Quantitative Analysis of a Common Audio Similarity Measure

Jesper Højvang Jensen, *Member, IEEE*, Mads Græsbøll Christensen, *Member, IEEE*, Daniel P. W. Ellis, *Senior Member, IEEE*, and Søren Holdt Jensen, *Senior Member, IEEE*

***Abstract*—For music information retrieval tasks, a nearest neighbor classifier using the Kullback–Leibler divergence between Gaussian mixture models of songs' melfrequency cepstral coefficients is commonly used to match songs by timbre. In this paper, we analyze this distance measure analytically and experimentally by the use of synthesized MIDI files, and we find that it is highly sensitive to different instrument realizations. Despite the lack of theoretical foundation, it handles the multipitch case quite well when all pitches originate from the same instrument, but it has some weaknesses when different instruments play simultaneously. As a proof of concept, we demonstrate that a source separation frontend can improve performance. Furthermore, we have evaluated the robustness to changes in key, sample rate, and bitrate.**

***Index Terms*—Melody, musical instrument classification, timbre recognition.**

I. Introduction

MEL-FREQUENCY cepstral coefficients (MFCCs) are extensively used in music information retrieval algorithms [1]–[12]. Originating in speech processing, the MFCCs were developed to model the spectral envelope while suppressing the fundamental frequency. Together with the temporal envelope, the spectral envelope is one of the most salient components of timbre [13], [14], which is "that attribute of auditory sensation in terms of which a listener can judge that two sounds similarly presented and having the same loudness and pitch are dissimilar" [15], i.e., what makes the same note played with different instruments sound different. Thus, the MFCCs in music information retrieval applications are commonly used to model the timbre. However, even though MFCCs have experimentally been shown to perform well in instrument recognition, artist recognition and genre classification [7], [8], [16], a number of questions remain unanswered. For instance, being developed for speech recognition in a single-speaker environment, it is not obvious how the MFCCs are affected by different instruments playing simultaneously and by chords where the fundamental frequencies have near-integer ratios. Furthermore, as shown in [17], MFCCs are sensitive to the spectral perturbations that result from low bitrate audio compression.

In this paper, we address these issues and more. We analyze the behavior of the MFCCs when either a single instrument or different instruments play several notes simultaneously, thus violating the underlying assumption of a single voice. In relation to the album effect [18], where MFCC-based distance measures in artist recognition rate songs from the same album as much more similar than songs by the same artist from different albums, we investigate how MFCCs are affected by different realizations of the same instrument. Finally, we investigate how MFCCs are affected by transpositions, different sample rates and different bitrates, since this is relevant in practical applications. A transposed version of a song, e.g., a live version that is played in a different key than the studio version, is usually considered similar to the original, and collections of arbitrary music, such as encountered by an internet search engine, will inevitably contain songs with different sample rates and bitrates.

To analyze these topics, we use MIDI synthesis, for reasons of tractability and reproducibility, to fabricate wave signals for our experiments, and we employ the distance measure proposed in [4] that extracts MFCCs and trains a Gaussian mixture model for each song and uses the symmetrized Kullback–Leibler divergence between the models as distance measure. A nearest-neighbor classification algorithm using this approach won the International Conference on Music Information Retrieval (ISMIR) genre classification contest in 2004 [6]. Genre classification is often not considered a goal in itself, but rather an indirect means to verify the actual goal, which is a measure of similarity between songs. In most comparisons on tasks such as genre identification, distributions of MFCC features have performed as well or better than all other features considered—a notable result [7], [8]. Details of the system, such as the precise form or number of MFCCs used, or the particular mechanism used to represent and compare MFCC distributions, appear to have only a secondary influence. Thus, the distance measure studied in this paper, a particular instance of a system for comparing music audio based on MFCC distributions, is both highly representative of most current work in music audio comparison, and is likely close to or equal to the state of the art in most tasks of this kind.

In Section II, we review MFCCs, Gaussian modeling and computation of the symmetrized Kullback–Leibler divergence. In Section III, we describe the experiments before discussing the results in Section IV and giving the conclusion in Section V.

Manuscript received December 15, 2006; revised October 28, 2008. Current version published March 27, 2009. This work was supported in part by the Intelligent Sound project, Danish Technical Research Council, under Grant 26-04-0092, and in part the Parametric Audio Processing project, Danish Research Council for Technology and Production Sciences, under Grant 274-06-0521. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Michael M. Goodwin.

The authors are with the Section for Multimedia Information and Signal Processing, Department of Electronic Systems, Aalborg University, DK-9220 Aalborg, Denmark, and also with LabROSA, Department of Electrical Engineering, Columbia University, New York, NY 10027 USA (e-mail: jhj@es.aau.dk; mgc@es.aau.dk; shj@es.aau.dk; dpwe@ee.columbia.edu).

Digital Object Identifier 10.1109/TASL.2008.2012314

II. Mel-Frequency Cepstral Coefficients-Based Timbral Distance Measure

In the following, we describe the motivation behind the MFCCs, mention some variations of the basic concept, discuss their applicability to music, and discuss the use of the Kullback–Leibler divergence between multivariate Gaussian mixture models as a distance measure between songs.

A. Mel-Frequency Cepstral Coefficients

MFCCs were introduced as a compact, perceptually based representation of speech frames [19]. They are computed as follows.

1) Estimate the amplitude or power spectrum of 20–30 ms of speech.
2) Group neighboring frequency bins into overlapping triangular bands with equal bandwidth according to the melscale.
3) Sum the contents of each band.
4) Compute the logarithm of each sum.
5) Compute the discrete cosine transform of the bands.
6) Discard high-order coefficients from the cosine transform.

Most of these steps are perceptually motivated, but some steps also have a signal processing interpretation. The signal is divided into 20–30 ms blocks because speech is approximately stationary within this time scale. Grouping into bands and summing mimics the difficulty in resolving two tones closely spaced in frequency, and the logarithm approximates the human perception of loudness. The discrete cosine transform, however, does not directly mimic a phenomenon in the human auditory system, but is instead an approximation to the Karhunen–Loève transform in order to obtain a compact representation with minimal correlation between different coefficients.

As the name of the MFCCs imply, the last three steps can also be interpreted as homomorphic deconvolution in the cepstral domain to obtain the spectral envelope (see, e.g., [20]). Briefly, this approach employs the common model of voice as glottal excitation filtered by a slowly-changing vocal tract, and attempts to separate these two components. The linear filtering becomes multiplication in the Fourier domain, which then turns into addition after the logarithm. The final Fourier transform, accomplished by the discrete cosine transform, retains linearity but further allows separation between the vocal tract spectrum, which is assumed smooth in frequency and thus ends up being represented by the low-index cepstral coefficients, and the harmonic spectrum of the excitation, which varies rapidly with frequency and falls predominantly into higher cepstral bins. These are discarded, leaving a compact feature representation that describes the vocal tract characteristics with little dependence on the fine structure of the excitation (such as its period). For a detailed description of homomorphic signal processing see [21], and for a discussion of the statistical properties of the cepstrum see [22]. For a discussion of using the MFCCs as a model for perceptual timbre space for static sounds, see [23].

