ACCESSING MUSIC COLLECTIONS VIA REPRESENTATIVE CLUSTER PROTOTYPES IN A HIERARCHICAL ORGANIZATION SCHEME

Markus Dopler Markus Schedl Tim Pohle Peter Knees
Department of Computational Perception
Johannes Kepler University
Linz, Austria

ABSTRACT

This paper addresses the issue of automatically organizing a possibly large music collection for intuitive access. We present an approach to *cluster tracks in a hierarchical manner* and to *automatically find representative pieces of music* for each cluster on each hierarchy level. To this end, audio signal-based features are complemented with features derived via Web content mining in a novel way. Automatic hierarchical clustering is performed using a variant of the *Self-Organizing Map*, which we further modified in order to *create playlists* containing similar tracks.
The proposed approaches for playlist generation on a hierarchically structured music collection and finding prototypical tracks for each cluster are then integrated into the *Traveller's Sound Player*, a mobile audio player application that organizes music in a playlist such that the distances between consecutive tracks are minimal. We extended this player to deal with the hierarchical nature of the playlists generated by the proposed structuring approach.
As for evaluation, we first assess the quality of the clustering method using the measure of entropy on a genre-annotated test set. Second, the goodness of the method to find prototypical tracks for each cluster is investigated in a user study.

1 INTRODUCTION AND MOTIVATION

Increased storage capabilities, high-speed Internet access, and novel techniques for music compression have tremendously increased the size of private as well as commercial music collections over the past few years. Due to the large number of audio tracks in private repositories, it is virtually impossible for users to keep track of every piece. Hence, elaborate methods for organizing large music collections become increasingly important. The enormous economic success of high-capacity mobile music players, such as Apple's "iPod", necessitates intelligent methods for automated structuring of music collections.
The paper at hand presents a possible solution to this problem. The approach proposed here employs an unsupervised hierarchical clustering algorithm, more precisely the *Growing Hierarchical Self-Organizing Map* (GHSOM) [3], on audio features extracted from the music collection. Each cluster of the resulting hierarchical organization is then labeled automatically with a typical track from the cluster to facilitate exploration. To this end, we elaborated an original approach that not only relies on audio features, but incorporates Web-based information on the prototypicality of artists to determine a representative track for each cluster. The main motivation for this is the bad quality of straightforward methods to find a representative for a set of data items. Indeed, choosing for example the music track whose audio feature representation is closest to the average feature vector of the cluster may give a mathematically sound prototype for the cluster. However, in preliminary experiments that employed this selection approach, we encountered many representatives that were either songs by barely known artists (even though the cluster contained very well known artists) or barely known tracks by popular artists. In any case, using such tracks as representative for a cluster seems counterintuitive to guide the user in exploring his music collection.
The main contributions of this work are the novel approach to *determine cluster prototypes*, based not only on the audio features, but also on Web mining techniques, in order to omit "average" cluster representatives that are unknown to most users. In addition, we reimplemented the GHSOM algorithm and modified it in order to *generate playlists*. Furthermore, a novel version of the Traveller's Sound Player application, which uses the cluster prototypes to facilitate navigating the collection, is proposed. The approaches to generate playlists and finding cluster prototypes are further evaluated thoroughly.
The remainder is structured as follows. Section 2 briefly discusses related work. In Section 3, the acquisition of feature data to describe artists and pieces of music is elaborated. Our approaches to hierarchically cluster a given music collection and creating playlists from these clusters can be found in Section 4, whereas finding representative prototypes for each cluster is addressed in Section 5. The experiments conducted to evaluate the proposed approaches are detailed in Section 6. Finally, Section 7 gives some remarks on the integration of the presented approaches in the Traveller's Sound Player application, and in Section 8, conclusions are drawn.

