Scalable Discovery of Audio Fingerprint Motifs in Broadcast Streams with Determinantal Point Process based Motif Clustering

Haotian Xu, Zhijian Ou, *Senior Member, IEEE*

***Abstract*—In this paper, we study the scalable discovery of audio repetitive patterns/motifs in long broadcast streams, where two segments are said to be repetitive if their audio fingerprints are close to each other. In this task, as we are confined to handle limited variability, we can adapt an audio hashing technique, originally proposed for searching a given music clip in music tracks, to successfully devise a linear complexity similarity matching method with a new step of repeated interval formation. This is the first contribution of this paper. As the similarity matching is super fast and thus coarse, there are false alarms in the large number of pairwise matches generated, which constitute a major source of noise. We propose applying subset selection to the original set of pairwise matches based on determinantal point processes (DPPs), as a filtering step, to reduce the noise. The selected subset of pairwise matches is then subjected to motif clustering. We successfully apply DPP-based subset selection to improve motif clustering, which has a nice property that favors both quality and diversity. This is the second contribution of this paper. The proposed method is thoroughly evaluated on a 9-hour real-world audio stream and is compared with several reference methods. The bootstrap technique is used for the significance test. It is shown that the similarity matching is computationally very efficient (above 100 times faster than real time), and the filtering step with DPPs can significantly improve the precision of motif discovery, without sacrificing the recall performance.**

***Index Terms*—Audio motif discovery, audio fingerprinting, determinantal point process, motif clustering.**

I. Introduction

AMONG the rich information types in audio, various pattern repetitions are of particular importance because they reveal some structural properties of the audio data. For different sources of audio data, the definition of what constitute repetitive patterns, often called *motifs*[1], varies significantly with different targeted applications. A variety of research efforts have targeted discovering such repetitive patterns. Related studies include extracting repeating objects from long multimedia streams [1], discovering near-duplicates/motifs in human speech and animal sounds [2], finding the repetitive structure of a music recording [3][4], or extracting repeated spoken terms in speech without automatic speech recognition (ASR) [5].

The authors are with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China. Correspondence should be addressed to Z.O. (ozj@tsinghua.edu.cn).

This work is supported by the National Natural Science Foundation of China (NSFC) via grant 61473168, and Tsinghua Initiative.

[1]As in many studies, the terms motifs and repetitive patterns are interchangeable in this paper.

Although there are differences between the definitions of audio motifs, the technical methods for discovering audio motifs could be conceptually summarized to consist of three main modules - feature extraction, similarity matching and motif clustering. Feature extraction transforms the raw audio into some low-dimensional features. Similarity matching refers to performing comparisons based on the extracted features and generating pairwise matches of segments. The two segments in a pairwise match are hypothesized to be two instances of a motif. We hope that all instances belonging to a common motif could be grouped into one cluster corresponding to one motif, which is the aim of *motif clustering*.

In this paper, we are interested in discovering repeated segments in broadcast radio and television streams in a scalable and unsupervised fashion, where two segments are said to be repetitive if their audio fingerprints [2] are close to each other. In other words, the repetitions defined by audio fingerprints, which we call *audio fingerprint motifs*, are our targets. This task of fingerprint motif discovery from broadcast streams was studied in the pioneering work of [1]. As noted in [1], a difficulty of this task is the large diversity of the repeated segments appearing in the broadcast streams, with respect to their types, durations and occurring frequencies. The repeated segments could be of various types, e.g., commercials, station call-signs, signature tunes of particular television programs, and even entire programs. Their durations can vary from a few seconds or minutes to hours, and the gaps between successive copies can be as short as seconds or minutes or as long as hours. Considering such diversity, we can hardly make any particular assumptions about the nature of repetitions as they appear in the stream.

In terms of scalability, we successfully devise a hashing-based similarity matching method that has *linear complexity* and is scalable for processing long broadcast streams (hours or days), which is in contrast to the quadratic complexity of most existing similarity matching methods. This is the first contribution of this paper. As we are confined to handle limited variability (because we target fingerprint motifs), we can use fingerprint hashing for similarity matching. The hashing technique, which is originally proposed in [8] to search a given music clip in music tracks, is adapted (to our knowledge, the first ever reported and thoroughly validated

[2]An audio fingerprint is a compact signature that summarizes a piece of audio and is currently often used in near-duplicate detection and mainly designed to be consistent under limited distortions such as compression, transmission, play-and-record, equalization and so on [6][7][8][9][10][11].

attempt) to discover fingerprint motifs from long streams in an unsupervised fashion.

