

VERY QUICK AUDIO SEARCHING : INTRODUCING GLOBAL PRUNING TO THE TIME-SERIES ACTIVE SEARCH

Akisato Kimura, Kunio Kashino, Takayuki Kurozumi and Hiroshi Murase

NTT Communication Science Laboratories
3-1, Morinosato-Wakamiya, Atsugi-shi, Kanagawa, 243-0198, Japan
E-Mail : {akisato, kunio, kurozumi, murase}@eye.brl.ntt.co.jp

ABSTRACT

Previously, we proposed a histogram-based quick signal search method called Time-Series Active Search (TAS). TAS is a method of searching through long audio or video recordings for a specified segment, based on signal similarity. TAS is fast; it can search through a 24-hour recording in 1 second after a query-independent preprocessing. However, an even faster method is required when we consider huge amount of audio archives, for example a month's worth of recordings. Thus, we propose a preprocessing method that significantly accelerates TAS. The core part of this method comprises a global histogram clustering of long signals and a pruning scheme using those clusters. Tests using broadcast recording indicate that the proposed algorithm achieves the search speed approximately 3 to 30 times faster than TAS. In these tests, the search results are exactly the same as with TAS.

1. INTRODUCTION

This paper proposes a method for quick searching through a long audio stream (we call this a *stored signal*) to detect and locate a known audio signal (a *reference signal* or a *query*) based on signal similarity. The prospective applications include searching broadcast recordings for the use of specific music titles with audio keys.

Previously, we proposed a histogram-based signal search method called Time-Series Active Search (TAS) [3]. TAS takes less than 1 second to detect a 15 second reference signal in a 24-hour stored signal on a standard PC, under the assumption that the signal segments to be detected preserve the same spectral pattern as the reference signal, except for minor distortions or noises. However, when we consider much longer stored signals, for example TV broadcasting recorded over a 1 month period, we need a quicker method than TAS.

Here is proposed an algorithm that can significantly accelerates TAS, while guaranteeing the search accuracy with respect to a certain similarity standard.

This paper is organized as follows: Section 2 overviews the TAS algorithm. Section 3 explains the core part of our

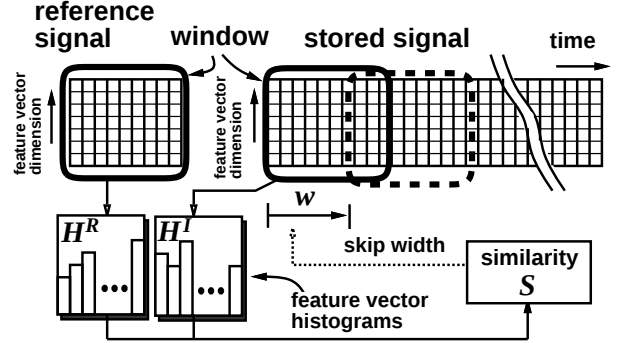


Fig. 1. Overview of TAS algorithm

new algorithm. Section 4 evaluates the speed of the algorithm using a recording of real TV broadcasting. Finally, Section 5 gives conclusions.

2. TIME-SERIES ACTIVE SEARCH (TAS)

Fig.1 shows an overview of the TAS algorithm. In the preparation stage, the feature vectors are calculated from both the reference signal and the stored signal. The feature vector $\mathbf{f}(k)$ is written as

$$\mathbf{f}(k) = (f_1(k), f_2(k), f_3(k), \dots, f_N(k)),$$

where N is the number of frequency channels and k is the sample time. Each element of $\mathbf{f}(k)$ is a normalized short-time power spectrum, which is given as

$$\begin{aligned} f_j(k) &= \alpha(k) Y_j(k), \\ Y_j(k) &= \sum_{t=k-(M+1)}^k y_j^2(t), \\ k &= lM \quad (l = 1, 2, \dots), \end{aligned}$$

where $y_i(t)$ is the output waveform of a second order band-pass IIR filter j at time t , M is the time support of the feature vector, and $\alpha(k)$ is a normalization constant defined as

$$\alpha(k) = \{\max_j(Y_j(k))\}^{-1}.$$

The feature vectors are then quantized using a certain vector quantization (VQ) algorithm. In the search stage, the windows are applied both to the reference feature vectors and to the stored ones. Next, histograms, one for the reference signal and one for the stored signal, are created by counting the number of the feature vectors over the window for each VQ codeword. The similarity between these histograms is then calculated. When the similarity exceeds a given value (a *search threshold*), the reference signal is detected. In the last step, the window on the stored signal is shifted forward in time and the search proceeds.

