Content-based Information Retrieval in Streaming Music*

Maria Kontaki, Ioannis Karydis and Yannis Manolopoulos

Data Engineering Lab, Aristotle University, 54124, Thessaloniki, Greece
{kontaki, karydis, manolopo}@csd.auth.gr

Abstract

In this paper we examine searching by content in broadcasted streams of musical data, where the querier defines a set of preferred musical pieces and receives a list of broadcasting feeds that contain music similar to the preferred set. Streaming environments impose challenging requirements for content-based music information retrieval as memory limitations do not allow for buffering, data accumulation "on-the-fly" makes pre & post processing not a possibility while high response time is a necessity. To address these requirements we devise an incremental version of an award winning feature extraction and similarity process and propose a song boundary detection method in order to increase similarity accuracy and reduce costly feature extraction and similarity calculations. Extensive experimental results verify our claims and illustrate the superiority of the proposed method, over a baseline approach, as well as the suitability of the method for the streaming environment.

Keywords: music information retrieval, content-based similarity, continuous querying, web radio, music podcasting, incremental feature extraction, streaming data.

1. Introduction

Nowadays, WWW is developing as a key means of information dissemination, even in daily routine activities. Most already existing forms of information in the "old" world are attempting a transformation to the known forms of the webbed world while in other cases, the developments of the web pioneer in launching brand new forms of information. Accordingly, as their volume and usage increase, the necessity of their management emerges as a prominent activity, especially for the new forms that introduce new requirements.

Internet music broadcasting falls within both previously mentioned categories. On the one hand, much like the traditional radio service, Internet radio is a broadcasting service transmitted via the Internet in a 24-hour, 365 days per year manner, suggesting a streaming media that delivers to listeners a continuous stream of audio, by adopting an already existing technique, the webcasting. As with the traditional

* Research supported by the PAVET 2005 program, funded by the GSRT, Greece.

radio, listeners have no control on the broadcasted media. On the other hand, the new and trendy [PR Newswire (2006)] practice of *podcasting* refers to the release of a digital recording of a radio broadcast or similar program, on the Internet for downloading to a personal audio player. Podcasting's initial appeal aimed at allowing individuals to distribute own "radio shows", following the previously developed web syndication techniques. Podcasts may be created and posted at any time and may be as large in terms of number of musical pieces included as desired by the creator. Nowadays, numerous websites exist enlisting tens of thousands of feeds [Directory of 55.000 radio stations (2006)] of Internet radio stations and several thousands of podcasts [Podcast mvyradio's Blues@8 (2006)] (henceforth the term "feed" describes both Internet radio stations and podcasts). Moreover, characteristics of radio such as non stopping stream of songs mixed with other non musical content may appear as well, though standard high-performance methodologies [Ajmera at. Al. (2003)] for music-speech segmentation do exist.

It should be noted that, in this work, broadcasted data are confronted as streaming data, where streaming data are time-series received from a feed. That is, data is modelled best not as persistent relations but rather as transient data streams. In that sense, broadcasted music shares a number of common characteristics with streaming data such as being time ordered, data arriving in segments at an unknown incoming rate and data storage/post-processing is not wanted/possible but instead an "on-the-fly" processing is required. Thus, the method utilised for the similarity should be incremental in order to deal with memory limitations as well as high response time requirements. Moreover, streaming data methods are required in order to satisfy the need to use *continuous querying* (cf. see Example scenario 2). Traditional content based music information retrieval (CBMIR) persistent relation models do not require a method for the identification of song boundaries, as each song's boundaries are clearly defined from a container file in the database used. Accordingly, if traditional CBMIR methods are used in a data stream music model, search for similarity cannot be stopped given that a song is early found to exceed similarity threshold as no means of song ending is provided therein.

Example scenario 1. An example scenario of the key idea of this research includes a website that contains a number of feeds and a user with a set of preferred songs. The user activates a client-side program that implements the method proposed in this paper, registers the preferred music and the URL of the website listing the feeds. Accordingly, the user selects a monitoring period for the program to monitor a certain number of feeds and receives back the feeds that for the monitoring period included musical pieces similar to the user preferences.

