A FAST AUDIO SEARCH METHOD BASED ON SKIPPING IRRELEVANT SIGNALS BY SIMILARITY UPPER-BOUND CALCULATION

Hidehisa Nagano, Ryo Mukai, Takayuki Kurozumi, Kunio Kashino

NTT Communication Science Laboratories
Nippon Telegraph and Telephone Corporation
3-1, Morinosato-Wakamiya, Atsugi-shi, 243-0198, Japan

ABSTRACT

In this paper, we describe an approach to accelerate fingerprint techniques by skipping the search for irrelevant sections of the signal and demonstrate its application to the divide and locate (DAL) audio fingerprint method. The search result for the applied method, DAL3, is the same as that of DAL mathematically. Experimental results show that DAL3 can reduce the computational cost of DAL to approximately 25% for the task of music signal retrieval.

Index Terms— Audio fingerprint, audio search, information retrieval

1. INTRODUCTION

Signal identification or retrieval for audio/video using fingerprint technology has been playing an important role in the world. In particular, audio fingerprint technologies are being used in many services such as broadcast monitoring for commercial messages and music, music information retrieval via mobile phones, content monitoring on the Internet, and second-screen services [1, 2]. In addition, due to the explosion of multimedia data, the expectations for fingerprint technology are increasing.

Audio fingerprint technology is the generic term for the technique of retrieving an audio signal identical to that given as the query (the query signal) from a database (the stored signal) by comparing fingerprints, which are representations of the features extracted from these signals. According to the objectives of the applications, discriminability of the signals, quick retrieval from huge databases, or robustness against the degradation of the signals due to the noise or distortion is required for audio fingerprints. Therefore, these have been major research issues in the field of fingerprint research.

In the literature, a signal search method called time-series active search (TAS) was proposed [3, 4]. TAS uses histograms of features, currently called Bag of Features (BoF), as the fingerprint data. TAS has been used for broadcast monitoring for commercial messages. Another well-known approach is to employ a binary representation of a sound spectrogram as the audio fingerprint [5, 6, 7, 8, 9]. Audio fingerprints based on pairs of peaks in a spectrogram were also proposed [10]. These approaches are also widely used in industry. In addition, to cope with severe additive noise such as in the case of background music detection in the TV programs, the divide-and-locate (DAL) method was proposed [11]. The basic idea of DAL is to divide a spectrogram into a number of small regions and undertake matching for each region to locate it in the database. The small regions are quantized by vector quantization (VQ), and the matching operations are executed by comparing VQ codes. In addition to the above mentioned methods, audio search techniques using computer vision approaches have also been proposed [12, 13].

The basic procedure in audio fingerprint techniques is to search stored signal for segments similar to the query signal. This is simply accomplished by comparing the fingerprint data of the query signal with that of each segment of the stored signal. However, if we can know that a section in the stored signal has no segment similar to the query signal beforehand, we can skip the search in this section as shown in Fig. 1. For example, BoF is often used as the fingerprint data. The BoF, which is given by summing up the BoFs for segments in a section, is informative for predicting whether this section has similar segments or not.

It may possible to accelerate several fingerprint techniques by using this approach. In this paper, we demonstrate the application of this approach to DAL. We call the applied search method DAL3. The main idea of DAL3 is to divide the stored signal into sections and skip the searches for those that, mathematically, have no segment similar to the query signal .

This paper is organized as follows. Section 2 describes the fingerprint data used for DAL and DAL3 and explains these methods. In Sect. 3, the experimental results show the search accuracy of DAL scheme, and the computational efficiency of DAL3 is shown. Section 4 concludes the paper.

2. METHODS

In this section, we describe the fingerprint data used for DAL and DAL3 first. After that, we describe the DAL and DAL3 methods.

2.1. Fingerprint data

In the DAL scheme, fingerprint data are extracted in the same way for both a query signal and a stored signal.

