

# Music Recommendation and Query-by-Content Using Self-Organizing Maps

Kyle B. Dickerson and Dan Ventura  
Computer Science Department  
Brigham Young University  
kyle\_dickerson@byu.edu, ventura@cs.byu.edu

**Abstract**—The ever-increasing density of computer storage devices has allowed the average user to store enormous quantities of multimedia content, and a large amount of this content is usually music. Current search techniques for musical content rely on meta-data tags which describe artist, album, year, genre, etc. Query-by-content systems allow users to search based upon the acoustical content of the songs. Recent systems have mainly depended upon textual representations of the queries and targets in order to apply common string-matching algorithms. However, these methods lose much of the information content of the song and limit the ways in which a user may search. We have created a music recommendation system that uses Self-Organizing Maps to find similarities between songs while preserving more of the original acoustical content. We build on the design of the recommendation system to create a musical query-by-content system. We discuss the weaknesses of the naive solution and then implement a quasi-supervised design and discuss some preliminary results.

## I. INTRODUCTION

The ability to purchase music in digital formats has caused a dramatic increase in the music collections of even casual computer users. Many personal libraries contain thousands of songs which the user needs to search through when looking for a particular song. Current search techniques typically rely on meta-data tags which describe artist, album, year, genre, or similar information. These tags must be created by a human and attached to each file—an error-prone process which is, at best, inconvenient.

Much work has been done to create systems which try to automatically tag a song with genre information [1]. Having accurate and automatically generated meta-data is helpful, but only if the user can remember the information stored in the tags. If, however, the user can only remember the tune of the song it is necessary to search by content rather than by meta-data. Systems that perform this type of search, which rely on information retrieved from audio files, are generally referred to as Music Information Retrieval (MIR) systems. Unfortunately, no system yet exists that searches audio by content and which is accurate, fast, robust, and intuitive.

Any MIR system will require a method for determining the similarity of songs. In fact, the system is heavily dependent on this distance function. Many current systems first transcribe the audio content to a text representation and then use common string-matching techniques as the distance function [2], [3], [4], [5], [6]. This process, however, is difficult to perform

accurately and reduces the content-rich music to a simple text string.

Instead, one could in principle extract various acoustic features from the audio using signal processing techniques with the distance function dependent upon which musical features are used. Determining which features to extract is a difficult problem. Whether or not a set of features is useful depends upon the context in which they will be used [7], [8], [9], [10], [11], [12], [13].

Once a good feature set is found, it is still necessary to determine a suitable distance function. The choice of distance function has also been heavily studied, resulting in varying levels of success [14], [15], [16], [17], [18]. Rather than attempt to design a specific distance function, we will use a Self-Organizing Map (SOM) to create a lower dimensional space, allowing us to use simpler distance metrics.

Any set of features can be used to train a SOM. SOMs create a new  $n$ -dimensional space (usually two) from any higher dimensionality, creating a map, while preserving as much similarity among training data as possible. A single SOM trained on song data can be used to perform music recommendation based on similarity. We present a basic recommendation system and use it to drive the development of a query-by-content system. We then present the preliminary results of our work on the query-by-content system.

## II. SELF-ORGANIZING MAPS

The Self-Organizing Map is an unsupervised learning algorithm which creates an  $n$ -dimensional space (usually two) while attempting to preserve as much of the intrinsic similarity in the training data as possible [19]. The algorithm begins by initializing a grid (map) of random feature vectors. These feature vectors are the same size as the data feature vectors. This grid may be considered to wrap around both horizontally and vertically, creating a toroid, to prevent unusual edge effects. For each training datum, the closest matching grid location is found, and a neighborhood around the matching location is updated to become more like that datum. Over time the size of the neighborhood shrinks and the influence of the update decreases (see Figure 1). In effect, these neighborhood alterations create smooth interpolations between data points across the map, a desirable property which allows us to train on a subset of the available data and still get a useful map.

```

 $V \leftarrow m^2$  real-valued vectors of length  $n$ 
Arrange  $V$  onto an  $m \times m$  grid
Choose  $\alpha, j, k \in (0, 1)$ 
Choose  $\rho \in (1, m)$ 
while NOT DONE do
  for training datum  $\bar{x} \in X$  do
     $\bar{v} \leftarrow \operatorname{argmin}_{\bar{v} \in V} \text{vector\_distance}(\bar{v}, \bar{x})$ 
     $v_i \leftarrow \alpha x_i + (1 - \alpha) v_i$ 
    for  $\bar{u}$  in neighborhood( $\bar{v}, \rho$ ) do
       $\alpha_u \leftarrow \alpha \times \frac{\rho - \text{grid\_distance}(\bar{v}, \bar{u})}{\rho}$ 
       $u_i \leftarrow \alpha_u x_i + (1 - \alpha_u) u_i$ 
    end for
  end for
   $\alpha \leftarrow k\alpha$ 
   $\rho \leftarrow j\rho$ 
end while

