices, as suggested in the MIREX task [1]. If no score is provided, only the time
points will be presented. In order to overcome this limitation in future work, the audio should be source-separated to identify the different lines and perform an F0 estimation to correctly identify the exact melody that defines the pattern (and not just the time points at which it occurs). Progress toward this goal has been presented in [2].

3.5 Time Complexity Analysis

Once the rhythm-synchronous chromagram is computed, the process of calculating the transposition-invariant SSM is $O(13N^2) = O(N^2)$, where $N$ is the number of time frames of the chromagram. The procedure to compute the score given a path has a time complexity of $O(2\rho M) = O(\rho M)$, where $\rho$ is the required parameter for the computation of the score, and $M$ is the length of the path from which to compute the score. The total process of identifying segments is $O\left(\frac{(N-\nu+1)(N-\nu)}{2} \rho M\right) = O((N - \nu)^2 \rho M)$, where $\nu$ is the minimum number of time frames that a pattern can have. Asymptotically, we can neglect the clustering of the segments, since the length of $Z$ will be much less than $N$. Therefore, the total time complexity of the proposed algorithm is $O(N^2 + (N - \nu)^2 \rho M)$.

4. EVALUATION

We use the JKU Patterns Development Dataset¹ to evaluate our algorithm. This dataset is comprised of five classical pieces annotated by various musicologists and researchers [1]. This dataset is the public subset of the one employed to evaluate the Pattern Discovery task at MIREX, using the metrics described below.

4.1 Metrics

Two main aspects of this task are evaluated: the patterns discovered and the occurrences of the identified patterns across the piece. Collins and Meredith proposed metrics to quantify these two aspects, which are detailed in [1]; all of these metrics use the standard $F_1$ accuracy score, defined as $F_1 = 2PR/(P + R)$, where $P$ is precision (such that $P = 1$ if all the estimated elements are correct), and $R = 1$ is recall (such that $R = 1$ if all the annotated elements are estimated).

Establishment $F_1$ Score ($F_{est}$): Determines how the annotated patterns are established by the estimated output. This measure returns a score of 1 if at least one occurrence of each pattern is discovered by the algorithm to be evaluated.

Occurrence $F_1$ Score ($F_o$): For all the patterns found, we want to estimate the ability of the algorithm to capture all of the occurrences of these patterns within the piece independently of how many different patterns the algorithm has identified. Therefore, this score would be 1 if the algorithm has only found one pattern with all the correct occurrences. A parameter $c$ controls when a pattern is considered to have been discovered, and therefore whether it counts toward the occurrence scores. The higher the $c$, the smaller the tolerance. In this evaluation, as in MIREX, we use $c = .75$ and $c = .5$.

Three-Layer $F_1$ Score ($F_3$): This measure combines both the patterns established and the quality of their occurrences into a single score. It is computed using a three-step process that yields a score of 1 if a correct pattern has been found and all its occurrences have been correctly identified.

4.2 Results

The results of the proposed algorithm, computed using the open source evaluation package mir_eval [19], are shown in Table 1, averaged for the entire JKU Dataset, along with an earlier version of our algorithm submitted to MIREX [15], another recent algorithm called SIARCT-CFP [2] that is assessed using both audio and symbolic representations as input in [3], and “COSIATEC Segment”, a method that only uses symbolic inputs [12]. We use this latter method for comparison because it is the only symbolic method in which we have access to all of the resulting metrics, and SIARCT-CFP since it is the most recent method that uses audio as input. The parameter values used to compute these results, $\nu = 8$, $\theta = 0.33$, $\rho = 2$, and $\Theta = 4$, were found empirically. We can see how our algorithm is better than [15] in all the metrics except running time; it also finds more correct patterns than [3] (the current state-of-the-art when using audio as input).

Our algorithm obtains state-of-the-art results when extracting patterns from audio, obtaining an $F_{est}$ of 49.80%. This is better than the symbolic version of [2] and almost as good as the algorithm described in [12]. The fact that our results are superior or comparable to the two other algorithms using symbolic representations indicates the potential of our method.

When evaluating the occurrences of the patterns, we see that our algorithm is still better than [15], but worse than [2] (at least for $c = .5$, which is the only reported result). Nevertheless, the numbers are much lower than [12]. In this case, working with symbolic representations (or estimating the F0 in order to apply a symbolic algorithm as in [2]) yields significantly better results. It is interesting to note that when the tolerance increases (i.e. $c = .5$), our results improve as opposed to the other algorithms. This might be due to the fact that some of the occurrences found in the SSM were actually very similar (therefore they were found in the matrix) but were slightly different in the annotated dataset. A good example of this would be an occurrence that contains only one melodic voice. Our algorithm only finds points in time in which an occurrence might be included, it does not perform any type of source separation in order to identify the different voices. If the tolerance decreases sufficiently, a polyphonic occurrence would be accepted as similar to a monophonich one corresponding to the same points in time.

Our three layer score ($F_3$) is the best result when using audio recordings, with an F-measure of 31.74% (unfortunately this metric was not reported in [2]). This metric aims to evaluate the quality of the algorithm with a single

Table 1. Results of various algorithms using the JKU Patterns Development Dataset, averaged across pieces. The top rows of the table contain algorithms that use deadpan audio as input. The bottom rows correspond to algorithms that use symbolic representations as input.

<table>
<thead>
<tr>
<th>Alg</th>
<th>$P_{est}$</th>
<th>$R_{est}$</th>
<th>$P_{est}$</th>
<th>$R_{est}$</th>
<th>$P_{est}$</th>
<th>$R_{est}$</th>
<th>$P_{est}$</th>
<th>$R_{est}$</th>
<th>$P_{est}$</th>
<th>$R_{est}$</th>
<th>$P_{est}$</th>
<th>$R_{est}$</th>
<th>$P_{est}$</th>
<th>$R_{est}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed</td>
<td>54.96</td>
<td>51.73</td>
<td>49.80</td>
<td>37.58</td>
<td>27.61</td>
<td>31.79</td>
<td>35.12</td>
<td>35.28</td>
<td>32.01</td>
<td>45.17</td>
<td>34.98</td>
<td>38.73</td>
<td>62.9</td>
<td>51.9</td>
</tr>
<tr>
<td>[3]</td>
<td>14.9</td>
<td>60.9</td>
<td>23.94</td>
<td>32.08</td>
<td>21.24</td>
<td>24.87</td>
<td>30.43</td>
<td>31.92</td>
<td>28.23</td>
<td>26.60</td>
<td>20.94</td>
<td>23.18</td>
<td>56.87</td>
<td>–</td>
</tr>
<tr>
<td>[15]</td>
<td>40.83</td>
<td>46.43</td>
<td>41.43</td>
<td>65.40</td>
<td>76.40</td>
<td>68.40</td>
<td>40.40</td>
<td>54.40</td>
<td>44.20</td>
<td>57.00</td>
<td>71.60</td>
<td>63.20</td>
<td>7297</td>
<td>–</td>
</tr>
<tr>
<td>[3]</td>
<td>21.5</td>
<td>78.0</td>
<td>33.7</td>
<td>78.3</td>
<td>74.7</td>
<td>76.5</td>
<td>–</td>
<td>78.3</td>
<td>74.7</td>
<td>76.5</td>
<td>–</td>
<td>78.3</td>
<td>74.7</td>
<td>76.5</td>
</tr>
<tr>
<td>[12]</td>
<td>43.60</td>
<td>63.80</td>
<td>50.20</td>
<td>43.60</td>
<td>63.80</td>
<td>50.20</td>
<td>43.60</td>
<td>63.80</td>
<td>50.20</td>
<td>43.60</td>
<td>63.80</td>
<td>50.20</td>
<td>43.60</td>
<td>63.80</td>
</tr>
</tbody>
</table>

5. CONCLUSIONS

We presented a method to discover repeating polyphonic patterns using audio recordings as input. The method makes use of various standard techniques typically used for music segmentation. We evaluated our method using the JKU Pattern Development Dataset and showed how it obtains competent results when retrieving all the occurrences of the patterns and state-of-the-art results when finding patterns. When the algorithm is compared to others that use symbolic representations, we see that it is comparable or superior in terms of the correct patterns found. In future work, source separation might be needed to successfully identify patterns that only comprise a subset of the different musical lines.

6. REFERENCES


score, including both pattern establishment and occurrence retrieval. Our results are still far from perfect (32.01%), but when compared to an algorithm that uses symbolic representations [12] (44.21%), it appears our results are not far from the state-of-the-art for symbolic representations.

Finally, our algorithm takes more than twice as long as [15]. However, our method is over 16 times faster than [12], indicating it is efficient in terms of computation time. This algorithm is implemented in Python and available for public download.2

2 https://github.com/urinieto/MotivesExtractor
BOUNDARY DETECTION IN MUSIC STRUCTURE ANALYSIS USING CONVOLUTIONAL NEURAL NETWORKS

Karen Ullrich Jan Schlüter Thomas Grill
Austrian Research Institute for Artificial Intelligence, Vienna
{karen.ullrich,jan.schlueter,thomas.grill}@ofai.at

ABSTRACT

The recognition of boundaries, e.g., between chorus and verse, is an important task in music structure analysis. The goal is to automatically detect such boundaries in audio signals so that the results are close to human annotation. In this work, we apply Convolutional Neural Networks to the task, trained directly on mel-scaled magnitude spectrograms. On a representative subset of the SALAMI structural annotation dataset, our method outperforms current techniques in terms of boundary retrieval F-measure at different temporal tolerances: We advance the state-of-the-art from 0.33 to 0.46 for tolerances of ±0.5 seconds, and from 0.52 to 0.62 for tolerances of ±3 seconds. As the algorithm is trained on annotated audio data without the need of expert knowledge, we expect it to be easily adaptable to changed annotation guidelines and also to related tasks such as the detection of song transitions.

1. INTRODUCTION

The determination of the overall structure of a piece of audio, often referred to as musical form, is one of the key tasks in music analysis. Knowledge of the musical structure enables a variety of real-world applications, be they commercially applicable, such as for browsing music, or educational. A large number of different techniques for automatic structure discovery have been developed, see [16] for an overview. Our contribution describes a novel approach to retrieve the boundaries between the main structural parts of a piece of music. Depending on the music under examination, the task of finding such musical boundaries can be relatively simple or difficult, in the latter case leaving ample space for ambiguity. In fact, two human annotators hardly ever annotate boundaries at the exact same positions. Instead of trying to design an algorithm that works well in all circumstances, we let a Convolutional Neural Network (CNN) learn to detect boundaries from a large corpus of human-annotated examples.

The structure of the paper is as follows: After giving an overview over related work in Section 2, we describe our proposed method in Section 3. In Section 4, we introduce the data set used for training and testing. After presenting our main results in Section 5, we wrap up in Section 6 with a discussion and outlook.