2 RELATED WORK

This paper was mainly motivated by the work done in [9], where Pohle et al. employ different heuristics to solve a Traveling Salesman Problem on a track-level audio similarity matrix. The resulting tour represents a playlist with minimized distances between consecutive pieces of music. The playlist generated for a complete music collection is then made accessible via a special music player. In [8], this approach is modified in that the audio similarity matrix is adapted according to artist similarity estimations, which are derived from $TF \times IDF$ features computed on artist-related Web pages. In contrast to [9, 8], we do not model a Traveling Salesman Problem to generate playlists. Instead we follow a recursive approach that builds upon the GHSOM algorithm to hierarchically break down the tracks of a music collection until every track is represented by a sole map unit. The playlist is then generated by successively visiting each such unit.

As we build our playlist generation approach upon a variant of the GHSOM, a hierarchical version of the Self-Organizing Map, this work is certainly also related to [3, 4], where the GHSOM is presented.

3 FEATURE EXTRACTION

As we combine features extracted from the audio signal with cultural features from the Web in order to determine prototypical pieces of music for each cluster, we perform the following two feature extraction stages.

3.1 Audio-based Features

First, we extract audio features using a well-established approach. For each piece of music, Mel Frequency Cepstral Coefficients (MFCCs) are extracted on short-time audio segments to obtain a coarse description of the spectral shape. Details on this procedure can be found, for example, in [2, 1, 7]. Gaussian Mixture Models (GMMs) are then used to aggregate the MFCCs of each individual piece of music. The similarity between two arbitrary pieces is calculated as the inverse of the distance between their GMMs. For our experiments, we used parameters similar to those proposed in [1] (19 MFCCs, 30 clusters per GMM). Performing this procedure for a collection of n music pieces eventually yields an $n \times n$ distance matrix.

3.2 Web-based Features

In order to estimate the prototypicality of each artist in the collection, and thus his or her suitability to serve as representative for a cluster, we make use of information found on artist-related Web pages. To this end, we follow the approach *backlink/forward link ratio with exorbitant popularity penalization* as proposed in [11, 12]. The basic assumption underlying this approach is that the probability that a popular artist is mentioned on a Web page about a rather unknown artist is higher than vice versa. This seems reasonable if one thinks, for example, of references of the form "Band X sounds like band Y.". Such phrases are more likely to occur on a Web page of a rather unknown band X, which points to a popular band Y than vice versa.

For each artist, we retrieve up to 150 Web pages returned by Google for queries of the form `"artist name" +music+review`. This query scheme was originally proposed in [13] for retrieving artist-related Web pages. Subsequently, the Web pages of each artist are analyzed for occurrences of all other artist names in the collection, and the respective document frequency $df_{a,B}$ (i.e., the number of Web pages in the set B of retrieved pages for artist b on which the name of artist a occur) is stored for all combinations of artists.

To estimate the prototypicality of an artist a for a specific set of tracks given by cluster u [1] , we compare the document frequencies $df_{a,B}$ and $df_{b,A}$ for all b's in the same cluster u as a. We count for how many of these b's the inequality $df_{a,B} \geq df_{b,A}$ holds. The higher this count, the more prototypical artist a. Performing this calculation for all artists of a specific cluster u, we obtain an artist prototypicality ranking $r^u(a)$. However, this ranking is usually quite distorted as artist names like "Genius", "Kiss", or "Bush" are always overrated. Addressing this issue, a penalization term that downranks artists whose prototypicality is extremely high for all clusters (which actually cannot be the case) is introduced. This eventually gives a corrected ranking function $r^u_w(a)$. For a more detailed elaboration on the prototypicality ranking function, please see [12].

4 ADAPTING THE GHSOM FOR PLAYLIST GENERATION

Structuring the music collection under consideration is performed using an unsupervised neural network algorithm that automatically organizes a given data set in a hierarchical manner. More precisely, we reimplemented in Java a hierarchical extension of the Self-Organizing Map, namely the *Growing Hierarchical Self-Organizing Map* (GHSOM), as proposed in [3]. The GHSOM has been integrated in the CoMIRVA framework [10].

