RESEARCH ARTICLE OPEN ACCESS

A Review on Audio Fingerprinting and the Methods Behind

Akansha Rastogi[1], Avinash Kumar Mishra[2], Trishla Shrivastava[3], Gaurav Agarwal[4]
[1,2,3]Student,[4] Assistant Professor
Department of Computer science and engineering ABES Institute of Technology, Ghaziabad, Uttar Pradesh

Abstract:

An Audio Fingerprinting is a content - based compact signature that summarizes an audio recording. This technology has attracted a lot of attention since they allow the identification of audio uniquely. Other uses of fingerprinting include: broadcasting, monitoring, filtering technology. The different approaches to fingerprinting have been described are: Pattern matching, Robust Hashing. In this paper, we survey and evaluate different techniques and previous algorithms.

***Keywords* — Audio fingerprinting, content-based audio identification, robust hashing, integrity verification, audio information retrieval, watermarking.**

I.INTRODUCTION

An Audio Fingerprint is essentially a hash function that maps an audio object of a large number of bits to a 'fingerprint' of only a few number of bits. The audio object can be uniquely identified from this bit string. Audio fingerprinting provides the ability to derive a compact representation which can be efficiently matched against other audio clips. With smart phones becoming ubiquitous, there are several applications of audio fingerprinting on mobile devices. It works by combining two basic elements: a database of unique audio fingerprinting of millions of audio files, and software that can process and analyze entries while searching that the database for match. The most important applications for audio fingerprinting are content - based audio identification (CBID), content-based integrity verification (CBV) and watermarking support.[3]

A common use case is query-by-example music recognition; a user listens to a song in a restaurant, shopping mall or in a car, and wants to know more information about the song. This is usually achieved by computing a fingerprint for every known audio piece beforehand, resulting in a fingerprint database with the related meta-data; then, when confronted with an unlabeled input file, the corresponding fingerprint is computed and subsequently matched against the fingerprinting database, linking it to the formerly missing meta-data in case of a match with the database. This recognition works through a process called audio fingerprinting. Examples include: Shazam, SoundHound/Midomi, Chroma print, Eco print.[2]

Mobile query-by-example applications pose a unique set of challenges. First, the application has to be low-latency to provide users with an interactive experience. Second, the length of the required to get a match should be short for mobile applications. This paper aims to give a peek through the difficulties faced by the audio fingerprinting in mobiles and a comparison between the different algorithms developed.[2]

An ideal fingerprinting system should fulfill the following requirements. It should be able to precisely identify an item, despite of the

level of distortion and compression or interference in the transmission channel. It should be able to identify the titles from the unknown audio clip of only a few seconds. The fingerprinting system should be made such so that it has high computational efficiency. In real time application, efficiency is critical both in the generation of the fingerprint of the audio excerpt and, in the search for a best match in large database of fingerprints.

II. LITERATURE SURVEY

Before continuing on to the survey lets see how the fingerprinting algorithm works:
First, a set of fingerprints are extracted from the query song. The extraction of the fingerprint could be done ether way firstly it could be done at the uniform sampling rate or the focus could be kept only round the point of interest in the spectrogram. For mobile applications, it is critical that the individual fingerprint be robust against surrounding noise, compared to the corresponding database fingerprint.
Next step is that the query is compared with a database of tracks to find the corresponding match.
Database hashes actually contain more information than the hashes that are produced for unlabeled input audio. The database hashes that are later on used for matching are actually 64-bits structs, as they are additionally containing a track ID.[2]
Following are the points that differentiate among approaches done previously:

A. Accuracy*:* Tells number of correct identifications done, missed identifications identified, and wrong identifications (false positives).

B. Reliability*: A way* for assessing that a query is present or not in the database to identify is of utmost importance for copyright enforcement organizations. A song should not be identified as a match, if it has not been broadcasted, even if there exist an actual match.

C. Robustness*:* Ability to precisely identify an item, despite of the interference or distortion in the transmission channel. Other sources for distortion of the track are equalization, pitching, D/A-A/D conversion, background noise, audio coders (such as GSM and MP3), etc.

D. Granularity*:* Ability to identify whole titles from excerpts which is a few second long. It requires to deal with shifting, which lacks synchronization between the calculated fingerprint and those stored in the repository of items and it adds complexity to the search. It needs to compare track in all possible alignments.

E. Security*:* When dealt with the robustness requirement, the manipulations to deal with are designed to mislead the fingerprint identification algorithm.

F. Versatility*:* Ability to identify audio clip despite of the audio format. Ability to use the same repository for various applications.

G. Scalability: The ability of a computing process to be used or produced in a range of capabilities. This affects the accuracy of the system and the complexity.