2. RELATED WORK

In the overview paper to audio structure analysis by Paulus et al. [16], three fundamental approaches to segmentation are distinguished: Novelty-based, detecting transitions between contrasting parts, homogeneity-based, identifying sections that are consistent with respect to their musical properties, and repetition-based, building on the determination of recurring patterns. Many segmentation algorithms follow mixed strategies. Novelty is typically computed using Self-Similarity Matrices (SSMs) or Self-Distance Matrices (SDMs) with a sliding checkerboard kernel [4], building on audio descriptors like timbre (MFCC features), pitch, chroma vectors and rhythm features [14]. Alternative approaches calculate difference features on more complex audio feature sets [21]. In order to achieve a higher temporal accuracy in rhythmic music, audio features can be accumulated beat-synchronously. Techniques capitalizing on homogeneity use clustering [5] or state-modelling (HMM) approaches [1], or both [9, 11]. Repeating pattern discovery is performed on SSMs or SDMs [12], and often combined with other approaches [13, 15]. Some algorithms combine all three basic approaches [18].

Almost all existing algorithms are hand-designed from end to end. To the best of our knowledge, only two methods are partly learning from human annotations: Turnbull et al. [21] compute temporal differences at three time scales over a set of standard audio features including chromagrams, MFCCs, and fluctuation patterns. Training Boosted Decision Stumps to classify the resulting vectors into boundaries and non-boundaries, they achieved significant gains over a hand-crafted boundary detector using the same features, evaluated on a set of 100 pop songs. McFee et al. [13] employ Ordinal Linear Discriminant Analysis to learn a linear transform of beat-aligned audio features (including MFCCs and chroma) that minimizes the variance within a human-annotated segment while maximizing the distance across segments. Combined with a repetition feature, their method defines the current state of the art in boundary retrieval, but still involves significant manual engineering.

For other tasks in the field of Music Information Retrieval, supervised learning with CNNs has already proven
to outperform hand-designed algorithms, sometimes by a large margin [3, 6, 8, 10, 17]. In this work, we investigate whether CNNs are effective for structural boundary detection as well.

3. METHOD

We propose to train a neural network on human annotations to predict likely musical boundary locations in audio data. Our method is derived from Schlüter and Böck [17], who use CNNs for onset detection. We also train a CNN as a binary classifier on spectrogram excerpts, but we adapt their method to include a larger input context and respect the higher inaccuracy and scarcity of segment boundary annotations compared to onset annotations. In the following, we will describe the features, neural network, supervised training procedure and the post-processing of the network output to obtain boundary predictions.

3.1 Feature Extraction

For each audio file, we compute a magnitude spectrogram with a window size of 46 ms (2048 samples at 44.1 kHz) and 50% overlap, apply a mel filterbank of 80 triangular filters from 80 Hz to 16 kHz and scale magnitudes logarithmically. To be able to train and predict on spectrogram excerpts near the beginning and end of a file, we pad the spectrogram with pink noise at -70 dB as needed (padding with silence is impossible with logarithmic magnitudes, and white noise is too different from the existing background noise in natural recordings). To bring the input values to a range suitable for neural networks, we follow [17] in normalizing each frequency band to zero mean and unit variance. Finally, to allow the CNN to process larger temporal contexts while keeping the input size reasonable, we subsample the spectrogram by taking the maximum over 3, 6 or 12 adjacent time frames (without overlap), resulting in a frame rate of 14.35 fps, 7.18 fps or 3.59 fps, respectively. We will refer to these frame rates as high, std and low.

We also tried training on MFCCs and chroma vectors (descriptors with less continuity in the ‘vertical’ feature dimension to be exploited by convolution), as well as fluctuation patterns and self-similarity matrices derived from those. Overall, mel spectrograms proved the most suitable for the algorithm and performed best.

3.2 Convolutional Neural Networks

CNNs are feed-forward neural networks usually consisting of three types of layers: Convolutional layers, pooling layers and fully-connected layers. A convolutional layer computes a convolution of its two-dimensional input with a fixed-size kernel, followed by an element-wise nonlinearity. The input may consist of multiple same-sized channels, in which case it convolves each with a separate kernel and adds up the results. Likewise, the output may consist of multiple channels computed with distinct sets of kernels. Typically the kernels are small compared to the input, allowing CNNs to process large inputs with few learnable parameters. A pooling layer subsamples its two-dimensional input, possibly by different factors in the two dimensions, handling each input channel separately. Here, we only consider max-pooling, which introduces some translation invariance across the subsampled dimension. Finally, a fully-connected layer discards any spatial layout of its input by reshaping it into a vector, computes a dot product with a weight matrix and applies an element-wise nonlinearity to the result. Thus, unlike the other layer types, it is not restricted to local operations and can serve as the final stage integrating all information to form a decision.

In this work, we fix the network architecture to a convolutional layer of 16 $8 \times 6$ kernels (8 time frames, 6 mel bands, 16 output channels), a max-pooling layer of $3 \times 6$, another convolution of $32 \times 3 \times 6$ kernels, a fully-connected layer of 128 units and a fully-connected output layer of 1 unit. This architecture was determined in preliminary experiments and not further optimized for time constraints.

3.3 Training

The input to the CNN is a spectrogram excerpt of $N$ frames, and its output is a single value giving the probability of a boundary in the center of the input. The network is trained in a supervised way on pairs of spectrogram excerpts and binary labels. To account for the inaccuracy of the ground truth boundary annotations (as observable from the disagreement between two humans annotating the same piece), we employ what we will refer to as target smearing: All excerpts centered on a frame within $\pm E$ frames from an annotated boundary will be presented to the network as positive examples, weighted in learning by a Gaussian kernel centered on the boundary. Figure 1 illustrates this for $E = 10$. We will vary both the spectrogram length $N$ and smearing environment $E$ in our experiments. To compensate for the scarceness of positive examples, we increase their chances of being randomly selected for a training step by a factor of 3.

Training is performed using gradient descent on cross-
entropy error with mini-batches of 64 examples, momentum of 0.95, and an initial learning rate of 0.6 multiplied by 0.85 after every mini-epoch of 2000 weight updates. We apply 50% dropout to the inputs of both fully-connected layers [7]. Training is always stopped after 20 mini-epochs, as the validation error turned out not to be robust enough for early stopping. Implemented in Theano [2], training a single CNN on an Nvidia GTX 780 Ti graphics card took 50–90 minutes.

3.4 Peak-picking
At test time, we apply the trained network to each position in the spectrogram of the music piece to be segmented, obtaining a boundary probability for each frame. We then employ a simple means of peak-picking on this boundary activation curve: Every output value that is not surpassed within ±6 seconds is a boundary candidate. From each candidate value we subtract the average of the activation curve in the past 12 and future 6 seconds, to compensate for long-term trends. We end up with a list of boundary candidates along with strength values that can be thresholded at will. We found that more elaborate peak picking methods did not improve results.

4. DATASET
We evaluate our algorithm on a subset of the Structural Analysis of Large Amounts of Music Information (SALAMI) database [20]. In total, this dataset contains over 2400 structural annotations of nearly 1400 musical recordings of different genres and origins. About half of the annotations (779 recordings, 498 of which are doubly-annotated) are publicly available.1 A part of the dataset was also used in the “Audio Structural Segmentation” task of the annual MIREX evaluation campaign in 2012 and 2013.2 Along with quantitative evaluation results, the organizers published the ground truth and predictions of 17 different algorithms for each recording. By matching the ground truth to the public SALAMI annotations, we were able to identify 487 recordings. These serve as a test set to evaluate our algorithm against the 17 MIREX submissions. We had another 733 recordings at our disposal, annotated following the SALAMI guidelines, which we split into 633 items for training and 100 for validation.

5. EXPERIMENTAL RESULTS
5.1 Evaluation
For boundary retrieval, the MIREX campaign uses two evaluation measures: Median deviation and Hit rate. The former measures the median distance between each annotated boundary and its closest predicted boundary or vice versa. The latter checks which predicted boundaries fall close enough to an unmatched annotated boundary (true positives), records remaining unmatched predictions and annotations as false positives and negatives, respectively, then computes the precision, recall and F-measure. Since not only the temporal distance of predictions, but also the figures of precision and recall are of interest, we opted for the Hit rate at as our central measure of evaluation, computed at a temporal tolerance of ±0.5 seconds (as in [21]) and ±3 seconds (as in [9]). For accumulation over multiple recordings, we follow the MIREX evaluation by calculating F-measure, precision and recall per item and averaging the three measures over the items for the final result. Note that the averaged F-measure is not necessarily the harmonic mean of the averaged precision and recall. Our evaluation code is publicly available for download.3

5.2 Baseline and upper bound
Our focus for evaluation lies primarily on the F-measure. Theoretically, the F-measure is bounded by $F \in [0, 1]$, but for the given task, we can derive more useful lower and upper bounds to compare our results to. As a baseline, we use regularly spaced boundary predictions starting at time 0. Choosing an optimal spacing, we obtain an F-measure of $F_{inf,3} \approx 0.33$ for ±3 seconds tolerance, and $F_{inf,0.5} \approx 0.13$ for a tolerance of ±0.5 seconds. Note that it is crucial to place the first boundary at 0 seconds, where a large fraction of the music pieces has annotated segment boundaries. Many pieces have only few boundaries at all, thus the impact can be considerable. An upper bound $F_{sup}$ can be derived from the insight that no annotation will be perfect given the fuzzy nature of the segmentation task. Even though closely following annotation guidelines,4 two annotators might easily disagree on the existence or exact po-

---

3 http://ofai.at/research/impml/projects/audiostreams/ismir2014/

Figure 2. Optimization of the threshold shown for model 8s_std_3s at tolerance ±0.5 seconds. Boundary retrieval precision, recall and F-measure are averaged over the 100 validation set files.
5.3 Threshold optimization

Peak-picking, described in Section 3.4, delivers the positions of potential boundaries along with their probabilities, as calculated by the CNN. The application of a threshold to those probabilities rejects part of the boundaries, affecting the precision and recall rates and consequently the F-measure we use for evaluation. Figure 2 shows precision and recall rates as well as the F-measure as a function of the threshold for the example of the $8s_{std \_3s}$ model ($8$ seconds of context, standard resolution, target smearing $3$ seconds) at $\pm 0.5$ seconds tolerance, applied to the $100$ files of the validation data set. By locating the maximum of the F-measure we retrieve an estimate for the optimum threshold which is specific for each individual learned model. Since the curve for the F-measure is typically flat-topped for a relatively wide range of threshold values, the choice of the actual value is not very delicate.

5.4 Temporal context investigation

It is intuitive to assume that the CNN needs a certain amount of temporal context to reliably judge the presence of a boundary. Furthermore, the temporal resolution of the input spectra (Section 3.1) and the applied target smearing (Section 3.3) is expected to have an impact on the temporal accuracy of the predictions. See Figure 3 and Figure 4 for comparisons of these model parameters, for tolerances $\pm 0.5$ seconds and $\pm 3$ seconds, respectively. Each bar in the plots represents the mean and minimum-maximum range of five individual experiments with different random initializations. For the case of only $\pm 0.5$ seconds of acceptable error, we conclude that target smearing must also be small: A smearing width of $1$ to $1.5$ seconds performs best. Low temporal spectral resolution tends to diminish results, and the context length should not be shorter than $8$ seconds. For $\pm 3$ seconds tolerance, context length and target smearing are the most influential parameters, with the F-measure peaking at $32$ seconds context and $4$ to $6$ seconds smearing. Low temporal resolution is sufficient, keeping the CNN smaller and easier to train.