First, time-frequency power spectra are extracted for the signals. Then each logarithmic power $p(f,t)$ at frequency f and time t is normalized to $\hat{p}(f,t)$ as follows:

$$p'(f,t) = \frac{p(f,t) - \bar{p}(f,t)}{\sigma(f,t)} \tag{1}$$

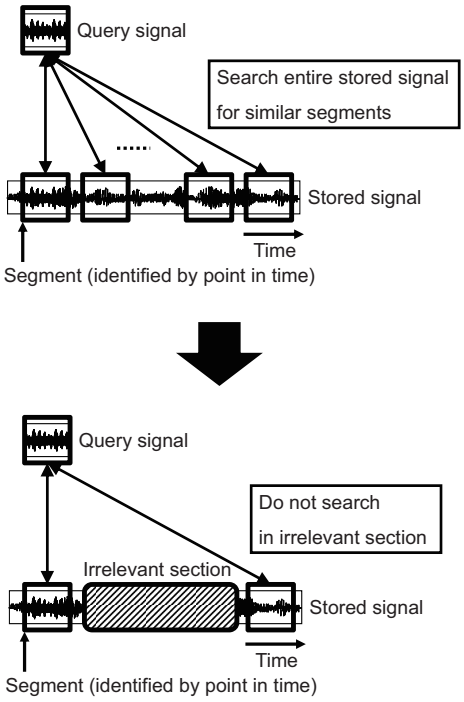

Fig. 1. Overview of signal search and our approach for its acceleration.

$$\hat{p}(f,t) = \begin{cases} \log_{10} p'(f,t) & \text{if } p'(f,t) > r \\ 0 & \text{otherwise} \end{cases}, \qquad (2)$$

where $\bar{p}(f,t)$ and $\sigma(f,t)$ are the average and the standard deviation of the logarithmic powers around (f,t), respectively, and r is a threshold value.

The normalized spectrum is then decomposed into a number of small time-frequency components of uniform size. Here, $F(i,t)$ denotes the decomposed component at frequency band i $(1 \leq i \leq b)$ and time t. Next, the spectrum corresponding to each component is classified by VQ. A VQ codebook is prepared for each frequency band. Then, let

$$V = \begin{pmatrix} v(1,1) & v(1,2) & \cdots \\ v(2,1) & v(2,2) & \cdots \\ \vdots & \vdots & \ddots \\ v(b,1) & v(b,2) & \cdots \end{pmatrix} \qquad (3)$$

be the obtained VQ code matrix, where $v(i,t)$ is the VQ code for $F(i,t)$. The VQ codes are positive integers. This matrix of VQ codes is used as the fingerprint data.

The fingerprint data V can be thinned out to reduce memory usage and computational cost. In our experiments described in Sect. 3, V is thinned out to become a matrix of a single row by interleaving the elements as follows:

$$\begin{aligned} V \quad = \quad & (v(1,1), v(2,2), \ldots, \\ & v(b,b), v(1,b+1), v(2,b+2), \ldots). \end{aligned} \qquad (4)$$

This thinning out is dispensable. In the following descriptions of DAL and DAL3 in this section, we do not assume this thinning out. However, considering that $b = 1$ for the fingerprint data, using such thinned fingerprint data in DAL and DAL3 is straightforward.

2.2. DAL

Let a b-by-m matrix X be the fingerprint data of the query signal; let a b-by-n matrix Y be the fingerprint data of the stored signal; and let

$$Y(t:t+m-1) = \begin{pmatrix} Y_{1,t} & Y_{1,t+1} & \cdots & Y_{1,t+m-1} \\ Y_{2,t} & Y_{2,t+1} & \cdots & Y_{2,t+m-1} \\ \vdots & \vdots & & \vdots \\ Y_{b,t} & Y_{b,t+1} & \cdots & Y_{b,t+m-1} \end{pmatrix}. \qquad (5)$$

We also assume $m \leq n$. Let M be a b-by-m matrix such that some assigned elements are 1 and others are 0. M is used for masking the fingerprint data.

