



Audio Engineering Society Convention Paper

Presented at the 123rd Convention
2007 October 5–8 New York, NY

The papers at this Convention have been selected on the basis of a submitted abstract and extended precis that have been peer reviewed by at least two qualified anonymous reviewers. This convention paper has been reproduced from the author's advance manuscript, without editing, corrections, or consideration by the Review Board. The AES takes no responsibility for the contents. Additional papers may be obtained by sending request and remittance to Audio Engineering Society, 60 East 42nd Street, New York, New York 10165-2520, USA; also see www.aes.org. All rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Journal of the Audio Engineering Society.

Using Audio Classifiers as a Mechanism for Content Based Song Similarity

Benjamin Fields¹, Michael Casey¹

¹*Goldsmiths College, University of London, New Cross, London, SE14 6NW, United Kingdom*

Correspondence should be addressed to Benjamin Fields (b.fields@ieee.org)

ABSTRACT

As collections of digital music become larger and more widespread, there is a growing need for assistance in a user's navigation and interaction with a collection and with the individual members of that collection. Examining song to song relationships and similarities, based upon content derived features, provides a useful tool to do so. This paper looks into a means of modifying a song classification algorithm to provide song to song similarity information. In order to evaluate the effectiveness of this method, the similarity data is used to cluster the songs into genres, the result of which is compared against the original genre sorting algorithm.

1. OVERVIEW

Through various technical innovations of recent years, people have greater access to larger collections of music than ever before. This overwhelming availability of music, and indeed, media in general, while providing a listener potentially greater freedom of access and availability of material, is not without problems. How does a user efficiently navigate a large collection? Can the discovery of new material of interest to a given listener be facilitated based on known likes and dislikes? These questions and others demand a means of quantifying the space between songs, e.g. their similarity, so that we may better un-

derstand and take advantage of inherent structures and relationships across a collection of digital music.

This paper details such a system for use in quantifying the distance between pairs of songs along multiple dimensions based upon features extracted from signal. This system takes the genre classification algorithm described in [1], changes the training portions of the algorithm to have a narrow single song focus and exhaustively expands the testing phase of the algorithm. This is achieved by changing the scope with which the models used to define a genre in the training process are selected. In its original use, a sufficient number of songs are selected to

both thoroughly and exhaustively define the genre by example. However in changing the classification-by-example system for similarity a much more narrow focus is necessary, leading to every song being used in isolation to create a descriptive model of that song. This leads to the generation of N models being created to describe a digital music collection containing N songs. In what was originally the testing phase of the audio classifier, the distance between songs is determined. This is done by comparing every song in a collection with every model that was generated previously. Returning to the collection of N songs with N models, a total $(N \times N)$ comparisons are made to generate an exhaustive song similarity matrix. Since the genre classification scheme being used exploits multiple extracted features to create multiple models, the resulting similarity matrix reflects this multidimensional space, in manner similar to [2] and discussed in [3].

In order to determine the effectiveness of the proposed audio based similarity measure, a means of evaluation is required. The obvious mechanism for this is through the use of human listeners, however, this would require a prohibitively long time commitment from anyone participating in such a study. As a means to automate this, nearest neighbor clustering as a genre classifier is proposed in [4, pp.61-62] as an efficient means of evaluating music similarity measures. This method is particularly useful in this case as these results are directly compared to the original result of genre classification decisions made by the algorithm in [1].

2. SONG CLASSIFICATION SYSTEM

The classification system that is the base for the similarity analysis use multiple-decision chains to predict genre of a song [1]. In an effort to make a system robust enough to maintain or improve its accuracy with large and musically diverse datasets[5], more features are extracted while making a classification decision. These features are each used independently to determine a genre. After each feature chain has made a decision about a song's genre, a final genre is selected based on the derived prevalent genre. If a clear genre is not determined, a confidence measure produced by each feature chain is used to assign the final genre of the test song.