The GHSOM extends the widely used Self-Organizing Map [6] in a way that during training, new map units are automatically inserted into the existing grid between the map unit whose data items show the highest deviation from the map unit's model vector and its most dissimilar neighboring unit. This is performed until the mean quantization error of all map units is reduced to a certain threshold τ_1. Hence, τ_1 controls the final size of the individual SOMs. A second threshold τ_2 determines the quantization error a particular map unit must exceed in order to reorganize its data items on

[1] In our case, the clusters are given by the map units of the Self-Organizing Map that we use to organize the collection (cf. Section 4).

a newly generated SOM at a deeper hierarchy level. A more detailed explanation of the GHSOM algorithm is given, for example, in [3].

In order to suit our need for playlist generation, we had to modify the original GHSOM algorithm in two regards. First, we redefined the neighborhood function used in the training process. In particular, we defined a *cyclic neighborhood relation* that extends its range over the map's borders, i.e., considers the last row/column of the map whenever a map unit in the first row/column is modified and vice versa. Second, each time a map unit u is expanded to a new SOM at a deeper hierarchy level, this new SOM is initialized using the model vectors of u's neighboring units.

These two modifications allow for a very simple playlist generation approach in that we just have to train a *1-dimensional GHSOM* on the input data and ensure the recursive expansion of map units until each track is represented by its sole unit. Traversing the resulting GHSOM tree in preorder while successively storing all tracks represented by the leave nodes eventually yields a playlist containing all pieces of the collection. As for training the GHSOM, we use the audio similarity matrix, whose creation is described in Subsection 3.1. For performance reasons, however, we apply *Principal Components Analysis* (PCA) [5] to reduce the dimensionality of the feature space to 30. [2]

5 FINDING CLUSTER PROTOTYPES

One of the main goals of this work is determining prototypical tracks that can serve as representatives to describe each cluster, i.e., map unit, of the GHSOM. However, preliminary experiments showed that relying solely on audio-based information to calculate a representative piece of music for each cluster yields unsatisfactory results, as already explained in Section 1. To alleviate this problem, we incorporate artist prototypicality information extracted as described in Subsection 3.2 into an audio-based prototypicality ranking. The resulting ranking function thus combines audio features of all tracks in the cluster under consideration with Web-based artist prototypicality information as detailed in the following.

Denoting the map unit under consideration as u, the ranking function $r^u(p)$ that assigns a prototypicality estimation to the representation in feature space of each [3] piece of music p by artist a is calculated as indicated in Equation 1, where $r_s^u(p)$ denotes the audio signal-based part of the ranking function (cf. Equation 2) and $r_w^u(a)$ is the corrected Web-based ranking function from Subsection 3.2. In the signal-based ranking function, $\overline{u}$ denotes the mean of the feature vectors represented by u, $|\cdot|$ is the Euclidean distance, and $norm(\cdot)$ is a normalization function that shifts the range to $[1, 2]$. Finally, each piece of music p represented by map unit u is assigned a prototypicality value, which is the higher, the more prototypical p is for u.

$$r^u(p) = r_s^u(p) \cdot r_w^u(a) \tag{1}$$

$$r_s^u(p) = norm\left(\frac{1}{1 + \ln(1 + |p - \overline{u}|)}\right) \tag{2}$$

6 EVALUATION

We conducted two sets of evaluation experiments. The first aimed at comparing our GHSOM-based playlist generation approach to the ones proposed in [9, 8]. In accordance with [9], the quality of the playlist was evaluated by calculating the genre entropy for a number of consecutive tracks. Second, we assessed the quality of our approach to detecting prototypical tracks via a user study.

For evaluation, we used the same collection as used in [8]. This collection contains 2,545 tracks by 103 artists, grouped in 13 genres. For details on the genre distribution, please consider [8].