DAL searches Y for segments $Y(t:t+m-1)$ similar to X such that

$$S(X, Y(t:t+m-1)) > S_{th}, \qquad (6)$$

where S_{th} is a search threshold and $S(X, Y(t:t+m-1))$ is the similarity between X and $Y(t:t+m-1)$ calculated as follows:

$$\begin{aligned} & S(X, Y(t:t+m-1)) \\ & \quad = \tfrac{1}{w(M)} u(X \circ M, Y(t:t+m-1) \circ M)), \end{aligned} \qquad (7)$$

where $w(M)$ is the number of non-zero elements of M, $\circ$ is the element-wise product operation, and $u(A,B)$ is a function that returns the number of elements such that $A_{i,j} = B_{i,j} \neq 0$. We say that $Y(t:t+m-1)$ is *similar* to X if inequality (6) holds.

The above procedure is usually performed using the inverted index constructed for the VQ codes in Y for the quick search [14]. M is introduced for the computational cost reduction.

2.3. DAL3

Let VQ codes be positive integers $1, 2, \ldots, q$; let $h(X)$ be the histogram of the VQ codes in X; let $h_i(X)$ be the number of VQ codes i in X; let

$$L^h(h(X), h(Y)) = \sum_{i=1}^{q} \min(h_i(X), h_i(Y)). \qquad (8)$$

In addition, we define *the overlapped histogram* $h^*(Y(t_s : t_s + l - 1))$ for the section $Y(t_s : t_s + l - 1)$ in Y, where $l \geq m$. $h^*(Y(t_s : t_s + l - 1))$ is a histogram of VQ codes, but the i-th bin $h_i^*(Y(t_s : t_s + l - 1))$, that is the number of VQ codes i, is calculated as

$$h_i^*(Y(t_s : t_s+l-1)) = \max_{t_s \leq t \leq t_s+l-m} (h_i(Y(t : t+m-1) \circ M)). \qquad (9)$$

Now, let

$$S^*(X, Y(t_s : t_s + l - 1)) = \frac{1}{w(M)} L^h(h(X \circ M), h^*(Y(t_s : t_s + l - 1))).$$

Then, for any t such that $t_s \leq t \leq t_s + l - m$, the following inequality holds.

$$\begin{aligned} &S^*(X, Y(t_s : t_s + l - 1)) \\ &= \frac{1}{w(M)} L^h(h(X \circ M), h^*(Y(t_s : t_s + l - 1))) \\ &\geq \frac{1}{w(M)} L^h(h(X \circ M), h(Y(t : t + m - 1) \circ M)) \\ &\geq S(X, Y(t : t + m - 1)). \end{aligned}$$

This means that, if

$$S_{th} \geq S^*(X, Y(t_s : t_s + l - 1)), \tag{10}$$

then, for any t such that $t_s \leq t \leq t_s + l - m$,

$$S_{th} \geq S(X, Y(t : t + m - 1)). \tag{11}$$

That is, if $S_{th} \geq S^*(X, Y(t_s : t_s + l - 1))$, no segment similar to X exists in the section $Y(t_s : t_s + l - 1)$.

In DAL3, Y is divided into sections of length l, then the overlapped histogram $h^*(Y(t_s : t_s + l - 1))$ is calculated for each section as preprocessing. Then, given X, $S^*(X, Y(t_s : t_s + l - 1))$ is calculated. If $S_{th} < S^*(X, Y(t_s : t_s + l - 1))$, DAL is performed for the section $Y(t_s : t_s + l - 1)$. If $S_{th} \geq S^*(X, Y(t_s : t_s + l - 1))$, DAL for it is skipped.

Note that the search result of DAL3 is the same as that of DAL.

3. EXPERIMENTS

For DAL3, the search accuracy and computational efficiency depend on the search threshold S_{th}. In this section, first, we explore the search threshold in terms of the search accuracy in the playlist-generation task. Then, we investigate the computational cost reduction with DAL3.

3.1. Search accuracy vs. search threshold

The experiment assumed the playlist-generation task on television, radio, or broadcasting on the Internet. Since it is difficult to simulate a variety of signal distortions due to the codec, the signal transmission, or other factors, we prepared compressed music signals as input signals. The task was to classify the input signals by retrieving a music piece identical to the input signal from the database and generate the playlist.