Because similarity matching is super fast and thus coarse, the large number of pairwise matches generated is *noisy and redundant*, which is shown in Fig. 1. First, there are false alarm pairwise matches that mostly are short in length (often several seconds) - a major source of noise. Second, for every ground-truth motif interval, there are a number of hypothesized matched intervals produced from similarity matching (being redundant), and they rarely agree on starting and ending times - another source of noise. This presents some difficulty for motif clustering. We propose applying subset selection to the original set of pairwise matches based on determinantal point processes (DPPs) [12] as a filtering step to reduce the noise and redundancy. The selected subset of pairwise matches is then subjected to motif clustering. DPP-based subset selection has a nice property that favors both quality and diversity [12]. In this paper, we successfully apply DPP-based subset selection to improve motif clustering and demonstrate its effectiveness. This is the second contribution of this paper.

The remainder of this paper is organized as follows. In Section II, we discuss related work. Our audio motif discovery system follows the conceptual decomposition of three modules - feature extraction, similarity matching and motif clustering. Section III describes the first two modules together, as hash extraction and hashing-based similarity matching are closely related in our system. Section IV describes motif clustering with DPP-based subset selection. Section V describes our experimental evaluation. Section VI concludes the paper with a discussion of potential future directions. Along with this paper, a continuous 9-hour real-world audio stream recorded from a public broadcast radio in Beijing is manually annotated with motif labels and will be made publicly available for motif discovery research purposes.

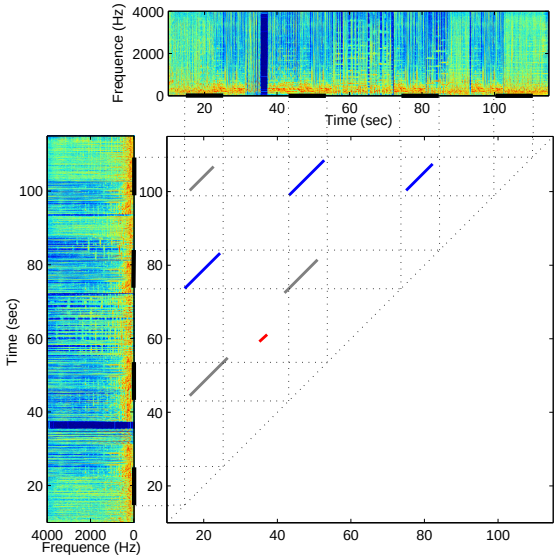


Fig. 1. It is shown that an audio example contains a motif with four occurrences, which are marked in black in the spectrograms. Through similarity matching, a total of seven hypothesized matched pairs of intervals are detected, as shown by the seven diagonal stripes. The selected subset of matched pairs based on DPP is shown with the three blue stripes. The false alarm match is shown by the red stripe.

II. Related Work

A variety of research efforts have investigated unsupervised audio motif discovery. The studies most closely related to our contribution are the methods of similarity matching to generate pairwise matches and motif clustering to derive clusters of mutually similar segments, which will be briefly reviewed below. Although we are interested in methods useful for discovering audio fingerprint motifs from long broadcast streams, we also discuss those matching and clustering methods that were successfully used in other motif discovery or similar tasks, e.g., repetition-based music structure analysis [3][13][14] and spoken term discovery [5][15][16].

A. Similarity Matching

In similarity matching, a self-similarity matrix (SSM) is a widely used data structure in many studies, e.g., [13] in music tasks (see [3] for more survey), [16] in spoken term discovery, and [10] in near-duplicate/motif discovery. A crucial observation is that repeating patterns in the feature sequence appear as diagonal stripes parallel to the main diagonal in a SSM. These stripes manifest the high similarity of a pair of segments. One possible approach is computing a SSM and searching for the diagonal stripes. This approach has been studied and improved extensively to meet different requirements in various tasks. We briefly describe two improvement directions, representing two extremes of balancing between allowing more variations and achieving fast matching.

On the one hand, in order to accommodate more variations (e.g., time warping, tempo) between repetitions, dynamic time warping (DTW) or variants [5][14] have been developed for similarity matching to tolerate more variability. This tolerance is crucial for music structure analysis and spoken term discovery, but is not necessary for detecting near-duplicates/fingerprint motifs, as pursued in this work. Moreover, the drawback is that the scalability is limited by the quadratic computational complexity.

