Effects of Album and Artist Filters in Audio Similarity Computed for Very Large Music Databases

Arthur Flexer[1] and Dominik Schnitzer[1,2]
(1) Austrian Research Institute for Artificial Intelligence, Vienna, Austria
(2) Department of Computational Perception, Johannes Kepler University Linz, Austria

In audio based music recommendation, a well known effect is the dominance of songs from the same artist as the query song in recommendation lists. We verify that this effect also exists in very large databases (> 250000 songs). Since our data set contains multiple albums from individual artists, we can also show that the album effect is relatively bigger than the artist effect.

Introduction

In Music Information Retrieval, one of the central goals is to automatically recommend music to users based on a query song or query artist. This can be done using expert knowledge (e.g. `pandora.com`), social meta-data (e.g. `last.fm`), collaborative filtering (e.g. `amazon.com/mp3`) or by extracting information directly from the audio (e.g. `muffin.com`). In audio based music recommendation, a well known effect is the dominance of songs from the same artist as the query song in recommendation lists.

This effect has been studied mainly in the context of genre classification experiments. Since usually no ground truth with respect to music similarity exists, genre classification is widely used for evaluation of music similarity. Each song is labelled as belonging to a music genre using e.g. music expert advice. High genre classification results indicate good similarity measures. If in genre classification experiments songs from the same artist are allowed in both training and test sets, this can lead to over-optimistic results since usually all songs from an artist have the same genre label. It can be argued that in such a scenario one is doing artist classification rather than genre classification. One could even speculate that the specific sound of an album (mastering and production effects) is being classified. In (10) the use of a so-called "artist filter" ensuring that all songs from an artist are in either the training or the test set is proposed. The authors found that the use of such an artist filter can lower the classification results quite considerably (with one of their music collection even from 71% down to 27%). These over-optimistic accuracy results due to not using an artist filter have been confirmed in other studies (8) (2). Other

results suggest that the use of an artist filter not only lowers genre classification accuracy but may also erode the differences in accuracies between different techniques (3).

All these results were achieved on rather small databases (from 700 to 15000 songs). Often whole albums from an artist were part of the data base, maybe even more than one. These specifics of the databases are often unclear and not properly documented. In extending these results, we analyse a very large data set (> 250000 songs) with multiple albums from individual artists. We try to answer the following questions:

- Is there an album and artist effect even in very large databases?
- Is the album effect larger than the artist effect?
- What is the influence of the size of a database on music recommendation and classification?

Data

For our experiments we used a data set $D(ALL)$ of $S = 254398$ song excerpts (30 seconds) from a popular web-shop selling music. The freely available preview song excerpts were obtained with an automated web-crawl. All meta information (artist name, album title, song title, genres) is parsed automatically from the hmtl-code. The excerpts are from $U = 18386$ albums from $A = 1700$ artists. From the 280 existing different hierarchical genres, only the $G = 22$ general ones on top of the hierarchy are being kept for further analysis (e.g. "Pop/General" is kept but not "Pop/Vocal Pop"). The names of the genres plus percentages of songs belonging to each of the genres are given in Table 1. Please note that every song is allowed to belong to more than one genre, hence the percentages in Table 1 add up to more than 100%. The genre information is identical for all songs on an album. The numbers of genre labels per albums are given in Figure 1. Our database was set up so that every artist contributes between 6 to 29 albums (see Figure 2).

To study the influence of the size of the database on results, we created random non-overlapping splits of the entire data set: $D(1/2)$ - two data sets with mean number of song excerpts $= 127199$, $D(1/20)$ - twenty data sets with mean number of songs excerpts $= 12719.9$, $D(1/100)$ - one hundred data sets with mean number of songs excerpts $= 2543.98$. An artist with all their albums is always a member of a single data set.

Pop	Classical	Broadway
49.79	12.89	7.45
Soundtracks	Christian/Gospel	New Age
1.00	10.20	2.48
Miscellaneous	Opera/Vocal	Alternative Rock
6.11	3.24	27.13
Rock	Rap/Hip-Hop	R&B
51.78	0.98	4.26
Hard Rock/Metal	Classic Rock	Country
15.85	15.95	4.07
Jazz	Children's Music	International
6.98	7.78	9.69
Latin Music	Folk	Dance & DJ
0.54	11.18	5.24
Blues		
11.24		

Table 1. Percentages of songs belonging to the 22 genres with multiple membership allowed.