```

Fig. 1. *SOM algorithm pseudocode.*  $D$  is the set of all training data.  $\rho$  is the radius used in determining neighborhoods.  $\alpha$  is the weight given to the training datum when updating the vectors on the grid. The vector and grid distance functions can be any metric—we have used Euclidean distance for both. Typical parameter values:  $\alpha = 0.1$ ,  $\rho = \frac{m}{2}$ ,  $j$  and  $k$  are linear decay functions

SOMs have already been used successfully in MIR systems. One of the first such systems was presented by Feiten and Günzel [20]. Harford [21] uses a SOM to perform melody retrieval. Dittenbach, Merkl, and Rauber [22] introduce a growing hierarchical SOM which Rauber, Pampalk, and Merkl [23] use to create a musical archive based upon sound similarity. As far as we have been able to determine, no work has been done in applying SOMs to a musical query-by-content system. However, an image query-by-content system was created by Laaksonen, Koskela and Oja [24]. In this system the user does not input a free-form query but rather selects images from presented sets as the system locates the area of the SOM the user is interested in.

Our work will use a SOM to power a musical query-by-content system. We present the current state-of-the-art in musical query-by-content systems in the next section before we discuss our work.

### III. MUSICAL QUERY-BY-CONTENT SYSTEMS

The musical query-by-content field has almost exclusively been comprised of systems relying on monophonic queries which then use a form of melody extraction to represent the query textually before applying common string matching algorithms to find a match in a stored set of text representations of the target songs (Figure 2). One early system utilizing melody extraction uses only pitch change to represent the queries and songs [2]. Another system, presented by Kosugi *et al.* [3], uses beats instead of notes and relies on a MIDI format for stored songs. A number of similar string-matching based systems have been presented by Pauws [4], Raju, Sundaram, and Rao [5], and Birmingham, Dannenberg, and Pardo [6].

The textual representation of the query is a considerable constraint in all of the above systems. Music is a very rich

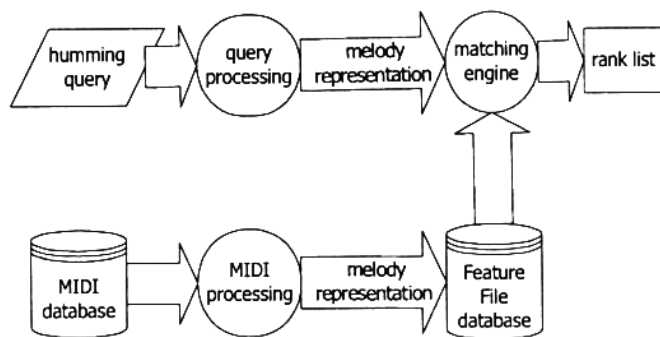


Fig. 2. *Typical organization of current musical query-by-content systems, borrowed from [25].* Queries and targets are first converted to text representations, and then compared using common string-matching techniques.

medium—songs often contain concurrent parts and a person may remember any single part while forgetting the others. Having a strong, unique bass with a vocal track accompanied by instrumentals is not uncommon, but by reducing the song to a single text representation much of that information is lost. Perhaps a person can remember how the bass sounded, but not the vocal or instrumentals. Current systems would be unable to help them find the correct song. There is one system which allows more comprehensive searching by extracting features directly from the MP3 encoding format which are segmented into a set of “phases” to which queries are matched [26]. This work, however, is limited to only songs in MP3 format.