Example scenario 2. In a differentiated scenario, the implementation of the proposed methodology could reside on a server-side program. That is, the user would visit a website that hosts a number of feeds as well as a large collection of songs for the user to define preferences. Accordingly, after selecting both possible feeds and preferences

the site could “on-the-fly” select the feed that for the current broadcast is most similar to the user’s preferences and offer the possibility for on-line listening. As the user listens, the site monitors the remaining feeds in order to ensure that no other feed is more similar, in which case the user would be notified (*continuous querying*).

In this work, we focus on the example scenario 2. To our best knowledge, the implementation of CBMIR in streaming musical data has not been attempted before in the literature. Nevertheless, existing works, that follow similar research direction, are neither considering the streaming character of data nor the necessity for continuous querying [Velivelli et. Al. (2003)]. The contribution of this work is summarised as follows:

- An incremental feature extraction process leading to significantly less processing cost.
- A song boundary detection methodology which is used to avoid costly (a) similarity distance calculations and (b) full feature extraction.
- The performance evaluation of the proposed method based on real-life data sets.

The rest of the paper is organised as follows. Section 2 describes background and related work. Section 3 provides a complete account of the proposed method. Subsequently, Section 4 presents and discusses the experimentation and results obtained, while the paper is concluded in Section 5.

2. Background & Related Work

2.1 Content-based Music Information Retrieval

The field of music information retrieval has received increased attention during the last decade. Numerous surveys examine the state of the art developments in the area [Karydis et. Al. (2006); Typke et. Al. (2005)] while a litany of works spawns rapidly in all directions of MIR. Most research on MIR is content based due to limitations imposed by metadata approaches, such as: they are not always available, their use in MIR requires knowledge for the query not provided by listening while their descriptive capability lacks customisation as it relies on predefined descriptors.

Content-based approaches assume that documents are described by features extracted directly from the content of musical documents. The selection of appropriate features is very important in music information retrieval. In this work, we do not concentrate on devising new features. Instead, we are interested in an incremental feature extraction methodology in order to fulfil the requirements posed by the data stream music model. Thus, we devise an incremental feature extraction process based on the Single Gaussian Combined (G1C) [Pampalk (2006)] as submitted to the MIREX 2006 contest [MIREX (2006)] bearing in mind that it achieved the highest score.

Initially, for each piece of music the *Mel Frequency Cepstrum Coefficients* (MFCCs) are computed, the distribution of which is summarised using a single Gaussian (G1) with full covariance matrix. The distance between two Gaussians is computed using a symmetric version of the Kullback Leibler divergence. Then, the fluctuation patterns (FPs) of each song are calculated. A FP describes the modulation of the loudness amplitudes per frequency bands, while to some extent it can describe periodic beats. All FPs computed for each window are combined by computing the median of all patterns. Accordingly, two features are extracted from the FP of each song, the "gravity" (FP.G) which is the centre of gravity of the FP along the modulation frequency dimension and the "bass" (FP.B) which is computed as the fluctuation strength of the lower frequency bands at higher modulation frequencies. For the four distance values (G1, FP, FP.B and FP.G) the overall similarity of two pieces is computed as a weighted linear combination (normalised in [0,1]) as described in detail in [Pampalk (2006)].

2.2 Streams and Continuous Querying

Nowadays, a plethora of applications, as one would expect, produce data streams as opposed to data sets: financial data, network monitoring, data feeds from sensor applications, etc. The theme of this research considers in particular continuous queries. Typical examples in financial applications include stock value monitoring where continuous queries may be used to examine similarities between stocks and trends.

Methodologies developed for content-based retrieval in streams are specific to the nature of the data under examination and thus not directly applicable. The key differentiation relies on the usage of metric distance similarities that take advantage of the triangular inequality and perform early pruning, thus minimising the processing cost of unwanted distance calculations [Gao et. Al. (2002); Kontaki et. Al. (2004)]. Typical such distance metrics are the Euclidean and Manhattan distance, which are not suitable for musical data similarity [Berenzweig et. Al. (2004)].