B. Variations

When computing MFCCs from a signal, there are a number of free parameters. For instance, both the periodogram, linear prediction analysis, the Capon spectral estimator, and warped versions of the latter two have been used to estimate the spectrum, and the number of meldistributed bands and their lower and upper cutoff frequency may also differ. For speech recognition, comparisons of different such parameters can be found in [24] and [25]. For music, less exhaustive comparisons can be found in [5] and [12]. It is also an open question how many coefficients should be kept after the discrete cosine transform. According to [17], the first five to fifteen are commonly used. In [26], as many as 20 coefficients, excluding the 0th coefficient, are used with success. In the following, we will use the term "MFCC order" to refer to the number of coefficients that are kept. Another open question is whether to include the 0th coefficient. Being the DC value, the 0th coefficient is the average of the logarithm of the summed contents of the triangular bands, and it can thus be interpreted as the loudness averaged over the triangular bands. On the one hand, volume may be useful for modeling a song, while on the other hand it is subject to arbitrary shifts (i.e., varying the overall scale of the waveform) and does not contain information about the spectral shape as such.

C. Applicability to Music

In [27], it is verified that the melscale is preferable to a linear scale in music modeling, and that the discrete cosine transform does approximate the Karhunen–Loève transform. However, a number of uncertainties remain. In particular, the assumed signal model consisting of one excitation signal and a filter only applies to speech. In polyphonic music there may, unlike in speech, be several excitation signals with different fundamental frequencies and different filters. Not only may this create ambiguity problems when estimating which instruments the music was played by, since it is not possible to uniquely determine how each source signal contributed to the spectral envelopes, but the way the sources combine is also very nonlinear due to the logarithm in step 4. Furthermore, it was shown in [17] that MFCCs are sensitive to the spectral perturbations that are introduced when audio is compressed at low bitrates, mostly due to distortion at higher frequencies. However, it was not shown whether this actually affects instrument or genre classification performance. A very similar issue is the sampling frequency of the music that the MFCCs are computed from. In a real-world music collection, all music may not have the same sampling frequency. A downsampled signal would have very low energy in the highest melbands, leaving the logarithm in step 4 in the MFCC computation either undefined or at least approaching minus infinity. In practical applications, some minimal (floor) value is imposed on channels containing little or no energy. When the MFCC analysis is applied over a bandwidth greater than that remaining in the compressed waveform, this amounts to imposing a rectangular window on the spectrum, or, equivalently, convolving the MFCCs with a sinc function. We will return to these issues in Section III.

D. Modelling MFCCs by Gaussian Mixture Models

Storing the raw MFCCs would take up a considerable amount of space, so the MFCCs from each song are used to train a parametric, statistical model, namely a multivariate Gaussian mixture model. As distance measure between the Gaussian mixture models, we use the symmetrized Kullback–Leibler divergence. This approach was presented in [4], but both [2] and [28] have previously experimented with very similar approaches. The probability density function for a random variable $\boldsymbol{x}$ modeled by a Gaussian mixture model with K mixtures is given by

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} c_k \frac{1}{\sqrt{|2\pi\boldsymbol{\Sigma}_k|}} \exp\Big(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_k)^T{\boldsymbol{\Sigma}_k}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_k)\Big) \tag{1}$$

where K is the number of mixtures and $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$, and c_k are the mean, covariance matrix, and weight of the k'th Gaussian, respectively. For $K=1$, the maximum-likelihood estimates of the mean and covariance matrix are given by [29]

$$\boldsymbol{\mu}_{\text{ML}} = \frac{1}{M}\sum_{n=1}^{M} \boldsymbol{x}_n \tag{2}$$

and

$$\boldsymbol{\Sigma}_{\text{ML}} = \frac{1}{M}\sum_{n=1}^{M} (\boldsymbol{x}_n - \boldsymbol{\mu}_{\text{ML}})(\boldsymbol{x}_n - \boldsymbol{\mu}_{\text{ML}})^T. \tag{3}$$

For $K>1$, the k-means algorithm followed by the expectation-maximization algorithm (see [30] and [31]) is typically used to train the weights c_k, means $\boldsymbol{\mu}_k$, and covariance matrices $\boldsymbol{\Sigma}_k$. As mentioned, we use the symmetrized Kullback–Leibler divergence between the Gaussian mixtures as distance measure between two songs. The Kullback–Leibler divergence is an asymmetric information theoretic measure of the distance between two probability density functions. The Kullback–Leibler divergence between $p_1(\boldsymbol{x})$ and $p_2(\boldsymbol{x})$, $d_{\text{KL}}(p_1,p_2)$, is given by

$$d_{\text{KL}}(p_1,p_2) = \int p_1(\boldsymbol{x}) \log \frac{p_1(\boldsymbol{x})}{p_2(\boldsymbol{x})} d\boldsymbol{x}. \tag{4}$$

For discrete random variables, $d_{\text{KL}}(p_1,p_2)$ is the penalty of designing a code that describes data with distribution $p_2(\boldsymbol{x})$ with shortest possible length but instead use it to encode data with distribution $p_1(\boldsymbol{x})$ [32]. If $p_1(\boldsymbol{x})$ and $p_2(\boldsymbol{x})$ are close, the penalty will be small and vice versa. For two multivariate Gaussian distributions, $p_1(\boldsymbol{x})$ and $p_2(\boldsymbol{x})$, the Kullback–Leibler divergence is given in closed form by

$$d_{\text{KL}}(p_1,p_2) = \frac{1}{2}\Bigg[\log\bigg(\frac{|\boldsymbol{\Sigma}_1|}{|\boldsymbol{\Sigma}_2|}\bigg) + \text{tr}({\boldsymbol{\Sigma}_1}^{-1}\boldsymbol{\Sigma}_2) + (\boldsymbol{\mu}_1-\boldsymbol{\mu}_2)^T{\boldsymbol{\Sigma}_1}^{-1}(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2) - D\Bigg] \tag{5}$$

where D is the dimensionality of $\boldsymbol{x}$. For Gaussian mixtures, a closed form expression for $d_{\text{KL}}(p_1,p_2)$ does not exist, and it must be estimated, e.g., by stochastic integration or closed form approximations [10], [33], [34]. To obtain a symmetric distance measure, we use $d_{\text{sKL}}(p_1,p_2) = d_{\text{KL}}(p_1,p_2) + d_{\text{KL}}(p_2,p_1)$.

Collecting the two Kullback–Leibler divergences under a single integral, we can directly see how different values of $p_1(\boldsymbol{x})$ and $p_2(\boldsymbol{x})$ affect the resulting distance

$$d_{\text{sKL}}(p_1,p_2) = \int d'_{\text{sKL}}(p_1(\boldsymbol{x}),p_2(\boldsymbol{x}))d\boldsymbol{x} \tag{6}$$

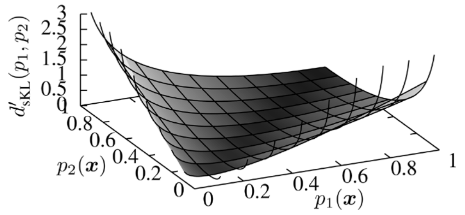

Fig. 1. Symmetrized Kullback–Leibler divergence. When either $p_1(\boldsymbol{x})$ or $p_1(\boldsymbol{x})$ approaches zero, $d'_{\text{sKL}}(p_1,p_2)$ approach infinity.