We prepared 211 music pieces from the RWC Music Database (Popular, Classical, and Jazz Music Database) [15]. We trimmed the silent parts at the beginning and end of each piece for the experiments. We compressed these music pieces by Ogg Vorbis using sox on a linux computer with two compression levels, 3 (approximately 112 kbps for stereo) and -1 (approximately 48 kbps for stereo), and we used these signals for the input signals. The total length of the input signals is approximately 945 minutes. For the database, we prepared about 0.39 million music pieces, including the original music pieces of the input signals. The total length of the pieces in the database is approximately 1,518,177 minutes (25,303 hours).

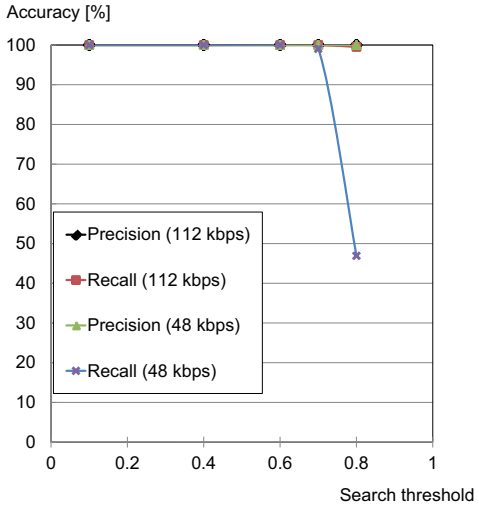

Fig. 2. Search accuracy vs. search threshold. The search accuracy was evaluated for each generated title.

In the experiments, the power spectrum was extracted every 10 ms with 60 band-pass filters equally spaced on the log-scale frequency axis from 140 to 4500 Hz. Then, after the normalization process, the spectrum was decomposed into three components of uniform size on the log-scale frequency axis. This means that $b = 3$. The size of each VQ codebook for each band was 256. Furthermore, the fingerprint data were thinned as in Eq. (4). We used M such that the element of the column every 50 ms was 1 and others were 0.

The playlist was created as follows: First, a 5-s segment was chosen every second of the input signal, and each segment was used as the query (the query signal). For each query signal, the database was searched for a match. A series of continuous hits on the same music piece was concatenated under time consistency filtering to create a playlist.

The accuracy was expressed in terms of the precision rate and recall rate, which we first measured for the titles in the generated playlist . Here,

$$\begin{aligned} \text{Precision} &= \#(\text{correct retrieved title})/\#(\text{retrieved title}), \\ \text{Recall} &= \#(\text{retrieved target title})/\#(\text{target title}). \end{aligned}$$

The target title is the title that should appear in the playlist. We evaluated the accuracy for search threshold values of 0.1, 0.4, 0.6, 0.7, and 0.8.

Figure 2 shows the results. Precision is constantly 100% for every search threshold and for both compression levels. With 112-kbps compression, recall is 100% for the search threshold of 0.7 or less and 99.5% for the search threshold of 0.8. With 48-kbps compression, it is 100% for the search threshold of 0.6 or less and 99.1% for the search threshold of 0.7. When the search threshold value is 0.8, it goes down to 46.2%.

We also evaluated precision and recall for the playlist for every second of the input signals. The results are shown in Fig. 3. In this evaluation, precision is again constantly 100%

for every search threshold and for both compression levels. With 112-kbps compression, recall is more than 99% for the threshold of from 0.1 to 0.7 and 92.6% for the search threshold of 0.8. With 48-kbps compression, it is more than 99% for the threshold of 0.1 and 0.4. For the threshold of 0.6, it is 95.7%. For larger thresholds, it suddenly goes down. The drop in recall for 48 kbps with the search threshold of 0.6 is mostly due to the degradations around the quiet sections of the music, including fade-in and fade-out, and these are not serious false negatives.

Considering these results, the search threshold of 0.6 is appropriate for this task. If the duration of each title in the playlist is not necessary, the search threshold of 0.7 is adequate.