3. The Proposed Method

3.1 System Architecture

Elaborating further on the example scenario 2, as discussed in Section 1, after the user has decided the preferred songs on which to determine stream likeness and their features have been extracted, the system must extract MFCCs of each broadcasted feed, identify the parts that contain music, extract features, calculate similarities with preferred songs and assign a score to each feed based on the each current playing song. The scores are then presented to the user, while after the initial score

calculation, the system must continuously monitor the feeds and perform the same procedures in order to determine if scores change as the feeds keep broadcasting. The proposed architecture is depicted in Figure 1.

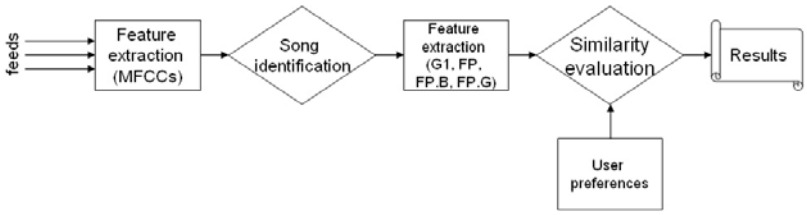

Figure 1. *CQiSM architecture.*

Accordingly, the proposed system, the *C*ontinuous *Q*uerying *i*n *S*treaming *M*usic (CQiSM) can be divided into three subsections: the incremental feature extraction, the song boundary detection and the overall feed evaluation. The next sections provide details on each of them.

3.2 Incremental Feature Extraction

One of the "sine qua non" characteristics of streaming data is that in order to abide by the input rate time constrains all processes on data must be fully optimised. In other words, previously calculated results, that can be used in order to alleviate burden off next calculations, need be taken advantage of. Accordingly, as one of the most time consuming steps of the proposed method is the feature extraction process, this section details an incremental version of the G1C method by Pampalk [Pampalk (2006)]. It should be noted that the proposed incremental extraction solely refers to the extraction of the MFCCs.

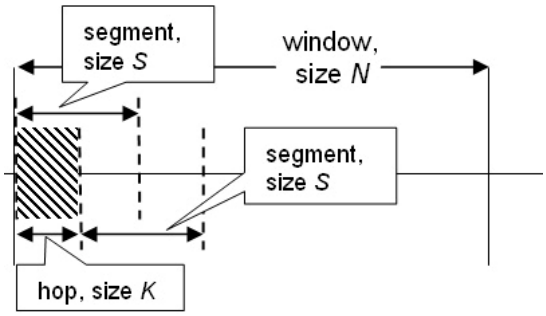

Figure 2. *Song segmentation.*

Each song is divided into windows of *N* elements length, while each window is divided in segments of length *S* overlapping (hop) by *K* elements (see Figure 2).

Every *K* values received define a new segment while the first segment of the window is discarded, collectively henceforth "update". For each new segment its MFCCs need be calculated to perform song boundary identification. The calculation of the MFCCs of the new segment comprises of applying Hann window operation over the segment values, *Fast Fourrier Transform* and numerous other static calculations. Then the results obtained are multiplied with triangular filters (Mel-filters) in order to transform the power spectrum to the Melscale using 34 (Mel-spaced) frequency bands.

Under the assumption that the method is applied on the full window (non-incremental version), Mel-filters and the previous results are two-dimensional matrices. Bearing in mind that the multiplication of two matrices $A(m \times n)$ and $B(n \times z)$ results in a matrix $C(m \times z)$, we note that each column *j* of the matrix *C* is the result of a combination of all the elements of matrix *A* with the elements of the column *j* of the matrix *B*. In the incremental version, the result obtained is a one-dimensional matrix as it originates from one segment only.

Finally, more static calculations occur to compute MFCCs of the new segment that later on are combined with the MFCCs of previous segments of the current window, in order to calculate the G1, FPs, FP.G and FP.B features. Thus, when an update occurs, we can calculate the MFCCs of the new segment only and combine them with the MFCCs of previous segments to form the MFCCs for the whole window, so the C1G method is implemented in an incremental manner.

