International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Impact Factor (2012): 3.358

An Efficient Hash Count Indexing and Searching Scheme for Audio Fingerprinting

Simarjeet Singh Bhatia[1], Rupali Bhartiya[2]

[1]Research Scholar, CSE Department, Shri Vaishnav Institute of technology and Science, Indore (M.P), India

[2]Assistant Professor, CSE Department, Shri Vaishnav Institute of technology and Science, Indore (M.P), India

Abstract: In today's dynamically changing world music databases are the one which are updating at a great speed and so is the identification process. In order to recognize a song rapidly and accurately we use an audio fingerprint. An audio fingerprint summarizes an audio recording or a song. A song can be recognized or identified by matching its fingerprint to the fingerprints present in a database .It has a wide variety of applications in broadcast monitoring, connected audio(FM/AM), music library and many more. This process of audio fingerprinting is divided into two phases namely fingerprint extraction and fingerprint matching. In this paper focus on fingerprint matching phase has been kept. We propose a new method of indexing and search algorithm. This indexing method will provide a base for our search algorithm to make it hybrid and more efficient. Theoretically and practically our algorithm provides better results as compared to Haitsma and Kalker method.

Keywords: sub-fingerprint; audio fingerprint; hashing; bunching; indexing

1. Introduction

In today's world, digital music libraries are becoming very popular and are growing with a considerable rate. With their demand in consumer based systems it has become quite important to analyze. The proficiency to find most similar or exact song from a large audio database is particularly a very important task for a number of applications. Broadcast monitoring is one of the applications for audio fingerprinting. It refers to the automatic generation of playlist of radio, TV, or web broadcasts. Another application can be described as searching or retrieving the name of the artist and title of the song given a short clip of that audio, filtering technology for file sharing is yet another application, automatic music generation is also one of the most important applications. Since an audio contains many features, extracting a representative audio fingerprint is the most important task that describes auditory content of each song. There are many techniques given by many authors to extract audio fingerprint namely Haitsma and Kalker, M.L Miller, Avery Wang, collectively by Yu Liu, Kiho Cho, Hwan Sik Yun, Jong Won Shin and Nam Soo Kim, Guang Ho-Cha. Haitsma and Kalker introduced the concept of look up table to store and searched the audio fingerprint whereas M.L Miller proposed the concept of 256-ary tree, Avery Wang presented landmark hashing technique, Yu Liu emphasized on discrete cosine transform to improve accuracy, and Guang Ho Cha proposed the concept of inverted file in order to improve the recognition rate. In our method we presume a representation of song by Philips Robust Hash [1] method in which a 3 second interval is represented by 256 sub-fingerprints of 32 bits each and provided this fingerprint our main focus would be on indexing as well as searching a song from the database. This problem can be characterized by the nearest neighbor search problem of a very high dimensional (i.e. 256*32=8192) space.

2. Literature Review

Haitsma and Kalker introduced an indexing scheme which works on the principle of look up table. A lookup table consisted of all the combinations of fingerprints. The method lets the entries in the lookup table point to the song or songs where the respective sub-fingerprint value occurs. A sub-fingerprint value can occur at multiple points in a song or multiple songs, the song pointers are stored in a linked list. Therefore a single sub-fingerprint can point to multiple pointers at the same time and thus all the 256 blocks of fingerprint block are compared with the entries in database.

The first problem in the Haitsma and Kalker method is that there are 2^{32} entries in lookup table which is practically too large. This problem is solved to some extent in the next work.

The second problem in this method is that Haitsma and Kalker make assumption that if a signal undergoes a "mild" degradation then also at least one sub fingerprint is error free which is quite unavoidable to consider [2].