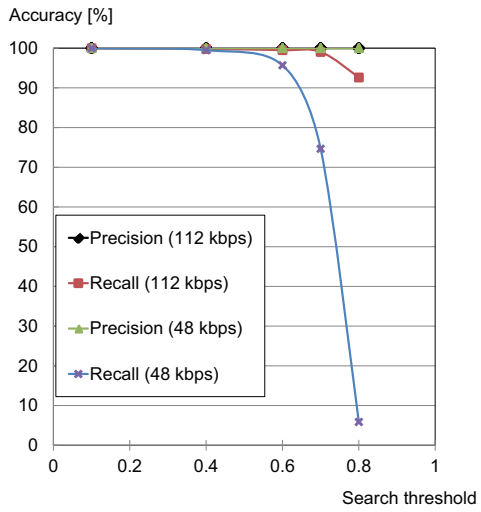

Fig. 3. Search accuracy vs. search threshold. The search accuracy was evaluated every second of the input signal.

3.2. Computational efficiency

We also evaluated the computational cost of DAL3 compared with DAL for the task in Sect. 3.1. Here, we used the same 211 input signals and the database of 1009 music pieces, including the original signals of the input signals. The total length of the music pieces in the database is approximately 5,097 minutes.

We measured the total cost of DAL operations as the total computational cost. To avoid confusion, we denote DAL operation performed in DAL3 as sub-DAL.

For usual DAL, for each query signal, DAL searches are performed through all the stored signals. The computational cost in this case is

$$\#(\text{query signal}) \times (\text{total length of stored signals}). \quad (12)$$

On the other hand, for DAL3, if $S_{th} \geq S^*(X, Y(t_s : t_s + l - 1))$, the sub-DAL operation for the query signal X and the irrelevant section $Y(t_s : t_s + l - 1)$ is skipped as described in Sect. 2.3. Therefore, for DAL3, we measured the total length of the sections on which sub-DAL search was performed. Note that, if sub-DAL search is performed twice for a section for two different query signals, the length of this section is added twice.

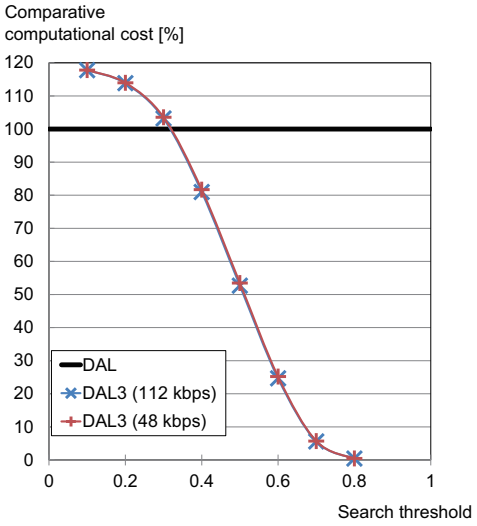

Fig. 4. Comparative computational cost vs. search threshold.

In this experiment, we divided each stored signal into sections of 30-s length with 5-s overlaps for DAL3. The purpose of the overlaps is to avoid segments on the dividing border, which are not retrieved with a query signal of 5-s length. We determined the length of the sections, 30 s, experimentally. Figure 4 shows the comparative computational costs assuming the computational cost for DAL is 1. There is no appreciable difference between 112 and 48-kbps compression. For the threshold value of 0.3 or less, computational costs for DAL3 are larger than those for DAL because of the overlaps of the divided sections. However, for the larger thresholds, we can see that computational costs for DAL3 are quite small. For the threshold value of 0.6, which is adequate for this task, the computational cost is approximately 25%. For the threshold value of 0.7, the computational cost is approximately 6%. DAL3 can considerably accelerate the search by DAL.

Note that the computational cost mentioned above does not include the computational cost for the calculations of $S^*(X, Y(t_s : t_s + l - 1))$ in inequality (10). However, this histogram intersection can be computed efficiently using SIMD instructions of recent processors.