On the other hand, a major concern in some tasks is speedup matching. For this efficiency purpose, hashing techniques are particularly attractive. One approach, as used in [11], first builds a similarity trellis based on hash collisions and then uses a Viterbi-like algorithm to find the continuous paths in the trellis (much like finding diagonal stripes in a SSM). The other approach, as used in [10], builds a histogram of differences of time offsets of every pair of hash collisions. Then, the pair-wise match detection is implemented by detecting local maximums in the hashing histogram, which is even more efficient than the Viterbi-like algorithm. Both works [10][11] use Haitsma and Kalker hashing [7].

Finally, it is worthwhile to comment on some similarity matching methods that do not rely on the use of SSMs. The work in [1] proposes a sequential searching and buffering strategy, which is adapted to spoken term discovery in [15]. This strategy is of linear complexity but makes assumptions on the motifs' durations and occurring frequencies. In [2], the top-K closest pairs are found by using probabilistic early

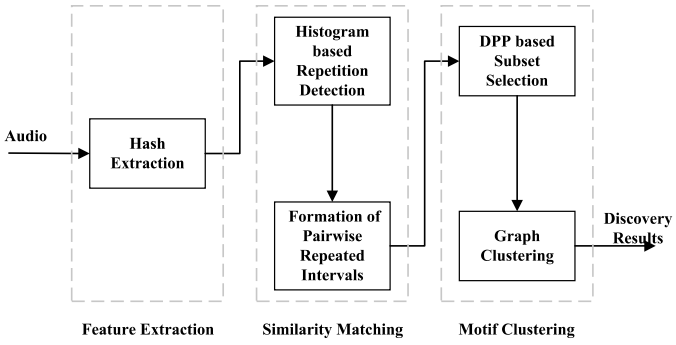


Fig. 2. The architecture of our fingerprint motif discovery system

abandoning. However, these methods operate only around real-time, whereas we aim for processing large volumes of broadcast streams in super real-time.

B. Motif Clustering

Compared to similarity matching, motif clustering is less studied in the existing literature. Most near-duplicate detection systems do not produce clustered results [11][17] or simply use transitive closure to combine matched pairs into clusters in [10].

In music structure analysis [3], the task of thumbnailing is mainly concerned with finding the one segment that best represents a music recording. In the case that a more comprehensive structural description is required, the spectral clustering method [13] and transitive closure reasoning [18] are used to group similar segments. An interesting work with similar motivation for studying motif clustering is [14], with two steps, stripe extraction from a SSM of a music recording and grouping pairwise matched segments into musically meaningful parts. The problem addressed in [14] is that the stripe structures in a SSM are noisy and fragmented, which makes both steps prone to error. This problem is similar to what we discuss in the Introduction: the pairwise match results are noisy and redundant. The proposed method of jointly performing the stripe extraction and grouping step for a given segment is good for analyzing a music recording, but may not be appropriate for processing long broadcast streams to find *all* motifs with a large diversity in types, durations and occurring frequencies.

In spoken term discovery, the spectral clustering of hypothesized intervals is studied in [19] and shown to work better than graph clustering in [5]. Noting that it is hard to determine the number of clusters, [19] proposes producing only a fixed number of the biggest clusters. This somewhat meets the task requirement of discovering word motifs with high recurrence. However, it is not good for the task of discovering *all* fingerprint motifs that do not necessarily have high recurrence as long as there are repeats, which is our primary concern in this paper.

III. Hashing-based Similarity Matching

The architecture of our fingerprint motif discovery system is presented in Fig. 2. It follows the general decomposition of having three modules: feature extraction, similarity matching and motif clustering. In our system, the first two modules together are responsible for producing pairwise match results and will be introduced together in this section. Motif clustering is further decomposed into two submodules, which are described in Section IV.