There are three independent feature-decision chains in this sorting system. The first is based heavily on

the SoundModelDS system of categorization[6]. The second is a feature chain based around an extraction of MFCC from the music audio file. The third is based on the beat and tempo related information as extracted by AudioBpmD (this feature extraction method is from the MPEG-7 standard). Each of these three chains outputs two pieces of data: the genre estimate and the confidence measure for that estimate. The confidence measure is generated for each of the feature-decision chains that use Hidden Markov Models (HMM) based on the likelihood of the selected statistical model. A similar process is used for the tempo feature chain, based upon distance from the average tempo. This decision process can be seen in Figure 1.

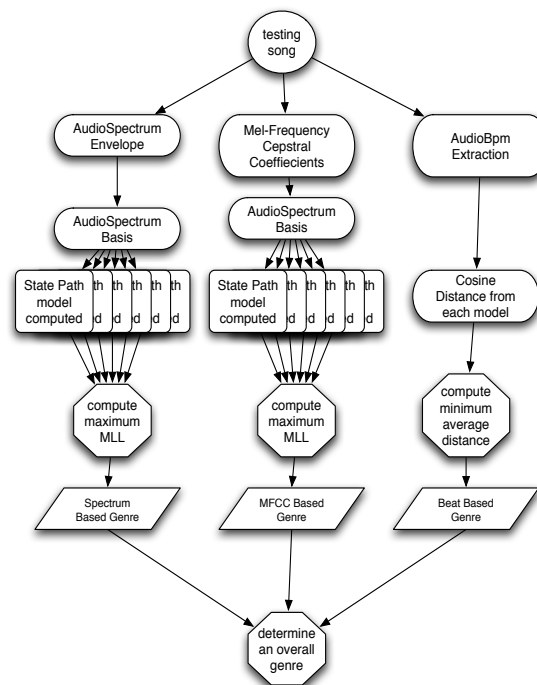


Fig. 1: An overview of all three testing decision chains and how they connect with each other.

2.1. Spectral Envelope Chain

This chain, as is true with all three feature chains, has two distinct processes. The first is a training by example process and the second is a testing process. The training examples can be as broad or focused as the particular application requires though it is

important that whatever the methodology used for training sample selection, the entirety of the given genre is covered so as to minimize false negatives. For each of these songs the AudioSpectrumEnvelope MPEG-7 descriptor is obtained. This is a logarithmic representation of the frequency spectrum on multiples of the octave and is the basic feature vector used in this entire process. Then the output of all of these AudioSpectrumEnvelope descriptor instantiations is passed into the AudioSpectrumBasis descriptor, see Figure 2. This descriptor is a wrapper for a group of basis functions that are used to project the AudioSpectrumEnvelope descriptors onto a lower dimensionality to facilitate classification. This descriptor is generated through a matrix multiplication of the AudioSpectrumEnvelope and the matrix produced by the basis functions. The output retains the maximum energy of the feature vectors, while reducing its dimensionality, helping to alleviate the dimensionality curse[7]. The output from this point contains the feature vectors that are used to create the HMM that will be used to make the determination as to which of the available genres our test information will fit. The HMM used in this implementation is informative to the MPEG-7 specification for the SoundModel descriptor scheme [8] and uses the standard solutions to the 3 critical HMM problems as can be found in [9]. It uses the Baum-Welch re-estimation algorithm to optimize likelihood.

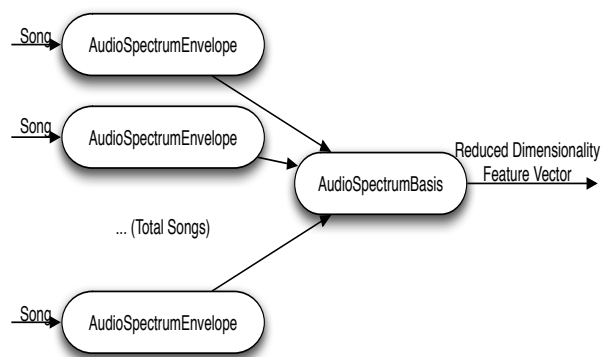


Fig. 2: The dimensional reduction process with the AudioSpectrumEnvelope feature

The likelihood is defined as $P(x|\omega_k)$, where ω_k is the

given genre class and x is the extracted feature data. This probability is found as a reduced proportion of Bayes' rule (Equation 1).