5.5 Model bagging

As described in Section 5.4, for each set of parameters we trained five individual models. This allows us to improve the performance on the given data using a statistical approach: Bagging, in our case averaging the outputs of multiple identical networks trained from different initializations before the peak-picking stage, should help to reduce model uncertainty. After again applying the above described threshold optimization process on the resulting boundaries, we arrived at improvements of the F-measure of up to $0.03$, indicated by arrow tips in Figures 3 and 4. Tables 1 and 2 show our final best results after model bagging for tolerances $\pm 0.5$ seconds and $\pm 3$ seconds, respectively. Each bar in the plots represents the mean and minimum-maximum range of five individual experiments with different random initializations.
Table 1. Boundary recognition results on our test set at ±0.5 seconds tolerance. Our best result is emphasized and compared with results from the MIREX campaign in 2012 and 2013.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>F-measure</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper bound (est.)</td>
<td>0.68</td>
<td>0.7059</td>
<td></td>
</tr>
<tr>
<td>16s_std_1.5s</td>
<td>0.4646</td>
<td>0.5944</td>
<td>0.7059</td>
</tr>
<tr>
<td>MP2 (2013)</td>
<td>0.3280</td>
<td>0.5648</td>
<td>0.6675</td>
</tr>
<tr>
<td>MP1 (2013)</td>
<td>0.5213</td>
<td>0.4793</td>
<td>0.6443</td>
</tr>
<tr>
<td>OYZS1 (2012)</td>
<td>0.2899</td>
<td>0.4561</td>
<td>0.2583</td>
</tr>
<tr>
<td>32s_low_6s</td>
<td>0.2884</td>
<td>0.3592</td>
<td>0.2680</td>
</tr>
<tr>
<td>KSPP2 (2012)</td>
<td>0.2866</td>
<td>0.2262</td>
<td>0.4622</td>
</tr>
<tr>
<td>SP1 (2012)</td>
<td>0.2788</td>
<td>0.2202</td>
<td>0.4497</td>
</tr>
<tr>
<td>KSPP3 (2012)</td>
<td>0.2788</td>
<td>0.2202</td>
<td>0.4497</td>
</tr>
<tr>
<td>KSPP1 (2012)</td>
<td>0.2788</td>
<td>0.2201</td>
<td>0.4495</td>
</tr>
<tr>
<td>RBH1 (2013)</td>
<td>0.2683</td>
<td>0.2493</td>
<td>0.3360</td>
</tr>
<tr>
<td>RBH2 (2013)</td>
<td>0.267</td>
<td>0.2043</td>
<td>0.3936</td>
</tr>
<tr>
<td>RBH4 (2013)</td>
<td>0.2567</td>
<td>0.2043</td>
<td>0.3936</td>
</tr>
<tr>
<td>CF5 (2013)</td>
<td>0.2128</td>
<td>0.1677</td>
<td>0.3376</td>
</tr>
<tr>
<td>CF6 (2013)</td>
<td>0.2101</td>
<td>0.2396</td>
<td>0.2239</td>
</tr>
<tr>
<td>SMGA1 (2012)</td>
<td>0.1968</td>
<td>0.1573</td>
<td>0.2943</td>
</tr>
<tr>
<td>MHRFA1 (2012)</td>
<td>0.1910</td>
<td>0.1941</td>
<td>0.2081</td>
</tr>
<tr>
<td>SMGA2 (2012)</td>
<td>0.1770</td>
<td>0.1425</td>
<td>0.2618</td>
</tr>
<tr>
<td>SBV1 (2012)</td>
<td>0.1546</td>
<td>0.1308</td>
<td>0.2129</td>
</tr>
<tr>
<td>Baseline (est.)</td>
<td>0.13</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Boundary recognition results on our test set at ±3 seconds tolerance. Our best result is emphasized and compared with results from the MIREX campaign in 2012 and 2013.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>F-measure</th>
<th>Precision</th>
<th>Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper bound (est.)</td>
<td>0.76</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32s_low_6s</td>
<td>0.6164</td>
<td>0.5944</td>
<td>0.7059</td>
</tr>
<tr>
<td>16s_std_1.5s</td>
<td>0.5726</td>
<td>0.5648</td>
<td>0.6675</td>
</tr>
<tr>
<td>MP2 (2013)</td>
<td>0.5213</td>
<td>0.4793</td>
<td>0.6443</td>
</tr>
<tr>
<td>MP1 (2013)</td>
<td>0.5188</td>
<td>0.5040</td>
<td>0.5849</td>
</tr>
<tr>
<td>CF5 (2013)</td>
<td>0.5052</td>
<td>0.3990</td>
<td>0.7862</td>
</tr>
<tr>
<td>SMGA1 (2012)</td>
<td>0.4985</td>
<td>0.4021</td>
<td>0.7258</td>
</tr>
<tr>
<td>RBH1 (2013)</td>
<td>0.4920</td>
<td>0.3922</td>
<td>0.7482</td>
</tr>
<tr>
<td>RBH2 (2013)</td>
<td>0.4920</td>
<td>0.3922</td>
<td>0.7482</td>
</tr>
<tr>
<td>RBH4 (2013)</td>
<td>0.4920</td>
<td>0.3922</td>
<td>0.7482</td>
</tr>
<tr>
<td>SP1 (2012)</td>
<td>0.4891</td>
<td>0.3854</td>
<td>0.7842</td>
</tr>
<tr>
<td>KSPP3 (2012)</td>
<td>0.4891</td>
<td>0.3854</td>
<td>0.7842</td>
</tr>
<tr>
<td>KSPP1 (2012)</td>
<td>0.4888</td>
<td>0.3850</td>
<td>0.7838</td>
</tr>
<tr>
<td>KSPP2 (2012)</td>
<td>0.4885</td>
<td>0.3846</td>
<td>0.7843</td>
</tr>
<tr>
<td>SMGA2 (2012)</td>
<td>0.4815</td>
<td>0.3910</td>
<td>0.6965</td>
</tr>
<tr>
<td>RBH3 (2013)</td>
<td>0.4804</td>
<td>0.4407</td>
<td>0.6076</td>
</tr>
<tr>
<td>CF6 (2013)</td>
<td>0.4759</td>
<td>0.5305</td>
<td>0.5102</td>
</tr>
<tr>
<td>OYZS1 (2012)</td>
<td>0.4401</td>
<td>0.6354</td>
<td>0.4038</td>
</tr>
<tr>
<td>SBV1 (2012)</td>
<td>0.4352</td>
<td>0.3694</td>
<td>0.5929</td>
</tr>
<tr>
<td>MHRFA1 (2012)</td>
<td>0.4192</td>
<td>0.4342</td>
<td>0.4447</td>
</tr>
<tr>
<td>Baseline (est.)</td>
<td>0.33</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

6. DISCUSSION AND OUTLOOK

Employing Convolutional Neural Networks trained directly on mel-scaled spectrograms, we are able to achieve boundary recognition F-measures strongly outperforming any algorithm submitted to MIREX 2012 and 2013. The networks have been trained on human-annotated data, considering different context lengths, temporal target smearing and spectrogram resolutions. As we did not need any domain knowledge for training, we expect our method to be easily adaptable to different ‘foci of annotation’ such as, e.g., determined by different musical genres or annotation guidelines. In fact, our method is itself an adaption of a method for onset detection [17] to a different time focus.

There are a couple of conceivable strategies to improve the results further: With respect to the three fundamental approaches to segmentation described in Section 1, the CNNs in this work can only account for novelty and homogeneity, which can be seen as two sides of the same medal. To allow them to leverage repetition cues as well, the vectorial repetition features of McFee et al. [13] might serve as an additional input. Alternatively, the network could be extended with recurrent connections to yield a Recurrent CNN. Given suitable training data, the resulting memory might be able to account for repeating patterns. Secondly, segmentation of musical data by humans is not a trivially sequential process but inherently hierarchical. The SALAMI database actually provides annotations on two levels: A coarse one, as used in the MIREX campaign, but also a more fine-grained variant, encoding subtler details of the temporal structure. It could be helpful to feed both levels to the CNN training, weighted with respect to the significance. Thirdly, we leave much of the data preprocessing to the CNN, very likely using up a considerable part of its capacity. For example, the audio files in the SALAMI collection are of very different loudness, which could be fixed in a simple preprocessing step, either on the whole files, or using some dynamic gain control. Similarly, many of the SALAMI audio files start or end with noise or background sounds. A human annotator easily recognizes this as not belonging to the actual musical content, ignoring it in the annotations. The abrupt change from song-specific background noise to our pink noise padding may be mistaken for a boundary by the CNN, though. Therefore it could be worthwhile to apply some intelligent padding of appropriate noise or background to provide context at the beginnings and endings of the audio. And finally, we have only explored a fraction of the hyperparameter space regarding network architecture and learning, and expect further improvements by a systematic optimization of these.

Another promising direction of research is to explore the internal processing of the trained networks, e.g., by visualization of connection weights and receptive fields [19]. This may help to understand the segmentation process as well as differences to existing approaches, and to refine the network architecture.
7. ACKNOWLEDGMENTS

This research is funded by the Federal Ministry for Transport, Innovation & Technology (BMVIT) and the Austrian Science Fund (FWF): TRP 307-N23. Many thanks to the anonymous reviewers for your valuable feedback!

8. REFERENCES


Oral Session 6

Cultures
TRACKING THE “ODD”: METER INFERENCE IN A CULTURALLY DIVERSE MUSIC CORPUS

Andre Holzapfel  
New York University Abu Dhabi  
andre@rhythmos.org

Florian Krebs  
Johannes Kepler University  
Florian.Krebs@jku.at

Ajay Srinivasamurthy  
Universitat Pompeu Fabra  
ajays.murthy@upf.edu

ABSTRACT

In this paper, we approach the tasks of beat tracking, downbeat recognition and rhythmic style classification in non-Western music. Our approach is based on a Bayesian model, which infers tempo, downbeats and rhythmic style, from an audio signal. The model can be automatically adapted to rhythmic styles and time signatures. For evaluation, we compiled and annotated a music corpus consisting of eight rhythmic styles from three cultures, containing a variety of meter types. We demonstrate that by adapting the model to specific styles, we can track beats and downbeats in odd meter types like 9/8 or 7/8 with an accuracy significantly improved over the state of the art. Even if the rhythmic style is not known in advance, a unified model is able to recognize the meter and track the beat with comparable results, providing a novel method for inferring the metrical structure in culturally diverse datasets.