The main feature of TAS is that it models a signal using feature histograms. The similarity between the reference and stored feature vector histograms over the windows can be determined in several ways. The similarity measure we specifically use is histogram intersection [4]. Histogram intersection S_1 is defined as

$$S_1 = S_1(H_R, H_S) \stackrel{\text{def}}{=} \frac{1}{D} \sum_{i=1}^L \min(h_{Ri}, h_{Si}), \quad (1)$$

where H_R and H_S are the histograms for the reference and the stored signal, h_{Ri} and h_{Si} are the number of feature vectors contained in the i -th bin of H_R and H_S , L is the number of histogram bins, and D is the total number of feature vectors voted in the histogram.

When the similarity value is calculated for one segment, the upper bound of the similarity $\bar{S}_1$ can be determined by

$$\bar{S}_1(n) = S_1(n_1) + \frac{n - n_1}{D},$$

where $\bar{S}_1(n)$ is the upper bound of the similarity when the stored signal window is at the n -th frame, and $S_1(n_1)$ is the similarity when the window stored signal is at the n_1 -th frame [3]. Thus, the histogram matching for the sections that give the upper bound of the similarity not greater than the search threshold can be omitted while guaranteeing no segment to be detected is missed. The skip width w is given by:

$$w = \begin{cases} \text{floor}(D(\theta_1 - S_1)) + 1 & (\text{if } S_1 < \theta_1) \\ 1. & (\text{otherwise}) \end{cases}$$

where $\text{floor}(x)$ means the greatest integer less than x , and θ_1 is the search threshold.

3. INTRODUCING GLOBAL PRUNING

3.1. Framework

TAS can be viewed as a method that accelerates the exhaustive search by using local redundancies in the histogram sequence. In our new method, the basic idea is to remove global redundancies by adding another preprocessing stage.

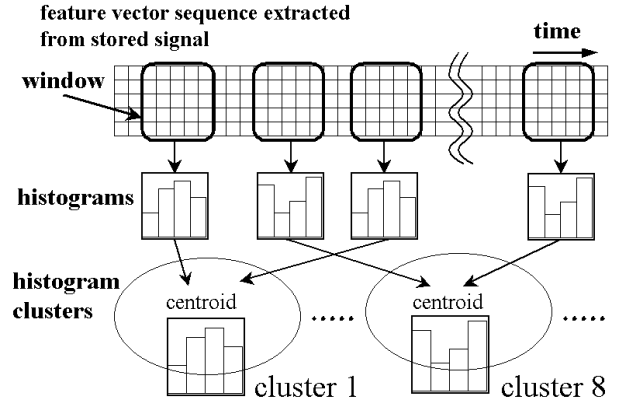


Fig. 2. Outline of the clustering method

3.2. Clustering of Histograms

Fig. 2 outlines the histogram clustering method. In this method the window length (i.e. the reference signal duration) is fixed (ex. created from a 15-second audio signal), while in TAS the length is variable from one search to another. In the TAS case, histograms are created during the search. In our new method, histograms for the stored signal are created prior to the search. Thus, the time-series of histograms with fixed size are created from the stored signal by sliding the window frame by frame. Those histograms are then classified based on a certain distance measure between histograms and using a certain VQ algorithm. Here, we use L_2 -distance (Euclidean distance) as the distance measure of VQ, which is defined as

$$d_2(H_1, H_2) = \sqrt{\sum_{i=1}^L (h_{1i} - h_{2i})^2}. \quad (2)$$

We call the resulting clusters *histogram clusters*. Finally, a *cluster table* is generated, which is a table that lists sections where each histogram cluster covers.

The processing mentioned above is query-independent and is done prior to the search.

3.3. Global Pruning Using the Histogram Clusters

The next step is finding relevant sections on the stored signal that must be searched. The following processing is done after a reference signal is provided.

When the reference signal is provided, a histogram of that signal (a *reference histogram*) is created. The reference histogram belongs to a certain histogram cluster (a *reference cluster*) whose centroid has the minimum distance to the reference histogram in the sense of L_2 -distance [5]. Then, clusters close to the reference cluster are chosen in case the segments to be detected are in different clusters from the reference cluster. Finally, sections that must be searched are determined by using the cluster table.