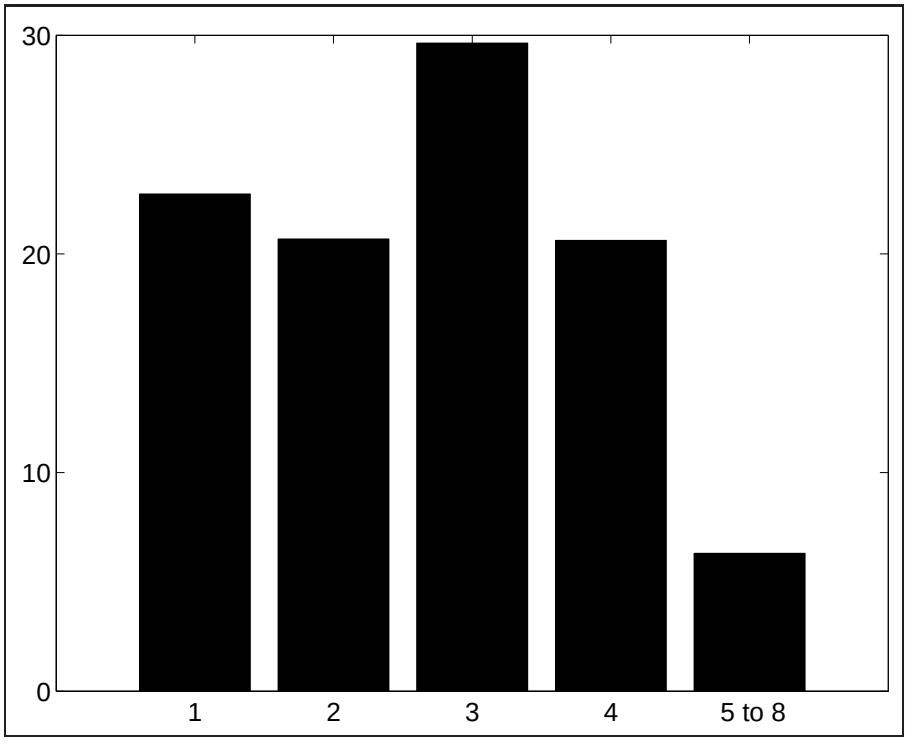

Figure 1. Percentages (y-axis) of albums having 1,2,3,4 or 5 to 8 genre labels (x-axis).

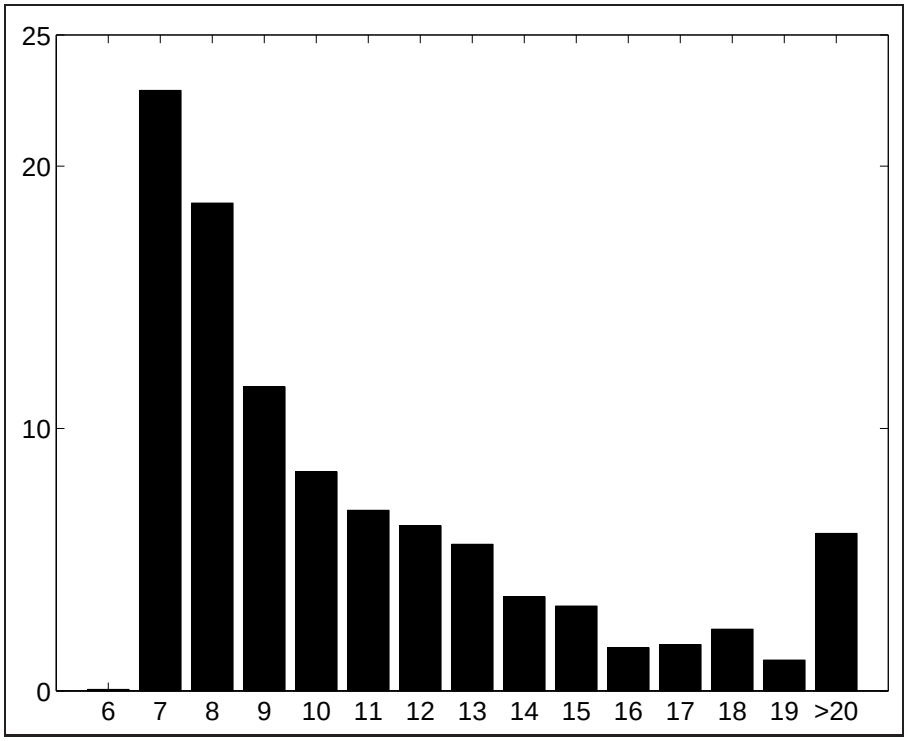

Figure 2. Percentages (y-axis) of artists having 6, 7, ..., 19, 20 to 29 albums (x-axis).

Methods

We compare two approaches based on different parametrisations of the data. Whereas Mel Frequency Cepstrum Coefficients (MFCCs) are a quite direct representation of the spectral information of a signal and therefore of the specific "sound" or "timbre" of a song, Fluctuation Patterns (FPs) are a more abstract kind of feature describing the amplitude modulation of the loudness per frequency band. It is our hypothesis that MFCCs are more prone to pick up production and mastering effects of a single album as well as the specific "sound" of an artist (voice, instrumentation, etc).

Mel Frequency Cepstrum Coefficients and Single Gaussians (G1)

We use the following approach to music similarity based on spectral similarity. For a given music collection of songs, it consists of the following steps:

1. for each song, compute MFCCs for short overlapping frames
2. train a single Gaussian (G1) to model each of the songs

3. compute a similarity matrix between all songs using the symmetrised Kullback-Leibler divergence between respective G1 models

The 30 seconds song excerpts in mp3-format are recomputed to 22050Hz mono audio signals. We divide the raw audio data into non-overlapping frames of short duration and use Mel Frequency Cepstrum Coefficients (MFCC) to represent the spectrum of each frame. MFCCs are a perceptually meaningful and spectrally smoothed representation of audio signals. MFCCs are now a standard technique for computation of spectral similarity in music analysis (see e.g. (5)). The frame size for computation of MFCCs for our experiments was $46.4ms$ (1024 samples). We used the first 25 MFCCs for all our experiments.

A single Gaussian (G1) with full covariance represents the MFCCs of each song (6). For two single Gaussians, $p(x) = \mathcal{N}(x; \mu_p, \Sigma_p)$ and $q(x) = \mathcal{N}(x; \mu_q, \Sigma_q)$, the closed form of the Kullback-Leibler divergence is defined as (11):

$$KL_N(p\|q) = \frac{1}{2}\left(\log\left(\frac{\det(\Sigma_p)}{\det(\Sigma_q)}\right) + Tr\left(\Sigma_p^{-1}\Sigma_q\right) + (\mu_p - \mu_q)'\,\Sigma_p^{-1}\,(\mu_q - \mu_p) - d\right) \tag{1}$$

where $Tr(M)$ denotes the trace of the matrix M, $Tr(M) = \Sigma_{i=1..n} m_{i,i}$. The divergence is symmetrised by computing:

$$KL_{sym} = \frac{KL_N(p\|q) + KL_N(q\|p)}{2} \tag{2}$$

Fluctuation Patterns and Euclidean Distance (FP)