1. INTRODUCTION

Musical rhythm subordinated to a meter is a common feature in many music cultures around the world. Meter provides a hierarchical time structure for the rendition and repetition of rhythmic patterns. Though these metrical structures vary considerably across cultures, metrical hierarchies can often be stratified into levels of differing time spans. Two of these levels are, in terminology of Eurogenetic music, referred to as beats, and measures. The beats are the pulsation at the perceptually most salient metrical level, and are further grouped into measures. The first beat of each measure is called the downbeat. Determining the type of the underlying meter, and the alignment between the pulsations at the levels of its hierarchy with music performance recordings – a process we refer to as meter inference – is fundamental to computational rhythm analysis and supports many further tasks, such as music transcription, structural analysis, or similarity estimation.

The automatic annotation of music with different aspects of rhythm is at the focus of numerous studies in Music Information Retrieval (MIR). Müller et al [5] discussed the estimation of the beat (called beat tracking), and the estimation of higher-level metrical structures such as the measure length. Approaches such as the one presented by Klapuri et al [3] aim at estimating structures at several metrical levels, while being able to differentiate between certain time signatures. In [7] beats and downbeats are estimated simultaneously, given information about the tempo and the meter of a piece. Most of these approaches assume the presence of a regular metrical grid, and work reasonably well for Eurogenetic popular music. However, their adaptation to different rhythmic styles and metrical structures is not straightforward.

Recently, a Bayesian approach referred to as bar pointer model has been presented [11]. It aims at the joint estimation of rhythmic pattern, the tempo, and the exact position in a metrical cycle, by expressing them as hidden variables in a Hidden Markov Model (HMM) [8]. Krebs et al. [4] applied the model to music signals and showed that explicitly modelling rhythmic patterns is useful for meter inference for a dataset of Ballroom dance music.

In this paper, we adapt the observation model of the approach presented in [4] to a collection of music from different cultures: Makam music from Turkey, Cretan music from Greece, and Carnatic music from the south of India. The adaption of observation models was shown to be of advantage in [4, 6], however restricted to the context of Ballroom dance music. Here, we extract rhythmic patterns from culturally more diverse data, and investigate if their inclusion into the model improves the performance of meter inference. Furthermore, we investigate if a unified model can be derived that covers all rhythmic styles and time signatures that are present in the training data.

2. MOTIVATION

The music cultures considered in this paper are based on traditions that can be traced back for centuries until the present, and were documented by research in ethnomusicology for decades. Rhythm in two of these cultures, Carnatic and Turkish Makam music, is organized based on potentially long metrical cycles. All three make use of rhythmic styles that deviate audibly from the stylistic paradigms of Eurogenetic popular music. Previous studies on music collections of these styles have shown that the current state of the art performs poorly in beat tracking [2, 9] and the recognition of rhythm class [9]. As suggested in [9], we explore a unified approach for meter inference that can rec-
ognize the rhythmic style of the piece and track the meter at the same time.

The bar pointer model, as described in Section 4, can be adapted to rhythmic styles by extracting possible patterns using small representative downbeat annotated datasets. This way, we can obtain an adapted system for a specific style without recoding and parameter tweaking. We believe that this is an important characteristic for algorithms applied in music discovery and distribution systems for a large and global audience. Through this study, we aim to answer crucial questions: Do we need to differentiate between rhythmic styles in order to track the meter, or is a universal approach sufficient? For instance, can we track a rhythmic style in Indian music using rhythmic patterns derived from Turkish music? Do we need to learn patterns at all? If a particular style description for each style is needed, this has some serious consequences for the scalability of rhythmic similarity and meter inference methods; while we should ideally aim at music discovery systems without an ethnocentric bias, the needed universal analysis methods might come at a high cost given the high diversity in the musics of the world.

3. MUSIC CORPORA

In this paper we use a collection of three music corpora which are described in the following.

The corpus of Cretan music consists of 42 full length pieces of Cretan leaping dances. While there are several dances that differ in terms of their steps, the differences in the sound are most noticeable in the melodic content, and we consider all pieces to belong to one rhythmic style. All these dances are usually notated using a 2/4 time signature, and the accompanying rhythmical patterns are usually played on a Cretan lute. While a variety of rhythmical patterns exist, they do not relate to a specific dance and can be assumed to occur in all of the 42 songs in this corpus.

The Turkish corpus is an extended version of the annotated data used in [9]. It includes 82 excerpts of one minute length each, and each piece belongs to one of three rhythm classes that are referred to as usul in Turkish Art music. 32 pieces are in the 9/8-usul Aksak, 20 pieces in the 10/8-usul Curcuna, and 30 samples in the 8/8-usul Düyüek.

The Carnatic music corpus is a subset of the annotated dataset used in [10]. It includes 118 two minute long excerpts spanning four tālas (the rhythmic framework of Carnatic music, consisting of time cycles). There are 30 examples in each of ādi tāla (8 beats/cycle), rāpaka tāla (3 beats/cycle) and mishaṇa chāpu tāla (7 beats/cycle), and 28 examples in khanda chāpu tāla (5 beats/cycle).

All excerpts described above were manually annotated with beats and downbeats. Note that for both Indian and Turkish music the cultural definition of the rhythms contain irregular beats. Since the irregular beat sequence is a subset of the (annotated) equidistant pulses, it can be derived easily from the result of a correct meter inference. For further details on meter in Carnatic and Turkish makam music, please refer to [9].

4. METER INFERENCE METHOD

4.1 Model description

To infer the metrical structure from an audio signal we use the bar pointer model, originally proposed in [11] and refined in [4]. In this model we assume that a bar pointer traverses a bar and describe the state of this bar pointer at each audio frame k by three (hidden) variables: tempo, rhythmic pattern, and position inside a bar. These hidden variables can be inferred from an (observed) audio signal by using an HMM. An HMM is defined by three quantities: A transition model which describes the transitions between the hidden variables, an observation model which describes the relation between the hidden states and the observations (i.e., the audio signal), and an initial distribution which represents our prior knowledge about the hidden states.

4.1.1 Hidden states

The three hidden variables of the bar pointer model are:

- Rhythm pattern index \( r_k \in \{r_1, r_2, ..., r_R\} \), where \( R \) is the number of different rhythmic patterns that we consider to be present in our data. Further, we denote the time signature of each rhythmic pattern by \( \theta(r_k) \) (e.g., 9/8 for Aksak patterns). In this paper, we assume that each rhythmic pattern belongs to a rhythmic class, and a rhythmic class (e.g., Aksak, Duyek) can hold several rhythmic patterns. We investigate the optimal number of rhythmic patterns per rhythm class in Section 5.
- Position within a bar \( m_k \in \{1, 2, ..., M(r_k)\} \): We subdivide a whole note duration into 1600 discrete, equidistant bar positions and compute the number of positions within a bar with rhythm \( r_k \) by \( M(r_k) = 1600 \cdot \theta(r_k) \) (e.g., a bar with 9/8 meter has 1600 \cdot 9/8 = 1800 bar positions).
- Tempo \( n_k \in \{n_{\text{min}}(r_k), ..., n_{\text{max}}(r_k)\} \): The tempo can take on positive integer values, and quantifies the number of bar positions per audio frame. Since we use an audio frame length of 0.02s, this translates to a tempo resolution of 7.5 \((= \frac{60}{1/4 \cdot 1600 \cdot 0.02})\) beats per minute (BPM) at the quarter note level. We set the minimum tempo \( n_{\text{min}}(r_k) \) and the maximum tempo \( n_{\text{max}}(r_k) \) according to the rhythmic pattern \( r_k \).

4.1.2 Transition model

We use the transition model proposed in [4, 11] with the difference that we allow transitions between rhythmic pattern states within a song as shown in Equation 3. In the following we list the transition probabilities for each of the three variables:

- \( P(m_k|m_{k-1}, n_{k-1}, r_{k-1}) \): At time frame \( k \) the bar pointer moves from position \( m_{k-1} \) to \( m_k \) as defined by \( m_k = [(m_{k-1} + n_{k-1} - 1) \mod (M(r_{k-1}))] + 1 \). (1)
- Whenever the bar pointer crosses a bar border it is reset to 1 (as modeled by the modulo operator).
- \( P(n_k|n_{k-1}, r_{k-1}) \): If the tempo \( n_{k-1} \) is inside the allowed tempo range \( \{n_{\text{min}}(r_{k-1}), ..., n_{\text{max}}(r_{k-1})\} \),
there are three possible transitions: the bar pointer remains at the same tempo, accelerates, or decelerates:

\[
P(n_k | n_{k-1}) = \begin{cases} 
1 - p_n, & n_k = n_{k-1} \\
p^m_n, & n_k = n_{k-1} + 1 \\
p^s_n, & n_k = n_{k-1} - 1 
\end{cases}
\] (2)

Transitions to tempi outside the allowed range are assigned a zero probability. \(p_n\) is the probability of a change in tempo per audio frame, and was set to \(p_n = 0.02\), the tempo ranges \((n_{\text{min}}(r), n_{\text{max}}(r))\) for each rhythmic pattern are learned from the data (Section 4.2).

- \(P(r_k | r_{k-1})\): Finally, the rhythmic pattern state is assumed to change only at bar boundaries:

\[
P(r_k | r_{k-1}, m_k < m_{k-1}) = p_r(r_{k-1}, r_k) \quad (3)
\]

\(p_r(r_{k-1}, r_k)\) denotes the probability of a transition from pattern \(r_{k-1}\) to pattern \(r_k\) and will be learned from the training data as described in Section 4.2. In this paper we allow transitions only between patterns of the same rhythm class, which will force the system to assign a piece of music to one of the learned rhythm classes.

### 4.1.3 Observation model

In this paper, we use the observation model proposed in [4]. As summarized in Figure 1, a Spectral Flux-like onset feature, \(y\), is extracted from the audio signal (sampled with 44100 Hz) using the same parameters as in [4]. It summarizes the energy changes that are likely to be related to instrument onsets in two dimensions related to two frequency bands, above and below 250 Hz. In contrast to [4] we removed the normalizing step at the end of the feature computations, which we observed not to influence the results.

![Figure 1: Computing the onset feature y from the audio signal](image)

As described in [4], the observation probabilities \(P(y_k | m_k, n_k, r_k)\) are modeled by a set of Mixture of Gaussian distributions (GMM). As it is infeasible to specify a GMM for each state (this would result in \(N \times M \times R\) GMMs), we make two assumptions: First, we assume that the observation probabilities are independent of the tempo and second, we assume that the observation probabilities only change each 64th note (which corresponds to 1600/64=25 bar positions). Hence, for each rhythmic pattern, we have to specify 64 \(\times \theta(r)\) GMMs.

#### 4.1.4 Initial distribution

For each rhythmic pattern, we assume a uniform state distribution within the tempo limits and over all bar positions.

### 4.2 Learning parameters

The parameters of the observation GMMs, the transition probabilities of the rhythm pattern states, and the tempo ranges for each rhythmic style are learned from the data described in Section 3. In our experiments we perform a two-fold cross-validation, excluding those files from the evaluation that were used for parameter learning.