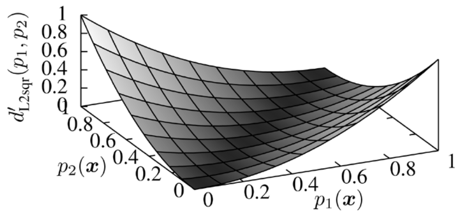

Fig. 2. Squared L2 distance. Note that unlike $d'_{\text{sKL}}(p_1,p_2)$ in Fig. 1, $d'_{\text{L2sqr}}(p_1,p_2)$ behaves nicely when $p_1(\boldsymbol{x})$ or $p_2(\boldsymbol{x})$ approach zero.

where

$$d'_{\text{sKL}}(p_1(\boldsymbol{x}),p_2(\boldsymbol{x})) = (p_1(\boldsymbol{x})-p_2(\boldsymbol{x}))\log\frac{p_1(\boldsymbol{x})}{p_2(\boldsymbol{x})}. \tag{7}$$

In Fig. 1, $d'_{\text{sKL}}(p_1(\boldsymbol{x}),p_2(\boldsymbol{x}))$ is shown as a function of $p_1(\boldsymbol{x})$ and $p_2(\boldsymbol{x})$. From the figure and (7), it is seen that for $d_{\text{sKL}}(p_1,p_2)$ to be large, there has to be $\boldsymbol{x}$ where both the difference and the ratio between $p_1(\boldsymbol{x})$ and $p_2(\boldsymbol{x})$ is large. High values are obtained when only one of $p_1(\boldsymbol{x})$ and $p_2(\boldsymbol{x})$ approach zero. In comparison, consider the square of the L2 distance, which is given by

$$d_{\text{L2}}(p_1,p_2)^2 = \int d'_{\text{L2sqr}}(p_1(\boldsymbol{x}),p_2(\boldsymbol{x}))d\boldsymbol{x} \tag{8}$$

where

$$d'_{\text{L2sqr}}(p_1(\boldsymbol{x}),p_2(\boldsymbol{x})) = (p_1(\boldsymbol{x})-p_2(\boldsymbol{x}))^2. \tag{9}$$

In Fig. 2, $d'_{\text{L2sqr}}(p_1(\boldsymbol{x}),p_2(\boldsymbol{x}))$ is plotted as a function of $p_1(\boldsymbol{x})$ and $p_2(\boldsymbol{x})$. Experimentally, using the L2 distance between Gaussian mixture models does not work well for genre classification. In unpublished nearest-neighbor experiments on the ISMIR 2004 genre classification training set, we obtained 42% accuracy using the L2 distance compared to 65% using the symmetrized Kullback–Leibler divergence (in the experiments, nearest-neighbor songs by the same artist as the query song were ignored). From this it would seem that the success of the symmetrized Kullback–Leibler divergence in music information retrieval is crucially linked to it asymptotically going towards infinity when one of p_1 and p_2 goes towards zero, i.e., it highly penalizes differences. This is supported by the observation in [10] that only a minority of a song's MFCCs actually discriminate it from other songs.

A disadvantage of using Gaussian mixture models to aggregate the MFCCs is that the temporal development of sounds is not taken into account, even though it is important to the perception of timbre [13], [14]. As noted in [10], a song can be modeled by the same Gaussian mixture model whether it is played

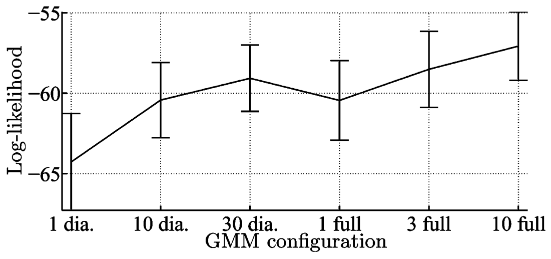

Fig. 3. Log-likelihood for various Gaussian mixture model configurations. The number denotes the number of Gaussians in the mixture, and the letter is "d" for diagonal covariance matrices and "f" for full covariance matrices.

forwards or backwards, even though it clearly makes an audible difference. Another disadvantage is that when two instruments play simultaneously, the probability density function (pdf) of the MFCCs will in general change rather unpredictably. If the two instruments only have little overlap in the melfrequency domain, they will still be approximately linearly mixed after taking the logarithm in step 4 in Section II-A and after the discrete cosine transform, since the latter is a linear operation. However, the pdf of a sum of two stochastic variables is the convolution of the pdf of each of the variables. Only if the instruments do not play simultaneously will the resulting pdf contain separate peaks for each instrument. To make matters even worse, such considerations also apply when chords are being played, and in this case it is almost guaranteed that some harmonics will fall into the same frequency bands, removing even the possibility of nonoverlapping spectra.

With Gaussian mixture models, the covariance matrices are often assumed to be diagonal for computational simplicity. In [7] and [8], it was shown that instead of a Gaussian mixture model where each Gaussian component has diagonal covariance matrix, a single Gaussian with full covariance matrix can be used without sacrificing discrimination performance. This simplifies both training and evaluation, since the closed form expressions in (2), (3), and (5) can be used. If the inverse of the covariance matrices are precomputed, (5) can be evaluated quite efficiently since the trace term only requires the diagonal elements of $\left(\boldsymbol{\Sigma}_1{}^{-1}\boldsymbol{\Sigma}_2\right)$ to be computed. For the symmetric version, the log terms even cancel, thus not even requiring the determinants to be precomputed. In Fig. 3, the average log-likelihoods for 30 randomly selected songs from the ISMIR 2004 genre classification training database are shown for different Gaussian mixture model configurations. The figure shows that log-likelihoods for a mixture of ten Gaussians with diagonal covariances and one Gaussian with full covariance matrix is quite similar. Using 30 Gaussians with diagonal covariance matrices increases the log-likelihood, but as shown in [9], genre classification performance does not benefit from this increased modeling accuracy. Log-likelihoods indicate only how well a model has captured the underlying density of the data, and not how well the models will discriminate in a classification task.

III. Experiments

In this section, we present six experiments that further investigate the behavior of the MFCC–Gaussian–KL approach. The basic assumption behind all the experiments is that this approach is a timbral distance measure and that as such it is supposed to perform well at instrument classification. In all experiments, we thus see how the instrument recognition performance is affected by various transformations and distortions. To perform the experiments, we take a number of MIDI files that are generated with Microsoft Music Producer and modify them in different ways to specifically show different MFCC properties. To synthesize wave signals from the MIDI files, we use the software synthesizer TiMidity++ version 2.13.2 with the six sound fonts listed in Table I. As each sound font uses different instrument samples, this approximates using six different realizations of each instrument. To compute MFCCs, we use the implementation in the Intelligent Sound Project toolbox that originates from the VOICEBOX toolbox by Mike Brookes. This implementation is described in [17] and includes frequencies up to 11 025 Hz in the MFCCs. To aggregate the MFCCs from each synthesized MIDI file, we use the approach with a single Gaussian with full covariance matrix, since this would be the obvious choice in practical applications due to the clear computational advantages. All experiments have been performed with a number of different MFCC orders to see how it affects the results. We use $a{:}b$ to denote MFCCs where the ath to the bth coefficient have been kept after the discrete cosine transform. As an example, 0:6 is where the DC coefficient and the following six coefficients have been kept. The experiments are implemented in MATLAB, and the source code, MIDI files and links to the sound fonts are available online.[1]

TABLE I
Six Sound Fonts Used for the Experiments

Number	Sound font
1	AirFont 340 v1.01
2	Fluid R3 GM
3	GeneralUser GS 1.4
4	PersonalCopy 51f
5	RealFont 2.1
6	SGM-180 v1.5

A. Timbre Versus Melody Classification

The first experiment is performed to verify that the MFCC–Gaussian–KL approach described in Section II also groups songs by instrumentation when an instrument plays several notes simultaneously. Due to the simple relation between harmonics in chords, the MFCC–Gaussian–KL approach could equally well match songs with similar chords than songs with identical instrumentation. When we refer to melodies in this section, we are thus not concerned with the lead melody, but rather with the chords and combinations of notes that are characteristic to a particular melody.

