Music recognition using MPEG-7

Konstantin Brand
Department of Applied Media Systems
Ilmenau University of Technology
P. O. Box 100565, D-98684 Ilmenau, Germany
Email: konstantin.brand@tu-ilmenau.de

***Abstract* — With the prevalence of digital media, there is a substantial growth of digital audio, which encourages to find new methods for indexing and characterizing music. The goal of this project is to create an audio recognition framework using the Moving Picture Experts Group (MPEG)-7 standard. Its focus will be on gaining a unique content identifier, so called Fingerprint (FP). Hereby the low-level descriptor audio-spectral-flatness is used to generate a unique description of an audio file and then is further computed to the high-level description scheme called audio-signature. To find a match the euclidean distance of a sample and 50 references is calculated element wise and the lowest distance describes the prediction. With a 5 second interval an average accuracy of 96% can be achieved.**

***Index Terms*—Music Information Retrieval, metadata, MPEG-7, Fingerprint**

I. Introduction

The amount of music files available in the world grew very fast in the last decade. Therefore, it becomes more and more important to identify music in a unique way. As the main interest may be due to copyright issues, also for the private use music recognition can be very handy to handle own databases or for scientific research it can be used for musicological studies.

As there are many ways to calculate a fingerprint, presented for example from Google or Shazam [10], a standardized way can benefit building up a big database with the help of many individuals and gives an easy way to communicate. Therefore, the "metadata" standard of the Moving Picture Experts Group (MPEG) offers a way to achieve these goals.

In cooperation with the House of Research in Berlin the motivation of this paper yields from the project on analyzing radio stations from all over the world, in particular Germany. The goal is to profile different stations and find statistical information like: how much music is played, how often do the songs vary, what type of music is offered, numbers commercial brakes etc. As this consumes a lot of hours for a human, the objective is to automate the recognition of a song. To achieve this goal, this paper presents the MPEG-7 - Fingerprint algorithm and its implementation in Python. Thus, the music identification is provided.

Several previous work on MPEG-7 and Audio recognition has already been done [2], [3], [6]. They prove that the provided FP algorithm is robust to noise and can distinguish cover songs. The values proposed from the MPEG-7 standard have been crosschecked in [6], too. In Python [1] has rebuild a Fingerprint algorithm similar to shazam. Furthermore, he provides his own way of calculating, which unfortunately is not a standardizer way and builds up on his own expertise which has no fundamental background research.

This paper will start by describing the MPEG-7 metadata standard, then the calculation of the Fingerprint will be explained, additionally the comparison computation to the database will be discussed and at last the performance will be presented and reviewed.

II. Mpeg 7 Audio

In 2002, the Moving Picture Experts Group (MPEG) released the standard called ISO/IEC-15938 with actual name "Information Technology Multimedia Content Description Interface" [5]. Today, it is referred to as MPEG-7 and is a multimedia content description standard. The distinctive feature of this standard is that it does not deal with the actual encoding of audio. Furthermore, it uses the Extensible Markup Language (XML) to store metadata, also called additional descriptive data, for representing information about the content, but not the content itself. The advantage of this approach is that the additional information can be attached (multiplexed) to a time-code in order to tag certain events, or for example to synchronize lyrics to a song [8].

Following tools were designed in the standard [9] :

- **Discriptor (D):** Is a syntactically and semantically defined feature, so that many descriptors can be used on an unique object.
- **Description Scheme (DS):** Specifies the relations between its components and defines the structure and semantics.
- **Description Definition Language (DDL):** Uses the Extensible Markup Language (XML) to define the structural relations between descriptors. It allows the creation of new Descriptors and also the modification of Descriptopn Schemes.
- **System tools:** These are special tools to deal with synchronization, transport and storage of descriptors.

MPEG-7 is divided into 8 parts each concerning a different focus of metadata retrieval. For this paper part 4 which

concerns Audio is the most relevant, where all information like: author, genre, time segment etc. can be stored. Obviously, these information can not be retrieved directly from the sound waves, therefore descriptors are defined to describe the wave. They are separated in low-level-descriptors and high-level-description-schemes. Low-level descriptors are directly referring to the wave itself (wave form, envelop, spectral flatness, etc.) and high-level-description schemes evolve from low level descriptors and have a higher abstraction (audio signature, musical instrument timbre, melody, etc.) [4], [7]

III. Data model

For music recognition and Fingerprint calculation the MPEG-7 standard provides the audio spectral-flatness D and audio-signature DS. Below the calculation of these descriptors will be described including all parameters set by the standard.