$$P(\omega_k|x) \propto P(x|\omega_k) \quad (1)$$

This reduction from Bayes' rule to Equation 1 is achieved by taking both $P(x)$ and $P(\omega_k)$ to be equal to 1. The maximum value of $P(\omega_k|x)$ is found by estimating $P(x|\omega_k)$ through the iterative use of the Viterbi algorithm[10] with a test song's extracted feature against a genre class' HMM to find the maximum likelihood of a given path within a HMM.

2.2. The MFCC Chain

In many ways the MFCC chain is similar to the AudioSpectrumEnvelopeD based chain. The only major difference is that rather than using spectral envelopes describing the signal, a matrix of MFCC are extracted. After all the training files have their MFCC matrices extracted, the stacked matrix is sent to AudioSpectrumProjectionD and from there to the AudioSpectrumBasisD. This follows the same signal flow as the SoundModelDS. As seen in the SoundModelDS, the output of AudioSpectrumBasisD is used to create and train an HMM. This training process can be seen in Figure 1. The HMM is then used by the testing process, along with the song to be tested, in the same testing process seen in the ASE Chain. The only difference being that the data, both in the genre class model and the test song, are derived through the extracted matrices of MFCC.

2.3. The Beat Chain

The third genre decision chain is based around beats per minute and other related information produced by the AudioBpmD descriptor. The creation of this model is a statistically simpler process than the model creation of the two prior chains. The model is composed of three $2 \times N$ matrices, where N is equal to the number of songs used in each model. These matrices store the values of each of the scalar values that are produced as output by the MPEG-7 v2 AudioBpmD descriptor[8], BPM, correlation and reliability. The correlation and reliability are both byproducts of the filterbank into combfilter methodology employed by the tempo detection algorithm [11]. The first row contains values calculated at the beginning of each audio file; the second row contains values calculated from the midsection of each audio file. These three matrices are then

stored as the beat model for a given genre. The testing phase of the beat chain begins with the extraction of the six scalars using the AudioBpmD. Each pair of scalars (BPM, correlation and reliability) can then be thought of as a vector in a two dimensional space. Similarly, each pair of scalars in the model matrices can be considered in the same way. Then a distance measure is taken between the test vector and each vector in the corresponding training matrix. A cosine distance measure is used here over simple Euclidean distance as the cosine distance has been found to yield better results in music similarity tasks [12]. Once the distance measures have been taken they are normalized and averaged together to yield an overall score of the test song against the genre model. This score is taken for each genre model and the minimum score (smallest average distance) across all the models is taken to be the genre from the perspective of the beat chain. This process puts equal emphasis on the tempo itself, as well as the reliability and correlation of the tempo. This accounts for the diverse range of tempo across different genres. Where some genres may see large difference in tempo, these same genres may show a high degree of independence and correlation in the reliability of that tempo or the correlation measure of the tempo.

2.4. Confidence Measure

The confidence measure, as defined in Equation 2, is used as a means to measure how strongly the chosen genre class matches the test song.

$$C = 100 \frac{\lambda_g}{\sum_1^n \lambda_i} \quad (2)$$

Where λ_g is the selected genre's normalized MaximumLogLikelihood and λ_i is genre i 's normalized MaximumLogLikelihood. In both of these cases the normalization occurs by Equation 3.

$$\lambda_i = l_i - \min([l_1, \dots, l_n]) \quad (3)$$

Where l_i is the MaximumLogLikelihood that a song fits in the HMM representing genre class i . This is useful in the event that there is no agreement across the three feature chains in selecting an overall genre class for a given test song [1].