We propose to use a SOM to power a musical query-by-content system, thereby allowing us to retain more of the original audio content. By retaining more content in the target songs, we hope to allow users to search using a broader range of query types—for example, humming, whistling, or singing.

### IV. MUSIC RECOMMENDATION USING A SELF-ORGANIZING MAP

We have created a proof-of-concept, SOM-based music recommendation system similar to those mentioned above. Our system generates a 128x128 map using a randomly chosen 25-second segment from 20% of the 881 songs available in our personal music library (Figure 3). Before creating the SOM the audio is preprocessed to extract feature vectors. The selection of features is an important aspect of any machine learning algorithm, and when using audio signals, the task is even more difficult because we must also decide what size windows to extract the features from. For simplicity, rather than attempt to determine a most effective set of features and parameters, we choose features and parameters that are common in many MIR systems. The features we use include the power spectrum, strongest beat, beat sum, Mel-Frequency Cepstral Coefficients, and Linear Predictive Coding. The features were extracted for every 5-second window, with a 50% overlap of segments. This preprocessing work therefore excludes any system we create from being a real-time system without many hours of prior computation. Our goal, however, is to show that a query-by-content system can be built using SOMs and not, necessarily,

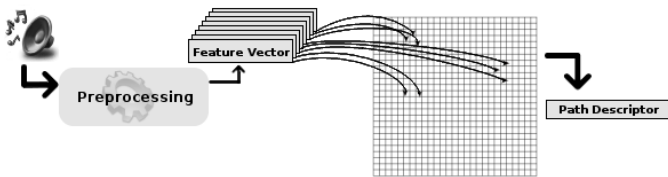


Fig. 3. System design for music recommendation. During training, 9 consecutive feature vectors (representing 25 seconds of audio) are taken from 176 (20%) of the preprocessed songs (1584 unique feature vectors). The feature vectors are used as the training data for the SOM algorithm. After training has completed, each song's complete set of feature vectors is mapped into the SOM creating unique path descriptors.

- 1) Haydn - Divertimento No. 1 in B flat major
- 2) Mozart - The Magic Flute: Aria of the Queen of Night
- 3) Bach - Keyboard Concerto No. 4 - Larghetto
- 4) Pachelbel - Canon in D major
- 5) Mozart - The Magic Flute: Zorastro's Aria

Fig. 4. Top 5 recommendations for Bach - Brandenburg Concerto No. 2, Allegro Moderato. All songs are orchestra pieces representing good matches to the seed song.

that it will be fast enough (yet) to use in real-time.

Once the SOM is trained, each of the 881 songs is completely mapped into it from start to end (one location for each 5 seconds of audio), creating a path within the map for each song. These paths act as unique descriptors for each song and are stored for later reference. We calculate the distance between two songs as the average Euclidean distance between the unique path descriptors. More advanced techniques could be used; however our proof-of-concept results are fairly good even using this simple algorithm. We assume that a more thorough algorithm that attempts segmentation and alignment matching would be more effective.

To test the recommendation system, we select various songs and subjectively decide if the top five recommendations seem similar. The system performs well with some types of songs, such as classical and rock, while doing poorly with others, such as dance club type music. For example, when we choose *Bach - Brandenburg Concerto No. 2, Allegro Moderato* we receive the results shown in Figure 4. Each of these songs has a common classical, orchestral sound, so, in our opinion, the recommender system does a fairly good job. The pieces from *The Magic Flute*, however, contain vocals while the others do not.

When we use a more contemporary piece as the seed song the system still performs well. Figure 5 shows the recommendations for the song *Dashboard Confessional - Hands Down*. These songs feature vocals over a strong rock sound. *Be Like That*, despite having the best rank, is probably the least like the others, having a slower tempo and softer feel.