To perform the experiment, we take 30 MIDI songs of very different styles and the 30 MIDI instruments listed in Table II. For all combinations of songs and instruments, we perform the following.

1) Read MIDI song i.
2) Remove all percussion.
3) Force all notes to be played by instrument j.
4) Synthesize a wave signal $s_{ij}(n)$.

[1]http://kom.aau.dk/~jhj/publications/

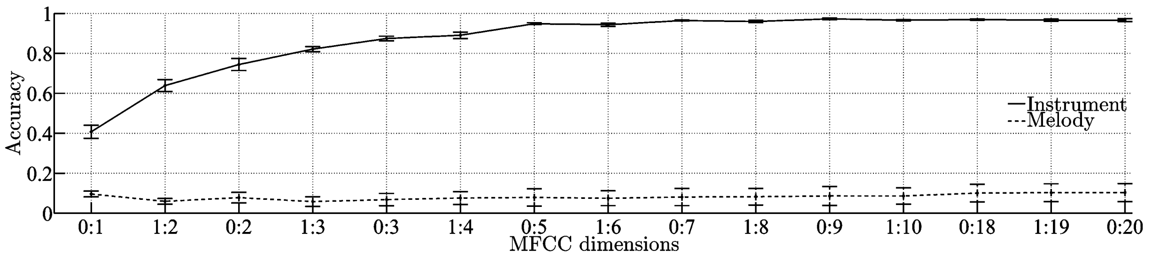

Fig. 4. Mean and standard deviation of instrument and melody classification accuracies, i.e., the fraction of songs that have a song with the same instrumentation, or the same melody as nearest neighbor, respectively. For moderate MFCC orders, the instrument classification accuracy is consistently close to 1, and the melody classification accuracy is close to 0.

5) Extract MFCCs.
6) Train a multivariate Gaussian probability density function $p_{ij}(\boldsymbol{x})$ on the MFCCs.

Next, we perform nearest-neighbor classification on the $30 \times 30 = 900$ songs, i.e., for each song we compute

$$(p,q) = \underset{\substack{k,l \\ (k,l)\neq(i,j)}}{\operatorname{argmin}}\ d_{\mathrm{sKL}}(p_{ij}, p_{kl}). \tag{10}$$

If the nearest neighbor to song $s_{ij}(n)$, played with instrument j, is $s_{pq}(n)$, and it is also played with instrument j, i.e., $j = q$, then there is a match of instruments. We define the instrument classification rate by the fraction of songs where the instrument of a song and its nearest-neighbor matches. Similarly, we define the melody classification rate by the fraction of songs where $i = p$. We repeat the experiment for the different sound fonts. Forcing all notes in a song to be played by the same instrument is not realistic, since, e.g., the bass line would usually not be played with the same instrument as the main melody. However, using only the melody line would be an oversimplification. Keeping the percussion, which depends on the song, i, would also blur the results, although in informal experiments, keeping it only decreases the instrument classification accuracy by a few percentage points. In Fig. 4, instrument and melody classification rates are shown as a function of the MFCC order and the sound font used. From the figure, it is evident that when using even a moderate number of coefficients, the MFCC–Gaussian–KL approach is successful at identifying the instrument and is almost completely unaffected by the variations in the note and chord distributions present in the different songs.

B. Ensembles

Next, we repeat the experiment from the previous section using three different instruments for each song instead of just one. We select 30 MIDI files that each have three nonpercussive tracks, and we select three sets with three instruments each. Let $\{a_1, a_2, a_3\}$, $\{b_1, b_2, b_3\}$, and $\{c_1, c_2, c_3\}$ denote the three sets, let $j, k, l \in 1, 2, 3$, and let i denote the MIDI file number. Similar to the experiment in Section III-A, we perform the following for all combinations of i, j, k, and l.

1) Read MIDI song i.
2) Remove all percussion.
3) Let all notes in the first, second, and third track be played by instrument a_j, b_k, and c_l, respectively.
4) Synthesize a wave signal $s_{ijkl}(n)$.
5) Extract MFCCs.

TABLE II
INSTRUMENTS USED TO SYNTHESIZE THE SONGS USED FOR THE EXPERIMENTS. ALL ARE FROM THE GENERAL MIDI SPECIFICATION

Number	Instrument name
1	Acoustic Grand Piano
11	Music Box
14	Xylophone
15	Tubular Bells
20	Church Organ
23	Harmonica
25	Acoustic Guitar (nylon)
37	Slap Bass 1
41	Violin
47	Orchestral Harp
53	Choir Aahs
54	Voice Oohs
57	Trumpet
66	Alto Sax
71	Bassoon
74	Flute
76	Pan Flute
77	Blown Bottle
79	Whistle
81	Lead 1 (square)
82	Lead 2 (sawtooth)
85	Lead 5 (charang)
89	Pad 1 (new age)
93	Pad 5 (bowed)
94	Pad 6 (metallic)
97	FX 1 (rain)
105	Sitar
110	Bag pipe
113	Tinkle Bell
115	Steel Drums

6) Train a multivariate Gaussian probability density function $p_{ijkl}(\boldsymbol{x})$ on the MFCCs.

As before, the nearest neighbor is found, but this time according to

$$(p', q', r', s') = \underset{\substack{p,q,r,s \\ p\neq i}}{\operatorname{argmin}}\ d_{\mathrm{sKL}}(p_{ijkl}, p_{pqrs}). \tag{11}$$

Thus, the nearest neighbor is not allowed to have the same melody as the query. This is to avoid that the nearest neighbor is the same melody with the instrument in a weak track replaced by another instrument. The fraction of nearest neighbors with the same three instruments, the fraction with at least two identical instruments and the fraction with at least one identical instrument is computed by counting how many of (q', r', s') equals (j, k, l).

In Fig. 5, the fractions of nearest neighbors with different numbers of identical instruments are plotted. The fraction

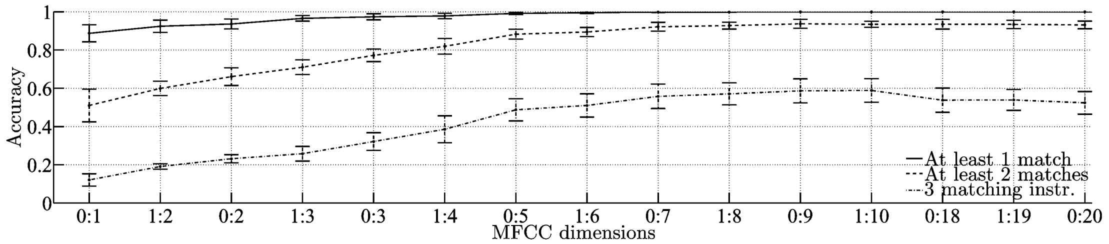

Fig. 5. Mean and standard deviation of instrument classification accuracies when the success criterion is that the nearest neighbor has at least one, two, or three identical instruments. Results are averaged over all six sound fonts.

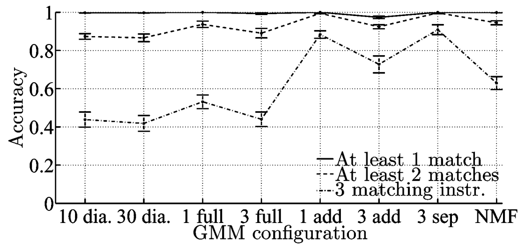

Fig. 6. Instrument classification rates for different configurations of the Gaussian mixture model. The numbers denote the number of Gaussians in the mixture, and "dia." and "full" refer to the covariance matrices. For both "add" and "sep," each instrument has been synthesized independently. For "add," the tracks were concatenated to a single signal, while for "sep," the three equally weighted Gaussians were trained separately for each track. For "NMF," an NMF source separation algorithm has been applied. Results are averaged over all six sound fonts.

of nearest neighbors with two or more identical instruments is comparable to the instrument classification performance in Fig. 4. To determine if the difficulties detecting all three instrument are caused by the MFCCs or the Gaussian model, we have repeated the experiments in Fig. 6 with MFCCs 0:10 for the following seven setups.