M.L Miller, M.C. Rodriguez, and I.J. Cox proposed 256-ary tree technique for fingerprint search. Each 8192 (256*32) bit fingerprint is represented as 1024 8-bit bytes. This technique adopts top down approach. The value of each successive byte determines the next step of where to find the fingerprint of 256 child to follow. The path of the root node to a leaf defines a fingerprint. The searching is done by comparing the first byte of query with the children of the root node by calculating the cumulative error rate with each child node. If the error rate seen so far is greater than threshold then the child is searched. This process is done continuously until the leaf node is reached. After this process the error rate is compared to the best error rate seen so far, if the error rate is less than it then it is considered as best candidate in the nearest neighbor. This technique's main loophole is same as that of Haitsma and Kalker method i.e. the size of the 256-ary tree is too large and the depth of the tree is also too big

for disk based database search. The second problem in Miller's method is that it uses probabilistic method based on binomial distribution to estimate the bit error rate in each tree node which leads the correct error up to 85% since it is difficult to find bit error rate in advance [3,4]. This drawback is removed to some extent in the next work.

Avery Wang introduced the scheme of landmark hashing. The basic operation of this scheme is that each audio track is analyzed to find prominent onsets concentrated in frequency, since these onsets are most likely to be preserved in noise and distortion. These onsets are formed into pairs, parameterized by the frequencies of the peaks and the time in between them. These values are quantized to give a relatively large number of distinct landmark hashes (about 1 million in number). Parameters are tuned to give around 20-50 landmarks per second. Each reference track is described by the (many hundreds) landmarks it contains, and the times at which they occur is stored in an inverted index. Similarly to identify a query it is converted to landmarks. This method again suffers from same drawback i.e. it can be seen that it contains about 1 million landmark hashes but again the problem is that to search a audio file it will go to the same position where the audio file is kept and therefore each time when a song is requested it will go through the same process [5]. It can be seen that the problem of accuracy is solved to a good extent but still one million landmarks is very high and thus extends the search domain still to a large area.

Yu Liu, Kiho Cho, Hwan Sik Yun, Jong Won Shin and Nam Soo Kim proposed a concept of DCT i.e. discrete cosine transform based multiple hashing technique which was quite robust in nature and had a good accuracy as well. It worked by including discrete cosine transform phase into its fingerprint extraction technique and making k hash tables in order to store them. As for their algorithm they used 4 hash tables in order to retrieve the song, thereby giving a good accuracy. Although the accuracy is quite good but still there is a major setback in this algorithm i.e. the number of hash tables which are 4. These hash tables at first look to seem small in number but when the number of songs increase then there length increases and so on thereby increasing the complexity of the algorithm therefore it then becomes quite complex to find a song which is present in 4^{th} hash table at a far position. Also this method concentrates more on accuracy rather than speed of search. However, the constraint of accuracy is removed in next work to a large extent [6].

Guang-Ho Cha proposed another concept in which he generated additional fingerprints with up to n-toggled bits. By generating additional fingerprints with errors he removed the limitations of Haitsma and Kalker method of "mild" degradation in signal also additionally he proposed a new scheme of indexing. He presented a new indexing scheme in which he stored a fingerprint in an Inverted File Structure. The inverted file structure consisted of sub-fingerprint, the number of times a particular sub-fingerprint present in a song, and the position of the sub-fingerprint in the song.

This method removes the curse of dimensionality to a very large extent since it deals with the large database containing sub-fingerprints with up to n toggled bits but still doesn't emphasizes on the searching technique because the number of sub fingerprints are now very large in number. So, instead of one sub fingerprint now 32 additional sub fingerprints are searched which overloads the searching domain and thus comes out as main limitation of this method [7].

Haitsma and Kalker method doesn't provide an efficient indexing method also the searching is quite ineffective because of large database, therefore in order to improve efficiency and increase searching speed we need to revise the Haitsma and Kalker algorithm. In order to overcome the above two problems we reuse the concept of Bunching with the help of Hash Count.

3. Proposed Scheme

The above mentioned problems of stated algorithms can be solved by introducing Hash Count [8] and Bunching into the proposed algorithm. Our main emphasis is on how to increase the speed of searching as well as decrease the search time by a considerable amount.