Fluctuation Patterns (FP) (7) (9) describe the amplitude modulation of the loudness per frequency band and are based on ideas developed in (4). For a given music collection of songs, computation of music similarity based on FPs consists of the following steps:

1. for each song, compute a Fluctuation Pattern (FP)
2. compute a similarity matrix between all songs using the Euclidean distance of the FP patterns

Closely following the implementation outlined in (8), an FP is computed by: (i) cutting an MFCC spectrogram into three second segments, (ii) using an FFT to compute amplitude modulation frequencies of loudness (range $0-10Hz$) for each segment and frequency band, (iii) weighting the modulation frequencies based on a model of perceived fluctuation strength, (iv) applying filters to emphasise certain patterns and smooth the result. The resulting FP is a 12 (frequency bands according to 12 critical bands of the Bark scale (12)) times 30 (modulation frequencies, ranging from 0 to $10Hz$) matrix for each song. The distance between two FPs i and j is computed as the Euclidean distance:

$$D(FP^i, FP^j) = \sum_{k=1}^{12} \sum_{l=1}^{30} (FP^i_{k,l} - FP^j_{k,l})^2 \tag{3}$$

Results

Album/Artist Precision

For the full database $D(ALL)$

For every song in the database $D(ALL)$, we computed the first nearest neighbour for both methods G1 and FP. For method G1, the first nearest neighbour is the song with minimum Kullback Leibler divergence (Equation 2) to the query song. For method FP, the first nearest neighbour is the song with minimum Euclidean distance of the FP pattern (Equation 3) to the query song. We then computed the percentage of instances, where the first nearest neighbour is from the same album (1st AL) or from other albums by the same artist (1st AR) as the query song (see Table 2).

Method	1st AL	1st AR	AL prec	AR prec
G1	27.87	35.76	13.86	8.14
FP	2.24	26.85	0.90	1.63

Table 2. Percentage of first nearest neighbour from same album (1st AL), from other albums from same artist (1st AR), album and artist precision (AL prec, AR prec) for G1 and FP.

For method G1, 27.87% are from the same album and 35.76% from other albums by the same artist. On average, there are 13.46 songs on an album and 131.2 songs from one artist. Considered that there are always more than 250000 songs from other artists, it is quite astonishing that only in 36.37% a song from a different artist turns up as a first

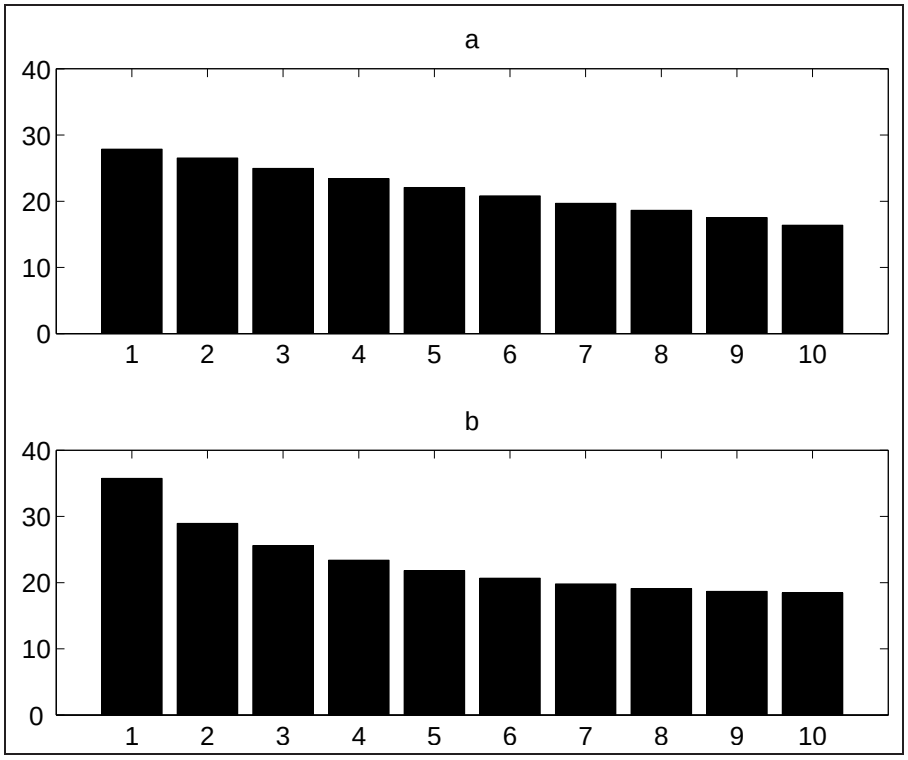

Figure 3. Percentage (y-axis) of $n = 1, \ldots, 10$ **nearest neighbours** *(x-axis) from same album (a), from other albums from same artist (b) for* **G1**.

nearest neighbour. For method FP, percentages are quite lower with only 2.24% from the same album and 26.85% from other albums by the same artist.

We also computed lists of $n = 1, \ldots, 10$ nearest neighbours for every song in the database $D(ALL)$ for both methods G1 and FP. We then computed the percentage of instances, where members of these lists of size $n = 1, \ldots, 10$ are from the same album or from other albums by the same artist as the query song (see Figure 3 for method G1 and Figure 4 for method FP). As can be seen, the degradation of percentages with growing list size for method G1 is quite graceful (see Figure 3): e.g. the percentage of instances where the five nearest neighbours are from the same album is at 22.07% compared to 27.87% for the first nearest neighbour. The percentage of instances where the five nearest neighbours are from other albums from the same artist is at 21.83% compared to 35.76% for the first nearest neighbour. There is a similar behaviour for method FP at a generally lower level (see Figure 4).