6.1 Entropy of the Playlist

To evaluate the coherence of the playlists generated by our GHSOM-based approach, we compute the entropy of the genre distribution on sequences of the playlists. More precisely, using each piece of the collection as starting point, we count how many of n consecutive tracks belong to each genre. The result is then normalized and interpreted as a probability distribution, on which the Shannon entropy is calculated, according to Equation 3, where $\log_2 p(x) = 0$ if $p(x) = 0$.

$$H(x) = -\sum_x p(x) \log_2 p(x) \tag{3}$$

In Figure 1, the entropy values for $n = 2 \ldots 318$ are given (plot "audio ghsom"), averaged over the whole playlist, i.e., each track of the playlist is chosen once as the starting track for a sequence of length n. [4]

Comparing the entropy values given by the GHSOM approach to the ones presented in [8] reveals that the GHSOM approach performs very similar to the SOM-based approach followed in [8]. This is obviously no surprise, but shows that the automated splitting of clusters by the GHSOM does not harm the entropy of the generated playlists, while at the same time offers the additional benefit of automatically structuring the created playlists in a hierarchical manner. Like those created by the recursive SOM approach in [8], the GHSOM-based playlists show quite good long-term entropy values (for large values of n), but rather poor values for short playlists.

[2] 30 dimensions seemed appropriate for our test collection of 2,545 pieces of music. Employing the approach on larger collections, most probably requires increasing this number.

[3] More precisely, "each" refers to those pieces of music that are represented by the map unit u under consideration.

[4] 318 tracks correspond to an angular extent of 45 degrees on the circular track selector in the Traveller's Sound Player application.

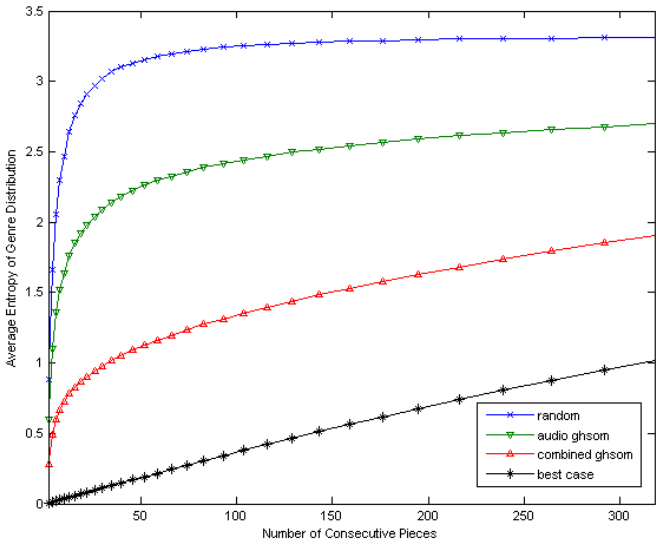

Figure 1. Genre entropy values for n consecutive tracks of the GHSOM-based playlist.

We also experimented with incorporating Web-based features directly into the playlist generation algorithm, as proposed in [8]. This improved on the one hand the entropy values (cf. plot "combined ghsom" in Figure 1). On the other hand, taking a critical look at the resulting playlists showed that using Web-based artist features as proposed in [8] yields "too perfect" playlists in that often one and the same artist contributes 20 or more consecutive tracks. Hence, this approach may be suited well for creating very consistent, but sometimes boring, playlists.

6.2 Quality of the Prototype Estimation

To evaluate the quality of the cluster representatives as determined by the approach proposed in Section 5, we performed a user study. The evaluation sets for the user study were created by training a 1-dimensional GHSOM with an initial number of 5 rows and gradient initialization. The growing threshold τ_1 was set to 0.5, the expansion threshold τ_2 to 0.1. We performed sequential training with a training length of 5 epochs. This setting resulted in a GHSOM with 11 clusters on the first level. [5]