3.3 Song Boundary Detection

The consideration of musical data in a continuous form (stream), which are at the same time additionally interleaved by other non musical data, introduces a twofold problem: the song boundary detection and the segmentation of music/speech. The later is required in order to avoid costly calculations in portions of the stream that will not contribute to the overall stream evaluation. Music/speech segmentation is a research direction of signal processing on its own, while high performance solutions already exist [Ajmera at. Al. (2003)]. As their performance would equally affect any compared methods, for simplicity we assume that the differentiation between speech and music is already known.

On the other hand, the song boundary detection refers to a song that is immediately followed by another song in the stream, with no intermission in between. In these cases, when a new song from the feed is identified, it is compared with the user's preferences and then no further similarity on this song need be done, as long as the beginning of the next song is identifiable. Thus, only a minimal feature extraction

process continues in order to be able to identify the song change, while both the rest feature extraction and similarity calculation are avoided.

In order to decide the change of song event, we monitor the progression of values of the MFCCs and look for high changes. As songs finish, their overall energy tends to reduce, an effect clearly evident in the first coefficient of the MFCCs, which is proportional to the audio energy. Accordingly, as the next successive song begins, its signal power is significantly higher and thus the first MFCC coefficient is considerably increased. In order to detect high changes of values of the MFCCs, any well-known similarity metric can be applied. In our experiments the Euclidean distance is used due to its simplicity and popularity. Having identified the beginning of a new song, we only perform a minimal feature extraction including only the MFCCs of the song leaving out the G1, FPs as well as the FP.G and FP.B features, which add-up a significant amount of calculation.

3.4 Overall Feed Evaluation

As the feature extraction and similarity methods utilised herein were designed so as to provide one number, given any two songs without any further context, we propose two alternative strategies leading to the overall feed evaluation.

Initially, in a quantitative approach, one could be interested in a dispersion or breadth of the similarity on a number of songs while not paying attention on the calculated similarity distance. Still, as feeds may contain different number of songs, normalisation is required in terms of the number of songs for which the result was obtained. A quantitative overall feed evaluation is given by Equation 1. That is, for every new song i in the feed, we may receive numerous matches j from the users' preferences with similarity T_{ij}. For every i and j we sum the corresponding similarity T_{ij} and divide it with the number L of songs considered in the feed.

$$\frac{\sum_i \sum_j T_{ij}}{L} \quad \textbf{(1)}$$

On the other hand, a qualitative approach would be concerned with the best match of each user's preference while disregarding the remaining matches. Once again, in order to account for the different number of songs for which the result was obtained across feeds, normalisation is required (see Equation 2). In this case, for every new song i only the most similar result is considered for the evaluation of each feed.

$$\frac{\sum_i \max\left(T_{ij}\right)}{L} \quad \textbf{(2)}$$

4. Performance Evaluation

All algorithms described have been implemented on a personal computer with 3,06GHz Intel Pentium IV processor, 1 GByte RAM, MS Windows XP operating system while the developing package utilised was MATLAB version 6.5.

The data sets employed for the experiments include real music objects. Each wav file was down-sampled at 11.025 Hz at 8 bits, mono channel. The music objects pertain to various genres such as Greek and English pop, rock as well as instrumental music. Each feed was created by concatenating 30 songs, while experiments assume 5 feeds unless otherwise stated. Each feed has been created by random selection of songs, while to examine exact matching capability, the user's preferences have been included in the feeds in random locations.

Table 1 summarises the default values that are used in all following experiments, unless otherwise stated. Due to space restrictions, we do not present results on the tuning of these parameters. In order to evaluate CQiSM, we utilised the method proposed by Pampalk [Pampalk (2006)] as a competitor (henceforth "baseline"), bearing in mind that although very efficient was not originally developed for streaming environments. To account for the continuous querying, we reapplied the baseline for each update.