Next we computed the album and artist precision at n. Album precision at n (AL prec) is the percentage of songs from the album in a list of the n nearest neighbours, with n being equal to the number of other songs in the same album as the query song. Artist precision at n (AR prec) is the percentage of songs from the artist in a list of the n nearest neighbours, with n being equal to the number of other songs from the same artist as the

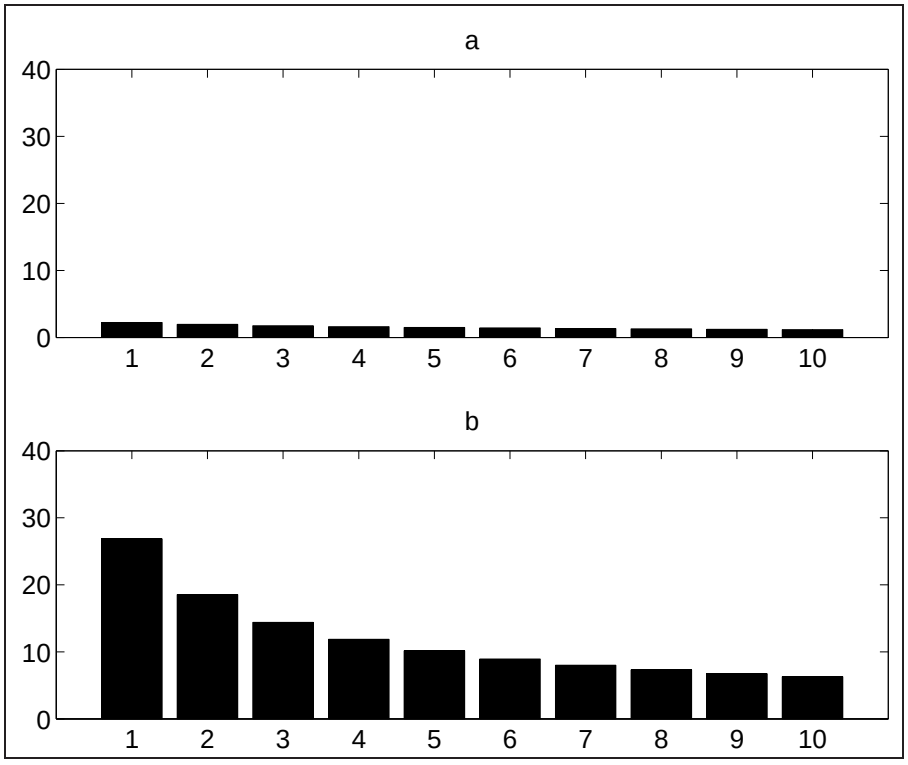

Figure 4. Percentage (y-axis) of $n = 1, \ldots, 10$ **nearest neighbours** *(x-axis) from same album (a), from other albums from same artist (b) for* **FP***.*

query song. For $D(ALL)$ and method G1, album precision is at 13.86% and artist precision at 8.14% (see Table 2). Precision values for method FP are very small.

To sum up, there is both an album and an artist effect in nearest neighbour based music recommendation for method G1. For this timbre based method, the album effect is even relatively bigger than the artist effect. For method FP, there is only a smaller artist effect but no album effect.

Influence of the size of the database

We repeated the experiments for all the subsets of the database as described in Section "Data". The results depicted in Figures 5, 6, 7 and 8 show mean values over 100 ($D(1/100)$), 20 ($D(1/20)$), 2 ($D(1/2)$) data sets or the respective single result for the full data set $D(ALL)$. Please note that in all these figures the x-axis is given in log-scale to better depict the large range of values of the different sizes of data sets (from 2543.98 for $D(1/100)$ to 254398.00 for $D(ALL)$). The percentage of the first nearest neighbour from the same album decreases from 38.91% for $D(1/100)$ to 27.87% for $D(ALL)$ for method G1 (Figure 5). There is a parallel decrease for the first nearest neighbour from other albums from the same artist for method G1 (Figure 5). A similar decrease at lower levels can be

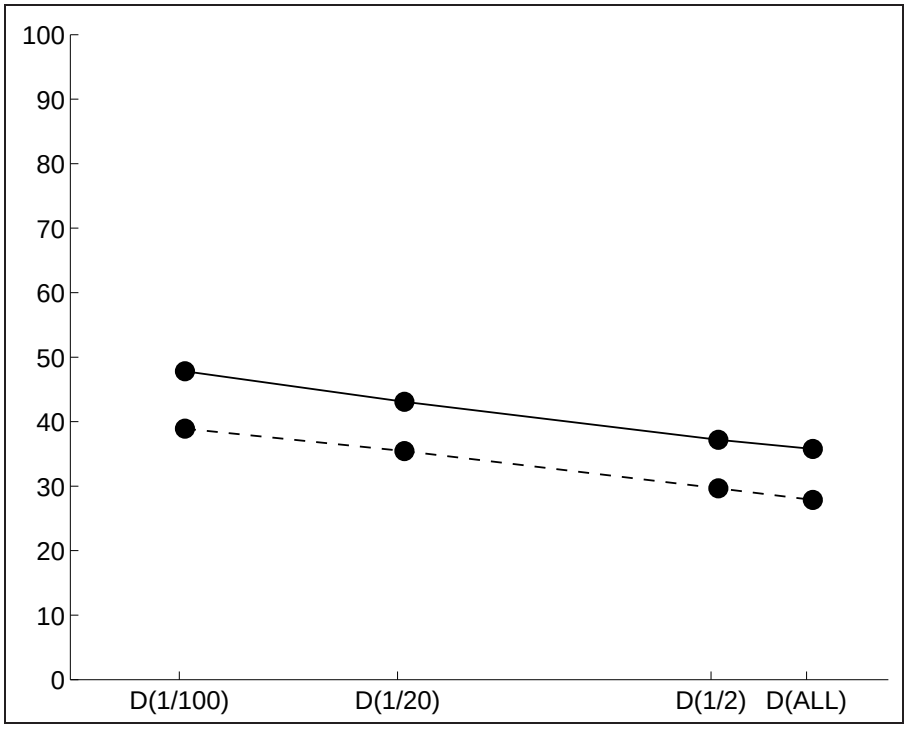

Figure 5. Percentage (y-axis) of **first nearest neighbour** *from same album (dashed line), from other albums from same artist (solid) for* **G1** *and different size of data set (x-axis, log-scale).*