First, the song is divided into window blocks of 30ms (1323 samples for a sampling rate of 44100). This offers a way to analyze the song in a much more detailed way, as every window can be investigated separately. The window size is proposed from the standard and is also called hop size or just hop. As every song varies in length, it is obvious that some songs can not be divided exactly into these window sizes, so it is necessary to add zeros at the end, so that every hop has full values.

By strictly cutting of the edges of the windows, a hamming-window of the same size has to be applied to each hop, to avoid the leakage effect.

Thus, the FFT with size of 2048 samples is calculated for every hop separately. As only real valued audio signals are considered, solely positive frequencies are of interest. The FFT is normalized by multiplying each element with:

$$\frac{1}{\sqrt{n_{fft} \cdot \sum_{i=0}^{w} (g_i)^2}} \tag{1}$$

Where n_{fft} is the FFT size of 2048 samples, w is the window size which corresponds to 1323 samples and g_i are the elements of the hamming window. The absolute value of the remaining FFT is then calculated to focus on real values only. [1]

To calculate the Spectral flatness, the Power Spectral Density (PSD) is needed which can be computed by squaring the FFT.

For a higher level of detail, the PSD is further divided into 1/4 octave bands, starting at 250Hz and ending at 16kHz with an overlap of 10%. This will lead to 24 subbands and will help to achieve robustness of the song recognition. If the band width is chosen smaller a slight pitch-shift could lead to a wrong recognition whereas broader bands will lead to mismatching if songs are similar [3].

The spectral-flatness for each subband b is calculated with [10]:

$$\boldsymbol{f}_{n,b} = \frac{\sqrt[\boldsymbol{h}(b)-\boldsymbol{l}(b)+1]{\prod_{i=\boldsymbol{l}(b)}^{\boldsymbol{h}(b)} \boldsymbol{c}(i)}}{(1/(\boldsymbol{h}(b)-\boldsymbol{l}(b)+1)) \sum_{i=\boldsymbol{l}(b)}^{\boldsymbol{h}(b)} \boldsymbol{c}(i)} \tag{2}$$

This describes the geometric mean divided by the arithmetic mean, $\boldsymbol{c}(i)$ is the PSD, $\boldsymbol{h}(b)$ and $\boldsymbol{l}(b)$ are the lower and upper indexes of $\boldsymbol{c}(i)$ within subband b and n is the corresponding hop.

The spectral-flatness will return a value between 0 and 1 for each subband, describing how "peaky" the spectrum is. Herby, 0 corresponds to white noise and 1 represents a single sinusoidal wave. Thus each subband contains information of the energy from the spectrum in these boundaries.

This will result into an array of 24 flatness values for each hop and will be combined to a matrix:

$$\boldsymbol{F} = \begin{bmatrix} f_{0,0} & f_{0,1} & \cdots & f_{0,23} \\ f_{1,0} & f_{1,1} & \cdots & f_{1,23} \\ \vdots & \vdots & \ddots & \vdots \\ f_{n,0} & f_{n,1} & \cdots & f_{n,23} \end{bmatrix} \tag{3}$$

for $\boldsymbol{F} \in \mathbb{R}^{n \times 24}$ and n is the number of divided hops. In the matrix $\boldsymbol{F}$ the columns represent the subbands and the rows the time in 30ms. This results in a unique representation of a song and is already enough to detect songs [3].

For further compression the audio-signature is proposed. Here, the mean of 34 rows in time direction is calculated. In the matrix $\boldsymbol{F}$ each line then represents 24 subbands of 1 sec from a song. This algorithm has been proposed in [10].

IV. Match Sample with Database

After determining the fingerprint the challenge is to detect a song with its help. Hence, a fingerprint of a short sample is calculated which leads to a smaller sample matrix $\boldsymbol{Q} \in \mathbb{R}^{m \times 24}$ with same number of columns but less rows m ($m \leq n$), due to a shorter time length of the sample.