Table 1. *Experimental parameters.*

Parameter	Default value
Window size	11,62 seconds (128128 stream elements)
Segment size	256 stream elements
Hop size	128 stream elements
New song event threshold	10
Similarity threshold	0,4

In our first experiment (Figure 3) we have examined the execution time for varying window sizes for both CQiSM and baseline approaches. Notice that in Figure 3 the abbreviation A denotes the CQiSM, B denotes the baseline while the execution time is given in a logarithmic scale. For each window size two bars describe each approach: the overall time of execution and the time required for the extraction of the MFCCs per update. Accordingly, their difference accounts for similarity time calculation. It can be clearly seen that the proposed methodology outperforms the baseline approach. It should be noted that, for the baseline approach, computation times become prohibitive in order to keep up with input rate of feeds.

In the next experiment (Figure 4), we tested the execution time for the two approaches for different values of the new song event detection threshold. To begin

with, the overall execution time and MFCC extraction time for the baseline approach is fixed due to the fact that it is not affected by the examined parameter. It is presented here for comparison reasons. The key observation is the confirmation of our claim for song boundary identification aiming at saving costly similarity calculations. In detail, for new song event detection threshold equal to zero, the difference between overall and MFCC extraction execution times is equal to the similarity calculations that the baseline performs (not clear due to logarithmic scale). Accordingly, the use of the new song event detection drastically reduces similarity calculations at the cost of potential lost song identifications. As the baseline method proves clearly not suitable for the examined framework, it is not examined further in the experimentation.

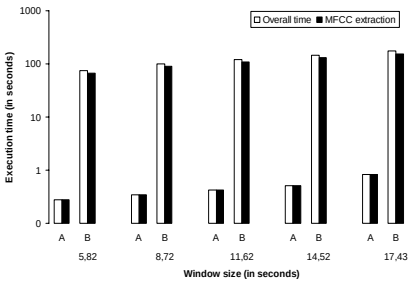

Figure 3. *Execution time over window size for CQiSM and baseline.*

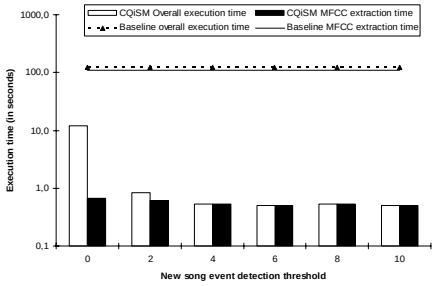

Figure 4. *Execution time for CQiSM and baseline approaches for different values of the new song event detection threshold.*

In our next experiment, Figure 5, memory requirements for different number of feeds and varying window sizes have been put to examination. Initially it should be noted that the memory requirements mainly depend on the size of the utilised window. The attitude of the CQiSM is linear for both the window size and number of feeds.

In the sequel, Figure 6 shows the achieved accuracy of CQiSM for different window sizes over the similarity threshold. In this case, the accuracy refers to exact matches identified in the feeds, in other words the ratio of exact matches found over the correct exact matches included in the feed. In this experiment the similarity threshold greater than zero is used for exact matching for the following two reasons: (a) comparison between windows might not reveal matching due to shifting and (b) the use of the hop size introduces areas from which features might not be extracted. Nevertheless, as it can be clearly seen, CQiSM attains over 89% accuracy in all cases. It is obvious that for increasing similarity threshold, the accuracy ameliorates as more matches accumulate. For varying window sizes the performance of the method is quite similar, while small changes appear due to the shifting effect introduced during window selection.

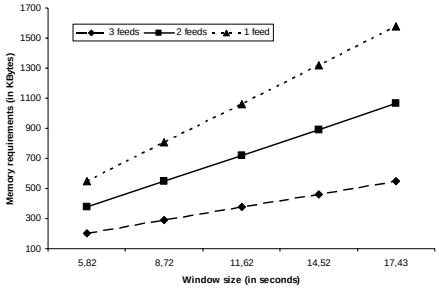

Figure 5. *CQiSM memory requirements for different number of feeds over window size.*