This can be explained as follows:
Suppose a song named 'x.mp3' has to be searched in the database, according to above mentioned algorithms a search is applied to it by first extracting its fingerprint, then those songs are retrieved which are likely to match with the query finger print. After this phase the fingerprint of query audio is matched with the fingerprint of retrieved audio from the database. Now in the best case scenario there can be only one song in the database which matches the first time and so the query finger print matches the very first time or there could be 'n' number of songs which would have the same fingerprint that of the query audio. After the exact song is matched it is fetched from the database as the result. The main problem as we stated is that the retrieved song is coming from that place where it is present in the database and next time when it is searched again it will come from that place only which will take the same time. In order to remove this flaw in Haitsma and Kalker method, in the proposed scheme we will give a Hash Count to the searched song. Each time when the song is searched and retrieved its Hash Count will be increased by a number and the song with the maximum number of Hash Count will be kept at initial or top position in the database, thus when the fine search is carried out for that song it can be retrieved as early as possible because it will be found at the top position and not on the same position in which it was kept at the first time. Thus with this process the overload can be reduced to some extent for random access [9] of disk.

There can be another scenario with the database containing songs. What happens when the database gets updated with a new song? [10] Since there is a new song then its hash count will be zero, also it could be possible that query fingerprint would exactly or near to exactly match this song. In this situation the algorithm will again have to scan till the end of the database for that song which is now updated. To solve this problem we introduce a further scheme of "Bunching". According to this scheme we will make a bunch of those songs which match with each other near to exact by making a bunch or group of them. Due to this process the song database will contain groups of songs which are similar with respect to the fingerprints, also a song can be grouped

according to their genre, pitch [11], level of noise content, zero crossing rate [12] etc..

On a broad view an audio fingerprinting algorithm is divided into two phases:
1.) The fingerprint extraction phase
2.) The search algorithm

The audio fingerprint is extracted with the help of Philips Robust Hash algorithm. The stages of fingerprint extraction phase are described as follows:

a) In the first stage of this phase the query audio is framed.
b) In the second stage Fourier transform is taken so as to change the signal from time domain to frequency domain.
c) Out of the result of second stage 33 energy bands are selected between the range of 300hz -2000hz because this is the range for human auditory system.
d) Up to this stage feature extraction takes place [13, 14, 15].
e) In the fourth stage filtering is done.
f) At the last stage we get a 32 bit sub fingerprint for every 11.6 millisecond therefore 256 sub fingerprints for every 3 seconds after performing threshold [16, 17] process. Figure 1 below shows various stages of sub-fingerprint extraction

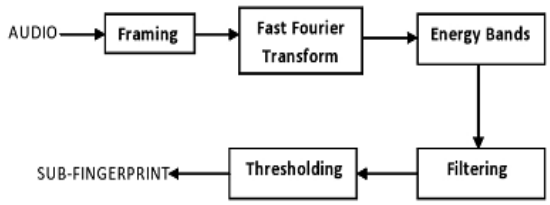

Fig. 1.Sub-Fingerprint Extraction Stages

The proposed search algorithm contains two phases as described below:

a.) In the first phase all those songs are searched and retrieved which could possibly contain the candidate's fingerprint.
b.) In the second phase each matched song's full fingerprint [18, 19] is fetched from the database and is compared with the query fingerprint, if the song matches with the query [20] fingerprint then it is retrieved and the song's hash count is now increased by one unit i.e. if the songs hash count [21] is 0 it is now 1 after the search and each time it is increased by 1 unit when it is retrieved after searching. After updating the hash count of the searched song, its position is updated in the database according to its hash count, the more the hash count the higher will be its position i.e. the top most position, so whenever this song's query comes next time then it will be retrieved from the updated position and not from the old position, which will decrease the time of retrieval thereby increasing the speed of searching. Figure 2 shows the flow chart structure of the proposed algorithm.