3. MEASURING DISSIMILARITY

In order for the the system described in section 2 to be used in measuring pairwise song relationships(e.g.

song similarity/dissimilarity), modifications must be made to the training and testing procedures. The primary change is in adjusting the scope of each model generated. When using this system for it's intended purpose of genre classification, each model represents a genre class and as such is built using examples which are members of the genre. However, when using a supervised system such as this one to examine song to song relationships, what each generated model represents narrows from a genre class to a single song. As such the examples used in training will narrow to only the song being described by the model. It is important to note that as the coverage of each model decreases, an increase is seen in the number of models necessary to describe the entire dataset. As each model approaches describing a single song, the number of models necessary to describe the entire dataset approaches the number of songs in said dataset (Equation 4).

$$M \propto \frac{1}{N_m} \quad (4)$$

Where M is the number of models necessary to describe every song in the dataset and N_m is the mean number of songs covered by each model.

Once a model has been created for each song in the dataset, the entire dataset is then used again as the test against these generated models, using the testing process described in 2.1 - 2.3. Every test song will generate a confidence score (Equation 2)along each feature-decision chain. The inverse of each of these scores gives a pairwise song distance for that particular feature-decision chain. These are used to build three dissimilarity matrices, one for each chain. These three matrices can be used either in isolation or as a combined multi-feature dissimilarity measure.

However, it is in the testing phase where issues of scale arise. For a dataset containing N songs, this testing algorithm must be run $3N^2$ times in order to fill the $3 N \times N$ dissimilarity matrices.

4. EXPERIMENT

Testing of this dissimilarity measure is a straightforward process, though careful consideration is necessary in compiling a dataset for this purpose as it is not difficult to inadvertently skew results through

various problems most notably overfitting[4, 13, p.34].

4.1. Dataset

The dataset used is a combination of both the testing and training sets used in [1], with 37 songs from an additional genre of *Dance* added, bring the total number of songs in the dataset to 436, across 11 prescribed genres labels. These *Dance* songs were gathered in the same manner as the songs used in [1], being the most popular songs downloaded from the iTunes Music Store in the genre on March 13, 2006.

4.2. Scaling and Clustering

In an effort to provide an objective means of evaluating this similarity mechanism, a form of k-means clustering is used on the data to attempt to explore groupings that present in the data, based on the relationships established via the three feature-decision chains. Further, this provides a quantitative means to evaluate the dissimilarity measure, by evaluating the number of clusters equal to the number of prescribed genre classes in the set. A confusion matrix of the resulting clusters against the genre class labels provides a simple numeric measure for how closely correlated the means clusters are to the genre labels.

In order to generate clusters, the dissimilarity matrices are used to put the songs in the dataset into a coordinate system. This is commonly done using classical MultiDimensional Scaling (MDS) [3, 7], though this method cannot be directly applied matrices generated by either the ASE chain or MFCC chain, as the dissimilarity matrices are neither euclidean or symmetrical, since the confidence measure is based upon the maximum likelihood of the best path to describe one song through another song's HMM (Section 2.1), an inherently asymmetrical process (Equation 5).

$$\lambda_{AB} \neq \lambda_{BA} \quad (5)$$

Where λ_{AB} is the maximum likelihood of the best path to describe song *B* through another song *A*'s HMM and λ_{BA} is the reverse.

Since any of the dissimilarity matrices generated from HMMs will be asymmetrical (due to Equation 5), a work around must be used. The simplest solution is take the average of each pair of entries across the

diagonal of a matrix and this as the new value in both positions (Equation 6).

$$X_{(ij,ji)} \Leftarrow \frac{X_{ij} + X_{ji}}{2} \quad (6)$$

Where X_{ij} is the the value in the matrix found at position (i, j) . This creates a symmetrical matrix which can then be used in classical MDS. The downside to this method is the loss of information contained the asymmetry that has been averaged out. In order to minimize this lose of information more complex approaches can be taken, such as in [14] where a dynamic weight model based upon an oscillating spring is used to keep more information whilst still moving into a coordinate space.