seen for method FP (Figure 6). As the data sets get larger, the probability that songs from other artists are more similar to the query song than songs from the same album or artist, clearly seems to increase.

Album and artist precision also decrease with increasing size of data set. For method G1, artist precision drops from 35.99% for $D(1/100)$ to 8.14% for $D(ALL)$ even falling below album precision (Fig 7). For method FP, artist precision drops from 19.19% for $D(1/100)$ to 1.63% for $D(ALL)$ which is at the same low level as album precision (Figure 8).

To sum up, both first nearest neighbour rates and precision values are over estimated when smaller data sets are used.

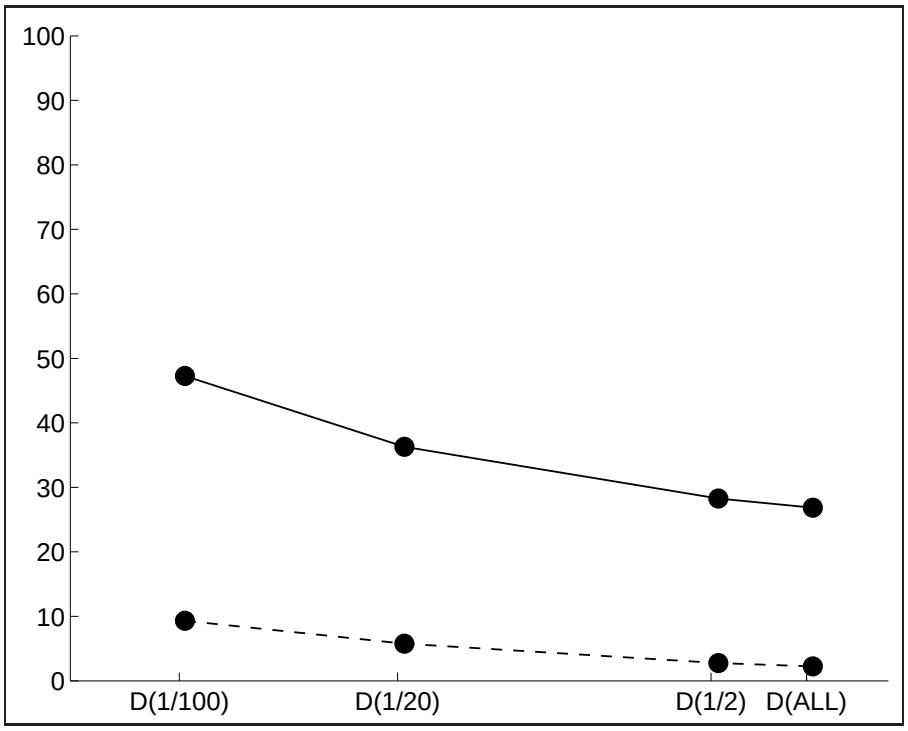

Figure 6. Percentage (y-axis) of **first nearest neighbour** *from same album (dashed line), from other albums from same artist (solid) for* **FP** *and different size of data set (x-axis, log-scale).*

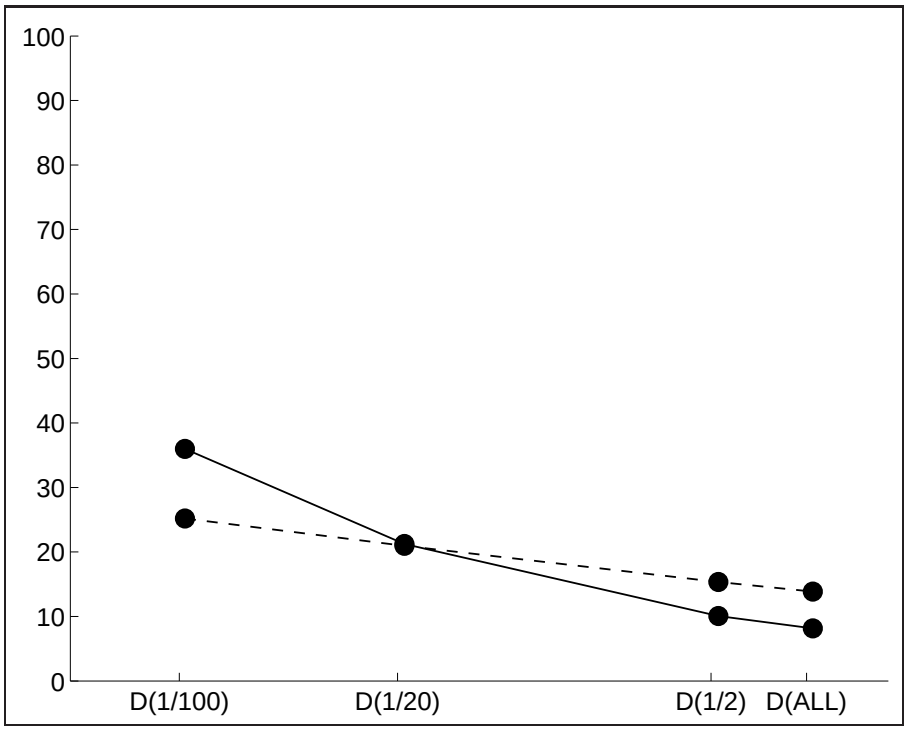

Figure 7. **Precision** *(y-axis) of album (dashed line) and artist (solid) for* **G1** *and different size of data set (x-axis, log-scale).*

