Detection of similarity in music files using signal level analysis

Conference Paper · November 2016

DOI: 10.1109/TENCON.2016.7848297

5 authors, including:

Mathew Thomas
New York University
3 PUBLICATIONS **3** CITATIONS
SEE PROFILE

Shashidhar Koolagudi
National Institute of Technology Karnataka
126 PUBLICATIONS **1,107** CITATIONS
SEE PROFILE

Y.V. Srinivasa Murthy
VIT University
27 PUBLICATIONS **70** CITATIONS
SEE PROFILE

Detection of Similarity in Music Files using Signal Level Analysis

Mathew Thomas, Mintu Jothish, Navin Thomas, Shashidhar G. Koolagudi and Y.V. Srinivasa Murthy
Deapartment of CSE, National Institute of Technology Karnataka,
Surathkal, Mangalore, India - 575 025.
mathewthomas1721@gmail.com, mintujot@gmail.com, navinthomas94@gmail.com, koolagudi@nitk.ac.in and yvsm@nitk.ac.in

***Abstract*—In today's age of digital media, the collection of music files available to the general public is extremely diverse. As with any such set of data, efforts must be made to classify and categorize these files in order to facilitate easy access and searching. Songs can be classified based on attributes available in the music file's metadata such as artist, album, year of release, length, etc. However, if the similarity between two songs is to be determined, a simple comparison of metadata is not only unsatisfactory, the metadata itself might not be available. Therefore, a method of comparison independent of the availability of metadata is required. In this work, a comparison method has been proposed involving the use of musical parameters such as tempo, key and signal envelope, which are extracted from the music file through signal level analysis. Genre is also computed using a support vector machine (SVM) classifier and used to estimate the similarity between two songs.**

***Index Terms*—Music metadata, tempo, music scale, chord progression, signal comparison, all common subsequences, and music similarity.**

I. Introduction

The digital music industry has constructed one of the most rapidly developing databases in the world today. Consumers are offered a vast selection of music through various streaming and shopping services [1]. In well managed databases, song files are assigned accurate metadata like song title, artist, album, genre, year of release and so forth. Music services can use these parameters to group similar songs together by artist, genre, etc [2], [3]. However, such groupings are not always accurate, as two songs of the same genre, or two songs by the same artist might not be similar. In such cases, and in cases where accurate metadata is not available, a different method of comparing song files is required [4]. Since the files to be compared are music files, it stands to reason that the general characteristics of musical compositions can be used for the comparison. A very fundamental property of a musical composition is *speed* or *tempo*. The tempo is measured in beats per minute (BPM), i.e. the number of quarter notes occurring in sixty seconds. Using a defining characteristic like tempo as a comparison factor allows us to select songs which are of similar speeds [5].

Another metric that characterizes a song is the *key* of the song. In music theory, the key of a musical composition is a group of pitches, or scale upon which a music composition is created. The group features a tonic note and its corresponding chords, also called a *tonic* or *tonic chord (TC)* [6]. It also includes their corresponding chords, pitches and chords outside the group. The key may be of a major or minor mode. The next metric considered for comparison is chord progression. A chord, in music theory, is any harmonic set of three or more notes that is heard as if sounding simultaneously [7]. Songs are usually composed of multiple chords played in series. These series tend to repeat in different sections of the song, with each section having a specific progression of chords. By detecting and comparing the progressions present in a song, they can be compared individually with chord progressions in other songs and checked for a match.

At the signal level, a very useful property for signal comparison is its envelope [8]. In physics and engineering, the envelope function of an oscillating signal is a smooth curve outlining its extremes. The envelope thus generalizes the concept of a constant amplitude. The envelope function may be a function of time, space, angle, or indeed of any variable. By comparing the envelopes of two song files, it is possible to see how similar their characteristic waveforms are. In order to further establish the similarity between two songs, an attempt can be made to classify both songs based on genre. Genre of a song is any category of music to which it belongs, based on certain stylistic properties. Genre itself is a rather subjective property, making it far more difficult to assign a definite value. Classification based on genre is achieved using machine learning principles [9]. A large database of songs whose genre is already known is fed to the system. Using this sample set, the system is able to determine the likelihood of a particular song being of a certain genre.