- Using Gaussian mixture models with ten and 30 diagonal covariance matrices, respectively.
- Gaussian mixture models with one and three full covariance matrices, respectively.
- Gaussian mixture models with one and three full covariance matrices, respectively, but where the instruments in a song are synthesized independently and subsequently concatenated into one song of triple length.
- Gaussian mixture models with three full covariance matrices where each instrument in a song is synthesized independently, and each Gaussian is trained on a single instrument only. The weights are set to 1/3 each.
- Gaussian mixture model with one full covariance matrix, where, as a proof of concept, a non-negative matrix factorization (NMF) algorithm separates the MFCCs into individual sources that are concatenated before training the Gaussian model. The approach is a straightforward adoption of [35], where the NMF is performed between steps 3 and 4 in the MFCC computation described in Section II-A. As we, in line with [35], use a log-scale instead of the melscale, we should rightfully use the term LFCC instead of MFCC. Note that, like the first two setups, but unlike the setups based on independent instruments, this approach does not require access to the original, separate waveforms of each instrument, and thus is applicable to existing recordings.

From the additional experiments, it becomes clear that the difficulties capturing all three instruments originate from the simultaneous mixture. As we saw in Section III-A, it does not matter that one instrument plays several notes at a time, but from Fig. 5 and the "1 full add" experiment in Fig. 6, we see that it clearly makes a difference whether different instruments play simultaneously. Although a slight improvement is observed when using separate Gaussians for each instrument, a single Gaussian actually seems to be adequate for modeling all instruments as long as different instruments do not play simultaneously. We also see that the NMF-based separation algorithm increases the number of cases where all three instruments are recognized. It conveniently simplifies the source separation task that a single Gaussian is sufficient to model all three instruments, since it eliminates the need to group the separated sources into individual instruments.

C. Different Realizations of the Same Instrument

In Section III-A, we saw that the MFCC–Gaussian–KL approach was able to match songs played by the same instrument when they had been synthesized using the same sound font. In this section, to get an idea of how well this approach handles two different realizations of the same instrument, we use synthesized songs from different sound fonts as test and training data and measure the instrument classification performance once again. To the extent that a human listener would consider one instrument synthesized with two different sound fonts more similar than the same instrument synthesized by the first sound font and another instrument synthesized by the second, this experiment can also be considered a test of how well the MFCC–Gaussian–KL approach approximates human perception of timbral similarity. The experimental setup is that of Section III-A, only we use two different sound fonts, sf_m and sf_n, to synthesize two wave signals, $s_{ij}^{(\mathrm{sf}_m)}(n)$ and $s_{ij}^{(\mathrm{sf}_n)}(n)$, and estimate two multivariate Gaussians probability density functions, $p_{ij}^{(\mathrm{sf}_m)}(\boldsymbol{x})$ and $p_{ij}^{(\mathrm{sf}_n)}(\boldsymbol{x})$. We perform nearest neighbor classification again, but this time with a query synthesized with sf_m and a training set synthesized with sf_n, i.e., (10) is modified to

$$(p,q) = \underset{\substack{k,l \\ (k,l)\neq(i,j)}}{\operatorname{argmin}} \; d_{\mathrm{sKL}}\left(p_{ij}^{(\mathrm{sf}_m)}, p_{kl}^{(\mathrm{sf}_n)}\right). \qquad (12)$$

We test all combinations of the sound fonts mentioned in Table I. The resulting instrument classification rates are shown in Fig. 7, and we see that the performance when using two different sound fonts are relatively low. We expect the low

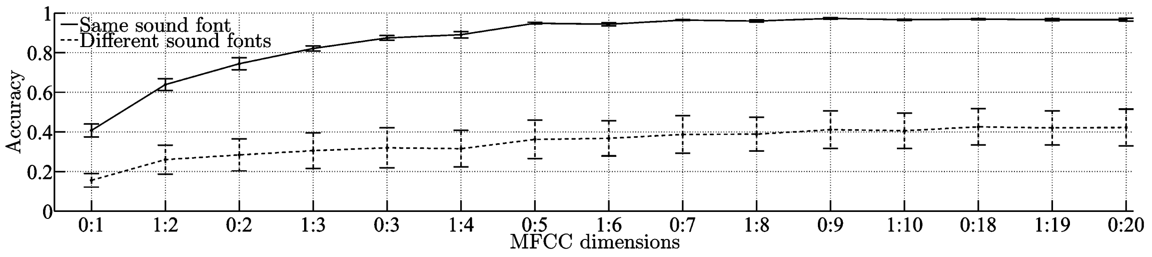

Fig. 7. Mean and standard deviation of instrument classification accuracies when mixing different sound fonts.

performance to have the same cause as the album effect [18]. In [36], the same phenomenon was observed when classifying instruments across different databases of real instrument sounds, and they significantly increased classification performance by using several databases as training set. However, this is not directly applicable in our case, since the MFCC–Gaussian–KL is a song-level distance measure without an explicit training step.

When using songs synthesized from the same sound font for query and training, it is unimportant whether we increase the MFCC order by including the 0th coefficient or the next higher coefficient. However, when combining different sound fonts, including the 0th MFCC at the cost of one of the higher coefficients has noticeable impact on performance. Unfortunately, since it is highly dependent on the choice of sound fonts if performance increases or decreases, an unambiguous conclusion cannot be drawn.

D. Transposition

When recognizing the instruments that are playing, a human listener is not particularly sensitive to transpositions of a few semitones. In this section, we experimentally evaluate how the MFCC–Gaussian–KL approach behaves in this respect. The experiment is built upon the same framework as the experiment in Section III-A and is performed as follows.

1) Repeat step 1–3 of the experiment in Section III-A.
2) Normalize the track octaves (see below).
3) Transpose the song T_m semitones.
4) Synthesize wave signals $s_{ij}^{(T_m)}(n)$.
5) Extract MFCCs.
6) Train a multivariate Gaussian probability density function $p_{ij}^{(T_m)}(\boldsymbol{x})$.

The octave normalization consists of transposing all tracks (e.g., bass and melody) such that the average note is as close to C4 (middle C on the piano) as possible, while only transposing the individual tracks an integer number of octaves relative to each other. The purpose is to reduce the tonal range of the songs. If the tonal range is too large, the majority of notes in a song and its transposed version will exist in both versions, hence blurring the results (see Fig. 8). By only shifting the tracks an integer number of octaves relative to each other, we ensure that all harmonic relations between the tracks are kept. This time, the nearest neighbor is found as

$$(p,q) = \underset{\substack{k,l \\ (k,l)\neq(i,j)}}{\operatorname{argmin}} \; d_{\text{sKL}}\left(p_{ij}^{(T_m)}, p_{kl}^{(T_0)}\right). \qquad (13)$$

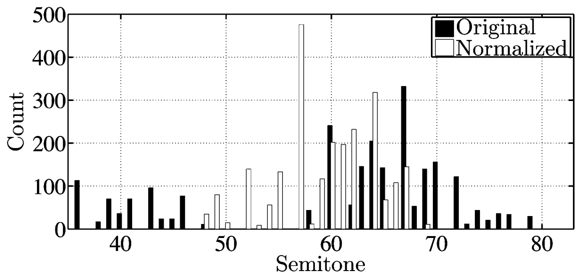

Fig. 8. Histogram of notes in a MIDI song before and after normalization. The x-axis is the MIDI note number, i.e., 64 is middle C on the piano. The tonal range of the original song is much larger than that of the normalized song.

