**International Journal of Advances in Engineering Research** http://www.ijaer.com

(IJAER) 2022, Vol. No. 23, Issue No. III, March **e-ISSN: 2231-5152, p-ISSN: 2454-1796**

# SURVEY AND EVALUATION OF AUDIO FINGERPRINT GENERATION AND SEARCHING ALGORITHMS

**Anshul Borawake, Nikhil Raul, Sangram Dhame, Richa Singh, Minal Shahakar**

*Computer Engineering, Dr. D. Y. Patil Institute Of Technology, Pimpri, Pune, INDIA*

## ABSTRACT

*We survey and evaluate popular audio fingerprinting algorithm implementation for hash generation method. We report and discuss results important for better performance, speed and accuray. We hope that the evaluation in this work will guide work towards reducing latency in fingerprint generation.*

## INTRODUCTION

We survey and evaluate popular audio fingerprinting algorithm implementation for hash generation method. We report and discuss results important for better performance, speed and accuray. We hope that the evaluation in this work will guide work towards reducing latency in fingerprint generation.

## PRIOR WORK AND MOTIVATION

State-of-the-art audio retrieval applications use a set of low level fingerprints extracted from the audio sample for retrieval. The fingerprints are typically computed on the spectrogram - a time frequency representation of the audio. Cano. T et al propose trade-off between dimensionality reduction and information loss. Miller, Rodriguez and Cox distortions, including additive noise, low pass filtering, subsampling, lossy compression and small offsets in the origin of the fingerprint. Distortions to a song will introduce distortions in the corresponding fingerprint. Obviously, the less the fingerprint is distorted, the easier the subsequent search will be. Kim et al propose a binary audio fingerprint that is compact for a fast DB search while minimizing the performance degradation. The audio fingerprint is based on the first-order normalized subband spectral moment. Jijun et al propose an audio fingerprinting algorithm based on spectral energy structure and NMF. Kamaladas et al access a novel fingerprinting technique based on wavelet transform that reduces the amount of audio fmgerprints. Ouali et al propose GPU implementation of a similarity search algorithm used in our audio fingerprinting system. To generate audio fingerprints, we convert the audio signal into a 2-D binary image derived from a spectrogram matrix by using the mean intensity values as a threshold.

The authors are not aware of a comprehensive evaluation of the different fingerprinting schemes in a common framework. In contrast, several such evaluations exist for image features in the computer vision community for contentbased image retrieval. Fingerprints developed like query by humming, singing and cover song detection are outside the scope of this paper. In particular, we are interested in factors such as the algorithms, speed improvement and performance boost. The questions that are most critical are:

- How much fingerprint data ( e.g. hash data ) does each algorithm generate ?
- Would GPU processing help in boosting the performance ?
- Are there other ways to store the fingerprint ?
- How do the different algorithm perform for really short (∼10 seconds) and noisy audio input captured from real life sources?
- How can various operations on the data help the performance of algorithm for generating and searching fingerprints?

## CONTRIBUTIONS

We survey and evaluate popular audio fingerprinting algorithm implementation for hash generation method. We report and discuss results important for better performance, speed and accuray. We hope that the evaluation in this work will guide work towards reducing latency in fingerprint generation. In Section 4, we identify various audio fingerprinting algorithms. In Section 5, we discuss the deep learning approach and provide experimental results.

## SURVEY OF AUDIO FINGERPRINTING ALGORITHM

Before we evaluate the audio fingerprint generation and searching algorithms, we will discuss the typical approach for fingerprint generation and searching approach. First, a group of fingerprints are extracted from the songs. The fingerprints could be extracted at uniform sampling rate, or only around points of interest in the spectrogram (e.g., spectrogram peaks). It is critical that individual fingerprints be robust against ambient noise, compared to the corresponding database fingerprint. Next the query is compared with a database of reference tracks to find candidate matches. To avoid pairwise comparison between the query and all of the reference tracks, the database is partitioned. The partitioning of the database is precomputed for the database, and each partition is associated with a list of database songs The partitioning on the database could be done by direct hashing of the fingerprints, Locality Sensitive Hashing or techniques based on Vector Quantization. This partitioning allows approximate-nearestneighbor-search as exact-nearest-neighbor search is infeasible in a database with billions of fingerprints. The inverted file for each cell consists of a list of song IDs and the timing offsets at which the fingerprints appear.

