SEMANTICALLY MEANINGFUL MUSIC RETRIEVAL WITH CONTENT-BASED FEATURES AND FUZZY CLUSTERING

A.S. Lampropoulos and G.A. Tsihrintzis

Department of Informatics
University of Piraeus
Piraeus 185 34
GREECE
{ArisLamp, Geoatsi}@unipi.gr

ABSTRACT

We propose a system which is based on fuzzy c-means clustering and automatically organizes a collection of music files according to musical surface characteristics (e.g., zero crossings, short-time energy, etc.) and tempo. Our system is complemented by semantic metadata which offers the user the ability to check the clustering result, correct existing textual meta-information (such as genre and album) and add additional semantic information (about lyrics, theme, other artists who perform similar music, musical instruments, etc). This second step is realized in a process of relevance feedback and imposes consistency and reliability between content and semantic information of the musical database.

1. INTRODUCTION

Today, there exist large music databases and digital music download activity is becoming an every day task for most computer users. This gives rise to a need for systems that have the ability to manage the content of stored music files.

There are two types of such Musical Information Retrieval Systems (M.I.R.S.). On one hand, there exist M.I.R.S. based on textual meta-information such as Kazaa. The best known example of this music file organization is probably the ID3 format, an extension to the popular mp3 format which allows the user to add specific tags such as song title, album name, artist or group name, etc. On the other hand, there are M.I.R.S. based on low-level content features such as MFCCs (Mel-frequency cepstral coefficients), musical surface characteristics (e.g. Zero crossings, Short time energy, etc.) and rhythmic characteristics (e.g. Tempo). These systems offer an objective and automatic way to organize music files and the possibility of Query-By-Example (Q.B.E.) in which the users provide an example of the content they seek.

However, such a scheme has some obvious limitations, since most users wish to search in terms of the music semantics rather than by content, especially when the search is done over the web. M.I.R.S., which are based on low-level features, cannot represent the semantics of music files effectively as the corresponding similarity measure does not account for high-level features such as information for lyrics or usual issues of an artist. In fact, there is always a gap between the feature vector and the semantic information.

Our aim is to provide a semi-automatic user interface that allows users to interact with their music files in a flexible way based on a combination of content and semantic information. Our approach is a two-step procedure: Initially, it uses fuzzy c-means clustering to organize music files according to content-based features. Then, it utilizes the existence of textual information to offer the user the possibility to check the clustering result, correct existing textual meta-information (such as genre and album) and add additional semantic information (about lyrics, theme, other artists who perform similar music, musical instruments, etc). This second step is realized in a process of relevance feedback and constructs consistency and reliability between content and semantic information of the musical database.

Consequently, the efficiency of the music retrieval process is significantly improved, since objective type of similarity computed from features from the acoustical signal is enhanced by inclusion of semantic information.

The paper is organized as follows: Section 2 describes a music/sound content-based feature extraction process, which maps each song into a feature vector of low dimensionality (43 features per song). Section 3 describes the relevance feedback process and presents our proposed system. Conclusions and some directions for future research are given in Section 4.

2. MUSIC/SOUND LOW-LEVEL CONTENT FEATURES

To be able to search through a collection of music pieces and make observations about the similarity between objects that are not directly comparable, we must trans-form raw data at a certain level of information granularity. Information granules refer to a collection of data that contain only essential information. Such granulation allows more efficient processing for extracting features and computing numerical representations that characterize a music file. As a result, the large amount of detailed information of each song is reduced to a limited collection of features. Each feature is a vector of low dimensionality, which captures some aspects of a song and can be used to determine song similarity.

An extensive variety of features can be extracted from raw audio data using either their time domain or the frequency domain (spectral) representation [4], [5], [6], [7], [8]. In our system, we used a 43-dimensional feature vector, comprised of the following features: Centroid, Roll-Off, Zero-Crossings, Energy, Spectral Flux (5 spectral bands), and Tempo. The mean, median, standard deviation of centroid, roll-off and zero-crossings features are calculated over a "texture window" of duration of 5 seconds consisting of 100 "analysis windows" of duration of 50 milliseconds. On the other hand, the short time Fourier trans-form (STFT) feature calculation is based on a Fast Fourier Transform (FFT) algorithm [1], [2]. In the following, we describe the feature computation in detail.