A real test of our recommendation system is if two different versions of the same song appear similar to each other. We have recordings of the song *You Raise Me Up* as performed by the group Celtic Woman as well as by the solo artist Josh Groban. The recommendations for these songs are a little

- 1) 3 Doors Down - Be Like That
- 2) Evanescence - Bring Me To Life
- 3) Linkin Park - Pushing Me Away
- 4) Matchbox Twenty - All I Need
- 5) Lifehouse - First Time

Fig. 5. Top 5 recommendations for Dashboard Confessional - Hands Down. Recommended songs reflect the prominent vocals and the strong rock background of the seed.

- 1) Harry Potter - Double Trouble
- 2) Phantom of the Opera - Angel of Music
- 3) Trans-Siberian Orchestra - God Rest Ye Merry Gentlemen
- 4) Mannheim Steamroller - Enchanted Forest IV
- 5) Mozart - The Marriage of Figaro: Duetting

Fig. 6. Top 5 recommendations for Celtic Woman - You Raise Me Up. These songs contain soft music mainly featuring vocals over light instrumentals. Compare with Figure 7.

bit more unusual and may not be considered very helpful as recommendations. They are, however, consistent between the two pieces. So while the recommendations may be less intuitive, they are at least not arbitrary and do contain strong similarities to the seed songs. The top five recommendations for the Celtic Woman and Josh Groban versions are presented in Figures 6 and 7, respectively.

The top three songs in each list all feature strong vocals over light instrumentals, which is consistent with the seed songs. The songs from Mannheim Steamroller, however, are not really what we would call music. They come from one of the artist's Halloween CDs and are simply spooky sounds to be played as sound effects. It is interesting that neither list contains the other song, yet they contain the same set of similar songs. It is not until the sixtieth song in the Josh Groban list that the Celtic Woman version appears. The Josh Groban version does not appear within the first 100 results of the Celtic Woman list. This effect may be due to the two songs having different length introductions and one may be trailing the other on their paths, a problem which could be overcome by an alignment mechanism. A contributing factor is that the area of the map in which the songs' paths mainly land is the same area in which several other songs are mapped to as well. This failure to differentiate well could potentially be addressed by selecting different features in the preprocessing stage.

- 1) Harry Potter - Double Trouble
- 2) Phantom of the Opera - Angel of Music
- 3) Trans-Siberian Orchestra - God Rest Ye Merry Gentlemen
- 4) Mannheim Steamroller - Enchanted Forest IV
- 5) Mannheim Steamroller - Enchanted Forest III

Fig. 7. Top 5 recommendations for Josh Groban - You Raise Me Up. Compare with Figure 6.

- 1) Bizet - *Carmen Suite No. 1*
- 2) Harry Potter - *Double Trouble*
- 3) *Phantom of the Opera* - *Angel of Music*
- 4) Creedence Clearwater Revival - *Susie Q*
- 5) Schubert - *Symphony No. 5 in B flat major*

Fig. 8. Top 5 recommendations for Sean Paul - *We Be Burnin'*. These are unusual recommendations which are unlike the seed song. This is probably a result of the training set not containing songs representative of the seed song.

Our system fails completely when we seed the search with the song *Sean Paul - We Be Burnin'*, a popular dance club type of music. The top five results for this song are presented in Figure 8.

In our opinion none of these songs are perceptually similar to the seed song. *Carmen Suite* is a traditional march piece, which does preserve the quick tempo and strong beat of the seed song, but contains no vocals. *Double Trouble* and *Angel of Music* both contain little instrumental and do not have a strong beat. *Susie Q* is closer to the seed song with a strong rock beat, instrumentals, and vocals. The Schubert piece, however, is a classical symphony song mainly featuring the violin section.

We believe the reason for the poor recommendations is that none of the songs from this album were selected in the 20% used to train the map; therefore the map failed to develop a neighborhood representative of this style of music. We did notice, however, that for ten of the eighteen songs on the album the song *Double Trouble* appeared within the top 3 results. This suggests that we may be seeing a similar effect as above—a crowded space requiring more discrimination. If information regarding each song’s genre is known, then a stratified selection approach could be used to drive the creation of representative neighborhoods to prevent this problem.