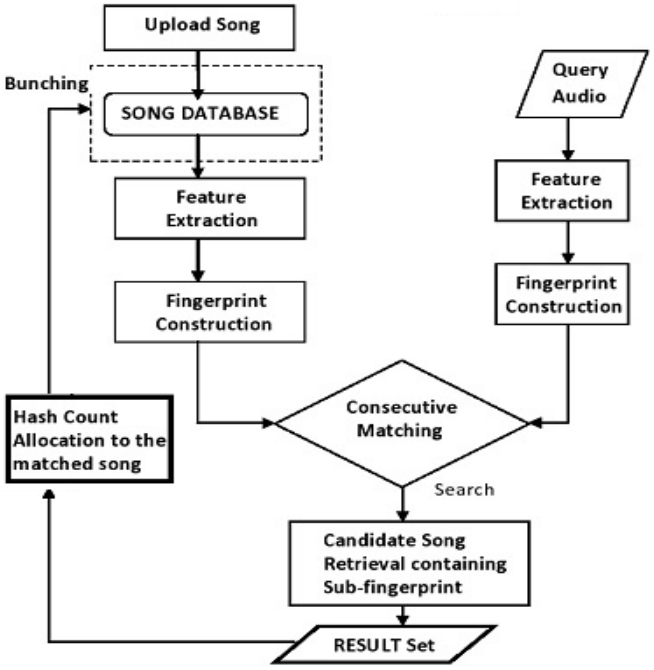

Figure 2: Proposed Hybrid Algorithm Flowchart

Figure 3 shows the song retrieval architecture for the above algorithm. This is the second phase of the proposed algorithm. It can be seen from the figure that a query fingerprint block contains 256 blocks of sub-fingerprints each of 32 bits. A fingerprint might be present at many positions in many songs. This fingerprint is projected [22,23] in database which then points to the pointers which further point to the position of a sub-fingerprint in a song.

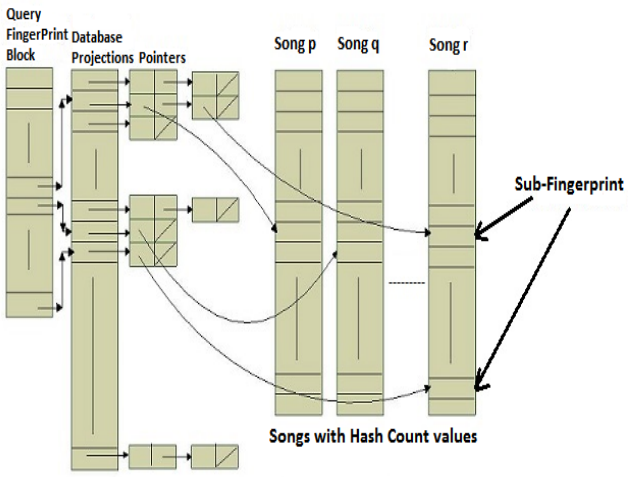

Figure 3: Proposed Retrieval Architecture

Pseudocode

Hybrid Algorithm for fingerprint search
Input Parameter: song clip 'c'.
Output Result: song 's' or 'no match'.
Procedure-

1.) Compute sub-fingerprint for query song clip 'c'.
2.) for i=1 to count(total songs) do {
if(sub-fingerprint('c')=sub-fingerprint('s')
then{
a=i; **//generalized search**
}
}
3.) for j=1 to count(a) do{
if(fingerprint(a)=fingerprint(s))

```
then{ //specialized search
write('song found');
increment hash_count(s) by 1;
}
else
write('no match');
}
```

4. Implementation Notes

The following figure 4 shows the GUI of our work in running phase, with each and every component in it. We have implemented our system in Matlab R2012b (8.0.0.783) 64 bit (win64) for GUI and fingerprint generation and for database purpose we have used MySQL GUI v5.1.7 (C)2002-2005 Webyog Softworks Pvt.Ltd.