#### 4.2.1 Observation model

The parameters of the observation model consist of the mean values, covariance matrix and the component weights of the GMM for each 64th note of a rhythmic pattern. We determine these as follows:

1. The two-dimensional onset feature \(y\) (see Section 4.1.3) is computed from the training data.
2. The features are grouped by bar and bar position within the 64th note grid. If there are several feature values for the same bar and 64th note grid point, we compute the average, if there is no feature we interpolate between neighbors. E.g., for a rhythm class which spans a whole note (e.g., Düyek (8/8 meter)) this yields a matrix of size \(B \times 128\), where \(B\) is the number of bars with Düyek rhythm class in the dataset.
3. Each dimension of the features is normalized to zero mean and unit variance.
4. For each of the eight rhythm classes in the corpus described in Section 3, a k-means clustering algorithm assigns each bar of the dataset (represented by a point in a 128-dimensional space) to one rhythmic pattern. The influence of the number of clusters \(k\) on the accuracy of the metrical inference will be evaluated in the experiments.
5. For each rhythmic pattern, at all 64th grid points, we compute the parameters of the GMM by maximum likelihood estimation.

#### 4.2.2 Tempo ranges and transition probabilities

For each rhythmic pattern, we compute the minimum and maximum tempo of all bars of the training fold that were assigned to this pattern by the procedure described in Section 4.2.1. In the same way, we determine the transition probabilities \(p_r\) between rhythmic patterns.

### 4.3 Inference

In order to obtain beat-, downbeat-, and rhythmic class estimations, we compute the optimal state sequence \(\{m^*_1, n^*_1, r^*_1\}\) that maximizes the posterior probability of the hidden states given the observations \(y_{1:L}\) and hence fits best to our model and the observations. This is done using the well-known Viterbi algorithm [8].
5. EXPERIMENTS

5.1 Evaluation metrics

A variety of measures for evaluating beat and downbeat tracking performance are available (see [1] for a detailed overview and descriptions of the metrics listed below)\(^1\). We chose five metrics that are characterized by a set of diverse properties and are widely used in beat tracking evaluation.

### F-measure (F-measure): The F-measure is computed from correctly detected beats within a window of ±70 ms by

\[
F\text{-measure} = \frac{2 \times p \times r}{p + r}
\]

where \(p\) (precision) denotes the ratio between correctly detected beats and all detected beats, and \(r\) (recall) denotes the ratio between correctly detected beats and the total number of annotated beats. The range of this measure is from 0\% to 100\%.

### CMLt (Allowed Metrical Levels with no continuity required): The same as AMLt, without the tolerance for off-beat, or doubling/halving errors.

### infGain (Information Gain): Timing errors are calculated between an annotation and all beat estimations within a one-beat length window around the annotation. Then, a beat error histogram is formed from the resulting timing error sequence. A numerical score is derived by measuring the K-L divergence between the observed error histogram and the uniform case. This method gives a measure of how much information the beats provide about the annotations. The range of values for the Information Gain is 0 bits to much information the beats provide about the annotations.

### Db-Fmeas (Downbeat F-measure): For measuring the downbeat tracking performance, we use the same F-measure as defined for beat tracking (using a ±70 ms tolerance window).

5.2 Results

In Experiment 1, we learned the observation model described in Section 4.2 for various numbers of clusters, separately for each of the eight rhythm classes. Then, we inferred the meter using the HMM described in Section 4.1, again separately for each rhythm class. The results of this experiment indicate how many rhythmic patterns are needed for each class in order to achieve an optimal beat and downbeat tracking with the proposed model.

### Tables (1a) to (1h) show the performance with all the evaluation metrics for each of the eight styles separately. For Experiment 1 (Ex-1), all significant increases compared to the previous row are emphasized using bold numbers (according to paired-sample t-tests with 5\% significance level). In our experiments, increasing the number \(R\) of considered patterns from one to two leads to a statistically significant increase in most cases. Therefore, we can conclude that for tracking these individual styles, more than one pattern is always needed. Further increase to three patterns leads to significant improvement only in the exceptional case of Ādī tāla, where measure cycles with long durations and rich rhythmic improvisation apparently demand higher number of patterns and cause the system to perform worse than for other classes. Higher numbers than \(R = 3\) patterns never increased any of the metrics significantly. It is important to point out again that a test song was never used to train the rhythmic patterns in the observation model in Experiment 1.

The interesting question we address in Experiment 2 is if the rhythm class of a test song is a necessary information for an accurate meter inference. To this end, we performed meter inference for a test song combining all the determined rhythmic patterns for all classes in one large HMM. This means that in this experiment the HMM can be used to determine the rhythm class of a song, as well as for the tracking of beats and downbeats. We use two patterns from each rhythm class (except ādī tāla), the optimally performing number of patterns in Experiment 1, to construct the HMM. For ādī tāla, we use three patterns since using 3 patterns improved performance in Experiment 1, to give a total of \(R = 17\) different patterns for the large HMM. The results of Experiment 2 are depicted in the rows labeled Ex-2 in Tables (1a) to (1h), significant change over the optimal setting in Experiment 1 are emphasized using bold numbers. The general conclusion is that the system is capable of a combined task of classification into a rhythm class and the inference of the metrical structure of the signal. The largest and, with the exception of ādī tāla, only significant decrease between the Experiment 1 and Experiment 2 can be observed for the downbeat recognition (Db-Fmeas). The reason for this is that a confusion of a test song into a wrong class may still lead to a proper tracking of the beat level, but the tracking of the higher metrical level of the downbeat will suffer severely from assigning a piece to a class with a different length of the meter than the test piece.

As described in Section 4.1, we do not allow transitions between different rhythm classes. Therefore, we can classify a piece of music into a rhythm class by evaluating to which rhythmic pattern states \(r_k\) the piece was assigned. The confusion matrix is depicted in Table 2, and it shows that the highest confusion can be observed within certain classes of Carnatic music, while the Cretan leaping dances and the Turkish classes are generally recognized with higher recall rate. The accent patterns in mishra chāpū and khanda chāpū can be indefinite, non-characteristic and non-indicative in some songs, and hence there is a possi-

\(^1\)We used the MATLAB code available at [http://code.soundsoftware.ac.uk/projects/beat-evaluation/] with standard settings.
In this paper we adapted the observation model of a Bayesian approach for the inference of meter in music of cultures in Greece, India, and Turkey. It combines the task of determining the type of meter with the alignment of the downbeats and beats to the audio signal. The model is capable of performing the meter recognition with an accuracy that improves over the state of the art, and is at the same time able to achieve for the first time high beat and downbeat tracking accuracies in additive meters like the Turkish Aksak and Carnatic mishra chāpū.

Our results show that increasing the diversity of a corpus means increasing the number of the patterns, i.e. a larger

Table 1: Evaluation results for each rhythm class, for Experiment 1 (separate evaluation per style, shown as Ex-1), and Experiment 2 (combined evaluation using one large HMM, shown as Ex-2). The last row in each Table, with row header as KL, shows the beat tracking performance using Klapuri beat tracker. For Ex-1, bold numbers indicate significant change compared to the row above, for Ex-2, bold numbers indicate significant change over the best parameter setting in Ex-1 (bold R parameter), and for KL the only differences to Ex-2 that are not statistically significant are underlined.

### Results for Experiment 1 (Ex-1)

<table>
<thead>
<tr>
<th>Rhythm Class</th>
<th>KL Mean</th>
<th>CMLt Mean</th>
<th>infGain Mean</th>
<th>Db-Fmeas Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ex-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(a) Turkish Music: Aksak (9/8)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>Fmeas</td>
<td>CMLt</td>
<td>AMLt</td>
<td>infGain</td>
</tr>
<tr>
<td>1</td>
<td>65.9</td>
<td>45.0</td>
<td>57.6</td>
<td>0.89</td>
</tr>
<tr>
<td>2</td>
<td>91.0</td>
<td>76.6</td>
<td>90.0</td>
<td>1.62</td>
</tr>
<tr>
<td>3</td>
<td>90.6</td>
<td>77.2</td>
<td>91.1</td>
<td>1.59</td>
</tr>
<tr>
<td>Ex-2</td>
<td>17</td>
<td>85.7</td>
<td>68.7</td>
<td>89.3</td>
</tr>
<tr>
<td>KL</td>
<td>69.38</td>
<td>41.24</td>
<td>64.60</td>
<td>1.46</td>
</tr>
<tr>
<td>(b) Turkish Music: Curcuna (10/8)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>Fmeas</td>
<td>CMLt</td>
<td>AMLt</td>
<td>infGain</td>
</tr>
<tr>
<td>1</td>
<td>71.4</td>
<td>47.8</td>
<td>50.3</td>
<td>0.68</td>
</tr>
<tr>
<td>2</td>
<td>89.1</td>
<td>75.6</td>
<td>75.6</td>
<td>1.04</td>
</tr>
<tr>
<td>3</td>
<td>87.7</td>
<td>73.0</td>
<td>73.0</td>
<td>0.99</td>
</tr>
<tr>
<td>Ex-2</td>
<td>17</td>
<td>89.3</td>
<td>74.8</td>
<td>77.5</td>
</tr>
<tr>
<td>KL</td>
<td>52.77</td>
<td>5.90</td>
<td>59.04</td>
<td>0.77</td>
</tr>
<tr>
<td>(c) Turkish Music: Düyeğ (8/8)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>Fmeas</td>
<td>CMLt</td>
<td>AMLt</td>
<td>infGain</td>
</tr>
<tr>
<td>1</td>
<td>57.2</td>
<td>33.5</td>
<td>42.2</td>
<td>0.68</td>
</tr>
<tr>
<td>2</td>
<td>85.2</td>
<td>70.1</td>
<td>82.7</td>
<td>1.51</td>
</tr>
<tr>
<td>3</td>
<td>83.3</td>
<td>63.3</td>
<td>81.9</td>
<td>1.45</td>
</tr>
<tr>
<td>Ex-2</td>
<td>17</td>
<td>86.6</td>
<td>75.8</td>
<td>87.2</td>
</tr>
<tr>
<td>KL</td>
<td>70.25</td>
<td>49.52</td>
<td>71.79</td>
<td>1.53</td>
</tr>
<tr>
<td>(d) Cretan leaping dances (2/4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>Fmeas</td>
<td>CMLt</td>
<td>AMLt</td>
<td>infGain</td>
</tr>
<tr>
<td>1</td>
<td>68.1</td>
<td>60.7</td>
<td>60.8</td>
<td>1.33</td>
</tr>
<tr>
<td>2</td>
<td>93.0</td>
<td>91.3</td>
<td>91.3</td>
<td>2.25</td>
</tr>
<tr>
<td>3</td>
<td>92.9</td>
<td>91.0</td>
<td>91.0</td>
<td>2.25</td>
</tr>
<tr>
<td>Ex-2</td>
<td>17</td>
<td>88.8</td>
<td>74.3</td>
<td>92.5</td>
</tr>
<tr>
<td>KL</td>
<td>35.87</td>
<td>34.42</td>
<td>72.07</td>
<td>1.57</td>
</tr>
<tr>
<td>(e) Carnatic music: Ādi (8/8)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>Fmeas</td>
<td>CMLt</td>
<td>AMLt</td>
<td>infGain</td>
</tr>
<tr>
<td>1</td>
<td>49.6</td>
<td>38.9</td>
<td>47.0</td>
<td>0.93</td>
</tr>
<tr>
<td>2</td>
<td>56.7</td>
<td>44.0</td>
<td>59.5</td>
<td>1.21</td>
</tr>
<tr>
<td>3</td>
<td>61.1</td>
<td>49.5</td>
<td>65.9</td>
<td>1.40</td>
</tr>
<tr>
<td>Ex-2</td>
<td>17</td>
<td>62.4</td>
<td>40.6</td>
<td>76.7</td>
</tr>
<tr>
<td>KL</td>
<td>59.42</td>
<td>45.90</td>
<td>64.91</td>
<td>1.53</td>
</tr>
<tr>
<td>(f) Carnatic music: Rūpaka (3/4)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>Fmeas</td>
<td>CMLt</td>
<td>AMLt</td>
<td>infGain</td>
</tr>
<tr>
<td>1</td>
<td>68.2</td>
<td>65.8</td>
<td>71.4</td>
<td>2.04</td>
</tr>
<tr>
<td>2</td>
<td>82.8</td>
<td>82.5</td>
<td>90.2</td>
<td>2.77</td>
</tr>
<tr>
<td>3</td>
<td>83.0</td>
<td>82.9</td>
<td>89.5</td>
<td>2.73</td>
</tr>
<tr>
<td>Ex-2</td>
<td>17</td>
<td>77.2</td>
<td>60.6</td>
<td>88.9</td>
</tr>
<tr>
<td>KL</td>
<td>53.42</td>
<td>29.17</td>
<td>60.37</td>
<td>1.30</td>
</tr>
<tr>
<td>(g) Carnatic music: Mishra chāpū (7/8)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>Fmeas</td>
<td>CMLt</td>
<td>AMLt</td>
<td>infGain</td>
</tr>
<tr>
<td>1</td>
<td>84.1</td>
<td>79.0</td>
<td>79.0</td>
<td>1.54</td>
</tr>
<tr>
<td>2</td>
<td>93.7</td>
<td>92.2</td>
<td>92.2</td>
<td>2.00</td>
</tr>
<tr>
<td>3</td>
<td>93.4</td>
<td>91.6</td>
<td>91.6</td>
<td>1.99</td>
</tr>
<tr>
<td>Ex-2</td>
<td>17</td>
<td>90.0</td>
<td>81.6</td>
<td>86.3</td>
</tr>
<tr>
<td>KL</td>
<td>74.61</td>
<td>44.99</td>
<td>68.71</td>
<td>1.25</td>
</tr>
<tr>
<td>(h) Carnatic music: Khanda chāpū (5/8)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>Fmeas</td>
<td>CMLt</td>
<td>AMLt</td>
<td>infGain</td>
</tr>
<tr>
<td>1</td>
<td>58.9</td>
<td>38.0</td>
<td>41.5</td>
<td>0.70</td>
</tr>
<tr>
<td>2</td>
<td>94.3</td>
<td>88.9</td>
<td>94.9</td>
<td>2.00</td>
</tr>
<tr>
<td>3</td>
<td>93.7</td>
<td>88.1</td>
<td>94.3</td>
<td>1.95</td>
</tr>
<tr>
<td>Ex-2</td>
<td>17</td>
<td>90.3</td>
<td>76.0</td>
<td>93.2</td>
</tr>
<tr>
<td>KL</td>
<td>76.16</td>
<td>57.76</td>
<td>66.34</td>
<td>1.18</td>
</tr>
</tbody>
</table>