H. Complexity*:* It refers to the computational costs of the fingerprint calculation, fingerprint size, the searching complexity, the complexity of the comparison of the fingerprint, the cost of adding new hashes to the repository, etc.

I. Fragility*:* Some applications like content-integrity verification systems, requires the detection of content changed. This is in contrast with the robustness requirement, as the fingerprint should be robust to content-preserving transformations. [1]

Generally, the fingerprint should be:
A never changing abstract of the recording: The fingerprint should maintain the maximum of track related information. This

abstract should allow the differentiation over a large number of fingerprints.

Invariant to distortions: It is derived from the robustness requirement. Content-integrity applications, however, relax this constraint for content-preserving distortions in order to detect intended manipulations.

Compact: A small size representation but not very small is good for complexity, since we need to store and compare a large number of fingerprints.

Easily computable: The extraction of the fingerprint should not be much time-consuming.[2]

Let us take a look on what has been in existence:

A. Ke, Hoiem and Sukthankar

Ke provides the deep understanding of the 1-D audio signals that can be processed as conventional images when we view them in the time-frequency spectrogram representation. The time-frequency spectrogram data is treated as a set of overlapping images. To compute a compact fingerprint on each image, the authors first train simple AdaBoost classifiers based on box-filters. What we get as an output is the binary value of each classifier. E.g., each classifier outputs either a 1 or a 0 based on the differences between values getting by the total sum of two sub-rectangular regions of the spectrogram image. The concatenated output of the set of classifiers is then used as a fingerprint of the spectrogram image.Ke and Haitsma use the same set of parameters for computing the spectrogram. The spectrogram, obtained by Short Term Fourier Transform (STFT).. Two fingerprints are considered to be a match if they have a Hamming distance <2, in the feature matching step of the retrieval pipeline.[2]

B. Cano , Kalker, Batle and Hatsima

Cano introduced a new framework completely different from the previous ones. In this framework, the fingerprint extraction have a front-end where the audio is been divided into frames and various number of discriminative and robust features are extracted from each frame. Later these features were transformed to a fingerprint by a fingerprint modelling unit which then compacts the representation of the fingerprint. The searching algorithm finds the best matching fingerprint in a huge database using various similarity measure. As we try to speed up the search process and try to avoid a sequential scanning of the repository, various strategies are used to quickly discard non-matching fingerprints. Various discussed audio fingerprinting algorithms are commercially brought into action, which shows the remarkable progress that has been made in this area. Of course, there is room for improvement in the quest for more robust, compact and discriminative fingerprints and efficient searching algorithms. The fact that how the identification framework can be extended to browsing and similarity retrieval of audio collections cannot be ignored at the same time. [1]

C. WANG

Wang proposes his theory looking only at spectrogram peaks. There are two reasons for choosing spectrogram peaks: First, spectrogram peaks are more likely to survive ambient noise. Second, spectrogram peaks satisfy the property of linear superposition, i.e., a spectrogram peak analysis of music and noise together will contain spectral peaks due to the music and the noise as if they were analyzed separately. Here we consider neighboring peaks.[2]

Shazam uses a smartphone's built-in microphone to capture an audio clip of the song being played. It creates a unique fingerprint based on the audio clip recorded and compares it against a central repository for a match. If a match is found, it sends back information like the artist name, song title, and album to the user.

The working is done by analysing the recorded audio clip and checking for a match based on the fingerprint in a database of more than a million songs. The identification of the song is based on an audio fingerprint which is

based on a spectrogram (time-frequency graph). Then it stores a catalogue of audio fingerprints in a repository. The user record a song for 10 seconds and the application generates an audio fingerprint. Once it generates the fingerprint of the audio clip, Shazam start searching for matches in the repository. If there exist a match, it returns the information back to the user; otherwise a "song not known" dialogue is returned.[4]

III. Conclusion

The following were observed in the study:

By defining more peaks in the Shazam's algorithm's frame (normally 5) would result in better accuracy but it will decrease the speed. In future we can try to implement the following:
Algorithms could be translated to survive a faster programing environment. More research should be done so that we can access the database more quickly. Other uses of the application should be investigated e.g. engine noise, gunshots etc.

IV. REFERNCES

[1].https://s3.amazonaws.com/academia.edu.documents/33553100/A_Review_of_Audio_Fingerprinting.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1522420202&Signature=IQbndkqhttD1S8MgESQPaZDvaR4%3D&response-content-disposition=inline%3B%20filename%3DA_Review_of_Audio_Fingerprinting.pdf

[2]. https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/37261.pdf

[3]. http://hpac.rwth-aachen.de/teaching/sem-mus-17/Reports/Froitzheim.pdf

[4]. https://en.wikipedia.org/wiki/Shazam_(company)