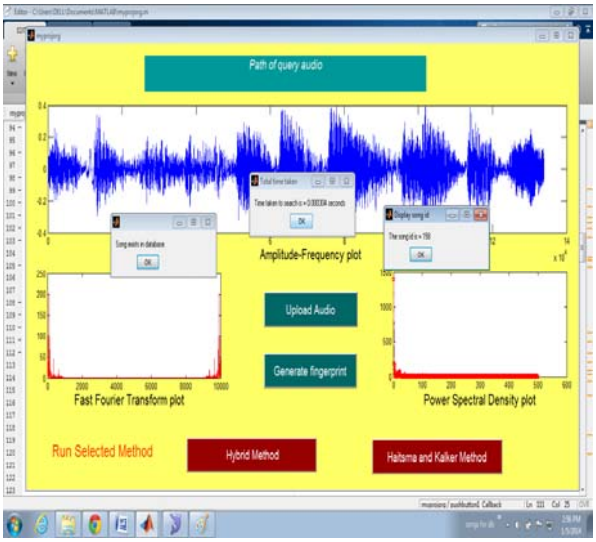

Figure 4: Interface for fingerprint generation and searching in Matlab R2012b

Our GUI consists of four buttons namely: Upload Audio, Generate Fingerprint, Hybrid Method and Haitsma and Kalker Method. The upload audio button saves the name of the song along with its initial counter value (0) in the database. The Generate fingerprint button generates the fingerprint (and sub-fingerprint) of the song and saves into the database. The Hybrid Method button searches the song with the our proposed algorithm and Haitsma and Kalker method button searches the song with their method. Each time when a song is searched its counter increases by 1 and the database is updated regularly each and every time. The Amplitude-Frequency plot graphs the amplitude of the song at various points whereas the Fast Fourier Transform plot and Power Spectral Density plot graphs the output of the respective functions used to generate sub fingerprint in matlab respectively.

5. Comparison of Existing Algorithm With Proposed Algorithm

As compared to the Haitsma and Kalker algorithm our algorithm elevates towards new scheme i.e. including Hash Count and Bunching. Surely in ideal conditions i.e. when no song is searched it will perform same as Haitsma and Kalker algorithm because to search a song whose hash count is 0 it has to go to its initial position, similarly if the song is searched for the first time then also the algorithm has to go to its initial position. This algorithm is much more advantageous for those songs which are searched more than any other song, this can be said so because a song which is searched more number of times will be on higher position in database and would be easily retrieved [24] as compared to other songs and also it will be retrieved early. So, there is an advantage of this algorithm that if a song has been searched [25, 26] more number of times it would be retrieved more rapidly.

6. Experimental Results

We evaluate our hash count scheme in search algorithm by generating about 5,00,000 fingerprints from over 700 songs and 1500 queries are generated by selecting a 3 second clip in any random order. We have taken into account a generalized dataset from marsyas.info which consists of 700 audio tracks each 30 seconds long. It contains 7 genres, each represented by 100 tracks. The tracks are all 22050Hz Mono 16-bit audio files in .wav format. Experimental results show that searching a song with respect to its hash count reduces much time as compared to Haitsma and Kalker method. Table 1 shows searching time in seconds taken by both the methods. There is a significant amount of improvement in time by the proposed algorithm.

Table 1: Comparison of both the Methods

Time Taken in Seconds			
Number of Elements	*Haitsma and Kalker Method*	*Hybrid(Proposed) Method*	*Difference of the time taken by both algorithm*
100	0.0135	0.0127	0.0008
150	0.0150	0.0142	0.0008
200	0.0153	0.0145	0.0008
250	0.0159	0.0133	0.0026
300	0.0160	0.0146	0.0014
350	0.0161	0.0157	0.0004
400	0.0164	0.0157	0.0007
450	0.0165	0.0162	0.0003
500	0.0169	0.0166	0.0003
550	0.0172	0.0168	0.0004
600	0.0174	0.0171	0.0003
650	0.0178	0.0175	0.0003
700	0.0181	0.0179	0.0002