We had 16 participants in the study, each of which evaluated 11 sets of tracks, consisting of 10 tracks each. 10 out of these 11 sets were constructed as follows. The most prototypical track of each cluster, according to our approach, was added to the corresponding set. The remaining 9 tracks were selected by drawing a random sample from their respective cluster. The 11^{th} set (for cluster 6) in contrast contained exactly the same pieces of music for all participants. [6]. The reason for this different creation method for cluster 6 is that we also wanted to take a qualitative look at some of the results, which would have been barely feasible for a large number of differing track sets, each of which is evaluated by only one user. We followed this rather complicated procedure of track set generation in order to raise the total number of evaluated sets. This procedure yielded 161 different track sets (16 participants · 10 sets + 1 invariant set for all participants). Each participant was instructed to select one track from each set that should be best suited to represent the corresponding set. We offered the participants the artist and song names as well as the audio files in case they were uncertain or unfamiliar with artist or track. In total, we received 176 prototype selections, which were analyzed as follows.

To obtain a general performance measure, we first calculated the overall concordance between the prototypes found by our automated approach and those indicated by the participants. Table 1 gives these overall concordance values over the 176 evaluated track sets. We assessed the concordances on different levels. The first row depicts the exact concordance of the tracks, i.e., in order to be considered concordant, the track given by our approach must be the same as the one selected by the user. This definition of concordance is weakened for rows two and three as our approach and the user judgment must, in this case, agree only on the artist or on the genre, respectively.

At first glance, these results may not seem very promising. However, taking the difficult nature of the user task into account (choosing between 10 probably quite similar tracks, many of which are by the same artists), it must be regarded a success that the concordance values considerably exceeded the baseline. The difficulty of the task is also reflected by the relatively high mean genre entropy of 1.95 over all track sets. This number is obviously higher than the entropy value given in Figure 1 for $n = 10$ because of the randomized selection process involved in the creation of the track sets.

Taking a closer look at the individual track sets, we often encountered the problem that the sets contained approximately equal numbers of tracks from two or three different genres. To analyze the influence this may have had on the evaluation results, Table 2 shows, for each group of sets (created from clusters 1 to 11 of the GHSOM), the absolute number of users that agreed with the prototype found by our approach (column "hits") and the mean genre entropy of the respective track set. Note that the entries are ordered by their average entropy. The baseline is 1.6 as each group was evaluated by 16 participants; a random guesser would have yielded a hit rate of 10% in the long run. From Table 2, we can see that high entropy values in fact tend to correspond to low hit rates and vice versa. The low hit rate of 2 for the group with the lowest entropy, however, can be explained by the fact

[5] Level 1 is the first distinctive level of the GHSOM as level 0 contains all data items.

[6] Cluster 6 also contained the prototype and a random sample of 9 other tracks, but this sample remained constant for all participants

level	baseline	concordance
track	10.00	17.61
artist	15.17	26.14
genre	35.34	50.57

Table 1. Overall concordance of the prototype finding algorithm and the user judgments, in percent.

that this group subsumed music by different, but all famous, metal bands. Furthermore, in most track sets of this group, 2 or 3 tracks by "Cannibal Corpse" were included, one of which was also rated the prototype by our approach. But the others were usually no less famous tracks by this band.

The identical composition of cluster 6 for all participants allowed us to take a qualitative look at the ranking results on the level of individual tracks. Table 3 summarizes these results. In this table, the evaluated tracks of cluster 6 are displayed in decreasing order of their prototypicality as determined by our approach. The leftmost column gives the number of times the respective track was selected as cluster representative by a participant. As for the composition of cluster 6, it mainly contained punk rock and folk rock songs, but also a metal song by "Pantera", which somehow also fits into this cluster, and a real outlier by the German electronic music producer "Scooter". From Table 3, it can be seen that about 56% (9 out of 16) of the user hits account for the top 3 rankings by our approach. On the other hand, also the lowest ranked two tracks by "Bad Religion" were chosen by 25% of the users. This is no surprise as this band may be seen as equally prototypical for the punk rock genre as the top ranked band "Offspring". The remaining 3 user hits were given to the tracks by "Subway to Sally". Since the randomly drawn sample included 3 tracks of this band in the evaluation set, it seems that some users considered the set best represented by a track by this artist. None of the participants, however, selected one of the outliers by "Pantera" or "Scooter" as representative.