The paper is organized into four main sections. Section II presents a detailed survey of existing work on feature extraction from music files, and attempts to determine similarity between songs. Section III illustrates the methodology of the proposed comparison method in detail. Section IV lists out the observations and calculations from the performed experimentation and Section V concludes the work with future directions.

II. Literature Survey

Computer music research is an emerging area of interest for researchers working on pattern recognition and machine learning techniques. Digital music databases containing digitized music files, and sequenced, or otherwise structurally represented music must be organized, indexed, and made

978-1-5090-2597-8/16/$31.00

easily accessible. A method of performing this task is known as Music Information Retrieval (MIR).
The descriptions for content-based search and retrieval of multimedia documents like MPEG-7 require standardization, and efforts are already being made to develop such standards.

Among the multiple tasks required to develop MIR, one significant task is modelization of music style. Using machine learning methods, computers can be trained to detect distinct properties that are characteristics of different music genres. So that, they can be used to search for that kind of music over vast musical databases. The same methodology is well suited to train systems to detect stylistic features of composers or even to model a user's own individual musical taste. One more interesting application of such a system would be its use with automatic composition algorithms to guide composition of new musical pieces in accordance with a given stylistic profile [10].

The proposal by *Pampalk et al.* postulates a new approach by combining information extracted from audio with meta-data like artist or genre [11]. Specifically, it extracts spectrum and periodicity histograms to approximately describe timbre and rhythm respectively. For each of these similarity parameters, the collection is organized via a self-organizing map (SOM). The SOM arranges the pieces of music on a map such that similar pieces are located closer to one another as compared to dissimilar pieces. It uses smoothed data histograms to describe the cluster structure and to create figurative Islands of Music, with groups of similar compositions being visualized as islands. The proposal by Mesaros et. al [12] evaluates methods for singer identification in polyphonic music, based on pattern classification together with an algorithm for vocal separation. The methods are evaluated using a database of songs where the difference in levels between the vocals and the accompaniment varies. It was found that separation of vocal lines enables robust singer identification down to 0dB and a -5dB singer-to-accompaniment ratio. *Shmulevich et. al* put forward several perceptual issues in the context of recognition of music patterns using machine learning [13]. The work discusses several metrics of rhythmic complexity which can be used to determine relative weights of pitch and rhythm errors. Further, a new method to determine localized tonal context is also discussed. This method involves the use of empirically derived key distances. The generated key assignments are then used to derive the perceptual pitch error criterion, which is based on note relatedness measures obtained from experiments with human listeners.

In this paper, an effort has been made to identify a similarity factor between songs based on fundamental properties of music such as tempo, key, and chord progression, as well as a more subjective characteristic like genre. The similarity factor between two songs has significant commercial application, such as the automatic generation of playlists, as well as copyright protection. This serves as motivation to design a system which can identify this similarity factor purely based on signal level analysis, without the use of existing metadata.

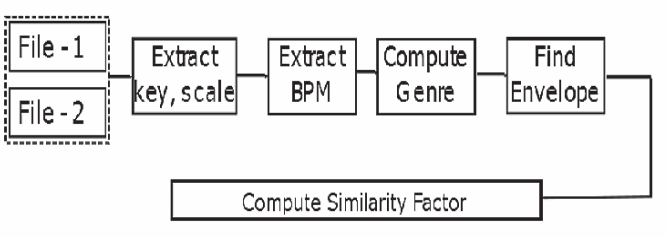

Fig. 1. Proposed methodology for finding the similarity between two song files

III. Methodology

For testing purposes, the sample songs are mostly pairs of songs with a high degree of similarity, for example, a song and an independent cover of the song, or the same song sung in different languages. Short instrumental pieces are also taken to check key and chord progression detection. Initial testing was performed almost exclusively on very short clips, around ten to fifteen seconds long. All songs were collected at high sampling frequency of 44.1 KHz and with a storage space of 16-bits per sample. Subsequently, the files were also converted from stereo to mono sound, in order to avoid any loss of information by analyzing only the left or right channel's audio input. The process of computing similarity among two song files is clearly depicted in Fig. 1. The process at every block is detailed in the subsequent subsections.