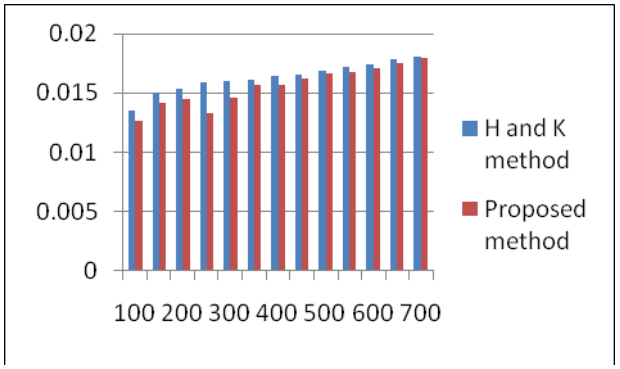

Figure 5:.shows the results in the form of graphical representation
x-axis- No. of elements/songs in database
y-axis- Time taken in seconds

Table 2: Efficiency Calculation

Number of Elements	*Difference of the time taken by both algorithm*	*Efficiency (in percentage)*
100	0.0008	5.9
150	0.0008	5.3
200	0.0008	5.2
250	0.0026	16.3
300	0.0014	8.7
350	0.0004	2.4
400	0.0007	4.2
450	0.0003	1.8
500	0.0003	1.7
550	0.0004	2.3
600	0.0003	1.7
650	0.0003	1.6
700	0.0002	1.1
Average overall Efficiency Gain*		4.4

***Average overall Efficiency Gain** can be defined as the average efficiency i.e. sum of all the efficiencies divided by the total number of iterations which in our case is 4.4 % which means our method is 4.4 % more efficient than the old technique (Haitsma and Kalker technique).

7. Future Scope

In this paper an alternative searching technique has been presented. When compared with the old technique a significant amount of decrease in time was achieved. On one hand the use of hash count decreased the amount of time to search and on the other hand it marginally increased the disk i/o for this extra operation. This algorithm has a great scope and can be used in military as well as telecommunication departments giving its good utilization and use by making it for speech datasets and for various other sound formats also in order to widen its scope of use.

8. Conclusion

In this paper we propose a new indexing scheme for large audio fingerprint databases. This indexing scheme theoretically and practically provides good results for large databases and achieves an overall gain of 4.4 % on Haitsma and Kalker technique This scheme is somehow more complex than Haitsma and Kalker algorithm as it introduces an additional concept of hashing and bunching but it sheds out advantages with respect to speed and accuracy as well which makes it useful method for audio fingerprinting.