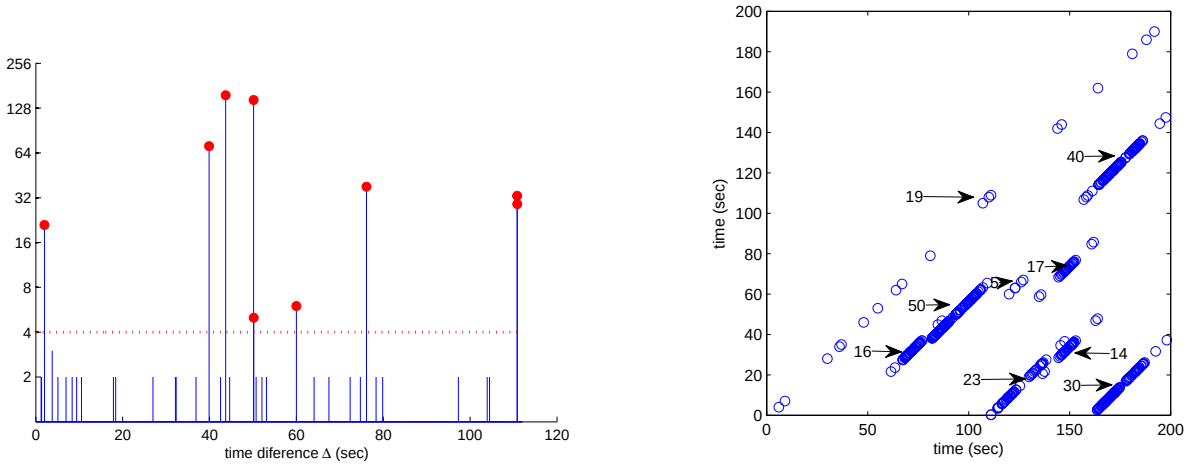
For the purpose of scalable similarity matching, hashing techniques are particularly attractive. Haitsma and Kalker hashing [7] extracts the signs of the energy band differences along both the time and frequency axes as a 32-bit hash and is used in [10][11] for commercial extraction. The Wang hashing [8] examines only spectrogram peaks and also generates 32-bit hashing. It is originally proposed for the task of searching a given music clip in song databases and shows superior performance in benchmarking evaluations of this task [20] when compared to Haitsma and Kalker hashing. However, to our knowledge, it has not yet been applied or thoroughly validated for the task of unsupervised motif discovery from long streams, although it is a well-known hashing technique in itself and mostly used in query-and-searching/retrieval tasks [17][21]. In this paper, the Wang hashing technique is adapted to our task. The resulting hashing-based similarity matching includes three steps: hash extraction, histogram-based repetition detection and formation of repeated intervals. While the first two steps may seem as a direct adaptation of Wang hashing and the histogram technique, the third step is *new* (see Section III-D for a discussion with related work).

A. Hashing Extraction

We review the main steps of the Wang hashing algorithm as follows. First, the magnitude spectrogram $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$ of the audio is created, with a frame length of 64 ms and 50% frame overlap. Then, we detect time-frequency peaks in the spectrogram $\mathbf{X}$, which are more likely to survive ambient noise. Finally, hashing is conducted for the pairs of peaks, called landmarks. Specifically, a fixed number of peaks (up to 5) is chosen for each frame. Each peak (as an anchor) is combinatorially paired with other following peaks (up to 3) within its target zone (32 frequency bins by 64 frames) to form landmarks, as shown in Fig. 3. For each landmark, i.e., a pair of associated peaks in the time-frequency plane, $L \triangleq \{(t_1, f_1), (t_2, f_2)\}$, $1 \leqslant t_1 < t_2 \leqslant T$, the numbers in the triplet of $(f_1; (t_2 - t_1); (f_2 - f_1))$ are quantized. The concatenated bit-string is used as the hash value for this landmark. The absolute time offset of the anchor peak from the beginning of the audio, t_1, is called the anchor time of the landmark and is not a part of the hash itself. The details can be found in [8].

B. Histogram-based Repetition Detection

After hash extraction, we are ready to find pairwise repeated intervals within the long audio $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$. An important data structure is the $T \times T$ sized self-similarity matrix (SSM). Pairwise repeated intervals in $\mathbf{X}$ manifest themselves as diagonal stripes in the visualization of this matrix. To detect diagonal stripes, a histogram of the time differences resulting from all hash collisions is constructed, as shown in Fig. 4a. At the same time, we can imagine a scatterplot



(a) (b)

Fig. 4. (a) The histogram of the time differences resulting from hash collisions. There are 9 local peaks that are higher than the threshold $\delta_{height} = 4$ (b) The scatterplot that displays hash collisions. The 9 diagonal line segments that correspond to the 9 local peaks in the left histogram are indicated by arrows, where the numbers at the end of the rows are the corresponding bin heights.