## V. MUSICAL QUERY-BY-CONTENT USING SELF-ORGANIZING MAPS

Because SOMs preserve similarity information while performing dimensionality reduction, they are well suited to powering a query-by-content system. In designing this system we hope to avoid imposing a single query style upon the user. We would like our system to be robust enough to allow the user to query in any way they choose, such as singing, humming, whistling, etc.

### A. Naïve Single-SOM Querying

The simplest and most obvious solution for querying a song library would be to train a SOM on the songs and then treat queries the same way, as if performing music recommendation as above. This is easily done by using a query as the seed for our recommendation system (Figure 9). However, this approach results in a degenerate solution in which the queries all map to a single area of the SOM which represents songs that have no instrumentation. This is an expected result because the query will not be similar to any of the songs based upon the musical content (other than the one theme expressed in the query). The queries tested were whistling queries and as

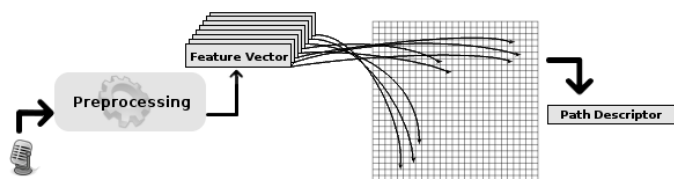


Fig. 9. Naïve single-SOM querying design. This system works identically to the recommender system. The source, however, is provided by the user rather than from music files (see Figure 3). Queries are preprocessed and converted into feature vectors. The feature vectors are then mapped into the SOM to create a path descriptor, which is used to calculate the similarity to stored path descriptors of songs in the library.

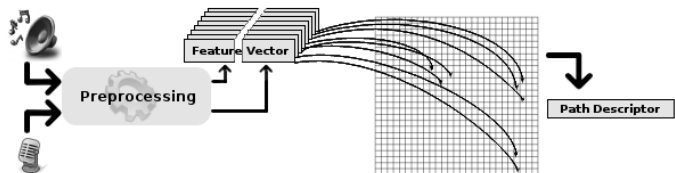


Fig. 10. Quasi-supervised SOM design. Sample queries are matched with their target songs. These pairs are individually preprocessed and the resulting pair of feature vectors are concatenated to create a single feature vector. This feature vector is used to train the SOM. After training, the songs in the library are mapped into the SOM using the first half of the stored feature vectors to create path descriptors. Queries are similarly mapped, using the second half of the stored feature vectors. The path descriptors of targets and queries are thus directly comparable while still explicitly modeling the query style.

such were unlikely to match themes occurring from guitars, singing, pianos, etc. The quasi-supervised approach allows us to compare songs and queries on a single SOM while avoiding this problem.

### B. Quasi-Supervised SOM Training

Traditionally SOM training is unsupervised—the feature vectors themselves determine the resulting map. In quasi-supervised training, we extend the feature vectors of the target songs with the feature vectors of matching sample queries (Figure 10). This will create a single interpolated map linking queries and targets. As before, the entire song library is then mapped into the SOM using only the set of features representing songs. Queries are then be matched to songs by mapping the queries into the SOM using the set of feature vectors representing queries and the location paths can be directly compared just as we did in section IV for the recommendation system.

### C. Quasi-Supervised SOM Preliminary Results

Our query collection system plays an audio clip to the user and then, while listening to the clip again, the user records a query to match that clip. We allow the user to listen to the sample while querying in order to elicit more accurate queries, helping to minimize errors in the dataset.

Our preliminary dataset consists of 71 matched song-clip and query pairs. Each clip is 20 seconds. We trained a small 32x32 SOM with 20% of the dataset (14 matched pairs). We altered the preprocessing to extract only the MFCC values for every 10ms. This resulted in 55,874 unique feature vectors in

	Training Set		Test Set	
	Expected Random	Trained SOM	Expected Random	Trained SOM
Percent Correct	7.1	14.3	1.8	1.8
Percent Within Top 5	35.7	50.0	8.8	10.5
Percent Within Top 10	71.4	78.6	17.5	22.8
Average Position of Match	7.5	6.1	29	25.9