The timing information is used in the final step of the pipeline. Based on the number of fingerprints they have in common with the query probe from the inverted index, a short list of potentially similar database songs is selected from the database. Finally, a temporal alignment step is applied to the most similar matches in the database. Techniques like Expectation Maximization, RANSAC, or Dynamic Time Warping are used for temporal alignment. In the case of linear correspondence, a simple and fast technique that looks for a diagonal in the time vs-time plot for matching database and query fingerprints. The existence of a strong diagonal indicates a valid match. The temporal alignment step is used to get rid of false positives, and enables very high precision retrieval. In this Section, we first discuss the details of the algorithms and the motivation behind the approach, followed by system parameters suggested by the authors that provide good trade-off between accuracy and computational complexity.

### A. Cano P., Batle E., Kalker T. and Haitsma J.

*Description*

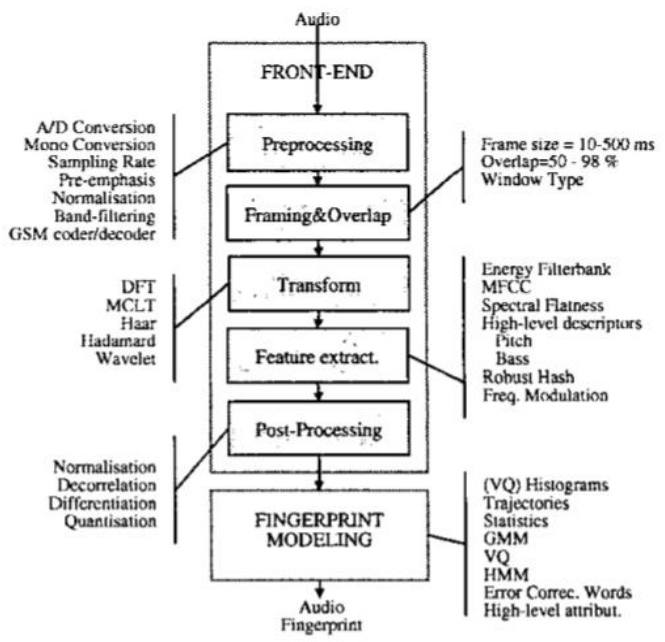
The solutions proposed to fulfill the above requirements imply a trade-off between dimensionality reduction and information loss. The fingerprint extraction consists of a front-end and a fingerprint modeling block. The front-end computes a set of measurements from the signal. The fingerprint model block defines the final fingerprint representation, e.g: a vector, a trace of vectors, a codehook, a sequence of indexes to HMM sound classes, a sequence of error correcting words or musically meaningful high-level attributes.

*System Parameters*

Given an audio input to the system, pre-processing is the first step where A/D conversion, mono conversion, sampling rate, pre-emphasis, normalisation, band-filtering and GSM coder/decoder operations are performed. The framing is performed for a frame size of 10.5ms with overlapping of 50- 98%. In the next step we transform the data using DFP, MCLT Haar Hadamard Wavelet, feature extraction parameters depend on Energy Filterbank MFCC, spectral flatness, high-level descipators, pitch, bass, robust hash and frequency modulation. Post preprocessing model include normalisation, decorrelation, differentiation and quantisation of audio and then finally the audio signal is passed to the fingerprint generation model.

The model generates histograms, trajectories, statistics GMM, VQ, HMM and error correction words and high-level attribute.