That is, we search for the nearest neighbor to $p_{ij}^{(T_m)}(\boldsymbol{x})$ among the songs that have only been normalized but have not been transposed any further. The instrument and melody classification rates are computed for 11 different values of T_s that are linearly spaced between -24 and 24, which means that we maximally transpose songs two octaves up or down.

In Fig. 9, instrument classification performance is plotted as a function of the number of semitones the query songs are transposed. Performance is hardly influenced by transposing songs ± 5 semitones. Transposing ten semitones, which is almost an octave, noticeably affects results. Transposing ± 24 semitones severely reduces accuracy. In Fig. 10, where instrument classification performance is plotted as a function of the MFCC order, we see that the instrument recognition accuracy generally increase with increasing MFCC order, stagnating around 10.

E. Bandwidth

Since songs in an actual music database may not all have equal sample rates, we examine the sensitivity of the MFCC–Gaussian–KL approach to downsampling, i.e., reducing the bandwidth. We both examine what happens if we mix songs with different bandwidths, and what happens if all songs have reduced, but identical bandwidth. Again, we consider the MFCCs a timbral feature and use instrument classification performance as ground truth.

1) Mixing Bandwidths: This experiment is very similar to the transposition experiment in Section III-D, only we reduce the bandwidths of the songs instead of transposing them. Practically, we use the MATLAB resample function to downsample the wave signal to $2 \cdot BW$ and upsample it to 22 kHz again. The nearest neighbor instrument classification rate is found as in (13) with $p_{ij}^{(T_m)}$ and $p_{kl}^{(T_0)}$ replaced by $p_{ij}^{(BW_m)}$ and $p_{kl}^{(BW_0)}$, respectively. The reference setting BW_0 is 11 kHz, corresponding to a sampling frequency of 22 kHz.

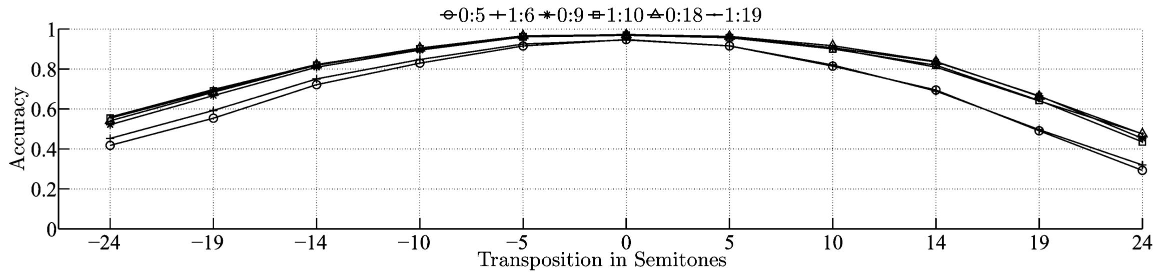

Fig. 9. Instrument classification rate averaged over all sound fonts as a function of the number of semitones that query songs have been transposed.

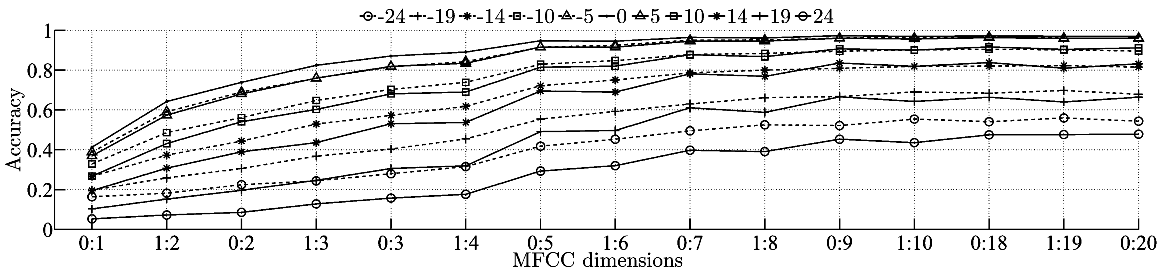

Fig. 10. Instrument classification rate averaged over all sound fonts as a function of the number of MFCCs. The numbers -19, -14 etc. denote the number of semitones songs have been transposed.

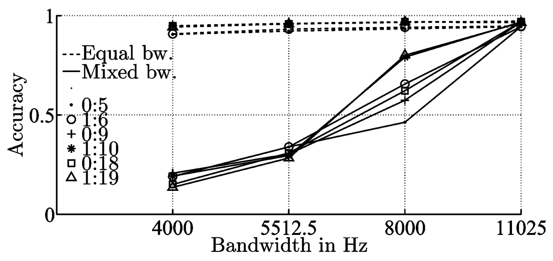

Fig. 11. Average instrument classification accuracy averaged over all sound fonts when reducing the songs' bandwidths. For the mixed bandwidth results, the training set consists of songs with full bandwidth, while for the equal bandwidth results, songs in both the test and training sets have equal, reduced bandwidth.

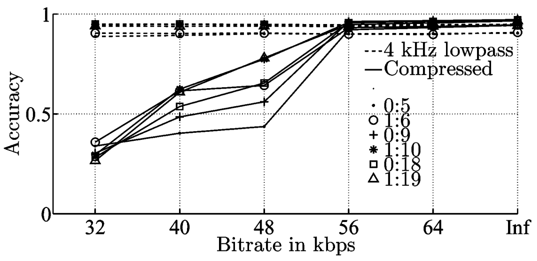

Fig. 12. Instrument classification rates averaged over all sound fonts with MP3 compressed query songs as a function of bitrate.

2) Reducing Bandwidth for All Files: This experiment is performed as the experiment in Section III-A, except that synthesized wave signals are downsampled to BW before computing the MFCCs for *both* test and training songs.

Results of both bandwidth experiments are shown in Fig. 11. It is obvious from the figure that mixing songs with different bandwidths is a bad idea. Reducing the bandwidth of the query set from 11 kHz to 8 kHz significantly reduces performance, while reducing the bandwidth to 5.5 kHz, i.e., mixing sample rates of 22 kHz and 11 kHz, makes the distance measure practically useless with accuracies in the range from 30%–40%. On the contrary, if all songs have the same, low bandwidth, performance does not suffer significantly. It is thus clear that if different sampling frequencies can be encountered in a music collection, it is preferential to downsample all files to e.g., 8 kHz before computing the MFCCs. Since it is computationally cheaper to extract MFCCs from downsampled songs, and since classification accuracy is not noticeably affected by reducing the bandwidth, this might be preferential with homogeneous music collections as well. The experiment only included voiced instruments, so this result might not generalize to percussive instruments that often have more energy at high frequencies. In informal experiments on the ISMIR 2004 genre classification training database, genre classification accuracy only decreased by a few percentage points when downsampling all files to 8 kHz.