2.1. Musical Surface Features

For pattern recognition purposes, we can use the statistics of the spectral distribution over time and represent the "musical surface" [4], [5], [6], [7], [8]. Some of these statistics are defined next.

2.1.1.Spectral Centroid.
This feature reflects the brightness of the audio signal. It is computed as the balancing point (centroid) of the spectrum. It can be calculated as

$$C = \frac{\sum_{n=1}^{N} M_t[n] \cdot n}{\sum_{n=1}^{N} M_t[n]}, \quad (1)$$

where $M_t[n]$ is the magnitude of the Fourier Transform at Frame t and frequency bin n.

2.1.2. Spectral Rolloff.
This feature describes the spectral shape. It is defined as the frequency R corresponding to r% of the magnitude distribution. It can be seen as a generalization of the spectral centroid, as the spectral centroid is the roll-off for r = 50%. In our system, we used a roll-off value of r = 95%.

$$\sum_{n=1}^{R} M_t[n] = r \sum_{n=1}^{N} M_t[n], \quad (2)$$

2.1.3. Spectral Flux.
This feature describes how the frequency evolves with time. It is computed as the difference of the magnitude of the short-time Fourier transform between the current and the previous frame. In other words, it is a measure of local spectral change and is defined as

$$SF = \sum_{n=1}^{N} \left(N_t[n] - N_{t-1}[n]\right)^2, \quad (3)$$

where $N_t[n]$ is the normalized magnitude of short-time Fourier transform at window t.

2.1.4. Zero-Crossings.
A zero-crossing occurs when successive samples in a digital signal have different signs. The corresponding feature is defined as the number of time domain zero-crossings of the signal. This feature is useful to detect the amount of noise in a signal. It can be calculated as

$$Z_n = \sum_m \left|\mathrm{sgn}[x(m)] - \mathrm{sgn}[x(m-1)]\right| w(n-m), \quad (4)$$

where

$$\mathrm{sgn}[x(n)] = \begin{cases} 1, & x(n) \geq 0, \\ -1, & x(n) < 0 \end{cases} \quad w(n) = \begin{cases} \frac{1}{2}, & 0 \leq n \leq N-1, \\ 0, & otherwise. \end{cases} \quad (5)$$

2.1.5. Short–Time Energy Function.
The short-time energy of an audio signal is defined as

$$E_n = \frac{1}{N} \sum_m \left[x(m) w(n-m)\right]^2, \quad w(n) = \begin{cases} 1, & 0 \leq n \leq N-1, \\ 0, & otherwise. \end{cases} \quad (7)$$

In Eqs.(7) , x(m) is the discrete-time audio signal, n is the time index of the short time energy and w(n) is a rectangular window. This feature provides a convenient representation of the temporal evolution of the amplitude variation.

2.2. Rhythm Feature

2.2.1. Tempo.
In order to extract tempo we use the beat detection algorithm provided at the site [9]. This algorithm is quite similar to a beat detection algorithm developed by the MIT Media Lab. It amounts to emphasizing the sudden impulses of sound in the song and then finding the fundamental period at which these impulses appear. This is accomplished by breaking the signal into frequency bands, extracting the envelope of the resulting frequency–banded signals, differentiating them to emphasize sudden

changes in sound, and running the signals through a bank of comb filters and choosing the highest energy result as tempo. A drawback of this algorithm is that it does not account for the possibility of a varying tempo in a music piece. Specifically, the algorithm extracts a piece of music of duration of 2.2 seconds from the middle section of a song, analyzes its tempo, and assumes that the computed tempo is associated with the entire music piece.

3. IMPLEMENTATION - PROPOSED SYSTEM

We describe a *relevance feedback* process, which is sometimes used in refining similar retrievals in image databases. This relevance feedback process allows the system to capture the user needs and preferences more accurately and, in future queries, retrieve music files that the user accepts as more relevant to his/her query.