6. CONCLUSIONS

In this paper we adapted the observation model of a Bayesian approach for the inference of meter in music of cultures in Greece, India, and Turkey. It combines the task of determining the type of meter with the alignment of the downbeats and beats to the audio signal. The model is capable of performing the meter recognition with an accuracy that improves over the state of the art, and is at the same time able to achieve for the first time high beat and downbeat tracking accuracies in additive meters like the Turkish Aksak and Carnatic mishra chāpū.

Our results show that increasing the diversity of a corpus means increasing the number of the patterns, i.e. a larger
Table 2: Confusion matrix of the style classification of the large HMM (Ex-2). The rows refer to the true style and the columns to the predicted style. The empty blocks are zeros (omitted for clarity of presentation).

<table>
<thead>
<tr>
<th></th>
<th>Turkish</th>
<th>Greek</th>
<th>Carnatic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Aksak</td>
<td>Düyek</td>
<td>Curcuna</td>
</tr>
<tr>
<td>Aksak</td>
<td>21</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>Düyek</td>
<td></td>
<td>23</td>
<td>2</td>
</tr>
<tr>
<td>Curcuna</td>
<td>1</td>
<td>3</td>
<td>13</td>
</tr>
<tr>
<td>Cretan</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adi</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rūpaka</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M.chāpu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K.chāpu</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Precision</td>
<td>84</td>
<td>61</td>
<td>76</td>
</tr>
</tbody>
</table>

amount of model parameters. In the context of the HMM inference scheme applied in this paper this implies an increasingly large hidden-parameter state-space. However, we believe that this large parameter space can be handled by using more efficient inference schemes such as Monte Carlo methods.

Finally, we believe that the adaptability of a music processing system to new, unseen material is an important design aspect. Our results imply that in order to extend meter inference to new styles, at least some amount of human annotation is needed. If there exist music styles where adaptation can be achieved without human input remains an important point for future discussions.

Acknowledgments

This work is supported by the Austrian Science Fund (FWF) project Z159, by a Marie Curie Intra-European Fellowship (grant number 328379), and by the European Research Council (grant number 267583).

7. REFERENCES


TRANSCRIPTION AND RECOGNITION OF SYLLABLE BASED PERCUSSION PATTERNS: THE CASE OF BEIJING OPERA

Ajay Srinivasamurthy∗ Rafael Caro Repetto† Harshavardhan Sundar‡ Xavier Serra∗
ajs.murthy@upf.edu rafael.caro@upf.edu harsha@ece.iisc.ernet.in xavier.serra@upf.edu

∗Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain
†Speech and Audio Group, Indian Institute of Science, Bangalore, India

ABSTRACT

In many cultures of the world, traditional percussion music uses mnemonic syllables that are representative of the timbres of instruments. These syllables are orally transmitted and often provide a language for percussion in those music cultures. Percussion patterns in these cultures thus have a well defined representation in the form of these syllables, which can be utilized in several computational percussion pattern analysis tasks. We explore a connected word speech recognition based framework that can effectively utilize the syllabic representation for automatic transcription and recognition of audio percussion patterns. In particular, we consider the case of Beijing opera and present a syllable level hidden markov model (HMM) based system for transcription and classification of percussion patterns. The encouraging classification results on a representative dataset of Beijing opera percussion patterns supports our approach and provides further insights on the utility of these syllables for computational description of percussion patterns.

1. INTRODUCTION

One common feature in traditional musics is the development of sets of predefined, identifiable melodic and rhythmic patterns. These patterns form a repository of structural elements for the composition or performance of the traditional repertoire. Certain entities of traditional music theory, like melodic modes, rhythmic cycles or musical forms, are instantiated by means of these patterns. The patterns function as key elements for the coordination of different musical elements, like instrumental and vocal sections, and the relationship with other art forms, like dance, theatrical acting, story-telling, etc. For the transmission of such patterns in these mostly oral traditions, particular systems of oral mnemonics have been developed. These systems often share common features across different cultures, so that general principles can be established. Computational analysis of these patterns is an important aspect in Music Information Research (MIR) for such music cultures. Further, their own traditional systems of transmission can offer a solid basis for their modeling.

1.1 Syllable based Percussion

Many music traditions around the world have developed particular systems of oral mnemonics for transmission of the repertoire and the technique. David Hughes [7] coined the term acoustic- iconic mnemonic systems for these phenomena, and described their use in different genres of traditional Japanese music. As he points out, the core aspect of these systems is that the syllables are chosen for the similarity of their phonetic features with the acoustic properties of the sounds they are representing, establishing an iconic relationship with them. Therefore, these systems are essentially different from those of solmization [6], like for instance the syllables of solfège, of the Indian svaras or the Chinese gongche notation, which are nonsensical in relation to the acoustic phenomena they represent. In this paper, we focus on the oral syllabic systems of mnemonics developed for percussion traditions.

The use of the aforementioned systems for the transmission of percussion is wide extended among traditional musics. David Hughes mentions in his paper, the shōga used for the set of drums of Noh theatre. In Korea, the young genre of samul nori, a percussion quartet of drums and gongs, draws on traditional syllabic mnemonics for the transmission of the repertoire. In the Indian subcontinent, both Hindustani and Carnatic music cultures have developed such oral syllabic systems of mnemonics for the percussion instruments, respectively the bols in the Hindustani tradition, employed mainly by tabla players, and the solkattu in the Carnatic tradition, where the main percussion instrument is the mridangam. The degree of sophistication that these systems have reached in India is such that the rhythmic recitation of the syllables, which requires high skills, are commonly inserted in concerts for musical appreciation. In Carnatic music, this practice has even been consolidated into a specific music form, called konnakol. Furthermore, these systems are also known to be used in Turkish traditional music and Javanese music. In this paper, we explore the use of oral syllabic system developed in the Beijing opera tradition for the computational analysis of its percussion patterns.

The benefits of using oral syllabic systems from an MIR perspective are both the cultural specificity of the approach and the accuracy of the representation of timbre, articulation and dynamics. The characterization of these percussion traditions need to consider elements that are essential to them such as the richness of their palettes of timbres, subtleties of articulation, and the different degrees and transitions of dynamics, all of which is accurately transmitted.

© Ajay Srinivasamurthy, Rafael Caro Repetto, Harshavardhan Sundar, Xavier Serra.
Licensed under a Creative Commons Attribution 4.0 International License (CC BY 4.0).
by the oral syllables.

We explore the use of oral syllables as a means of representation in the MIR tasks of percussion pattern transcription and classification in syllabic percussion systems, considering Beijing opera percussion patterns as a study case. The well defined oral syllabic system and the limited set of percussion patterns make it an ideal choice for a first exploration. Since these syllables have a clear analogy to speech and language, we present a speech recognition based approach to transcribe a percussion pattern into a sequence of syllables. We then use this transcription to classify the sequence into one of the predefined set of patterns that occur in Beijing Opera. We first provide an introduction to percussion patterns in Beijing Opera.