In order to detect a match, a sliding comparison [10] of matrix $\boldsymbol{F}$ and $\boldsymbol{Q}$ is computed, where all rows of Q are vertically slided over the matrix $\boldsymbol{F}$. This way $\boldsymbol{Q}$ can be mapped to $\boldsymbol{F}$, if at some point the matrices are the same or very similar, it is most likely that the sample is part of the song. Accordingly for each slide the euclidean distance is calculated as follows:

$$\boldsymbol{e}_{Q,F}(k) = \sum_{i=0}^{m} \sum_{j=0}^{23} |q_{i,j} - f_{i+k,j}| \tag{4}$$

[1] If zeros have been added make sure to correct the last hop by multiplying each element with: $\sqrt{\frac{w}{(w - n_{zeros})}}$ where w is the window size (1323 samples) and n_{zeros} are the number of zeros added.

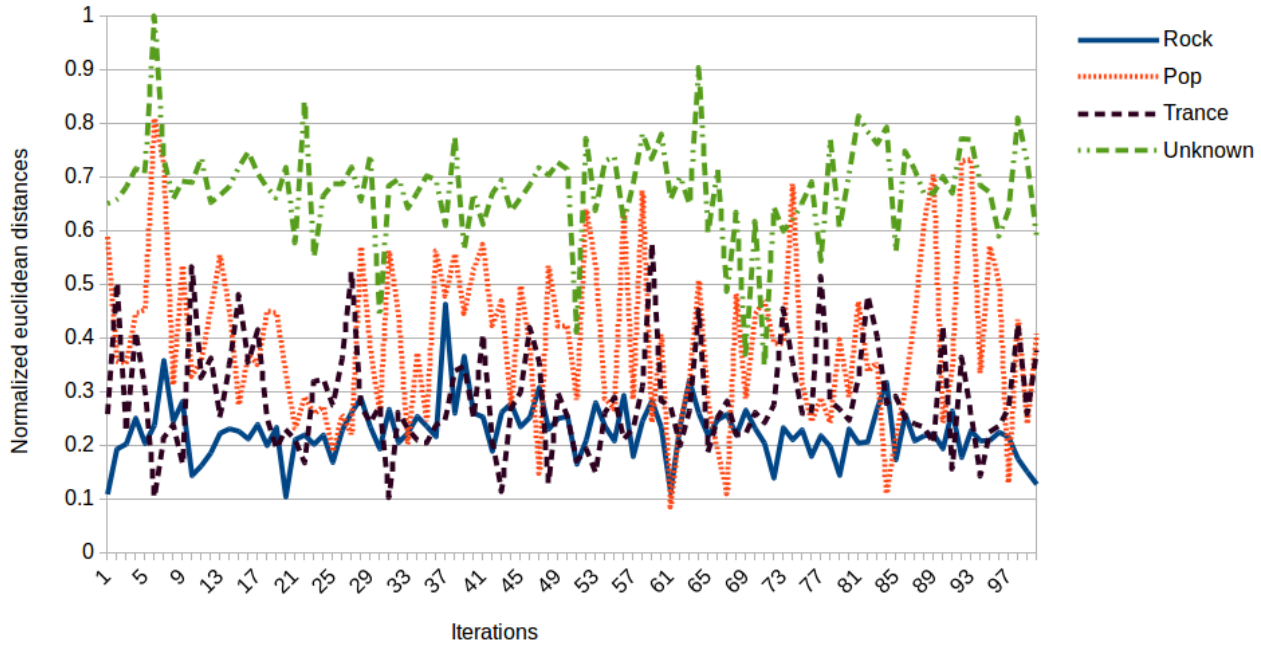

Fig. 1: Performance comparison of known songs and an unknown

which will result in a vector $\boldsymbol{e}_{Q,F} \in \mathbb{R}^{(m-n+1)}$ that contains the euclidean distances of every comparison. The minimum distance will tell how similar the sample and the song are:

$$d_{min} = \min \boldsymbol{e}_{Q,F}(k) \tag{5}$$

By comparing many songs the minimum will be the predicted song.