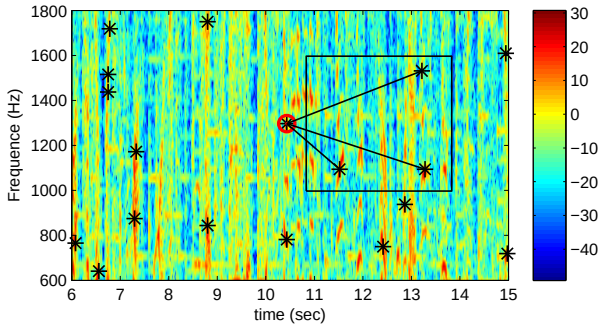


Fig. 3. An illustration of the hashing extraction technique [8]. The detected peaks in the spectrogram are denoted using the stars. The red circled star is the anchor peak, which is paired with other following peaks within its target zone to form landmarks.

for all hash collisions (i.e., a binarized SSM), as shown in Fig. 4b. If there are n landmarks with the same hash value and thus hashed into the same bucket, there are $\binom{n}{2}$ hash collisions for that hash bucket and $\binom{n}{2}$ points in the scatterplot. For two matched landmarks resulting from a hash collision, $L^a \triangleq \{(t_1^a, f_1^a), (t_2^a, f_2^b)\}$ and $L^b \triangleq \{(t_1^b, f_1^b), (t_2^b, f_2^b)\}$, the time difference between the two landmarks is defined as $|t_1^a - t_1^b|$, the absolute difference of the anchor times for the two landmarks. Accordingly, there is a point located at (t_1^a, t_1^b) in the scatterplot, and the height of the histogram bin at $|t_1^a - t_1^b|$ is incremented by one. Due to symmetry, we only need to consider points at the lower right half-plane of the scatterplot. All time differences resulting from all hash collisions are put into bins to form a histogram of time differences. For time difference $\triangle = 0, 1, \ldots, T-1$, the height of the histogram bin at $\triangle$ is exactly the number of points in the diagonal line with x-intercept $\triangle$. An example of the histogram and scatterplot for a 200-second audio is shown in Fig. 4.

To search for pairwise repeated intervals, we exploit the so-called *temporal consistency*, which means that the time differences between the matched landmarks within two matched intervals should be consistent. If an interval is matched with another interval $\triangle$ frames later, there should be a number of hash collisions between the landmarks within these two intervals, and the time differences resulting from these hash collisions are most likely to be $\triangle$. Consequently, the height of the histogram bin at $\triangle$ is most likely to be a (local) maximum.

Therefore, the pairwise repeated intervals could be detected by first searching for local maximums in the hashing histogram. It is important to note that for the simple task of identifying a short clip of music out of a song, it is enough to search for the global maximum in the histogram to declare if the clip appears in the song or not. To find all pairs of repeated intervals, we need to not only search for local maximums in the histogram but also conduct some further steps to link matched landmarks to form pairwise repeated intervals, described in the following.

C. Formation of Pairwise Repeated Intervals

Searching for local maximums in the histogram is a good starting point for searching for pairwise repeated intervals, but not adequate, especially for determining the boundaries of the repeated intervals. There are two main reasons. First, the matched pairs of landmarks with the same time difference are collapsed into the same bin, so the bin height carries no boundary information. There may be more than one repeated interval in a diagonal line. Second, accidental landmark matches usually co-exist together with expected matches, and the generation of landmarks does not guarantee that the landmarks from two matched intervals are continuously matched one after another. Therefore, there might be outliers and holes along the diagonal line corresponding to the selected local peaks in the histogram.

For efficiency, the following three steps are used in this paper to link matched landmarks to form pairwise repeated intervals. For easy reference, Table I lists the parameters and