F. Bitrate

Music is often stored in a compressed format. However, as shown in [17], MFCCs are sensitive to the spectral perturbations introduced by compression. In this section, we measure how these issues affect instrument classification performance. This experiment is performed in the same way as the transposition experiment in Section III-D, except that transposing has been replaced by encoding to an MP3 file with bitrate B_m and decoding. Classification is also performed as given by (13). For MP3 encoding, the constant bitrate mode of LAME version 3.97 is used. The synthesized wave signal is in stereo when encoding but is converted to mono before computing the MFCCs. Results of different bitrates are shown in Fig. 12. Furthermore, results of reducing the bandwidth to 4 kHz after decompression are also shown. Before compressing the wave signal, the MP3 encoder applies a lowpass filter. At 64 kbps, this lowpass filter has transition band from 10935 Hz to 11226 Hz, which is in the range of the very highest frequencies used when computing the MFCCs. Consequently, classification rates are virtually unaffected at a bitrate of 64 kbps. At 48 kbps, the transition band is between 7557 Hz and 7824 Hz, and at 32 kbps, the transition band is between 5484 Hz and 5677 Hz. The classification rates at 5.5 kHz and 8 kHz in Fig. 11 and at 32 kbps and 48 kbps in Fig. 12, respectively, are strikingly similar, hinting that bandwidth reduction is the major cause of the reduced accuracy. This is confirmed by the experiments where the bandwidth is always reduced to 4 kHz, which are unaffected by changing bitrates. So, if robustness to low bitrate MP3 encoding is desired, all songs should be downsampled before computing MFCCs.

IV. Discussion

In all experiments, we let multivariate Gaussian distributions model the MFCCs from each song and used the symmetrized Kullback–Leibler divergence between the Gaussian distributions as distance measures. Strictly speaking, our results therefore only speak of the MFCCs with this particular distance measure and not of the MFCCs on their own. However, we see no obvious reasons that other classifiers would perform radically different.

In the first experiment, we saw that when keeping as little as four coefficients while excluding the 0th cepstral coefficient, instrument classification accuracy was above 80%. We therefore conclude that MFCCs primarily capture the spectral envelope when encountering a polyphonic mixture of voices from one instrument and not e.g., the particular structure encountered when playing harmonies.

When analyzing songs played by different instruments, only two of the three instruments were often recognized. The number of cases where all instruments were recognized increased dramatically when instruments were playing in turn instead of simultaneously, suggesting that the cause is either the log-step when computing the MFCCs, or the phenomenon that the probability density functions of a sum of random variables is the convolution of the individual probability density functions. From this it is clear that the success of the MFCC–Gaussian–KL approach in genre and artist classification is very possible due only to instrument/ensemble detection. This is supported by [37] that showed that for symbolic audio, instrument identification is very important to genre classification. We hypothesize that in genre classification experiments, recognizing the two most salient instruments is enough to achieve acceptable performance.

In the third experiment, we saw that the MFCC–Gaussian–KL approach does not consider songs with identical instrumentation synthesized with different sound fonts very similar. However, with nonsynthetic music databases, e.g., [5] and [8], this distance measure seems to perform well even though different artists use different instruments. A possible explanation may be that the synthesized sounds are more homogeneous than a corresponding human performance, resulting in over-fitting of the multivariate Gaussian distributions. Another possibility is that what makes a real-world classifier work is the diversity among different performances in the training collection; i.e., if there are 50 piano songs in a collection, then a given piano piece may only be close to one or two of the other piano songs, while the rest, with respect to the distance measure, just as well could have been a trumpet piece or a xylophone piece. As observed in [8], performance of the MFCC–Gaussian–KL approach in genre classification increases significantly if songs by the same artist are in both the training and test collection, thus supporting the latter hypothesis. We speculate that relying more on the temporal development of sounds (for an example of this, see [38]) and less on the spectral shape and using a more perceptually motivated distance measure instead of the Kullback–Leibler divergence can improve the generalization performance.

In [5], it is suggested that there is a “glass ceiling” for the MFCC–Gaussian–KL approach at 65%, meaning that no simple variation of it can exceed this accuracy. From the experiments, we can identify three possible causes of the glass ceiling.

1) The MFCC–Gaussian–KL approach neither takes melody nor harmony into account.
2) It is highly sensitive to different renditions of the same instrument.
3) It has problems identifying individual instruments in a mixture.

With respect to the second cause, techniques exists for suppressing channel effects in MFCC-based speaker identification. If individual instruments are separated in a preprocessing step, these techniques might be applicable to music as well. As shown in Section III-B, a successful signal separation algorithm would also mitigate the third cause.

We measured the reduction in instrument classification rate when transposing songs. When transposing songs only a few semitones, instrument recognition performance was hardly affected, but transposing songs in the order of an octave or more causes performance to decrease significantly. When we compared MFCCs computed from songs with different bandwidths, we found that performance decreased dramatically. In contrast, if all songs had the same, low bandwidth, performance typically did not decrease more than 2–5 percentage points. Similarly, comparing MFCCs computed from low bitrate MP3 files and high bitrate files also affected instrument classification performance dramatically. The performance decrease for mixing bitrates matches the performance decrease when mixing bandwidths very well. If a song collection contains songs with different sample rates or different bitrates, it is recommended to downsample all files before computing the MFCCs.

V. Conclusion

We have analyzed the properties of a commonly used music similarity measure based on the Kullback–Leibler distance between Gaussian models of MFCC features. Our analyses show that the MFCC–Gaussian–KL measure of distance between songs recognizes instrumentation; a solo instrument playing several notes simultaneously does not degrade recognition accuracy, but an ensemble of instruments tend to suppress the weaker instruments. Furthermore, different realizations of instruments significantly reduces recognition performance. Our results suggest that the use of source separation methods in combination with already existing music similarity measures may lead to increased classification performance.

Acknowledgment

The authors would like to thank H. Laurberg for assistance with the non-negative matrix factorization algorithm used in Section III-B.

References

[1] J. T. Foote, "Content-based retrieval of music and audio," *Proc. SPIE Multimedia Storage and Archiving Syst. II*, pp. 138–147, 1997.

[2] B. Logan and A. Salomon, "A music similarity function based on signal analysis," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2001, pp. 745–748.

[3] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 5, pp. 293–301, Jul. 2002.

[4] J.-J. Aucouturier and F. Pachet, "Finding songs that sound the same," in *Proc. IEEE Benelux Workshop Model Based Process. Coding Audio*, 2002, pp. 91–98.

[5] J.-J. Aucouturier and F. Pachet, "Improving timbre similarity: How high's the sky?," *J. Negative Results Speech Audio Sci.*, vol. 1, no. 1, 2004.

[6] E. Pampalk, "Speeding up music similarity," in *Proc. 2nd Annu. Music Inf. Retrieval eXchange*, 2005.

[7] M. I. Mandel and D. P. Ellis, "Song-level features and support vector machines for music classification," in *Proc. Int. Symp. Music Inf. Retrieval*, 2005, pp. 594–599.

[8] E. Pampalk, "Computational models of music similarity and their application to music information retrieval," Ph.D. dissertation, Vienna Univ. of Technology, Vienna, Austria, 2006.

[9] A. Flexer, "Statistical evaluation of music information retrieval experiments," Inst. of Medical Cybernetics and Artificial Intelligence, Medical Univ. of Vienna, Vienna, Austria, Tech. Rep, 2005.

[10] J.-J. Aucouturier, "Ten experiments on the modelling of polyphonic timbre," Ph.D. dissertation, Univ. of Paris 6, Paris, France, 2006.

[11] J. Bergstra, N. Casagrande, D. Erhan, D. Eck, and B. Kégl, "Aggregate features and Adaboost for music classification," *Mach. Learn.*, vol. 65, no. 2–3, pp. 473–484, 2006.

[12] J. H. Jensen, M. G. Christensen, M. N. Murthi, and S. H. Jensen, "Evaluation of MFCC estimation techniques for music similarity," in *Proc. Eur. Signal Process. Conf.*, 2006.