V. Results

For conducting the experiment a database of 50 songs are collected and their fingerprints are calculated and stored. To evaluate the algorithm a random sample of one song is taken and the minimum euclidean distances to every song in the database are distinguished. For further inspection of the robustness a 5 sec, 10 sec, 15 sec, and 20 sec random interval of the song is chosen. The experiment was conducted 100 times. Furthermore this is rerun for different genres, rock, pop and trance.

Sample size	Rock	Pop	Trance	Average
5 sec	89%	100%	100%	96,33%
10 sec	99%	100%	100%	99,33%
15 sec	99%	100%	100%	99,33%
20 sec	100%	100%	100%	100%

Table 1: Recognition accuracy for different sample sizes and their genres

In Table 1 the results of the experiment are presented. It shows that for a 20 sec sample interval a song can be correctly detected by 100% for all genres. This proves that the algorithm works and that it is possible to do music recognition with the MPEG-7 fingerprint algorithm. For smaller samples the detection for the Rock samples looses accuracy, but Pop and Trance remain at 100%. This can be explained by the many silent or instrumental parts of the rock songs, where only few instruments play (mostly guitar, base and drums). As for the FP calculation, a song is divided in a window 30 ms (with the audio signature even 1 sec), the random sample can start at any time, for example at the middle or even the end of the window. This offset my lead to a wrong detection. Comparing the euclidean distances of a mismatch they are often off by the second digit.

As a further step, an unknown song has been taken and the performance of the algorithm is shown in Fig **??**. It can be seen that an unknown song has a higher euclidean distance then the known songs. This can lead to threshold to distinguish whether the song is contained in the database. As no noise has been added this theory is not fully determined.

VI. Conclusions

In this paper, the MPEG-7 fingerprint algorithm was explained for identifying music from a database. This shall help the House of Research Berlin to profile radio stations through out the world. It has been proven that the MPEG-7 standard provides a robust method to identify music for a 10 sec sample, with a detection rate of 99.33%.

VII. Future Work

Despite algorithm robustness the disadvantage of the MPEG-7 fingerprint method is that a large fingerprint database is needed. Therefor the fingerprint of every song needs to be calculated and this is obviously difficult to achieve. Therefore, a technique which distinguishes a beginning and end of a song from a radio program could be implemented, and the fingerprint could be calculated automatically out of the source. Unfortunately the song name would not be associated with the fingerprint, and human effort is still needed. Another problem is that the comparison of a sample to a database takes much computational effort. With millions of songs in the database the computational time can take hours because it hast to run a comparison with every line of each FP. Hence, a novel method is required to optimize the search algorithm.

References

[1] Will Drevo, "Audio Fingerprinting with Python and Numpy", 2013.

[2] Oliver Hellmuth, Eric Allamanche, Markus Cremer, Holger Grossmann, Jurgen Herre, and Thorsten Kastner, "Using MPEG-7 Audio Fingerprinting in Real-World Applications".

[3] Oliver Hellmuth, Jurgen Herre, Eric Allamanche, Markus Cremer, Thorsten Kastner, and Wolfgang Hirsch, "Advanced Audio Identification Using MPEG-7 Content Description".

[4] Michael Honig, "Extraktion und Beschreibung von Metadaten aus Audiodateien mittels MPEG-7".

[5] ISO/IEC JTC 1/SC 29, "Information Technology — Multimedia Content Description Interface — Part 4: Audio", 2001-06-9.

[6] Thorsten Kastner, Eric Allamanche, Jurgen Herre, Oliver Hellmuth, Markus Cremer, and Holger Grossmann, "MPEG-7 Scalable Robust Audio Fingerprinting".

[7] José M. Martínez, "International Organisation for Standardisation Organisation Internationale de Normalisation ISO/IEC JTC1/SC29/WG11 Coding of moving pictures and audio".

[8] S. Quackenbush and A. Lindsay, "Overview of MPEG-7 audio", *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 6, pp. 725–729, 2001.

[9] Anneke Winter, "MPEG-7: Überblick und Zusammenfassung".

[10] Shingchern D. You, Wei-Hwa Chen, and Woei-Kae Chen, "Music identification system using MPEG-7 audio signature descriptors", *TheScientificWorldJournal*, vol. 2013, pp. 752464, 2013.