*System Design*



**B. Matthew L. Miller, Manuel Acevedo Rodriguez and Ingemar J.**

*Description*

Audio fingerprinting falls within the domain of classical pattern recognition and must therefore solve both the representation and matching problems. The problem is more tractable than many pattern recognition problems (e.g. three dimensional face recognition) in that the range of distortions is relatively small. Nevertheless, the problem is still difficult because, for the applications we envision, the database must be very large, perhaps one million songs. If we assume approximately 10,000 unique fingerprints per song, this means we may wish to search through about 10 billion fingerprints. Ideally, the fingerprint should be a compact representation that is invariant to a variety of common 286 Miller, Rodriguez and Cox distortions, including additive noise, low pass filtering, subsampling, lossy compression and small offsets in the origin of the fingerprint. Distortions to a song will introduce distortions in the corresponding fingerprint. Obviously, the less the fingerprint is distorted, the easier the subsequent search will be.

In this paper, we develop an approximate search algorithm for high dimensional binary vectors. Section 2 first describes the algorithm. Section 3 then presents experimental results on a database of 1000 songs and 12,217,111 fingerprints. Finally, summarizes our results and discusses possible avenues of future work.

*System Parameters*

Given the set of known fingerprints, we first construct a 256-ary tree. Each 8192-bit fingerprint is represented as 1024 8-bit bytes. The value of each consecutive byte in the fingerprint determines which of the 256 possible children to descend. A path from the root node to a leaf defines a fingerprint. As the depth of the tree increases, it is common to find nodes with only a single child. This is because the number of actual fingerprints is very much less than the total possible number. For efficiency purposes, we compress such sequences of nodes with only one child into a single node that represents multiple bytes. We consider the level, l, of a node to be the number of bytes represented by the path from the root to that node. Our search algorithm is guided by a table, T, indexed by a node level, l, and a number of errors, e. During a search, when we visit any node nl,i, at level l in the tree, we will have examined b = 8l bits of the vector and seen e errors between the first b-bits of the query and the first b-bits represented by the path to node nl,i. The probability, p, of observing x errors in b-bits with a bit error rate (BER) of r is a binomial distribution.

*Project Formulae*
Binomial Distribution

$$p(x|\ b,r)\ =\ \binom{b}{x} r^x (1-r)^{b-x}$$

Cumulative Probability

$$p(e_0 \geq E|b,r)\ =\ 1\ -\sum_{0}^{E}\quad p(x|b,r)dx$$

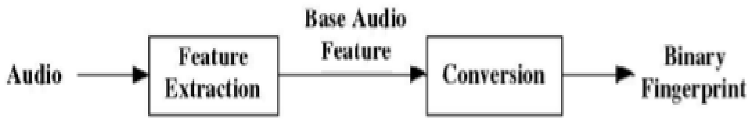
**C. Kim Sungwoong and Yoo Chang D.**

*Description*
In this paper, a binary audio fingerprint that is compact for a fast DB search while minimizing the performance degradation is proposed. The proposed audio fingerprint is based on the first-order normalized subband spectral moment. The first-order normalized moment is known to be not only reliable but also robust against common audio processing steps including lossy compression, random start, equalization, etc. A well-known boosting technique known as the AdaBoost [8] is sometimes used in the audio classification problem to improve the performance of the classifier through learning. The modified AdaBoost algorithm [8] is used in this paper to obtain a binary fingerprint from the first-order normalized subband spectral moment. The experimental results show that the proposed audio fingerprint outperforms other state-of-the-art binary fingerprint in the context of audio identification.