References

[1] J. Haitsma and T. Kalker, “A Highly Robust Audio Fingerprinting System,” Proc. Int. Symp. on Music Information Retrieval, pp. 107–115,2002.
[2] J. Haitsma and T. Kalker, “A Highly Robust Audio Fingerprinting System With an Efficient Search Strategy,” J. New Music Research, vol. 32, no. 2, pp. 211–221, 2003.
[3] M.L. Miller, M.C. Rodriguez, and I.J. Cox, “Audio Fingerprinting: Nearest Neighbor Search in High Dimensional Binary Spaces,” Proc. IEEE Multimedia Signal Processing Workshop, pp. 182–185, 2002.
[4] M.L. Miller, M.C. Rodriguez, and I.J. Cox, “Audio Fingerprinting: Nearest Neighbor Search in High Dimensional Binary Spaces,” J. VLSI Signal Processing, vol. 41, pp. 285–291, 2005.
[5] Avery Wang "An Industrial-Strength Audio Search Algorithm", Proc. 2003 ISMIR International Symposium on Music Information Retrieval, Baltimore, MD, Oct. 2003.
[6] Yu Liu, Kiho Cho, Hwan Sik Yun, Jong Won Shin, and Nam Soo Kim “DCT Based Multiple Hashing Technique for Robust Audio Fingerprinting” –IEEE trans. ICASSP, pp.61-64, 2009.
[7] Guang-Ho Cha, “An Effective and Efficient Indexing Scheme for Audio Fingerprinting”, IEEE trans. Fifth FTRA International Conference on Multimedia and Ubiquitous Engineering, pp.48-52.
[8] “Structural fingerprint based hierarchical filtering in song identification” Qiang Wang, Gang Liu, Zhiyuan Guo, Jun Guo, Xiaoyu Chen-IEEE-2011.
[9] F. Balado, N.J. Hurley, E.P. McCarthy, and G.C.M. Silvestre, “Performance analysis of robust audio hashing,”IEEE trans. Information forensics and security, vol. 2,no. 2, pp. 254–266, Jun. 2007.
[10] Chih-Chin Liu, Po-Jun Tsai, “Content-Based Retrieval of MP3 Music Objects”- CIKM'01, ACM, pp.-506-511, November 5-10, 2001.
[11] Anonymous ICME submission,” Audio-To-Tag Mapping: A Novel Approach for Music Similarity Computation”, IEEE trans.2011, pp.1-4.
[12] Arijit Ghosal, Bibhas Chandra Dhara, Sanjoy Kumar Saha, “Speech/Music Classification Using Empirical Mode Decomposition”- IEEE, Second International Conference on Emerging Applications of Information Technology, pp.49-52, 2011.
[13] C. Panagiotakis and G. Tziritas,“A speech/music discriminator based on rms and zero-crossings,” IEEE Transactions on Multimedia, vol. 7, pp. 155–166, 2005.
[14] Daniel P. W. Ellis, PLP and RASTA (and MFCC, and inversion) in Matlab, 2005. http://www.ee.columbia.edu/~dpwe/resources/matlab/rastamat/
[15] Notes on “Mel Frequency Cepstral Coefficients for music modeling” by Beth Logan.
[16] Notes on “Audio Signal Classification” by Hariharan Subramanian, M.Tech. Credit Seminar Report, Electronic Systems Group, EE. Dept, IIT Bombay, November2004.
[17] A. Wang, “The Shazam music recognition service,” Com. ACM, vol. 49, no. 8, pp. 44–48, 2006.
[18] J. J. Burred and A. Lerch, “Hierarchical automatic audio signal classification,” Journal of the Audio Engineering Society, vol. 52, pp. 724–739, 2004.
[19] L. Lu, H. J. Zhang, and H. Jiang, “Content analysis for audio classification and segmentation,” IEEE Transactions on Speech and Audio Processing, vol. 10, 2002.
[20] E. Wold, T. Blum, D. Keislar, and J. Wheaton, “Content-based classification, search and retrieval of audio,” IEEE Multimedia Mag., vol. 3, pp. 27–36, July 1996.
[21] Gal Chechik, Eugene Ie, Martin Rehn, Samy Bengio, Dick Lyon, “Large-Scale Content-Based Audio

Retrieval from Text Queries", MIR'08, October 30–31, ACM, pp.105-112, 2008.

[22] F. Balado, N.J. Hurley, E.P. McCarthy, and G.C.M. Silvestre, "Performance analysis of robust audio hashing," IEEE trans. Information forensics and security, vol. 2, no. 2, pp. 254–266, Jun. 2007.

[23] J.E. Mu˜ noz-Expósito, S. Garcia-Galán, N. Ruiz-Reyes, P. Vera-Candeas, F. Rivas-Pe˜ na, " Speech/Music Discrimination Using A Warped Lpc-Based Feature and A Fuzzy Expert System For Intelligent Audio Coding ", 14th European Signal Processing Conference (EUSIPCO 2006), Florence, Italy, September 4-8, 2006.

[24] S. Baluja and M. Covell, "Content fingerprinting using wavelets," in Proc. Conf. Visual Media Production, Nov. 2006, pp. 198–207.

[25] A. Gionis, P. Indyk, and R. Motwani, "Similarity Search in High Dimensions Via Hashing," Proc. VLDB Conf., pp. 518–529, 1999.

[26] Guodong Guo and Stan Z. Li, " Content-Based Audio Classification and Retrieval by Support Vector Machines" IEEE Transactions On Neural Networks, Vol. 14, No. 1,pp.-209-215,January 2003.