[13] T. D. Rossing, F. R. Moore, and P. A. Wheeler, *The Science of Sound*, 3rd, Ed. New York: Addison-Wesley, 2002.

[14] B. C. J. Moore, *An Introduction to the Psychology of Hearing*, 5th, Ed. New York: Elsevier Academic Press, 2004.

[15] *Acoustical Terminology SI*, Rev. 1–1960, American Standards Association Std., New York, 1960.

[16] A. Nielsen, S. Sigurdsson, L. Hansen, and J. Arenas-Garcia, "On the relevance of spectral features for instrument classification," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. ICASSP'07*, 2007, vol. 2, pp. II-485–II-488.

[17] S. Sigurdsson, K. B. Petersen, and T. Lehn-Schiøler, "Mel frequency cepstral coefficients: An evaluation of robustness of mp3 encoded music," in *Proc. Int. Symp. Music Inf. Retrieval*, 2006.

[18] Y. E. Kim, D. S. Williamson, and S. Pilli, "Understanding and quantifying the album effect in artist identification," in *Proc. Int. Symp. Music Inf. Retrieval*, 2006.

[19] S. B. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. ASSP-28, no. 4, pp. 357–366, Aug. 1980.

[20] A. V. Oppenheim and R. W. Schafer, "From frequency to quefrency: A history of the cepstrum," *IEEE Signal Process. Mag.*, vol. 21, no. 5, pp. 95–106, Sep. 2004.

[21] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 1st, Ed. Englewood Cliffs, NJ: Prentice-Hall, 1989.

[22] P. Stoica and N. Sandgren, "Smoothed nonparametric spectral estimation via cepsturm thresholding," *IEEE Signal Process. Mag.*, vol. 23, no. 6, pp. 34–45, Nov. 2006.

[23] H. Terasawa, M. Slaney, and J. Berger, "Perceptual distance in timbre space," in *Proc. Int. Conf. Auditory Display*, 2005, pp. 61–68.

[24] F. Zheng, G. Zhang, and Z. Song, "Comparison of different implementations of MFCC," *J. Comput. Sci. Technol.*, vol. 16, pp. 582–589, 2001.

[25] M. Wölfel and J. McDonough, "Minimum variance distortionless response spectral estimation," *IEEE Signal Process. Mag.*, vol. 22, no. 5, pp. 117–126, Sep. 2005.

[26] E. Pampalk, "A MatLab toolbox to compute music similarity from audio," in *Proc. Int. Symp. Music Inf. Retrieval*, 2004, pp. 254–257.

[27] B. Logan, "Mel frequency cepstral coefficients for music modeling," in *Proc. Int. Symp. Music Inf. Retrieval*, 2000.

[28] Z. Liu and Q. Huang, "Content-based indexing and retrieval-by-example in audio," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2000, pp. 877–880.

[29] P. Stoica and R. Moses, *Spectral Analysis of Signals*. Upper Saddle River, NJ: Prentice-Hall, 2005.

[30] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *J. R. Statist. Soc. Ser. B*, vol. 39, no. 1, pp. 1–38, 1977.

[31] R. A. Redner and H. F. Walker, "Mixture densities, maximum likelihood, and the EM algorithm," *SIAM Rev.*, vol. 26, no. 2, pp. 195–239, 1984.

[32] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York: Wiley, 1991.

[33] N. Vasconcelos, "On the complexity of probabilistic image retrieval," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2001, pp. 400–407.

[34] A. Berenzweig, "Anchors and hubs in audio-based music similarity," Ph.D. dissertation, Columbia Univ., New York, 2007.

[35] A. Holzapfel and Y. Stylianou, "Musical genre classification using nonnegative matrix factorization-based features," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 16, no. 2, pp. 424–434, Feb. 2008.

[36] A. Livshin and X. Rodet, "The importance of cross database evaluation in sound classification," in *Proc. Int. Symp. Music Inf. Retrieval*, 2003.

[37] C. McKay and I. Fujinaga, "Automatic music classification and the importance of instrument identification," in *Proc. Conf. Interdisciplinary Musicol.*, 2005.

[38] A. Meng, P. Ahrendt, J. Larsen, and L. Hansen, "Temporal feature integration for music genre classification," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, no. 5, pp. 1654–1664, Jul. 2007.

Jesper Højvang Jensen (M'08) was born in West Jutland, Denmark, in 1981. He received the M.Sc. degree in electrical engineering from Aalborg University, Denmark, in 2005. where he is currently pursuing the Ph.D. degree within the Intelligent Sound Project.

He has been a Visiting Researcher at Columbia University, New York, and his primary research interest is feature extraction for music similarity.

Mads Græsbøll Christensen (S'00–M'06) was born in Copenhagen, Denmark, in March 1977. He received the M.Sc. and Ph.D. degrees from Aalborg University, Aalborg, Denmark, in 2002 and 2005, respectively.

He is currently an Assistant Professor with the Department of Electronic Systems, Aalborg University. He has been a Visiting Researcher at Philips Research Labs, Ecole Nationale Supérieure des Télécommunications (ENST), and Columbia University, New York. His research interests include digital signal processing theory and methods with application to speech and audio, in particular parametric analysis, modeling, and coding of speech and audio signals.

Dr. Christensen received several awards, namely an IEEE International Conference Acoustics, Speech, and Signal Processing Student Paper Contest Award, the Spar Nord Foundation's Research Prize awarded anually for an excellent Ph.D. dissertation, and a Danish Independent Research Council's Young Researcher's Award.

Daniel P. W. Ellis (S'93–M'96–SM'04) received the Ph.D. degree in electrical engineering from the Massachusetts Institute of Technology (MIT), Cambridge, in 1996.

He was a Research Assistant at the Media Lab, MIT . He is currently an Associate Professor in the Electrical Engineering Department, Columbia University, New York. His Laboratory for Recognition and Organization of Speech and Audio (LabROSA) is concerned with extracting high-level information from audio, including speech recognition, music description, and environmental sound processing. He is an External Fellow of the International Computer Science Institute, Berkeley, CA. He also runs the AUDITORY e-mail list of 1700 worldwide researchers in perception and cognition of sound.

Søren Holdt Jensen (S'87–M'88–SM'00) received the M.Sc. degree in electrical engineering from Aalborg University, Aalborg, Denmark, in 1988, and the Ph.D. degree in signal processing from the Technical University of Denmark, Lyngby, in 1995.

Before joining the Department of Electronic Systems, Aalborg University, he was with the Telecommunications Laboratory of Telecom Denmark, Ltd, Copenhagen, Denmark; the Electronics Institute, Technical University of Denmark; the Scientific Computing Group, Danish Computing Center for Research and Education (UNI-C), Lyngby; the Electrical Engineering Department, Katholieke Universiteit Leuven, Leuven, Belgium; and the Center for PersonKommunikation (CPK), Aalborg University. His research interests include statistical signal processing, speech and audio processing, multimedia technologies, digital communications, and satellite based navigation.

Dr. Jensen was an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING, is Member of the Editorial Board of the *EURASIP Journal on Advances in Signal Processing*, is an is Associate Editor for *Elsevier Signal Processing* and the *Research Letters in Signal Processing*, and has also guest-edited two special issues for the *EURASIP Journal on Applied Signal Processing*. He is a recipient of an European Community Marie Curie Fellowship, former Chairman of the IEEE Denmark Section, and Founder and Chairman of the IEEE Denmark Section's Signal Processing Chapter.