*System Parameters*

First, an input audio is converted to mono and downsampled to 1025 Hz. Next, the downsampled signal is split into overlapping frames windowed by Hamming window. The window size is 4096 samples (0.372 s), and the adjacent frames are overlapped by 2048 samples (0.186 s). Then short-time Fourier transform (STFT) is applied to each frame to obtain the spectrum. The spectrum of each frame is divided into 16 critical bands from 300 to 5300 Hz, and finally the T7i s are computed at each critical band. As a result of the extraction, an 16 dimensional real-valued base audio feature is obtained every 2048 samples. The AdaBoost [8] is a learning algorithm to produce a strong classifier from a number of weak classifiers. We modified the AdaBoost to select 16 binary conversion methods from the candidate set. Selecting a weak classifier is equivalent to selecting a binary conversion method. In other words, selecting a weak classifier entails choosing one difference formula out of 4,264 possible candidates and selecting an appropriate threshold T. The Pairwise Boosting is used to select both robust and discriminative 16 difference formulas and those thresholds to generate an 16-dimensional binary audio fingerprint through 16 rounds.

*System Design*

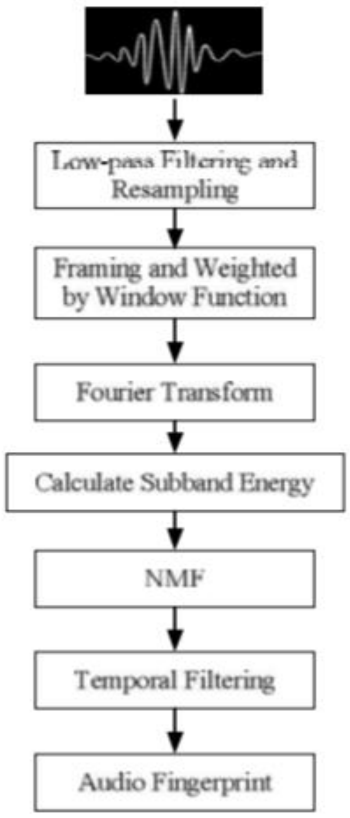


**D. Jijun Deng, Wanggen Wan, XiaoQing Yu andWei Yang**

*Description*
An audio fingerprinting algorithm based on spectral energy structure and NMF is proposed, Comparing other audio features, the spectral band energy structure is relatively stable and robust to many kinds of signal degradations. And NMF is a newly emerging technique for finding parts-based, linear representations of non-negative data. Experimental results have proved that audio fingerprints that are obtained by applying NMF on the spectral band energy structure are resistant to temporal and spectral distortions.

*System Parameters*
Given an audio sample, the audio fingerprinting algorithm passes from low-pass filtering and resampling to extract important data and drop remaining information. The audio is the passed through Fourier Tranform, calculating subband energy picking up non-negative matrix on the spectral band energy structure, applying temporal Filtering layer and generating the fingerprint

*System Design*



**E. Kamaladas M. D. and Dialin M. M.**

*Description*

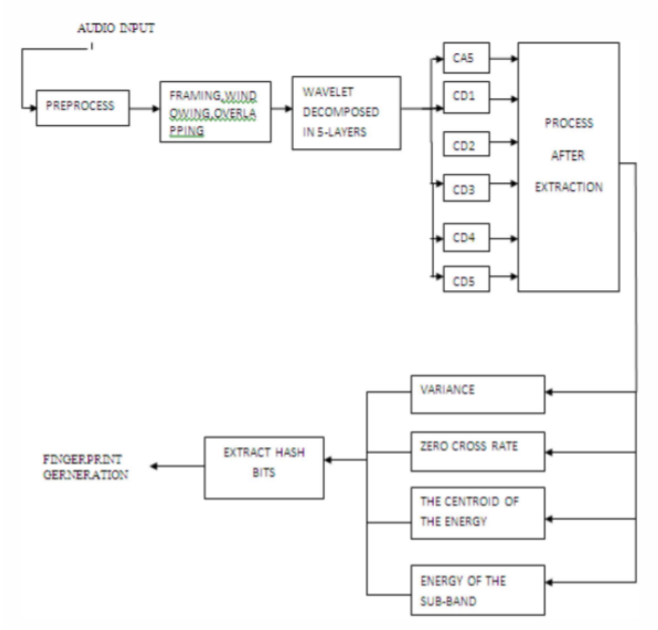
Thus a novel fingerprinting technique based on wavelet transform that reduces the amount of audio fmgerprints is presented. The existing methods created the large amount of fingerprints per audio file. Due to the increase in amount of fmgerprints the search time also increased. The proposed method could reduce the amount of fingerprints and thereby increase the amount of songs stored in the database is also increased. When dealing this algorithm with the A WGN, it outperforms the existing method for the low signal to noise ratio. In addition the proposed method can also identify the audio even the speed and pitch value of the audio content was changed.