5. RESULTS AND ANALYSIS

The results of this dissimilarity analysis will be examined in two ways, a narrow qualitative examination and a broad, quantitative analysis based on clustering. First, a brief qualitative look at the dissimilarity matrices directly through examination of top five nearest (e.g. smallest values in the dissimilarity matrix) songs to a selection of songs in the dataset. Second, K-means clusters will be apply to the songs after they have been put a coordinate space through MDS.

5.1. Nearest Neighbors

A simple way to evaluate the dissimilarity matrix is to examine the song that have lowest dissimilarity from a small subset of songs in the dataset. This check via audition, though informal, is a good means to confirm that the algorithm is producing sensible results, without needing to consider the viability of the ground truth data being used in the more quantitative collection wide evaluation methods discussed in Section 5.2. The top five nearest songs by each of the individual feature-decision chains and in a combined simple weighted metric (weighting is show in Equation 7) from the song "You're Beautiful" by James Blunt are shown in Table 5.1.

$$W_{ij} = 0.001T_{ij} + 0.500M_{ij} + 0.499S_{ij} \quad (7)$$

5.2. Clustered Confusion

As discussed in Section 4.2 moving from a distance space to a coordinate space and clustering via k-means can provide a meaningful quantitative measure for the entire dataset. Additionally since nearly

Spectral Feature	MFCC	Tempo	Weighted Avg.
<i>The Rockafeller Skank</i> - Fatboy Slim	<i>Soul Man</i> - The Blues Brothers	<i>Return to Innocence</i> - Enigma	<i>Drops of Jupiter</i> - Train
<i>Far Away</i> - Nickelback	<i>You Don't Mess Around with Jim</i> - Jim Croce	<i>Can't Help Falling in Love</i> - Andrea Bocelli	<i>Return to Innocence</i> - Enigma
<i>Goodies</i> - Ciara	<i>When It Don't Come Easy</i> - Patty Griffin	<i>The Way You Look Tonight</i> - Tony Bennett	<i>When Stars Go Blue</i> - The Corrs
<i>This Love</i> - Maroon 5	<i>Schindler's List: Theme</i> - John Williams	<i>I See Right Through You</i> - DJ Encore	<i>Teardrop</i> - Massive Attack
<i>Come on Closer</i> - Jem	<i>Planet Rock</i> - Afrika Bambaataa and Soulsonic	<i>Ramblin' Irishman</i> - Andy M. Stewart	<i>Just Feel Better</i> - Santana

Table 1: Nearest Neighbor results from the song "You're Beautiful" by James Blunt along the various feature-decision chains.

Genre/Cluster	1	2	3	4	5	6	7	8	9	10	11
alternative	27.50	10.00	17.50	5.00	0	2.50	17.50	10.00	2.50	5.00	2.50
blues	2.43	7.32	53.67	0	0	26.83	2.44	0	4.88	2.44	0
classical	0	17.50	72.50	0	0	10.0	0	0	0	0	0
dance	2.70	2.70	21.62	5.41	8.18	8.18	29.73	2.70	5.41	10.81	2.70
electronic	2.50	2.50	12.50	7.50	12.50	20.00	22.50	2.50	7.50	10.00	0
folk	2.56	2.56	35.90	2.56	10.26	41.03	2.56	0	0	0	2.56
hip-hop/rap	2.50	0	2.50	10.00	2.50	5.00	47.50	5.00	2.50	22.50	0
jazz	0	4.87	65.86	2.44	0	17.07	0	7.32	0	2.44	0
pop	24.39	7.31	9.76	4.88	0	2.44	19.51	2.44	4.88	21.95	2.44
R & B/Soul	7.89	2.63	13.16	7.89	5.26	2.63	21.05	2.63	0	36.84	0
rock	23.08	2.56	20.51	5.13	2.56	7.69	17.95	2.56	5.12	7.69	5.13

Table 2: Confusion matrix showing the overlap between the prescribed genre labels and the 11 cluster generated via k-means.

the same dataset was run using a related genre sorting method in [1], some meaning can be taken from comparing the cluster overlap seen with this methodology against the genre recognition rates of that work. In order to perform the clustering, classic MDS is employed (after the use of Equation 6 to make the dissimilarity matrix symmetrical) to move to an 8 dimensional coordinate space. K-means clustering into 11 clusters is then applied. The overlap seen between these clusters and the assigned genre labels can be seen in Table 5.2. If we take the average along the diagonal, it gives what, if the clusters were assigning genres, the overall accuracy of this assignment. In Table 5.2 this overall overlap is 24.38% (p-value of 0.049).