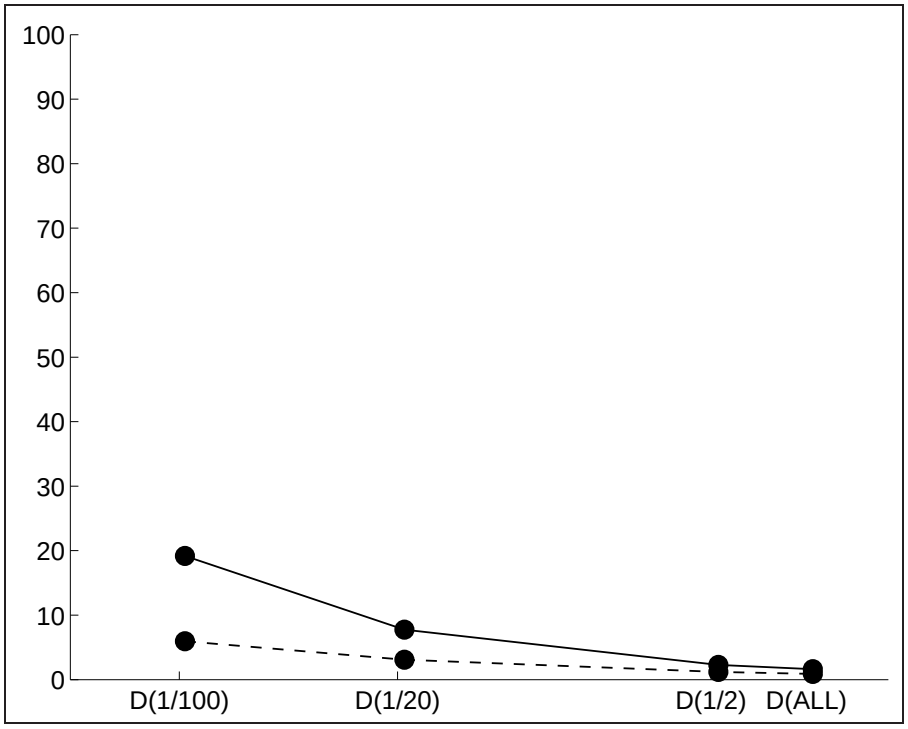

Figure 8. **Precision** *(y-axis) of album (dashed line) and artist (solid) for* **FP** *and different size of data set (x-axis, log-scale).*

Genre Classification

For the full database $D(ALL)$

We also did experiments on the influence of album and artist filters on genre classification performance. We used nearest neighbour classification as a classifier. For every song in the database $D(ALL)$, we computed the first nearest neighbour for both methods G1 and FP. For method G1, the first nearest neighbour is the song with minimum Kullback Leibler divergence (Equation 2) to the query song. For method FP, the first nearest neighbour is the song with minimum Euclidean distance of the FP pattern (Equation 3) to the query song. When using an album filter (ALF), all other songs from the same album as the query song were excluded from becoming the first nearest neighbour. When using an artist filter (ARF), all other songs from the same artist as the query song were excluded from becoming the first nearest neighbour. When using no filter (NOF), any song was allowed to become the first nearest neighbour. To estimate genre classification accuracy, the genre label of a query song s_{query} and its first nearest neighbour s_{nn} were compared. The accuracy is defined as:

$$acc(s_{query}, s_{nn}) = \frac{|g_{query} \cap g_{nn}|}{|g_{query} \cup g_{nn}|} \quad (4)$$

with g_{query} (g_{nn}) being a set of all genre labels for the query song (nearest neighbour song) and $|.|$ counting the number of members in a set. Therefore accuracy is defined as the number of shared genre labels divided by the set size of the union of sets g_{query} and g_{nn}. The latter is done to penalise nearest neighbour songs with high numbers of genre labels. The range of values for accuracy is between 0 and 1. The baseline accuracy achieved by always guessing the three most probable genres ("Rock", "Pop", "Alternative Rock", see Table 1) is 32.42%. We decided to use a number of three genres for this baseline accuracy because the majority of songs is labelled with three genres (see Figure 1). Average accuracy results for methods G1 and FP are given in Table 3. Without using any filter (NOF), G1 clearly outperforms FP (68.98% vs. 44.35%). Using an album filter (ALF) strongly degrades the performance of G1 down to 56.07%, but hardly impairs method FP. Using an artist filter (ARF) further degrades the performance of G1 but also of FP. The difference between G1 and FP is now much closer (35.98% vs. 28.96%). However, method G1 barely outperforms the baseline accuracy of 32.42% and method FP clearly falls below it.

Method	NOF	ALF	ARF
G1	68.98	56.07	35.98
FP	44.35	43.03	28.96

Table 3. Average accuracies for G1 and FP without (NOF) and with album filter (ALF) and artist filter (ARF).

To sum up, not using any filter yields very over-optimistic accuracy results. As a matter fact, results after artist filtering are very close or even below baseline accuracy. There is both an album and an artist filter effect for G1. There is only an album filter effect for FP. Using filters diminishes the differences in accuracies between methods G1 and FP, since filters have a bigger impact on G1 than FP.