*System Parameters*

Wavelet transform is a local transformation on a signal in Time and Frequency domains, which can effectively extract information from the signal, and do multi-scale detailed analysis on a function or

signal by functions such as scaling and translation, thereby can solve many difficult issues which cannot be solved by the Fourier transform. The proposed method uses the fingerprint extraction algorithm in time and frequency domains using wavelet transform. The audio signal was decomposed into 5-layer wavelet, and then calculated the energy distribution center, the energy of sub-band in wavelet domain, and the variance of wavelet coefficient.
By using this algorithm the number of fingerprints per audio file is reduced. This leads to reduced search time and less memory requirement. Also, using the technique audio identification can be done even when the quality of the audio waveform.

*System Design*



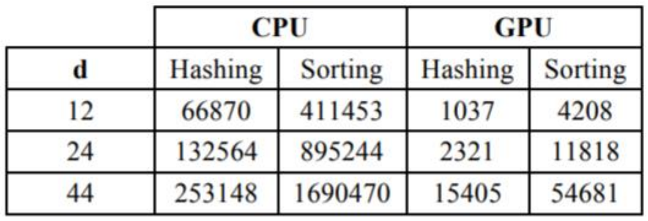
### F. Ouali Chahid, Dumouchel Pierre and Gupta Vishwa

*Description*
A GPU implementation of a similarity search algorithm used in our audio fingerprinting system. To generate audio fingerprints, we convert the audio signal into a 2-D binary image derived from a spectrogram matrix by using the mean intensity values as a threshold. Each fingerprint encodes the positions of selected salient regions from the binary image. The similarity between two fingerprints is defined as the intersection between their elements. To accelerate the search, we propose GPU implementation of two different intersection algorithms: hashing-based and sortingbased algorithms. We present run times of these two algorithms using varying number (d) of salient regions and different

GPU configurations. Specifically, we investigate the use of shared memory (SM) as a solution to reduce run time. The use of SM reduced significantly run times for both algorithms compared to their CPU implementations, yet showed mediocre performance for large d=44. The problem mainly comes from the higher dimensionality of fingerprints and the large number of query frames loaded to SM, reducing the number of concurrent threads. To improve performance we load only one query frame to SM instead of 256 and increase the number of threads from 256 to 512. The overall conclusion is to optimize SM usage while ensuring that maximum number of threads can be run concurrently. Compared to the CPU implementation, the best GPU implementation of the hashingbased algorithm accelerates the search by 150 times. Sortingbased algorithm is more parallelizable (although slower for this task) and accelerates the search by 379 times.