To summarize the results, even though they certainly leave enough room for improvement, most users at least roughly agreed with the prototypicality ratings given by the proposed approach. Moreover, taking into account the especially difficult task the participants in the user study had to perform and the quite heavy mixtures of different genres in the track sets, the results are surprisingly good.

7 A NOVEL VERSION OF THE TRAVELLER'S SOUND PLAYER

The basic idea of the *Traveller's Sound Player* (TSP) presented in [9] is to arrange the tracks in a music collection around a wheel that serves as track selector. The authors of [9] aim at performing this arrangement in a way such that consecutive tracks are maximally similar. To this end, a large circular playlist is created by solving a Traveling Salesman Problem on an audio similarity matrix. A drawback of the original version of the TSP is, however, that it does not guide the user in finding a particular style of music he or she is interested in. Indeed, the user has to explore different regions of the playlist by randomly selecting different angular positions with the wheel. It is also very hard to exactly position the wheel at a specific track as thousands of tracks are placed around one wheel, which corresponds to an angular representation in the magnitude of tenths of one degree for each track.

Addressing these shortcomings, we propose to integrate the prototype information found by our GHSOM-based approach presented in Section 5 into the TSP. To this end, we elaborated an extension to the TSP that allows for hierarchically exploring the music collection by means of the cluster prototypes. More precisely, the novel user interface maps the structure of the collection, as given by the trained GHSOM, to the wheel. It thus reflects the hierarchical organization in that the user is first only presented the prototypes of the first level clusters. He or she can then either opt for playing the representative track or for descending to a lower hierarchy level. On each level, the user is presented a playlist containing the most prototypical tracks of all clusters (be them either single map units or other GHSOMs at a deeper level) that make up the GHSOM at the current level.

group	hits	entropy
4	2	0.9609
5	7	1.3385
6	**4**	**1.6855**
3	4	1.7516
2	2	2.0189
7	5	2.1068
8	2	2.1785
1	3	2.2763
0	2	2.2772
9	0	2.4343
10	0	2.5372

Table 2. Number of correctly found prototypes and mean entropy values for each of the 11 playlists.

hits	ranking	track name
4	0	offspring - hypodermic
2	1	offspring - leave it behind
3	2	blink 182 - don't leave me
0	3	pantera - strength beyond strength
0	4	scooter - back in the u.k.
2	5	subway to sally - mephisto
1	6	subway to sally - ohne liebe
0	7	subway to sally - böses erwachen
3	8	bad religion - let them eat war
1	9	bad religion - markovian process

Table 3. Detailed results of the user study for group 6, i.e., the group containing identical tracks for all participants.

8 CONCLUSIONS AND FUTURE WORK

We presented an approach to automatically cluster the tracks in a music collection in a hierarchical fashion with the aim of generating playlists. Each track is represented by similarities calculated on MFCC features, which are used as input to the clustering algorithm. To this end, we reimplemented and modified the Growing Hierarchical Self-Organizing Map (GHSOM). We further proposed an approach to determine, for each of the clusters given by the GHSOM, the most representative track. This is achieved by combining the MFCC-based audio features with artist-related information extracted from the Web. We finally integrated the two approaches into a music player software called the "Traveller's Sound Player", for which we elaborated an extension to deal with the hierarchical data.

We carried out evaluation experiments in two ways. First, we aimed at assessing the coherence of the playlists generated by the GHSOM approach by measuring their genre entropy. We compared these entropy values to those given by other playlist generation approaches and found that they largely correspond to a similar approach based on standard SOMs. In addition, our approach also yields a hierarchically organization of the collection and determines a representative for each cluster, while at the same time the good long-term entropy values of the (GH)SOM-based playlist generation are retained. Second, we conducted a user study to assess the quality of the found cluster prototypes. Even though the results are still improvable, we could exceed the baseline considerably. Especially when taking into account the challenging setup of the user study, the results are promising.