Influence of the size of the database

We repeated the experiments for all the subsets of the database as described in Section "Data". The results depicted in Figures 9 and 10 show mean accuracy values over 100 ($D(1/100)$), 20 ($D(1/20)$), 2 ($D(1/2)$) data sets or the respective single result for the full data set $D(ALL)$. For both methods G1 and FP, the accuracy without using a filter (dotted lines in Figures 9 and 10) decreases with increasing size of data set. For G1, from 80.65% for $D(1/100)$ to 68.98% for $D(ALL)$. For FP, from 57.19% for $D(1/100)$ to 44.35% for $D(ALL)$. There is an almost parallel decrease in accuracy when using album filters (dashed lines in Figures 9 and 10). For both methods G1 and FP, the accuracy when using

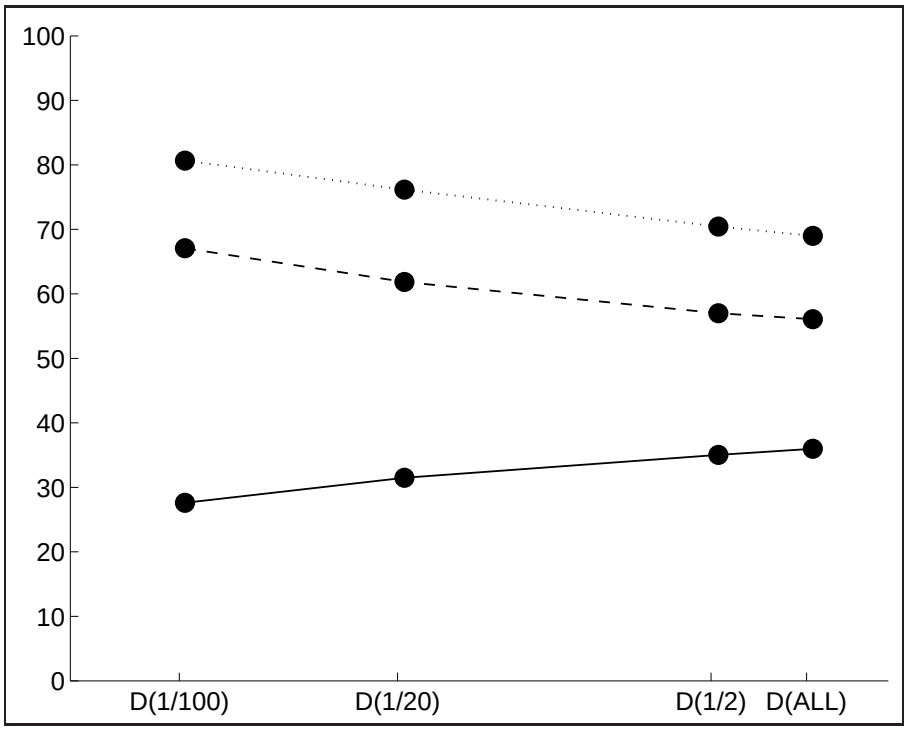

Figure 9. **Accuracy** *(y-axis) for no filter (dotted line), album filter (dashed line), artist filter (solid) for* **G1** *and different size of data set (x-axis, log-scale).*

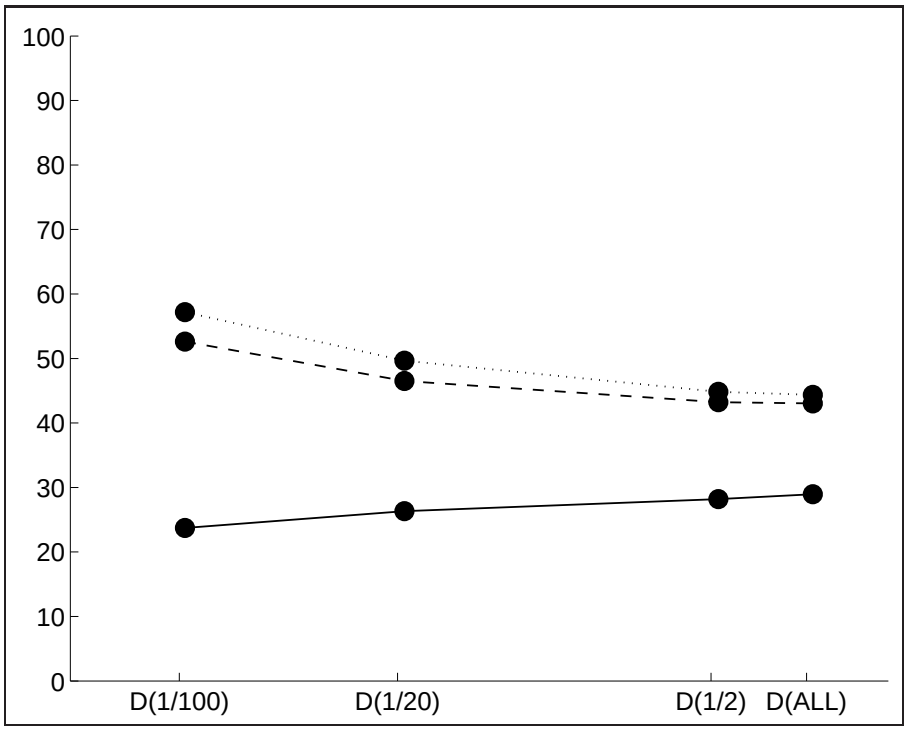

Figure 10. **Accuracy** *(y-axis) for no filter (dotted line), album filter (dashed line), artist filter (solid) for* **FP** *and different size of data set (x-axis, log-scale).*

an artist filter (solid lines in Figures 9 and 10) increases with increasing size of data set. For G1, from 27.60% for $D(1/100)$ to 35.98% for $D(ALL)$. For FP, from 23.72% for $D(1/100)$ to 28.96% for $D(ALL)$.