4. CONCLUSION

We have described an approach to accelerate fingerprint techniques and applied it to the DAL (divide-and-locate) method. The search result of the applied method is same as that of DAL mathematically. Experimental results show that DAL3 can reduce the computational cost of DAL to approximately 25%.

For future work, we plan to apply this approach to other fingerprint techniques and accelerate them.

5. REFERENCES

[1] P. Cano, E. Batlle, T. Kalker, and J. Haitsma, "A Review of Audio Fingerprinting," *Journal of Signal Processing Systems*, vol. 41, no. 3, pp. 271–284, Nov. 2005.

[2] V. Chandrasekhar, M. Sharifi, and D. A. Ross, "Survey and Evaluation of Audio Fingerprinting Schemes for Mobile Query-by-Example Applications," in *Proceedings of the 12th International Society for Music Information Retrieval Conference (ISMIR 2011)*, Oct. 2011, pp. 801–806.

[3] K. Kashino, G. Smith, and H. Murase, "Time-series Active Search for Quick Retrieval of Audio and Video," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP-99)*, Mar. 1999, vol. VI, pp. 2993–2996.

[4] K. Kashino, T. Kurozumi, and H. Murase, "A Quick Search Method for Audio and Video Signals Based on Histogram Pruning," *IEEE Trans. Multimedia*, vol. 5, no. 3, pp. 348–357, Sept. 2003.

[5] J. Haitsma and T. Kalker, "A Highly Robust Audio Fingerprinting System," in *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR 2002)*, Oct. 2002, pp. 107–115.

[6] B. Coover and J. Han, "A Power Mask based Audio Fingerprint," in *Proceedings of the 2014 International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2014)*, May 2014, pp. 1408–1412.

[7] Z. Raffi, B. Coover, and J. Han, "An Audio Fingerprinting System for Live Version Identification using Image Processing Techniques," in *Proceedings of the 2014 International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2014)*, May 2014, pp. 644–648.

[8] C. Ouali, P. Dumouchel, and V. Gupta, "A Robust Audio Fingerprinting Method for Content-Based Copy Detection," in *Proceedings of the 2014 12th International Workshop on Content-Based Multimedia Indexing (CBMI)*, June 2014, pp. 1–6.

[9] J. S. Seo, "An Asymmetric Matching Method for a Robust Binary Audio Fingerprinting," *IEEE Signal Processing Letters*, vol. 21, no. 7, pp. 844–847, June 2014.

[10] A. Wang, "An Industrial-Strength Audio Search Algorithm," in *Proceedings of The Fourth International Conference on Music Information Retrieval (ISMIR 2003)*, Oct. 2003, pp. 7–13.

[11] K. Kashino, A. Kimura, H. Nagano, and T. Kurozumi, "Robust Search Methods for Music Signals Based on Simple Representation," in *Proceedings of the 2007 International Conference on Acoustics, Speech and Signal Processing (ICASSP'07)*, Apr. 2007, vol. IV, pp. 1421–1424.

[12] Y. Ke, D. Hoiem, and R. Sukthankar, "Computer Vision for Music Identification," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, June 2005, vol. 1, pp. 597–604.

[13] H. Jégou, J. Delhumeau, J. Yuan, G. Gravier, and P. Gros, "BABAZ: A Large Scale Audio Search System for Video Copy Detection," in *Proceedings of the 2012 International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2012)*, Mar. 2012, pp. 2369–2372.

[14] R. Mukai, T. Kurozumi, K. Hiramatsu, T. Kawanishi, H. Nagano, and K. Kashino, "NTT Communication Science Laboratories at TRECVID 2010 Content-Based Copy Detection," in *TRECVID 2010 Notebook Papers*, Nov. 2010, `http://www-nlpir.nist.gov/projects/tvpubs/tv10.papers/ntt-csl.pdf`.

[15] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Popular, Classical, and Jazz Music Databases," in *Proceedings of the 3rd International Conference on Music Information Retrieval (ISMIR 2002)*, Oct. 2002, pp. 287–288.