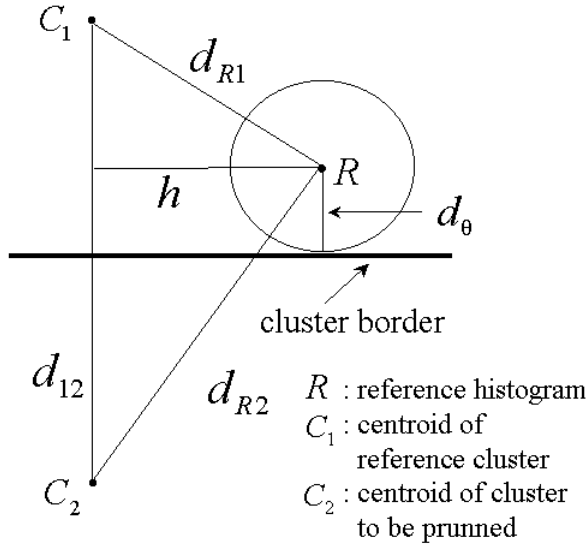


Fig. 3. The condition for selecting clusters to be searched

Here, we should note that the local pruning by the window skipping is based on L_1 -distance, while the global pruning uses L_2 -distance. This strategy is based on our preliminary investigation that L_1 -distance is suitable for distinguishing different signals, while L_2 -distance gives a tight condition for global pruning.

Also, it should be noted that this pruning method introduces no approximation when a proper condition is used in the cluster selection stage. In what follows, we further discuss this point.

Fig. 3 illustrates the situation that the reference signal is given in the search stage. Here, L -dimensional histogram space is sliced by a plane on which the three specific points, R , C_1 and C_2 , simultaneously reside, where R is the reference histogram, C_1 is the centroid of the reference cluster, and C_2 is a centroid of an other cluster. When the minimum distance of the reference histogram and the cluster with centroid C_2 , d_θ , exceeds a given value (*selection threshold*) $\bar{\theta}_2$, we define that histograms in the cluster with centroid C_2 must be matched. First, we would like to determine d_θ . We consider the following distance relationship:

$$\begin{aligned}
 h^2 &= d_{R1}^2 - \left(\frac{1}{2}d_{12} - d_\theta \right)^2 \\
 &= d_{R2}^2 - \left(\frac{1}{2}d_{12} + d_\theta \right)^2,
 \end{aligned}$$

where d_{R1} is the distance between the reference histogram and the centroid of the reference cluster, d_{R2} is the distance between the reference histogram and the centroid of the cluster to be checked, and d_{12} is the distance between the centroids of those clusters. The distances d_{R1} , d_{R2} and

d_{12} are in the L_2 -distance measure. Then,

$$d_\theta = \frac{d_{R2}^2 - d_{R1}^2}{2d_{12}}. \quad (3)$$

Here, we examine the relationship between the accuracy and two threshold values (θ_1 and $\bar{\theta}_2$). First, we convert the histogram intersection measure into its equivalent L_1 -distance form:

$$\begin{aligned}
 S_1 = S_1(H_1, H_2) &= \frac{1}{D} \sum_{i=1}^L \min(h_{1i}, h_{2i}) \\
 &= 1 - \frac{1}{2D} \sum_{i=1}^L |h_{1i} - h_{2i}| \\
 &= 1 - \frac{1}{2D} d_1(H_1, H_2), \quad (4)
 \end{aligned}$$

where d_1 is the L_1 -distance

$$d_1 = d_1(H_1, H_2) \stackrel{\text{def}}{=} \sum_{i=1}^L |h_{1i} - h_{2i}|.$$

Considering the relationships between the L_1 - and L_2 -distance measures, we find that

$$d_2 = d_2(H_1, H_2) \leq d_1 \leq \sqrt{L}d_2 \quad (5)$$

holds. From (4) and (5),

$$\frac{2D}{\sqrt{L}}(1 - S_1) \leq d_2 \leq 2D(1 - S_1). \quad (6)$$

Here, if

$$\bar{\theta}_2 \geq 2D(1 - \theta_1),$$

all the segments satisfying $S_1 \geq \theta_1$ satisfy $d_2 \leq \bar{\theta}_2$, which means that no segment to be detected is missed in terms of L_1 -distance (or histogram intersection). On the other hand, if

$$\bar{\theta}_2 \leq \frac{2D}{\sqrt{L}}(1 - \theta_1),$$

all the segments satisfying $d_2 \leq \bar{\theta}_2$ satisfy $S_1 \geq \theta_1$, which means that no segment to be detected is missed in terms of L_2 -distance. In this article, we chose

$$\bar{\theta}_2 = \frac{2D}{\sqrt{L}}(1 - \theta_1), \quad (7)$$

which guarantees the accuracy in terms of L_2 -distance and also enables efficient pruning.