A. Feature Extraction

The first characteristic feature to be detected is the *tempo*. As stated earlier, the tempo is measured in beats per minute (BPM). The unit itself somewhat describes the method of detecting tempo. The number of beats occurring in a particular time interval is detected and then the time interval is extrapolated to sixty seconds. The problem becomes how to detect beats in a song. The tempo in a song is usually maintained using a constant percussive pulse. Using this information, it was postulated that the tempo could be calculated by finding these percussive pulses and finding the time interval between them. The percussive pulses can be found by looking for peaks in the songs waveform. So, the final method involved detecting peaks in the songs waveform, then finding the interval of time between these peaks. Such a method gives us a fairly accurate estimate of the tempo of the song [14]–[16]. Another useful technique to identify the beats appearing at a particular time interval is autocorrelation. [17].

After detecting tempo, the next property to be extracted is the chord progression present in the song. Each chord is associated with a particular note in musical theory. For example a *C chord* is associated with the note C. Correspondingly, each note is equivalent to a particular fundamental frequency. If we take C6, its fundamental frequency is 1046.50 Hz. Similarly, there exist definite values for all chords that can be played in a song. By searching for these frequency values at regular intervals of the song, the

chords which have been played to make up the given sample song can be established. These chords are input to an array and stored. This array gives the chord progression of the song [18], [19]. After extracting key from this array, the repeating chord values are merged to a single value. It is possible to extract the *key* of a song from the chord progression array [20]. As stated, key is a group of pitches, or scale upon which a music composition is created. Therefore, it can be assumed that the key of the song is the same as the chord which is played most frequently. By analyzing the array containing the chord progression, it is easy to find most frequently occurring chord and establish it as the key of a song.

The genre of a song file can also be determined using machine learning techniques [21]. A large collection of songs with pre-determined genre were used as a training set for the machine learning model. This generates a usable support vector machine (SVM) [22], which can be used to detect patterns in songs that are characteristic of a particular genre. Passing the test song to the through the SVM generates probability values for each potential genre, and the most likely genre is selected as genre of the song. The process of genre extraction is simplified in Fig. 2.

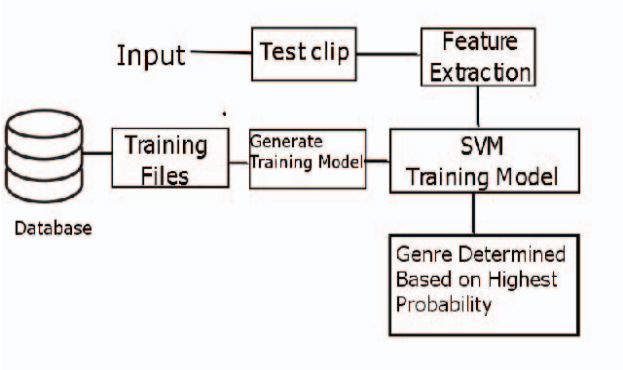

Fig. 2. The process of estimating the genre of a song clip using an SVM classifier.

The last feature considered is the envelope of a signal. The signal's envelope is equivalent to its waveform's outline. The function of an envelope detector is simply to connect all the peaks in the signal [23], [24]. This envelope detection method involves squaring the input signal and sending the resultant signal through a low pass filter. Squaring the signal acts to demodulate the input by using itself as its own carrier wave. This means that effectively half the energy of the signal is shifted up to higher frequencies and half is shifted down towards down conversion. This signal is then down-sampled to reduce the sampling frequency. Downsampling can be performed as long as the signal does not have any high frequencies which could potentially cause aliasing. Otherwise an finite impulse response (FIR) decimation should be used which passes the signal through a low pass filter before downsampling. After this, the signal is passed through a minimum phase, low pass filter to eliminate the high frequency energy [25]. Finally, the only component remaining is the signal envelope.

B. Algorithms for Comparison

Multiple algorithms are used in this tool to compare the above specified features of two songs. Two important algorithms are discussed here to compare the envelope of two signals and key characteristics.