TABLE I

*Preliminary Results.* OUR PRELIMINARY TEST USING A VERY SIMPLE 32X32 SOM YIELDED ENCOURAGING RESULTS. THE RESULTS ARE BETTER THAN THE EXPECTED VALUES OF RANDOMLY GENERATED SONG ORDERINGS.

our training set. The results are promising, but clearly indicate that adjustment of parameters, including feature selection, is going to be important. To evaluate the performance of our SOM we used four metrics: percent correct, percent in top five results, percent in top ten results, and average position of correct match in the results list. If the possible set of results were simply randomly ordered then we would expect to see  $(\frac{1}{NumItems} \times 100)\%$  correct,  $(\frac{5}{NumItems} \times 100)\%$  within the top 5,  $(\frac{10}{NumItems} \times 100)\%$  within the top ten, and an average position of  $\frac{NumItems+1}{2}$ . Our results are summarized in Table I.

The training set resulted in 14.3% correct (2/14), 50.0% within the top five results (7/14), 78.6% within the top ten results (11/14), and the average position of the matching result was 6.1. The testing set resulted in 1.8% correct (1/57), 10.5% within the top five results (6/57), 22.8% within the top ten results (13/57) and the average position of the matching result was 25.9. These results are not yet convincing; however, they are encouraging. This was a preliminary test, with a very small SOM, but it does show that there is improvement over a random ordering of songs. The trained SOM is completely saturated; that is, every location has multiple feature vectors mapping to it. A larger SOM should perform better by allowing further separation of feature vectors, improving the discriminating power of the SOM and increasing the distances between dissimilar points in the map.

## VI. CONCLUSION

Musical query-by-content systems help users search through large song libraries to find specific songs based on the acoustic content rather than on meta-data such as artist, title, genre, and lyrics. We have created a simple music recommendation system using Self-Organizing Maps to show that SOMs can be used successfully in musical applications. Based on our subjective analysis the recommendation system we present as proof-of-concept produces acceptable results. Many improvements could be made to the initial algorithm to increase the accuracy. Important improvements would be to implement an alignment mechanism and adjust the features extracted during preprocessing.

Our goal is to use SOMs as the basis of a query-by-content system. Current query-by-content systems mainly rely upon creating a textual representation of songs and queries

and using string-matching algorithms to find matches. In order to preserve more of the latent acoustic information and allow various query types we use a SOM to power such a system. The preliminary results of the SOM-based approach are encouraging, though much work is still to be done. To help drive our future work we are currently obtaining sample queries matched to song clips from a set of several individuals. With this data we will be able to more extensively test our design as we continue to tune the parameters and improve feature selection.