1.2 Percussion patterns in Beijing opera

Beijing opera (Jingju, 京剧), also called Peking Opera, is one of the most representative genres of Chinese traditional performing arts, integrating theatrical acting with singing and instrumental accompaniment. It is an active art form and exists in the current social and cultural contexts, with a large audience and significant musicological literature. One of the main characteristics of Beijing opera aesthetics is the remarkable rhythmics that governs the acting overall. From the stylized recitatives to the performers’ movements on stage and the sequence of events, every element presented is integrated into an overall rhythmic flow. The main element that keeps this rhythmics is the percussion ensemble, and the main means to fulfill this task is a set of predefined and labeled percussion patterns. Along with the main purpose of keeping the overall rhythmics of the performance, these patterns have different functions. They signal important structural points in the play. A performance starts and ends with percussion patterns, they generally introduce and conclude arias, and mark transition points within them. They accompany the actors’ movements on stage and set the mood of the play, the scene, the aria or a section of the aria. Therefore, the detection and characterization of percussion patterns is a fundamental task for the description of the music dimension in Beijing opera.

The percussion patterns in Beijing Opera music can be defined as sequences of strokes played by different combinations of the percussion instruments, and the resulting variety of timbres are transmitted using oral syllables as mnemonics. The percussion ensemble is formed mainly by five instruments played by four musicians. The ban clappers and the danpipeline drum are played by one single performer, and are therefore known by a conjoint name, bangu. The other three instruments are the xiao huo (small gong), the dahu (big gong) and the naobo (cymbals). Bangu has a high pitched drum-like sound while the rest of three instruments are metallophones with distinct timbres\(^1\). Each of the different sounds that these instruments can produce individually, either through different playing techniques or through different dynamics, as well as the sounds that are produced by a combination of different instruments have an associated syllable that represent them [9]. The syllables and their associated instrument combinations are shown in Table 1. Thus, each percussion pattern is a sequence of syllables in their pre-established order, along with their specific rhythmic structure and dynamic features. A particular feature of the oral syllabic system for Beijing opera percussion that makes it especially interesting is that the syllables that form a pattern refer to the ensemble as a whole, and not to particular instruments. Each particular pattern thus has a single unique syllabic representation shared by all the performers.

In practice, there is a library of limited set of named patterns (called luogu jing, 钟鼓经) that are played in a performance, with each of these having a specific role in the arias. These named patterns in the library can be referred to as “pattern classes” for the purpose of classification, and classifying an instance of a pattern occurring in the audio recording of an aria into one of these pattern classes is thus a primary task. Although a definite agreed number for the total number of these patterns is lacking, some estimations,

\(^1\) A few annotated audio examples of these instruments can be found at [http://compmusic.upf.edu/examples-percussion-bo]
like in [9], suggest the existence of around ninety of them.

Figure 1 shows the score for an example pattern shanchui. It also shows how a possible transcription in staff notation (adapted from [9]) can be simplified in a single line by the oral syllabic system. Hence, the use of these oral syllabic sequences simplify and unify the representation of these patterns played by an ensemble, making them optimal for the transcription and automatic classification of the patterns. Further, Figure 2 shows an audio example, along with time aligned markers to indicate the syllable onsets. The spectrogram shows the timbral characteristics of the percussion instruments xiaoluo (increasing pitch) and dahu (decreasing pitch). Some variation to the notated score can also be seen, such as expressive timing and additional insertion of syllables.

Though the patterns are limited in number and predefined, there are several challenges to the problem of percussion pattern transcription and classification. Being an oral tradition, the syllables used for the representation of the patterns lack full consistency and general agreement. The result being that one particular timbre might be represented by more than one syllable. Furthermore, the syllabic representation conveys information for the conjoint timbre of the ensemble, so only the main structural sounds are represented. In an actual performance, a particular syllable might be performed by different combinations of instruments - e.g. in Figure 1, the first occurrence of the syllable tāi is played just by the xiaoluo, but in the rest of the pattern is played by xiaoluo and the bangu together. In fact, generally speaking, the strokes of the bangu are seldom conveyed in the syllabic sequence (as can be seen in the third measure in Figure 1 for the second sixteenth-note of the bangu), except for the introductions and other structural points played by the drum alone. As indicated in Table 1, cāng is mostly a combination of all the three metallophones, but in some cases, cāng can be played with just the dahu, or just the dahu+naobo combination. A detailed description and scores for various patterns is available at http://compmusic.ufpe.br/bo-perc-patterns

Since one of the main functions of the patterns is to accompany the movements of actors on stage, the overall length and the relative duration of each stroke can vary notably, which makes it difficult to set a stable pulse or a definite meter. The time signature and the measure bars used in Figure 1, as suggested in [9], are only indicative and fail to convey the rhythmic flexibility of the pattern. Furthermore, many patterns (such as shanchui) accompany scenic movements of undefined duration. In these cases, certain syllable sub-sequences in the pattern are repeated indefinitely, e.g. the audio example in Figure 2 has two additional repetitions of the sub-sequence cāng-tāi-qíě-tāi in the pattern. This causes the same pattern in different performances to have variable lengths, and these repetitions need to be explicitly handled. Finally, although the patterns are usually played in isolation, in many cases the string instruments or even the vocals can start playing before the patterns end, presenting challenges in identification and classification.

1.3 Previous work

There is significant MIR literature on percussion transcription [4]. Nakano et al. [10] explored drum pattern retrieval using vocal percussion, using an HMM based approach. They used onomatopoeia as the internal representation for drum patterns, with a focus on retrieving known fixed sequences from a library of drum patterns with snare and bass drums. Kapur et al. explored query by BeatBoxing [8], aiming to map the BeatBoxing sounds into the corresponding drum sounds. A distinction to be noted here is that in vocal percussion systems such as BeatBoxing, the vocalizations form the music itself, and not a means for transmission as in the case of oral syllables. More recently, Paulus et al. proposed the use of connected HMMs for drum transcription in polyphonic music [12]. This approach is different from what we present in the sense that it aimed to transcribe individual drums (bass, snare, hi-hat) and not overall timbres due to combinations, and no reference to syllabic percussion was made. However all these approaches have indirectly and implicitly used some form of syllabic representations for drum patterns.

Chordia [2] explored the use of tabla böls in transcription of solo tabla sequences. Recently, tabla syllables were used for a predictive model for tabla stroke sequences [3]. Anantapadmanabhan et al. [1] used the syllables of the mir-dangam in a stroke transcription task. Unlike these works, we address a syllabic system that conveys information for a whole ensemble instead of individual instruments.

Despite the rich musical heritage and the size of audience, little work has been done for computational analysis of Beijing opera from an MIR perspective. It has been studied as a target in a some genre classification works [17] and the acoustical properties of Beijing opera singing has been studied [14]. Apart from a recent study [15] that explored the use of Non-negative matrix factorization for onset detection and onset classification into the different percussion instrument classes, no significant work has studied Beijing opera percussion from a computational perspective.

Similar to Nakano et al. [10], we explore a speech recognition based framework in this study. This approach is different to ours in the sense that these onomatopoeic representations were created by the authors, while we are relying on already existing oral traditions. Speech recognition is a well explored research area with many state of the art algorithms and systems [5]. Hence we can apply several available tools and knowledge for computational analysis of syllabic percussion patterns. To the best of our knowl-

<table>
<thead>
<tr>
<th>Syllables</th>
<th>Instruments</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>bā (八), bēn (本), dā (答), dà (大), dōng (冬), duō (多)</td>
<td>bangu</td>
<td>DA</td>
</tr>
<tr>
<td>lǎi (来), tái (台), líng (岭)</td>
<td>xiaoluo</td>
<td>TAI</td>
</tr>
<tr>
<td>qǐ (起), pù (扑)</td>
<td>naobo</td>
<td>Q1</td>
</tr>
<tr>
<td>qiě (撇)</td>
<td>naobo+xiaoluo</td>
<td>Q1E</td>
</tr>
<tr>
<td>cāng (仓), kuāng (匡), kǒng (空)</td>
<td>dahu+naobo</td>
<td>CANG</td>
</tr>
</tbody>
</table>

Table 1: Syllables used in Beijing opera percussion and their groupings used in this paper. Column 2 shows the instrument combination used to produce the syllable, instrument shown between <> is optional. Column 3 shows the symbol we use for the syllable group in this paper.
edge, this is the first work to explore transcription and classification of syllable based percussion patterns, as applied to Beijing opera.

2. PROBLEM FORMULATION

In Beijing opera, several syllables can be mapped to a single timbre. This many-syllable to one-timbre mapping is useful to reduce the syllable space for computational analysis of percussion patterns. We first mapped each syllable to one or several of the instrument categories considered for analysis, as explained in [15], without considering differences in playing technique or dynamics. Based on inputs from expert musicologists, we then grouped the syllables with similar timbres into five syllable groups - DA, TAI, QI, QIE, and CANG, as shown in Table 1. Every individual stroke of the bangzi, both drum and clapper, have been grouped as DA. In the rest of the syllable groups, the bangzi can be played simultaneously or not. The single strokes of the xiaolu and the naobo are called TAI and QI respectively, and the combined stroke of these two instruments together is the syllable QIE. Finally, any stroke of the daluo or any combination that includes daluo has been noted as CANG. This mapping to a reduced set of syllable groups is only for the purpose of computational analysis. For the remainder of the paper, we limit ourselves to the reduced set of syllable groups and use them to represent the patterns. For convenience, when it is clear from the context, we call the syllable groups as just syllables, and denote them by the common symbol in column 3 of Table 1. Hence, in the current task, there are five syllable groups. Further, in Beijing opera, the recognition of the pattern as a whole is more important that an accurate syllabic transcription of the pattern. Due to the limited set of pattern classes and owing to all the variations possible in a pattern, we are primarily interested in classifying an audio pattern into one of the possible pattern classes. Syllabic transcription is only considered as an intermediate step towards pattern classification.

We now present a formulation for transcription and recognition of syllable based audio percussion patterns. There is a significant analogy of this task to connected word speech recognition using word models. Syllables are analogous to words and a percussion pattern to a sentence - a sequence of words. There are language rules to form a sentence using a vocabulary, just as each percussion pattern is formed with a defined sequence of syllables from a vocabulary. However unlike in the case of speech recognition where infinitely many sentences are possible, in our case we have a small number of percussion patterns to be recognized.

Consider a set of N pattern classes \( \mathcal{P} = \{P_1, P_2, \ldots, P_N\} \), each of which is a sequence of syllables from the set of M syllables \( \mathcal{S} = \{S_1, S_2, \ldots, S_M\} \). So, \( P_k = [s_1, s_2, \ldots, s_{L_k}] \) where \( s_i \in \mathcal{S} \) and \( L_k \) is the length of \( P_k \). Given a test audio pattern \( x[n] \), the transcription task aims to obtain a syllable sequence \( P^* = [s_1, s_2, \ldots, s_{L*}] \) and the classification task aims to assign \( P^* \) into one of the patterns in the set \( \mathcal{P} \).