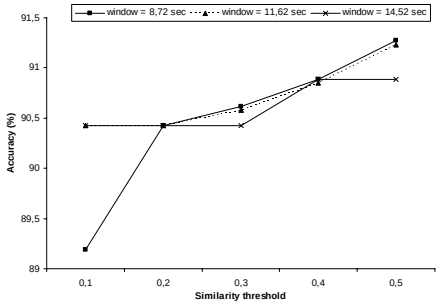

Figure 6. *CQiSM accuracy for different window sizes over similarity threshold.*

The following experiment (Table 2) studied the qualitative attributes of CQiSM. In this experiment for the running value of new song event detection threshold we examine the *recall* percentage accounting for the number of songs identified over the correct number of the received songs. The recall achieved by the CQiSM reaches 96,667%, which corresponds to one missed song. The next parameter pertains to the number of false alarms introduced with respect to the total number of updates, which would be the number of calculations processed by the baseline approach. The last parameter shows the number of new song events detected with respect to the total number of updates. It is obvious that CQiSM achieves high recall rates, despite the fact that our song detection approach prunes approximately 99,95% of the candidates, thus the savings produced in similarity calculations overwhelm the false alarm ratio.

Table 2. *Qualitative attributes of CQiSM*

Parameter	Value %
Recall	96,667
False alarms	0,0507
New song events	0,0555

Table 3. *Exact match average delay.*

Window Size	Average delay
8,72	7,08
11,62	7,17
14,52	7,27

In our final experiment (Table 3) we have tested the suitability of CQiSM to operate in a streaming environment. The results in Table 3 show that our proposed method manages to identify an exact match shortly after 7 seconds (on average) it has arrived. This means that even before the smallest window buffer has accumulated the next window data, results can be obtained and the user notified.

5. Conclusions

This paper deals with the development of methods for searching by content in broadcasted streams of musical data, where the querier defines a set of preferred

musical pieces and receives a list of broadcasting feeds that contain music similar to the preferred set. Notice that to our best knowledge, the implementation of CBMIR in streaming musical data has not been attempted before in the literature.

To address the requirements posed by the streaming environments, we propose the incremental version of an award winning feature extraction & similarity process and a song boundary detection in the stream in order to increase similarity accuracy and reduce costly feature extraction and similarity calculations. The aforementioned claims are verified through extensive experimental results.

References

Ajmera J., McCowan I., Bourlard H. (2003), *Speech/music segmentation using entropy and dynamism features in a HMM classification framework*, Speech Communication, vol. 40, no. 3, pp. 351-363.

Berenzweig A., Logan B., Ellis D. P., Whitman B.P., (2004), *A Large-Scale Evaluation of Acoustic and Subjective Music-Similarity Measures*, Computer Music Journal, vol. 28, no. 2 , pp. 63-76.

Directory of 55.000 radio stations, retrieved from http://radiotime.com/

Gao L., Wang X., (2002), *Continually evaluating similarity based pattern queries on a streaming time series*, in Proc. ACM SIGMOD, pp. 370-381, Wisconsin, USA.

Karydis I., Nanopoulos A., Manolopoulos, Y., (2006), *Mining in Music Databases*, Processing and Managing Complex Data for Decision Support, Idea Group Publishing, pp. 340-374.

Kontaki M., Papadopoulos A.N., (2004), *Efficient Similarity Search in Streaming Time Sequences*, in Proc. SSDBM, pp. 63-72, Santorini, Greece.

MIREX, Annual Music Information Retrieval eXchange, http://www.music-ir.org/mirex2006/

Pampalk E., (2006), *Audio-Based Music Similarity and Retrieval: Combining a Spectral Similarity Model with Information Extracted from Fluctuation Patterns*, implementation submitted to the 3rd Annual Music Information Retrieval eXchange (MIREX'06).

Podcast mvyradio's Blues@8, retrieved on December 31st 2006, from http://www.podcast.net/cat/18

PR Newswire, 'Podcast' Is the Word of the Year, retrieved on December 31st 2006, from http://www.prnewswire.com/cgi-bin/stories.pl?ACCT=104&STORY=/www/story/12-05-2005/0004228195

Typke R., Wiering F., Veltkamp R.C., (2005), *A Survey of Music Information Retrieval Systems*, in Proc. ISMIR-05, pp. 153-160, London, UK.

Velivelli A., Zhai C., Huang T.S., (2003), *Audio Segment retrieval using a synthesized HMM*, in Proc. ACM SIGIR, Multimedia Information Retrieval Workshop, Toronto, Canada.