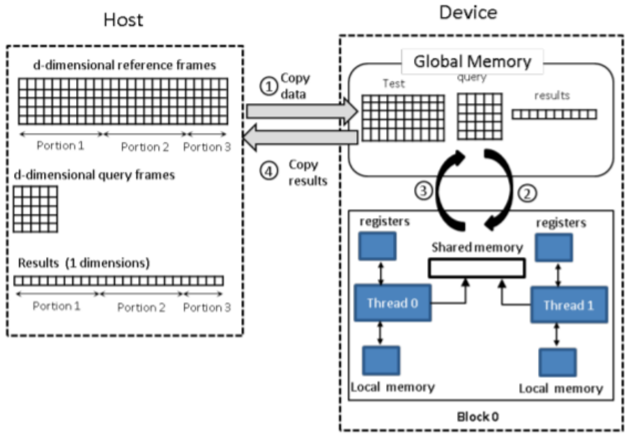
*System Parameters*

Each thread loads 1 d-dimensional reference frame into local memory (D-dimensional vector is used to hash the reference frame in case of hashing-based algorithm). Load query frames to shared memory: each thread in the block loads one d-dimensional query frame to shared memory. Compute the similarity between the reference and all query frames in shared memory using hashing-based or sorting- based algorithms. Save the number of the closest query frame. Repeat until no more query frames.
Processing run times (in seconds) of hashing-based and sorting-based algorithms on CPU and GPU using different dimensions of d. The GPU results are obtained by applying processing steps with D = 745, d = {12, 24, and 44}, nThreads (number of threads per block) = 256, k (number of query frames to be loaded to shared memory) = nThreads and nBlocks (numbers of thread blocks) = number of reference frames/nThreads.



| | CPU | | GPU | |
|---|---|---|---|---|
| **d** | Hashing | Sorting | Hashing | Sorting |
| 12 | 66870 | 411453 | 1037 | 4208 |
| 24 | 132564 | 895244 | 2321 | 11818 |
| 44 | 253148 | 1690470 | 15405 | 54681 |

*System Design*



## CONCLUSION

We perform a thorough survey and evaluation of popular audio fingerprinting algorithms for hash generation based on size, speed, range and performance of the algorithm.

## REFERENCES

[1] Shazam Music Recognition Service. http://www.shazam.com/.

[2] Cano, P.; Batle, E.; Kalker, T.; Haitsma, J. (2002). [IEEE 2002 IEEE 5th Workshop on Multimedia Signal Processing - St.Thomas, VI, USA (9-11 Dec. 2002)] 2002 IEEE Workshop on Multimedia Signal Processing. - A review of algorithms for audio fingerprinting. , (), 169–173.

[3] Matthew L. Miller; Manuel Acevedo Rodriguez; Ingemar J. Cox (2005). Audio Fingerprinting: Nearest Neighbor Search in High Dimensional Binary Spaces. , 41(3), 285–291.

[4] Kim, Sungwoong; Yoo, Chang D. (2007). [IEEE 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07 - Honolulu, HI, USA (2007.04.15-2007.04.20)] 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07 - Boosted Binary Audio Fingerprint Based on Spectral Subband Moments. , (), I-241–I-244.

[5] Jijun Deng, ; Wanggen Wan, ; XiaoQing Yu, ; Wei Yang, (2011). [IEEE 2011 IEEE 13th International Conference on Communication Technology (ICCT) - Jinan, China (2011.09.25-2011.09.28)] 2011 IEEE 13th International Conference on Communication Technology - Audio fingerprinting based on spectral energy structure and NMF. , (), 1103–1106.

[6] Kamaladas, M. D.; Dialin, M. M. (2013). [IEEE 2013 International Conference on Signal Processing, Image Processing, and Pattern Recognition (ICSIPR) - Coimbatore (2013.2.7-

2013.2.8)] 2013 International Conference on Signal Processing , Image Processing & Pattern Recognition - Fingerprint extraction of audio signal using wavelet transform. , (), 308–312.

[7] Ouali, Chahid; Dumouchel, Pierre; Gupta, Vishwa (2015). [IEEE 2015 13th International Workshop on Content-Based Multimedia Indexing (CBMI) - Prague, Czech Republic (2015.6.10-2015.6.12)] 2015 13th International Workshop on Content-Based Multimedia Indexing (CBMI) - GPU implementation of an audio fingerprints similarity search algorithm. , (), 1–6.

[8] Wu, P., & Zhao, H. (2011). Some Analysis and Research of the AdaBoost Algorithm. Intelligent Computing and Information Science, 1–5.