How can this contrary behaviour of decreasing accuracy for no filter and album filter versus increasing accuracy for artist filters be explained? Larger data sets allow for a larger choice of songs to become the first nearest neighbour. This larger choice of songs can come with the wrong or correct genre labels. If we use artist filters, this larger choice seems to make it more probable that a song with the correct genre label is first nearest neighbour. Otherwise we would not see the increase in accuracy. If we use no filter or only an album filter, the larger choice seems to interfere with the songs from the same artist still in the database. Songs from the larger choice sometimes end up being first nearest neighbour instead of a song from the same artist as the query song. Since most songs from an artist share the same labels, the larger choice in this case diminishes the accuracy.

To sum up, there clearly is an influence of the database size on accuracy performance. Small data sets are too pessimistic when artist filters are used. But they are over optimistic if no or only album filters are used.

Conclusion

There clearly is both an album and an artist effect in music recommendation even in very large databases. For the timbre based method G1, about one third of the first recommendations are from the same album and about another third from other albums from the same artist as the query song. Considering that every artist has multiple albums in the database and that an album contains only about 13 songs on average, the album effect is relatively bigger than the artist effect. This suggests that the direct representation of the spectral information is sensitive to production and mastering effects of individual albums. For method FP, there is only a smaller artist effect but no album effect. This suggests that the more abstract signal representation of the fluctuation patterns is not sensitive to production and mastering effects of individual albums. But it is still able to model the common musical sound of an artist across different albums. Please note that we have no way to know whether an artist is working together with the same recording studio or sound engineer for more than one album. Our experiments also show that album and artist effects in music recommendation are over estimated when smaller data sets are being used.

Since most research on artist filters so far concentrated on genre classification, we did large scale experiments on classification accuracy also. We corroborated earlier results that not using any filter yields very over-optimistic accuracy results. Using artist filters

even reduces results close to or even below baseline accuracy. As reported before, using artist filters also diminishes the differences in accuracies between methods that are effected distinctly by filtering. Additionally, there clearly is an influence of the data base size on accuracy performance.

As with all large scale performance studies, there remains the question as to how representative and universally valid our results are. We are convinced that our database is representative of music that is generally listened to and available in the Western hemisphere since it is a large and random subset of about 5 million songs from a popular web-shop. As to the methods employed, we chose one method that closely models the audio signal and one that extracts information on a somewhat higher level. It is our guess that other method's performance will be close to either of our methods depending on their level of closeness to the analysed audio signal. Systems using a Gaussian representation of spectral features together with the Kullback-Leibler divergence (like our method G1) regularly rank in the top places of the yearly Music Information Retrieval Evaluation Exchange (MIREX, http://www.music-ir.org/mirexwiki/, (1)), which is a community-based framework for the formal evaluation of Music Information Retrieval systems and algorithms. This shows that at least one of our chosen music recommendation methods belongs to the most successfull aproaches in music similarity. The choice of our methods was also influenced by considerations of computability. After all, 250000 song excerpts are a lot of data to analyse and both our methods can be implemented very efficiently. Using nearest neighbour methods for music recommendation seemed to be the obvious choice.

With audio based music recommendation maturing to the scale of the web, our work provides important insight into the behavior of music similarity for very large databases. Even with hundreds of thousands of songs, album and artist filtering remain an issue.

ACKNOWLEDGEMENTS: Parts of this work have been funded by the "Austrian National Science Foundation (FWF)" (Projects P21247 and L511-N15) and by the "Austrian Research Promotion Agency (FFG)" (Bridge-project 815474).

References

[1] Downie J.S.: The music information retrieval evaluation exchange (2005-2007): A window into music information retrieval research, Acoustical Science and Technology, vol. 29, no. 4, pp. 247-255, 2008.

[2] Flexer A.: Statistical Evaluation of Music Information Retrieval Experiments, Journal of New Music Research, Vol. 35, No. 2, pp.113-120, 2006.

[3] Flexer A.: A closer look on artist filters for musical genre classification, in Pro-

ceedings of the 8th International Conference on Music Information Retrieval (ISMIR'07), Vienna, Austria, 2007.

[4] Fruehwirt M., Rauber A.: Self-Organizing Maps for Content-Based Music Clustering, Proceedings of the Twelth Italian Workshop on Neural Nets, IIAS, 2001.

[5] Logan B.: Mel Frequency Cepstral Coefficients for Music Modeling, Proceedings of the International Symposium on Music Information Retrieval (ISMIR'00), Plymouth, Massachusetts, USA,2000.

[6] Mandel M.I., Ellis D.P.W.: Song-Level Features and Support Vector Machines for Music Classification, Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR'05), London, UK, 2005.

[7] Pampalk E.: Islands of Music: Analysis, Organization, and Visualization of Music Archives, MSc Thesis, Technical University of Vienna, 2001.

[8] Pampalk E.: Computational Models of Music Similarity and their Application to Music Information Retrieval, Vienna University of Technology, Austria, Doctoral Thesis, 2006.

[9] Pampalk E., Rauber A., Merkl D.: Content-based organization and visualization of music archives, Proceedings of the 10th ACM International Conference on Multimedia, Juan les Pins, France, pp. 570-579, 2002.

[10] Pampalk E., Flexer A., and Widmer G.: Improvements of Audio-Based Music Similarity and Genre Classification , Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR'05), London, UK, 2005.

[11] Penny W.D.: Kullback-Leibler Divergences of Normal, Gamma, Dirichlet and Wishart Densities, Wellcome Department of Cognitive Neurology, 2001.

[12] Zwicker E., Fastl H.: Psychoaccoustics, Facts and Models, Springer Series of Information Sciences, Volume 22, 2nd edition, 1999.