## REFERENCES

- [1] Nicolas Scaringella, Giorgio Zoia, and Daniel Mlynek, "Automatic genre classification of music content: a survey," *Signal Processing Magazine, IEEE*, vol. 23, no. 2, pp. 133–141, 2006.
- [2] Asif Ghias, Jonathan Logan, David Chamberlin, and Brian C. Smith, "Query by humming: Musical information retrieval in an audio database," in *Proceedings of ACM International Conference on Multimedia*, New York, NY, USA, 1995, pp. 231–236, ACM.
- [3] Naoko Kosugi, Yuichi Nishihara, Tetsuo Sakata, Masashi Yamamuro, and Kazuhiko Kushima, "A practical query-by-humming system for a large music database," in *Proceedings of ACM International Conference on Multimedia*, New York, NY, USA, 2000, pp. 333–342, ACM Press.
- [4] Steffen Pauws, "Cubyhum: A fully operational "query by humming" system.," in *Proceedings of International Conference on Music Information Retrieval*, 2002.
- [5] M. Anand Raju, Bharat Sundaram, and Preeti Rao, "Tansen: A query-by-humming based music retrieval system," in *Proceedings of Indian Institute of Technology National Conference on Communications*, 2003.
- [6] William Birmingham, Roger Dannenberg, and Bryan Pardo, "Query by humming with the vocalsearch system," *Communications of the ACM*, vol. 49, no. 8, pp. 49–52, 2006.
- [7] Kurt Jacobson, "A multifaceted approach to music similarity," in *Proceedings of International Conference on Music Information Retrieval*, October 2006, pp. 346–348.
- [8] P. Ahrendt, A. Meng, and J. Larsen, "Decision time horizon for music genre classification using short time features," in *Proceedings of European Signal Processing Conference*, Vienna, Austria, sep 2004.
- [9] Cory McKay and Ichiro Fujinaga, "Automatic music classification and the importance of instrument identification," in *Proceedings of Conference on Interdisciplinary Musicology*, Montreal, Canada, March 2005.
- [10] Tim Pohle, Elias Pampalk, and Gerhard Widmer, "Evaluation of frequently used audio features for classification of music into perceptual categories," in *Proceedings of International Workshop on Content-Based Multimedia Indexing*, Riga, Latvia, June 2005.
- [11] A. Meng and J. Shawe-Taylor, "An investigation of feature models for music genre classification using the support vector classifier," in *Proceedings of International Conference on Music Information Retrieval*, sep 2005, pp. 604–609, Final version : 6 pages instead of original 8 due to poster presentation.
- [12] George Tzanetakis and Perry Cook, "Musical genre classification of audio signals," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 5, pp. 293–302, July 2002.
- [13] E. Allamanche, Jürgen Herre, Oliver Hellmuth, T. Kastner, and C. Ertel, "A multiple feature model for musical similarity retrieval," in *Proceedings of International Conference on Music Information Retrieval*, 2003.
- [14] J. Foote, "A similarity measure for automatic audio classification," in *Proceedings of AAAI Symposium on Intelligent Integration and Use of Text, Image, Video and Audio Corpora*. American Association for Artificial Intelligence, March 1997.
- [15] Jouni Paulus and Anssi Klapuri, "Measuring the similarity of rhythmic patterns," in *Proceedings of International Conference on Music Information Retrieval*, Michael Fingerhut, Ed., Paris, France, Oct 2002, pp. 150–156.
- [16] B. Logan and A. Salomon, "A music similarity function based on signal analysis," in *Proceedings of IEEE International Conference on Multimedia and Expo*, 2001.
- [17] J. Foote, M. Cooper, and U. Nam, "Audio retrieval by rhythmic similarity," in *Proceedings of International Conference on Music Information Retrieval*, 2002.

- [18] Kris West, Stephen Cox, and Paul Lamere, "Incorporating machine-learning into music similarity estimation," in *Proceedings of ACM Workshop on Audio and Music Computing Multimedia*, New York, NY, USA, 2006, pp. 89–96, ACM.
- [19] T. Kohonen, *Self-Organizing Maps*, Springer, 2001.
- [20] B. Feiten and S. Günzel, "Automatic indexing of a sound database using self-organizing neural nets," *Computer Music Journal*, vol. 18, no. 3, pp. 53–65, 1994.
- [21] S. Harford, "Automatic segmentation, learning and retrieval of melodies using a self-organizing neural network," in *Proceedings of International Conference on Music Information Retrieval*, 2003.
- [22] Michael Dittenbach, Dieter Merkl, and Adreas Rauber, "The growing hierarchical self-organizing map," in *Proceedings of International Joint Conference on Neural Networks*, Washington, DC, USA, 2000, vol. 6, p. 6015, IEEE Computer Society.
- [23] A. Rauber, E. Pampalk, and D. Merkl, "Using psycho-acoustic models and self-organizing maps to create a hierarchical structuring of music by sound similarities," in *Proceedings of International Conference on Music Information Retrieval*, 2002.
- [24] Jorma Laaksonen, Markus Koskela, and Erkki Oja, "Content-based image retrieval using self-organizing maps," in *Proceedings of International Conference on Visual Information and Information Systems*, London, UK, 1999, pp. 541–548, Springer-Verlag.
- [25] Lie Lu, Hong You, and Hong-Jiang Zhang, "A new approach to query by humming in music retrieval," in *Proceedings of IEEE International Conference on Multimedia and Expo*, 2001.
- [26] Chih-Chin Liu and Po-Jun Tsai, "Content-based retrieval of mp3 music objects," in *Proceedings of International Conference on Information and Knowledge Management*, New York, NY, USA, 2001, pp. 506–511, ACM.