As for future work, we are thinking of integrating the artist distribution of the clusters into the ranking function, as the user study showed that users tend to prefer representative tracks by artists who occur often in the cluster under consideration. Similar considerations would probably also be suited to filter out outliers. Furthermore, we have to experiment with different user interfaces for hierarchically accessing music collections as the integration of the cluster representatives into the Traveller's Sound Player is currently done in a quite experimental fashion. Our special concern in this context will be the usability of the user interface for mobile devices.

9 ACKNOWLEDGMENTS

This research is supported by the Austrian Fonds zur Förderung der Wissenschaftlichen Forschung (FWF) under project numbers L112-N04 and L511-N15.

10 REFERENCES

[1] J.-J. Aucouturier and F. Pachet. Improving Timbre Similarity: How High is the Sky? *Journal of Negative Results in Speech and Audio Sciences*, 1(1), 2004.

[2] J.-J. Aucouturier, F. Pachet, and M. Sandler. "The Way It Sounds": Timbre Models for Analysis and Retrieval of Music Signals. *IEEE Transactions on Multimedia*, 7(6):1028–1035, 2005.

[3] M. Dittenbach, D. Merkl, and A. Rauber. The Growing Hierarchical Self-Organizing Map. In *Proc. of the Int'l Joint Conference on Neural Networks*, Como, Italy, 2000. IEEE Computer Society.

[4] M. Dittenbach, A. Rauber, and D. Merkl. Uncovering Hierarchical Structure in Data Using the Growing Hierarchical Self-Organizing Map. *Neurocomputing*, 48(1–4):199–216, 2002.

[5] H. Hotelling. Analysis of a Complex of Statistical Variables Into Principal Components. *Journal of Educational Psychology*, 24:417–441 and 498–520, 1933.

[6] T. Kohonen. *Self-Organizing Maps*, volume 30 of *Springer Series in Information Sciences*. Springer, Berlin, Germany, 3rd edition, 2001.

[7] B. Logan. Mel Frequency Cepstral Coefficients for Music Modeling. In *Proc. of the Int'l Symposium on Music Information Retrieval*, Plymouth, MA, USA, 2000.

[8] T. Pohle, P. Knees, M. Schedl, E. Pampalk, and G. Widmer. "Reinventing the Wheel": A Novel Approach to Music Player Interfaces. *IEEE Transactions on Multimedia*, 9:567–575, 2007.

[9] T. Pohle, E. Pampalk, and G. Widmer. Generating Similarity-based Playlists Using Traveling Salesman Algorithms. In *Proc. of the 8th Int'l Conference on Digital Audio Effects*, Madrid, Spain, 2005.

[10] M. Schedl, P. Knees, K. Seyerlehner, and T. Pohle. The CoMIRVA Toolkit for Visualizing Music-Related Data. In *Proc. of the 9th Eurographics/IEEE VGTC Symposium on Visualization*, Norrköping, Sweden, 2007.

[11] M. Schedl, P. Knees, and G. Widmer. Improving Prototypical Artist Detection by Penalizing Exorbitant Popularity. In *Proc. of the 3rd Int'l Symposium on Computer Music Modeling and Retrieval*, Pisa, Italy, 2005.

[12] M. Schedl, P. Knees, and G. Widmer. Investigating Web-Based Approaches to Revealing Prototypical Music Artists in Genre Taxonomies. In *Proc. of the 1st IEEE Int'l Conference on Digital Information Management*, Bangalore, India, 2006.

[13] B. Whitman and S. Lawrence. Inferring Descriptions and Similarity for Music from Community Metadata. In *Proc. of the 2002 Int'l Computer Music Conference*, Göteborg, Sweden, 2002.