1) All Common Subsequences (ACS): When comparing the characteristics of two songs for similarity, every portion of the first song must be compared to every portion of the second song using their respective extracted feature sets. To find the similarity, all common subsequences (ACS) between the two strings that are similar and longer than a predetermined threshold are considered. Similarity is based on a tolerance value which signifies the variation tolerance between two sequences of two songs. The length of these strings are used to score the sub-sequence and is calculated with the global similarity score. This score is converted to percentage scale and returned [26]–[31]. Fig. 3. describes the ACS algorithm in detail and Algorithm 1 shows the needful steps to implement it. If a song has multiple similar parts, this algorithm takes it into consideration. For example, if one portion of the first song has a similar pitch envelope as multiple portions of the second song, all those portions are detected and added to the global score for calculation [32].

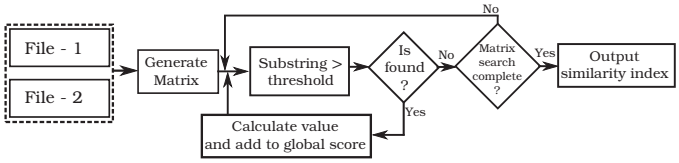

Fig. 3. Finding all common sub-sequences of two envelop signals.

2) Key Similarity: To find key similarity, values are provided to each key based on the circle of fifths (COF) [33], [34] This a cyclic list of keys which shows the progression of keys.

$COF : "C", "Em", "G", "Bm", "D", "F\#m",$
$"A", "C\#m", "E", "G\#m", "B", "D\#m", "F\#",$
$"A\#m", "C\#", "Fm", "G\#", "Cm", "D\#", "Gm",$
$"A\#", "Dm", "F", "Am"$

Based on the COF, the cyclic distance between two keys is found. This value is then converted to a percentage value shown in Eq. 1.

$$\% = \frac{(len(COF) - D)}{(len(COF))} * 100 \quad (1)$$

Where D is the difference computed from Algorithm 2 and $len(COF)$ is the total number of elements in the COF array.

IV. Results and Observations

Initially the envelope comparison was tested individually before utilizing tempo, key, chord progression and genre. After having established an accurate method of detecting the signal

TABLE I
COMPARISON OF SIX PAIRS OF MUSICAL CLIPS AND THEIR RESULTS FOR INDIVIDUAL FEATURES

ID	BPM-1	BPM-2	Key-1	Key-2	Genre - 1	Genre - 2	BPM (in %)	Key (in %)	Envelope (in %)
For similar song pairs									
1	87.08	87.52	F# Major	F# Major	Rhythm	Rhythm	98.21	100.00	100.00
2	131.34	131.27	E Major	E Major	Rhythm	Rhythm	99.94	100.00	100.00
3	143.79	143.55	A Minor	A Major	Hip Hop	Rhythm	99.83	100.00	100.00
4	106.43	105.98	A Minor	E Major	Rhythm	Hip Hop	99.58	91.67	100.00
For dissimilar song pairs									
5	142.96	140.09	A# Major	D# Minor	Hip Hop	Rhythm	98.00	91.67	17.58
6	143.69	143.79	B Minor	B Minor	Rhythm	Rhythm	99.93	100.00	6.36

Algorithm 1 ALL COMMON SUBSEQUENCES ALGORITHM

procedure ACS($a, b, thresh, tolerance$)
▷ The similarity score
  $alen \leftarrow len(a)$
  $blen \leftarrow len(b)$
  $c[\,] \leftarrow 0$
  for $i = 1$ to $alen$ **do**
    for $j = 1$ to $blen$ **do**
      $c(i, j) = abs(a(i) - b(j))$
    end for
  end for
  $sum \leftarrow 0$
  for $i = 1$ to $alen$ **do**
    for $j = 1$ to $blen$ **do**
      $len \leftarrow 0$
      $m \leftarrow i$
      $n \leftarrow j$
      while $m > 0$ and $n > 0$ and $c(m, n) >= 0$ and $abs(c(m, n) - c(i, j)) < tolerance$ **do**
▷ tolerance is the maximum difference in value that will be considered
        $len \leftarrow len + 1$
        if $len >= thresh$ **then**
          $c(m, n) \leftarrow -1$
        end if
        $m \leftarrow m - 1$
        $n \leftarrow n - 1$
      end while
      if $len >= thresh$ **then**
        $sum \leftarrow sum + len$
      end if
    end for
  end for
  $avg \leftarrow (alen + blen)/2$
  $score \leftarrow sum/avg$
  $n \leftarrow n - 1$
  return $score$
end procedure

envelope, and developing an algorithm to compare different signal envelopes, the other extra parameters were added to gain higher accuracy.