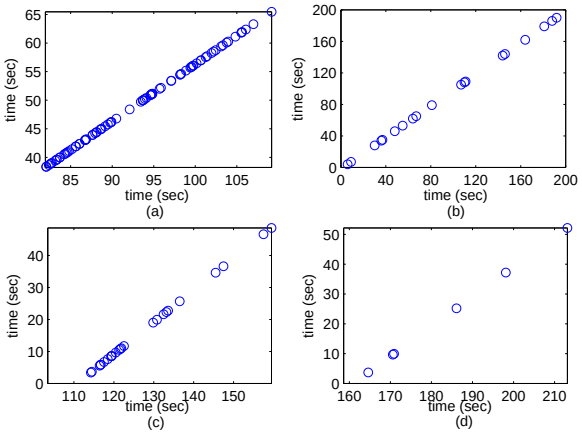


Fig. 5. Different cases for clustering the points over the diagonal lines in the scatterplot. (a) The desired case - hash collisions from two matched interval are consistently aligned. (b) Uninteresting hash collisions that are near the main diagonal of the scatterplot, to be filtered by δ_{bin}. (c) Outliers and holes along a diagonal due to accidental landmark matches. (d) The sporadic hash collisions should be discarded.

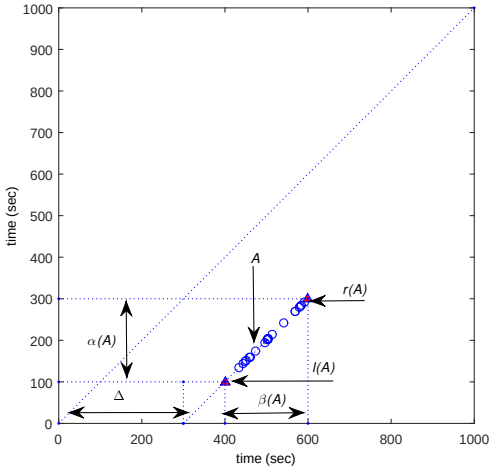


Fig. 6. An illustration of the math symbols used in this paper to denote the cluster of points A within a diagonal line, which essentially represents a matched pair of intervals $\alpha(A)$ and $\beta(A)$.

their values used in this paper for the experiments described in Section III. Fig. 5 shows the motivating cases that lead to the three-step method.

1) Pre-Filtering: In this step, first, the prominent local peaks where the corresponding bin heights are above a given threshold δ_{height} are selected from the histogram, which can be seen in Fig. 4a. Second, note that a landmark is highly probable to have the same hash value with some nearby landmark, which is generated only several frames away. There are a large number of points near the main diagonal of the scatterplot, as shown in Fig. 5b, implying that a large number of intervals are "repeated" only several frames away. Such overlapped repetitions are not of interest. Therefore, the bins corresponding to time differences less than δ_{bin} are excluded to avoid the overlapped repetitions. The above two steps remove the minor bins from the histogram, which are either of low height or for which the corresponding time differences are small. The resulting scatterplot is actually a filtered version of the original scatterplot, where all minor diagonal lines, mainly consisting of accidental matches, are eliminated. The remained diagonal lines are called major diagonal lines.

2) Clustering: For each major diagonal line in the lower right half-plane of the scatterplot, the points lying on the diagonal line are clustered to form intervals. Following the uphill direction along the diagonal line with x-intercept $\triangle > 0$, we denote the points as a sequence $\{p_i, i = 1, \ldots, K\}$, where $p_i = (p_{i,x}, p_{i,y})$ are the horizontal and vertical coordinates of point i in the scatterplot and $p_{i,x} - p_{i,y} = \triangle$. Then, a cluster of points, A, over the diagonal line with x-intercept $\triangle$ essentially represents a pair of repeated intervals with lag $\triangle$, which could be denoted as

$$\alpha(A) \triangleq [p_{l(A),y}, p_{r(A),y}], \tag{1}$$

$$\beta(A) \triangleq [p_{l(A),x}, p_{r(A),x}], \tag{2}$$

where

$$l(A) = \underset{i \in A}{argmin}\, p_{i,x}, \tag{3}$$

$$r(A) = \underset{i \in A}{argmax}\, p_{i,x}, \tag{4}$$

are the left-most and right-most points in the cluster A. An illustration of these math symbols is shown in Fig. 6.

To cluster the points over a diagonal line, we first sort the points in ascending order based on the frame number. Two adjacent points, p_i and p_{i+1}, are connected if the distance between them is below a given threshold,

$$\| p_i - p_{i+1} \| \leq \delta_{gap}, i = 1, \ldots, K-1 \tag{5}$$

Then, the groups of connected points are used as clusters.

3) Post-Filtering: After the clustering, we obtain a few pairwise repeated intervals corresponding to the clusters of points along each major diagonal line. These pairwise repeated intervals are subject to the following filtering to produce the final pairwise repeated intervals.