4. EXPERIMENTS

We tested the proposed search algorithm in terms of the search speed and the search accuracy. In the search speed

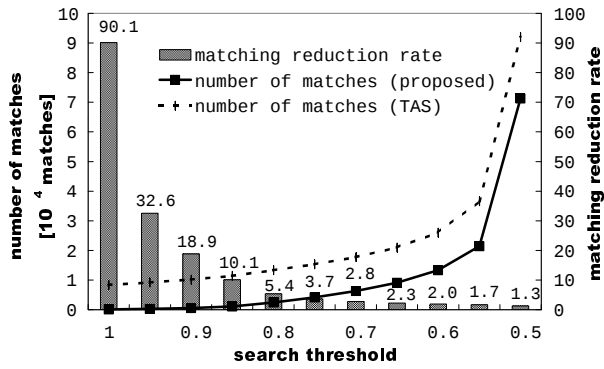


Fig. 4. Number of matches

test, two measures were used: the number of matches and the CPU time in the search. The CPU time includes only the time for the matching stage, and the time for the feature extraction, the feature vector quantization, the histogram clustering and the global pruning are not included. We used a recording of a real 24-hour TV broadcast as a stored signal. The tests were carried out on a PC (Pentium III 966 MHz). In the feature extraction, each signal was first digitized at 11.025 kHz sampling frequency and 8bit quantization accuracy, and then analyzed by a seven-channel ($N = 7$) second order IIR band-pass filter (the filter $Q = 10$) to extract feature vectors. The filter center frequencies were equally spaced in a log frequency scale. The feature vectors were calculated every 110 input samples ($M=110$). The VQ codebook size for feature vectors was 512, and the number of histogram clusters was 512. Those parameter values were empirically chosen. For simplicity, parts of the stored signal were used as reference signals: 12 segments of 15 seconds were randomly chosen every 2 hours from the stored signal.

Fig. 4 shows the number of matches averaged over the 12 reference signals. In this graph, the horizontal axis is the search threshold and the vertical axes are the number of matches (left) and matching reduction rate (MRR; right). MRR is the ratio of the number of matches. The proposed method reduces the number of matches, for example, to 1/10 when the search threshold is 0.85.

Fig. 5 shows the search time measured in the CPU time. The horizontal axis is the search threshold and the vertical axes are the CPU time in the search (left) and time reduction rate (TRR; right). TRR is the ratio of the CPU time in the search. We can see that the proposed algorithm detects the segments, for example, approximately 13 times faster than TAS when the search threshold is 0.85.

In the search accuracy test, on the other hand, the search results given by the proposed method and by TAS were compared in all the search conditions used in the above-mentioned search speed test. In this experiment, those results were completely identical, and also correct. This means that the L_2 -distance-based pruning standard given by (7) is reasonable.

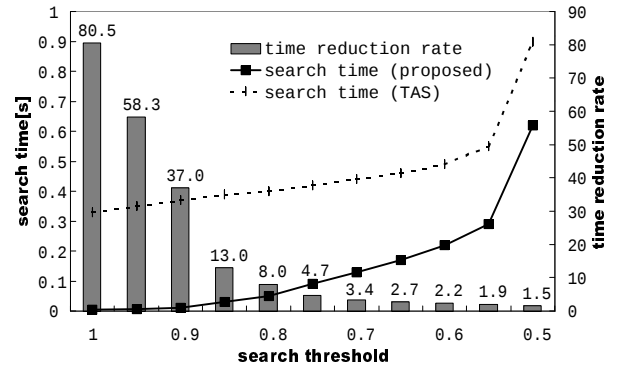


Fig. 5. Search CPU time

5. CONCLUSIONS

We have proposed a global pruning method that significantly accelerates the TAS method, which is a quick search method for audio signals. The proposed method reduces the number of matches to 1/10 (the search threshold = 0.85). We expect that this method could be applied to a search system that retrieves a segment from a month recording within several seconds. Though we focused on audio search in this paper, we will extend it also to video search.

6. ACKNOWLEDGMENT

The authors thank Drs. Ken-ichiro Ishii, Noboru Sugamura and Norihiro Hagita for their help and encouragement.

7. REFERENCES

- [1] R. Mohan: "Video Sequence Matching", *Proc. of ICASSP98*, Vol. 6, pp. 3697-3700, May. 1998.
- [2] N. Katayama and S. Satoh: "The SR-Tree: An Index Structure for High-Dimensional Nearest Neighbor Queries", *Proc. of the 1997 ACM SIGMOD*, Vol. 26, pp. 369-380, 1997.
- [3] K. Kashino, G. Smith, H. Murase: "Time-Series Active Search for Quick Retrieval of Audio and Video", *Proc. of ICASSP99*, Vol. 6, pp. 2993-2996, March. 1999.
- [4] V. V. Vinod and H. Murase: "Focused Color Intersection with Effective Searching for Object Extraction", *Pattern Recognition*, Vol. 30, No. 10, 1997.
- [5] M. Sugiyama: "Fast Segment Search Algorithms", *Technical Report of IEICE*, SP98-141, pp. 39-45, Feb. 1999 (in Japanese).