Algorithm 2 KEY SIMILARITY ALGORITHM

$val1 \leftarrow COF.index(key1)$
$val2 \leftarrow COF.index(key2)$
$D \leftarrow abs(val1 - val2)$
if $D > len(COF)/2$ **then**
  if $val1 == min(val1, val2)$ **then**
    $D = len(COF) - val2 + val1$
  else
    $D = len(COF) - val1 + val2$
  end if
end if

A high degree of accuracy was observed when the obtained results for tempo, key, chord progression, and genre were compared with their correct values. Genre, being a slightly subjective quantity, was vulnerable to some error, but tests proved accurate. In Table I the detected values for test pairs of song clips are. The first four pairs were clips taking from corresponding portions of songs which have been translated into different languages. These samples differ only in lyrical content, so a high degree of similarity is detected based on the chosen parameters. The last two test cases are completely different test samples, hence a lesser degree of similarity is detected, particularly in the envelope section. The detected values for BPM, Key, and Genre in sample 1 and 2 are shown in columns 2-7. The similarity percentages between the detected values of BPM, Key, and Envelope for each test pair are shown in columns 8-10

V. CONCLUSION AND FUTURE WORK

It has been shown that songs can be compared accurately using certain fundamental musical descriptors. Such a system can be used for a music recommendation system, providing far more accurate and useful results than groupings based purely on existing metadata.

This system can always be made more accurate by adding more features to the list of parameters used for comparison. As long as the features extracted are accurate, they can be used to make the similarity results more precise. The possibility of incorporating new features such as timbre, rhythmic patterns, and individual instrument extraction warrants further investigation.

REFERENCES

[1] G. Graham, G. J. Lewis, G. Graham, and G. Hardaker, "Evaluating the impact of the internet on barriers to entry in the music industry," *Supply Chain Management: An International Journal*, vol. 10, no. 5, 2005, pp. 349–356.

[2] E. Pampalk, A. Flexer, G. Widmer *et al.*, "Improvements of audio-based music similarity and genre classificaton." *ISMIR*, vol. 5. London, UK, 2005, pp. 634–637.

[3] L. Barrington, A. Chan, D. Turnbull, and G. Lanckriet, "Audio information retrieval using semantic similarity," *IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, vol. 2. IEEE, 2007, pp. II–725.

[4] B. Logan, D. P. Ellis, and A. Berenzweig, "Toward evaluation techniques for music similarity," *The MIR/MDL Evaluation Project White Paper Collection*, vol. 3, 2003, pp. 81–85.

[5] E. D. Scheirer, "Tempo and beat analysis of acoustic musical signals," *The Journal of the Acoustical Society of America*, vol. 103, no. 1, 1998, pp. 588–601.

[6] B. Maess, S. Koelsch, T. C. Gunter, and A. D. Friederici, "Musical syntax is processed in broca's area: an meg study," *Nature neuroscience*, vol. 4, no. 5, 2001, pp. 540–545.

[7] T. Fujishima, "Realtime chord recognition of musical sound: A system using common lisp music," *Proc. ICMC*, vol. 1999, 1999, pp. 464–467.

[8] J. Paulus and A. Klapuri, "Measuring the similarity of rhythmic patterns." *ISMIR*, 2002.

[9] T. Li, M. Ogihara, and Q. Li, "A comparative study on content-based music genre classification," *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, ACM, 2003, pp. 282–289.

[10] P. J. P. De Leon and J. M. Inesta, "Pattern recognition approach for music style identification using shallow statistical descriptors," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 37, no. 2, 2007, pp. 248–257.

[11] E. Pampalk, S. Dixon, and G. Widmer, "Exploring music collections by browsing different views," *Computer Music Journal*, vol. 28, no. 2, 2004, pp. 49–62.