A variety of relevance feedback schemes have been proposed in the literature which are generally classified into two approaches: query point movement (query refinement) and reweighting (similarity measure refinement). Both approaches are based on the feature vector model of information retrieval theory [15]. In our proposed system, we follow the framework proposed in [10], which unifies image semantics and image content-based features. This framework maintains the strength of low-level (content-based) feature retrieval while at the same time incorporating learning and annotations through the relevance feedback processes. Experiments have shown that this framework effectively not only improves the retrieval performance in a given query session, but also utilizes the knowledge learnt from previous queries to reduce the number of iterations in subsequent queries. The framework has been implemented into iFind, a web-based image retrieval system developed by Microsoft Research China [11]. iFind provides functionality of keyword-based image search, image query by example, and their combination. In our system, we allow relevance feedback on both music semantics keywords (herein artist name, genre, etc.) and the music content-based features of Section 2 along lines similar to those in [10], [11].

Our proposed system is developed as a desktop application and includes the inter-related processes described in the following. The feature extractor module which captures the low-level (content-based) features of music files. This module is complemented with the lame decoder [13] which normalizes the format of all music files, so that a query by example searches for data that are *globally* similar to the query independently of the scale of the query. In other words, all records in a database are re-sampled into a database-wide uniform sample rate.

Each music piece is allowed to be stored in any audio file format, such as *.mp3*, *.au*, or *.wav*. In order to extract features, first we apply format normalization. We decode each music file to raw Pulse Code Modulation (PCM), converting it to the .wav format with audio CD quality and resolution of 16 bit samples at a sampling rate of 44.1 kHz. Next, we reduce stereo sound quality to mono quality. This latter reduction implies no significant information loss as the two channels contain the same samples.

From each file, a randomly chosen sample of duration of 5 seconds is used as input to the feature extraction module. The actual features computed by our system are the running means, median and standard deviation over a number of analysis windows of the features of Section 2. In short time audio analysis, the signal is broken into small, possibly overlapping temporal segments of duration of 50 milliseconds and each segment is processed separately. These segments are called ‘‘analysis windows’’ and need be short enough for the frequency characteristics of the magnitude spectrum to be relatively stable. The term ‘‘texture window’’ describes the longest window that is necessary to identify music texture. This corresponds to the minimum amount of sound needed to identify music texture and was equal to 5 seconds in our system.

The second module is an ID3 tag extractor and an interface for the user to update the extracted semantic meta-data and add other possible semantic keywords. Semantic knowledge in our application includes knowledge about artist, composer, lyricist, genre, album or the collection that include a music file, title of a music file, keywords about impact on mood (e.g. relax, happy etc), keywords from lyrics and musical instruments

The third module realizes the process of fuzzy *c*-means (FCM) clustering, which is a method of data clustering that allows one piece of data to belong to two or more clusters at the same time [14]. FCM is used for the initialization of our system. Fuzzy clustering applied on low-level content-based features is used to reveal the predominant classes of the database. Moreover, having the advantage of the clustering result, the user is then given the possibility to annotate clusters with keywords of semantic knowledge, such as albums, artists, etc. In this way, we create semantic information even for those files that do not have initial semantic meta-data annotated to them, as for example in the case of empty ID3 tags.

The fourth module realizes the retrieval process which also embeds the relevance feedback process. The user can make queries, using either semantic keywords or by example or a combination of them. The user can submit a query consisting of one or more keywords (e.g. artist: “Terzis” and genre: “rebetica”). Then, the system uses the keywords to automatically search in the database for those music files relevant to them. There are two situations to consider at this point. In the first case, no music files in the system are tagged with the query keywords. In the second case, some music files have tags which either match the query or are included in the

cluster with those keywords. In the first case, the system does not return any results. In the second case, those files tagged with the keywords specified in the query (or similar to them) are retrieved and presented as top ranking matches. In addition, more music files are added to the result list, specifically the set of music files found on the basis of their low level content-based similarity to the files matched with the query.