3. DATASET

Since there was no available dataset of Beijing opera percussion patterns, we built a representative dataset of patterns from the audio recordings of arias in the CompMusic

<table>
<thead>
<tr>
<th>Pattern Class</th>
<th>ID</th>
<th>Instances</th>
<th>LEN (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>daobantu【呆板头】</td>
<td>1</td>
<td>66</td>
<td>8.70 (1.73)</td>
</tr>
<tr>
<td>man changchui【慢长锤】</td>
<td>2</td>
<td>33</td>
<td>13.99 (4.47)</td>
</tr>
<tr>
<td>duotuo【多头】</td>
<td>3</td>
<td>19</td>
<td>7.18 (1.49)</td>
</tr>
<tr>
<td>xiaolu duotuo【小锣多头】</td>
<td>4</td>
<td>11</td>
<td>8.16 (2.15)</td>
</tr>
<tr>
<td>shanchui【闪锤】</td>
<td>5</td>
<td>8</td>
<td>10.31 (3.26)</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td><strong>133</strong></td>
<td><strong>9.85 (3.69)</strong></td>
</tr>
</tbody>
</table>

Table 2: The Beijing Opera Percussion Pattern (BOPP) dataset. The last column is the mean pattern length and standard deviation in seconds.

Beijing opera research corpus [13], which is a curated collection of arias from commercially available releases spanning many different artists and recording conditions. For this study, we chose only the patterns at the beginning of the aria, which are characteristic and important. From all the pattern classes existing in the corpus, we chose five \( (N = 5) \) most frequently used ones. These five patterns are also the most widely used and hence hold a high degree of representativeness. The patterns were extracted from the audio recording and assigned to a pattern class by a musicologist. The dataset is described in Table 2 and comprises about 22 minutes of audio with over 2200 syllables in total. The audio samples are stereo recordings sampled at 44.1kHz. The syllabic transcription of each audio pattern is obtained directly from the score of the pattern class it belonged to. Hence the ground truth transcriptions available in the dataset are not time aligned. Since it is a significant effort to obtain time aligned transcriptions, we aim to develop algorithms which do not require the use of time-aligned transcriptions for training. This also ensures that the approaches scale when we add more pattern classes to the dataset. In case of patterns where a sub-sequence of the pattern can be repeated (e.g. \textit{man changchui} and \textit{shanchui}), the additional syllables that occur due to repetitions were manually added by listening to the pattern. Though most of the dataset consists of isolated percussion patterns, there are many audio examples that contain a melodic background apart from the percussion pattern. The dataset is available for research purposes through a central online repository\(^3\).

4. THE APPROACH

The syllables are non-stationary signals and to model their timbral dynamics, we build an HMM for each syllable (analogous to a word-HMM). Using these syllable HMMs and a language model, an input audio pattern is transcribed into a sequence of syllables using Viterbi decoding, and then classified to a pattern class in the library using a measure of distance.

A block diagram of the approach is shown in Figure 3. We first build syllable level HMMs \( \{A_m\} \), \( 1 \leq m \leq M \) (= 5), for each syllable \( S_m \) using features extracted from the training audio patterns. We use the MFCC features to model the timbre of the syllables. To capture the temporal dynamics of syllables, we add the velocity and the acceleration coefficients of the MFCC. The stereo audio is converted to mono, since there is no additional information in stereo channels. The 13 dimensional (including the 0th

\(^3\) More details at http://compmusic.ufu.edu/bopp-dataset
coefficient) MFCC features are computed from audio patterns with a frame size of 23.2 ms and a shift of 5.8 ms. We also explore the use of energy (as measured by the 0th MFCC coefficient) in classification performance. Hence we have two sets of features, MFCC 0, D A, the 39 dimensional feature including the 0th, delta and double-delta coefficients, and MFCC D A, the 36 dimensional vector without the 0th coefficient.

We model each syllable using a 5-state left-to-right HMM including an entry and an exit non-emitting states. The emission densities for each state is modeled with a four component Gaussian Mixture Model (GMM) to capture the timbral variability in syllables. We experimented with eight and sixteen component GMMs, but with little performance improvement. Since we do not have time aligned transcriptions, an isolated HMM training for each syllable is not possible. Hence we use an embedded model Baum-Welch re-estimation to train the HMMs using just the syllable sequence corresponding to each feature sequence. The HMMs are initialized with a flat start using all of the training data. All the experiments were done using the HMM Toolkit (HTK) [16].

For testing, since we only need a rough syllabic transcription independent of the pattern class, we treat the test pattern as a first order time-homogenous discrete Markov chain, which can consist of any finite length sequence of syllables, with uniform unigram and bi-gram (transition) probabilities, i.e. \( p(s_i = S_i) = 1/M \) and \( p(s_{i+1} = S_j/s_k = S_i) = 1/M, 1 \leq i, j \leq M \) and with \( k \) being the sequence index. This also forms the language model for forming the percussion patterns using syllables. Given the feature sequence extracted from test audio pattern, we use the HMMs \( \{ \lambda_m \} \) to do a Viterbi (forced) alignment, which aims to provide the best sequence of syllables \( P^* \), given a syllable network constructed from the language model.

Given the decoded syllable sequence \( P^* \), we compute the string edit distance [11] between \( P^* \) and elements in the set \( \mathcal{P} \). The use of edit distance is motivated by two factors. First, due to errors in Viterbi alignment, \( P^* \) can have insertions (I), deletions (D), substitutions (B), and transposition (T) of syllables compared to the ground truth. Secondly, to handle the allowed variations in patterns, an edit distance is preferred over an exact match to the sequences in \( \mathcal{P} \). We explore the use of two different string edit distance measures, Levenshtein distance \( (d_1) \) that considers I, D, B errors and the Damerau–Levenshtein distance \( (d_2) \) that considers I, D, B, T errors.

As discussed earlier, there can be repetitions of a subsequence in some patterns. Though the number of repetitions is indefinite, we observed in the dataset that there are at most two repetitions in a majority of pattern instances. Hence for the pattern classes that allow repetition of a subsequence, we compute the edit distance for the cases of zero, one and two repetitions and then take the minimum distance obtained among the three cases. This way, we can handle repeated parts in a pattern. Finally, the \( P^* \) is assigned to the pattern class \( \mathcal{P}_o \subset \mathcal{P} \) for which the edit distance \( d \) (either \( d_1 \) or \( d_2 \)) is minimum, as in Eqn 1.

\[
P_o = \text{argmin}_{1 \leq k \leq N} d(P^*, \mathcal{P}_k)
\]

\[
(1)
\]

5. RESULTS AND DISCUSSION

We present the syllable transcription and pattern classification results on the dataset described in Section 3. The results shown in Table 3 are the mean values in a leave-one-out cross validation. We report the syllable transcription performance using the measures of Correctness (C) and Accuracy (A). If \( L \) is the length of the ground truth sequence, \( C = (L - D - B)/L \) and \( A = (L - D - B - I)/L \). The Correctness measure penalizes deletions and substitutions, while Accuracy measure additionally penalizes insertions too. The pattern classification performance is shown for both edit distance measures \( d_1 \) and \( d_2 \) in Table 3. All the results are reported for both the features, MFCC 0, D A and MFCC D A. The difference in performance between the two features was found to be statistically significant for both Correctness and Accuracy measures in a Mann-Whitney U test at \( p = 0.05 \), assuming an asymptotic normal distribution.

As discussed earlier, there can be repetitions of a subsequence in some patterns. Though the number of repetitions is indefinite, we observed in the dataset that there are at most two repetitions in a majority of pattern instances. Hence for the pattern classes that allow repetition of a subsequence, we compute the edit distance for the cases of zero, one and two repetitions and then take the minimum distance obtained among the three cases. This way, we can handle repeated parts in a pattern. Finally, the \( P^* \) is assigned to the pattern class \( \mathcal{P}_o \subset \mathcal{P} \) for which the edit distance \( d \) (either \( d_1 \) or \( d_2 \)) is minimum, as in Eqn 1.

\[
P_o = \text{argmin}_{1 \leq k \leq N} d(P^*, \mathcal{P}_k)
\]

\[
(1)
\]

5. RESULTS AND DISCUSSION

We present the syllable transcription and pattern classification results on the dataset described in Section 3. The results shown in Table 3 are the mean values in a leave-one-out cross validation. We report the syllable transcription performance using the measures of Correctness (C) and Accuracy (A). If \( L \) is the length of the ground truth sequence, \( C = (L - D - B)/L \) and \( A = (L - D - B - I)/L \). The Correctness measure penalizes deletions and substitutions, while Accuracy measure additionally penalizes insertions too. The pattern classification performance is shown for both edit distance measures \( d_1 \) and \( d_2 \) in Table 3. All the results are reported for both the features, MFCC 0, D A and MFCC D A. The difference in performance between the two features was found to be statistically significant for both Correctness and Accuracy measures in a Mann-Whitney U test at \( p = 0.05 \), assuming an asymptotic normal distribution.

In general, we see a good pattern classification performance while syllable transcription accuracy is poor. We see that MFCC 0, D A has a better performance with syllable transcription, while both kinds of features provide a comparable performance for pattern classification. Though syllable transcription is not the primary task we focus on, an analysis of its performance provides several insights. The set of percussion instruments in Beijing Opera is fixed, but there can be slight variations across different instruments of the same kind. The training examples are varied and representative, and models built can be presumed to be source independent. Nevertheless, there can be unrepresented syllable timbres in test data leading to a poorer transcription performance. A bigger training dataset can improve the performance in such a case. The energy co-efficient provides significant information about the kind of syllables.

Table 3: Syllable transcription and Pattern classification performance, with Correctness (C) and Accuracy (A) measures for syllable transcription. Pattern classification results are shown for both distance measures \( d_1 \) and \( d_2 \). All values are in percentage.
and hence gives a better syllable transcription performance.

We see that the Correctness is higher than Accuracy showing that the exact sequence of syllables, as indicated in the score was never achieved in a majority of the cases, with several insertion errors. This is due to the combined effect of errors in decoding and allowed variations in patterns. An edit distance based distance measure for classification is quite robust in the present five class problem and provides a good classification performance, despite the low transcription accuracy. Both distance measures provide comparable performance, indicating that the number of transposition errors are low. To see if there are any systematic classification errors, we build a confusion matrix (Table 4) with one of the well performing configurations: MFCC_0 D A with $d_1$ distance. We see that duotou has a low recall, and gets confused with shanzhu (ID=5) often. A close examination of the scores showed that a part of the pattern duotou is contained within shanzhu, which explains source of confusion. Such confusions can be handled with better language models, which need further exploration.

6. CONCLUSIONS AND SUMMARY

We presented a formulation based on connected-word speech recognition for transcription and classification of syllabic percussion patterns. On a representative collection of Beijing opera percussion patterns, the presented approach provides a good classification performance, despite a simplistic language model and inadequate syllabic transcription accuracy. Though the approach is promising, the evaluation using a small dataset necessitates a further assessment of the generalization capabilities of the proposed approach. We intend to explore better language models that use sequence and rhythmic information more effectively, and extend the task to a much larger dataset spanning more pattern classes. We used isolated patterns in this study, but an automatic segmentation of patterns from audio is a good direction for future work. We also plan to extend this formulation for computational description of percussion patterns in other music cultures such as Hindustani and Carnatic music, which have more complex syllabic percussion systems.

Acknowledgments

This work is partly supported by the European Research Council under the European Union’s Seventh Framework Program, as a part of the CompMusic project (ERC grant agreement 267583)

7. REFERENCES