[12] A. Mesaros, T. Virtanen, and A. Klapuri, "Singer identification in polyphonic music using vocal separation and pattern recognition methods." *ISMIR*, 2007, pp. 375–378.

[13] I. Shmulevich, O. Yli-Harja, E. Coyle, D.-J. Povel, and K. Lemström, "Perceptual issues in music pattern recognition: Complexity of rhythm and key finding," *Computers and the Humanities*, vol. 35, no. 1, 2001, pp. 23–35.

[14] G. Tzanetakis and P. Cook, "Musical genre classification of audio signals," *IEEE Transactions on speech and audio processing*, vol. 10, no. 5, 2002, pp 293–302.

[15] G. Tzanetakis, "Tempo extraction using beat histograms," *Proceedings of the 1st Music Information Retrieval Evaluation eXchange (MIREX 2005)*, 2005.

[16] M. F. McKinney, D. Moelants, M. E. Davies, and A. Klapuri, "Evaluation of audio beat tracking and music tempo extraction algorithms," *Journal of New Music Research*, vol. 36, no. 1, 2007, pp. 1–16.

[17] J. C. Brown, "Determination of the meter of musical scores by autocorrelation," *The Journal of the Acoustical Society of America*, vol. 94, no. 4, 1993, pp. 1953–1957.

[18] J. Minamitaka, "Technique for selecting a chord progression for a melody," Jun. 8 1993, US Patent 5,218,153.

[19] H. Papadopoulos and G. Peeters, "Simultaneous estimation of chord progression and downbeats from an audio file," *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2008, pp. 121–124.

[20] D. Temperley, *Music and probability*, The MIT Press, 2007.

[21] N. Scaringella, G. Zoia, and D. Mlynek, "Automatic genre classification of music content: a survey," *IEEE Signal Processing Magazine*, vol. 23, no. 2, 2006, pp. 133–141.

[22] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "Liblinear: A library for large linear classification," *Journal of machine learning research*, vol. 9, no. Aug, 2008, pp. 1871–1874.

[23] C. Xu, Y. Zhu, and Q. Tian, "Automatic music summarization based on temporal, spectral and cepstral features," *Multimedia and Expo, 2002. ICME'02. Proceedings. 2002 IEEE International Conference on*, vol. 1, IEEE, 2002, pp. 117–120.

[24] B. Gold, N. Morgan, and D. Ellis, *Speech and audio signal processing: processing and perception of speech and music*, John Wiley & Sons, 2011.

[25] K. Larkin, "Efficient demodulator for bandpass sampled am signals," *Electronics Letters*, vol. 32, no. 2, 1996, pp. 101–102.

[26] D. S. Hirschberg, "A linear space algorithm for computing maximal common subsequences," *Communications of the ACM*, vol. 18, no. 6, 1975, pp. 341–343.

[27] J. W. Hunt and M. MacIlroy, *An algorithm for differential file comparison*, Citeseer, 1976.

[28] J. W. Hunt and T. G. Szymanski, "A fast algorithm for computing longest common subsequences," *Communications of the ACM*, vol. 20, no. 5, 1977, pp. 350–353.

[29] C. Rick, "A new flexible algorithm for the longest common subsequence problem," *Annual Symposium on Combinatorial Pattern Matching*, Springer, 1995, pp. 340–351.

[30] H. Goeman and M. Clausen, "A new practical linear space algorithm for the longest common subsequence problem," *Kybernetika*, vol. 38, no. 1, 2002, pp. 45–66.

[31] R. I. Greenberg, "Fast and simple computation of all longest common subsequences," *arXiv preprint cs/0211001*, 2002.

[32] T. En-Najjary, O. Rosec, and T. Chonavel, "A voice conversion method based on joint pitch and spectral envelope transformation." *INTERSPEECH*, 2004.

[33] J. C. Bartlett and W. J. Dowling, "Recognition of transposed melodies: a key-distance effect in developmental perspective." *Journal of Experimental Psychology: Human Perception and Performance*, vol. 6, no. 3, 1980, p. 501.

[34] J. Clough and G. Myerson, "Musical scales and the generalized circle of fifths," *The American Mathematical Monthly*, vol. 93, no. 9, 1986, pp. 695–701.