From the list of retrieved music files, the user may provide relevance feedback to inform the system which files he/she considers "relevant" or "irrelevant". For each of the untagged "relevant" music files, tags are attached which correspond to the query keywords (e.g. genre: "rebetica") and have an initial weight of 1. If the file tags have already been completed, the weight of this keyword is increased by a given increment of 1. For each of the irrelevant files, the weight of this keyword is decreased by a given decrement of ¼. If the weights become very small (e.g. less than 1), the keyword is removed from the annotation to this file. The process results in a set of keywords and weights associated with each individual file in the database. The links between keywords and files form a semantic network as in [10]. In the semantic network, each music file can be complemented with a set of semantically relevant keywords and, vice versa, the same keyword (e.g. genre: "rebetica") can be associated with more than one files.

Additionally, when the user adds a new music file into the database (or makes a database query by example), the system uses the new music file as a query by example and performs a content-based retrieval using the nearest neighbor criterion. The system presents the results and suggests to the user the cluster to which the new file belongs and keywords for its tags.

Our proposed system has been developed using *MatLab* as the computing engine for implementing the feature extractor and building a dynamically-linked library (.dll) with the *Com-Builder* tool. On the other hand, the user interface was developed in the *.NET* framework using the *C#* and *VB .NET* languages. Extracted features and other meta-data of music files are stored in *SQL SERVER 2000 RDBMS*.

4. CONCLUSIONS – FUTURE WORK

In this paper, we proposed the combination of content-based and semantic meta-data in a music retrieval system for searching and adding semantic meta-data in a music database. Our approach bridges the gap between low level (content-based) and high-level (semantics-based) music retrieval systems [12]. Our preliminary proposed system evaluation and experimental results indicate more accurate, robust and efficient music retrieval. We are in the process of improving further the clustering accuracy and perception efficiency of our system by considering additional low level music features. Further extensions to our system follow the implementation of a web-based retrieval system and related system architectures, as well as extensive performance evaluation. This and other work is currently in progress and the results will be presented shortly.

5. REFERENCES

[1] G.A. Tsihrintzis, and C. Douligeris, *Principles and Applications of Signals and Systems*, Varmar Publications, Piraeus, Greece, 2003.

[2] G.A. Tsihrintzis, *Pattern Recognition*, University of Piraeus 2002.

[3] R. Steinmetz, and K. Nahrstedt, *Multimedia Fundamentals Volume 1, Media Coding and Content Processing*, Prentice Hall 2002.

[4] J. Foote, "An overview of audio information retrieval" *Multimedia Systems*, 7(1):2-10, 1999.

[5] K. Kosina, Music Genre Recognition, PhD thesis, Hagenberg, 2002.

[6] G. Tzanetakis, Manipulation, Analysis and Retrieval Systems for Audio Signals, PhD thesis, Princeton University, 2002.

[7] M.Welsh, N. Borisov, R. von Behren, A. Woo, "Querying large collections of music for similarity", Technical report UCB/CSD00-1096, U.C. Berkeley, Computer Science, 1999.

[8] H. Deshpade, Unjung Nam, R. Singh, "MUGEC Automatic music genre classification", Technical report, Standford University, 2001.

[9] K. Cheng, et al, "Beat detection algorithm", http://www.owlnet.rice.edu/~elec30/beatalgo.html

[10] Y. Lu, et al, "A Unified Framework for Semantics and Feature Based Relevance Feedback in Image Retrieval Systems", In *Proceedings of ACM MULTIMEDIA 2000*, Los Angeles California, pp 31-38, 2000.

[11] Z. Chen, et al, "iFind: a web image search engine", In *Proceedings of the 24th annual international ACM SIGIR conference on Research and Development in Information Retrieval*, New Orleans, Louisiana, United States, pp.450, 2001.

[12] A.S. Lampropoulos, and G.A. Tsihrintzis, "Agglomerative Hierarchical Clustering For Musical Database Visualization and Browsing, To be published in *Proceedings of the 3rd Hellenic Conference on Artificial Intelligence*", Samos, Greece, 2004.

[13] http://lame.sourceforge.net/

[14] J.C. Bezdek, "FCM: The fuzzy c-means clustering algorithm", *Computers and Geosciences*, 10, 191-203 , 1984

[15] Y. Rui, T.S. Huang, M. Ortega, S. Mehrotra, "Relevance Feedback: A Power Tool in Interactive Content-Based Image Retrieval", *IEEE Transactions on Circuits and Systems for Video Technology*, 1998