6. CONCLUSION AND FUTURE WORK

This paper has examined a means to use audio content classifiers in order to build dissimilarity matrices to better understand and exploit pairwise song relationships across a collection of music. A small dataset ($N = 436$) was used to test this system and evaluate its performance. The performance was evaluated in two ways, a narrow qualitative examination and a broad collection wide quantitative measure. In the qualitative measure, fairly sensible results were shown for nearest neighbor clusters, with the output of the three independent feature chains compared with each other and against a weighted combination of all three. The quantitative measure showed clearly that the clusters produced, though perhaps reasonable groupings of songs, lacked strong correlation with the genre labeling of the dataset, making it difficult to determine how much semantic meaning these clusters have without further study.

This presents a few possible avenues of inquiry. Without changing the dissimilarity measure, a better evaluation method would be highly beneficial, allowing for improvements in qualitative ratings of this measure against others. One possibility, though labor intensive, would be to implement listening tests examining human listener's determination of pairwise song relationships.

7. REFERENCES

- [1] B. Fields, "Using mixed feature extraction with multiple statistical models to achieve song categorization by genre," in *Proc. Audio Engineering Society 122nd Int. Conv.*, (Vienna, Austria), Audio Engineering Society, May 2007.
- [2] K. Jacobson, "A multifaceted approach to music similarity," in *Proc. of Int. Symposium on Music Information Retrieval*, 2006.
- [3] E. Pampalk, A. Flexer, and G. Widmer, "Improvements of audio-based music similarity and genre classification," in *Proc. of Int. Symposium on Music Information Retrieval*, 2005.
- [4] E. Pampalk, *Computational Models of Music Similarity and their Application in Music Information Retrieval*. PhD thesis, Technischen Universität Wien, May 2006.
- [5] Z. Xiong, R. Radhakrishnan, A. Divakaran, and T. S. Huang, "Comparing mfcc and mpeg-7 audio features for feature extraction, maximum likelihood hmm and entropic prior hmm for sports audio classification," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2003.
- [6] M. Casey, "Mpeg-7 sound-recognition tools," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, pp. 737 – 747, June 2001.
- [7] J. Aucouturier and F. Pachet, "Tools and architecture for the evaluation of similarity measures : Case study of timbre similarity," in *Int. Symposium on Music Information Retrieval*, 2004.
- [8] J. Martinez, "Mpeg-7 overview (version 10)." <http://www.chiariglione.org/mpeg/standards/mpeg-7/mpeg-7.htm>, Oct. 2005.
- [9] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proc. of the IEEE*, vol. 77, pp. 257 – 286, Feb 1989.
- [10] G. D. Forney, "The viterbi algorithm," *Proc. of the IEEE*, vol. 61, pp. 268 – 278, March 1973.
- [11] E. Scheirer, "Tempo and beat analysis of acoustic musical signals," *J. Acoust. Soc. Am.*, vol. 103, pp. 588 – 601, Jan. 1998.

- [12] M. Cooper and J. Foote, “Automatic music summarization via similarity analysis,” in *Proc. of Int. Symposium on Music Information Retrieval*, pp. 81 – 85, 2002.
- [13] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*. San Francisco, CA, USA: Morgan Kaufmann, 2nd ed., 2005.
- [14] A. Muñoz and M. Martin-Merino, “New asymmetric iterative scaling models for the generation of textual word maps,” *JADT 2002 : 6es Journées internationales d’Analyse statistique des Données Textuelles*, 2002.