- The width of the interval should be larger than a given threshold δ_{width}.
- The size of the cluster, i.e., the number of points in the cluster, should be larger than a given threshold δ_{num}.

In summary, the formation of repeated intervals consists of three steps: pre-filtering using δ_{height} and δ_{bin}, the clustering using δ_{gap}, and the post-filtering using δ_{width} and δ_{num}. It is designed mainly to address the problem that there might be outliers and holes along the diagonal lines. Filtering removes unwanted outliers, and clustering enables us to link points to form intervals, although there are holes due to the fingerprints. The pairwise repeated intervals, which are formed from applying the three-step method to Fig. 4b, are shown in Fig. 7.

D. Related Work on Repeated Interval Formation

It is worthwhile discussing two other methods related to repeated interval formation and commenting on the novelty of

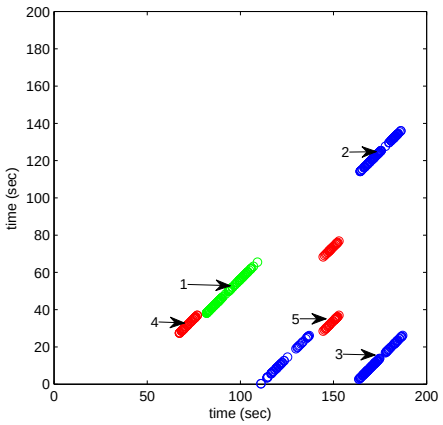


Fig. 7. The set of pairwise repeated intervals, $\mathcal{Y}$, which is formed by applying the three-step method to Fig. 4b. The DPP-based selection result Y is also shown, where the numbers at the end of the rows indicate the selected orders. The matched pairs from a common motif have the same color.

our method. These two methods are also based on the temporal consistency of repetitions and thus have some resemblance with our method.

The study by Duong et al. [22] is close to our method in spirit in that it also uses the Wang hashing [8], the scatterplot and the histogram to synchronize multiple versions of the same movie. Additional steps are introduced to refine the match and eliminate outliers. In particular, a hierarchical clustering-based approach is proposed, where the Euclidean distance between every pair of clusters is computed and the clusters for which the distance is smaller than a pre-defined threshold are merged. This implementation of clustering is computationally more expensive than our clustering implementation described above, although theoretically, the clustering results may not differ much.

Wu et al. [10] propose temporal recurrence hashing for detecting duplicate commercials. First, a scatterplot resulting from all hash collisions is constructed, similar to our method, but using the Haitsma and Kalker hashing technique with a fixed frame rate of 30. Then, a two-dimensional histogram is further built from the scatterplot, and repeated intervals are directly determined by detecting local peaks in the histogram and applying a fixed threshold. Specifically, for a hash collision between the fingerprints at times t^a and t^b, the bin at $\left(\mid t^a - t^b \mid, min(t^a, t^b)\right)$ within the two-dimensional histogram is incremented by 1. The first dimension, $\mid t^a - t^b \mid$, is the time difference of the hash collision and represents the first dimension of the histogram, and the second dimension, $min(t^a, t^b)$, is the first occurrence time of the matched fingerprint and represents the second dimension of the histogram. A remarkable treatment is that the resolutions of $\mid t^a - t^b \mid$ and $min(t^a, t^b)$ are set to be 1 second and 1 minute, respectively. Therefore, an equivalent view of their method is imagining a partition of the scatterplot into diagonal strips, each with width 1 second and height 1 minute. The number of points within each diagonal strip is then used as the bin height for the two-dimensional histogram, which is subjected to local peak selection and thresholding. Setting the width resolution to be 1 second is mainly done to accommodate the time resolution of the fingerprint used in their method. In [10], fingerprint hashing is conducted for a fragment formed by combining continuous frames with the same fingerprint. Temporal consistency is more likely to be satisfied at the resolution of 1 second. Setting the height resolution to be 1 minute mainly suits the specific application scenario studied in [10]. The duplicate audio/video to be detected is a TV commercial, the duration of which is assumed to be no more than 30 seconds. This special treatment is not necessary for our method. As the Wang hashing technique we used is more robust to distortions and accurate in timing, there is no need to sacrifice the time resolution. Additionally, there is no need to limit the length of a motif.

Algorithm 1 Greedy DPP-MODE Algorithm

Inilization: $\mathbf{L}$

Output: $\hat{C}$

1: Set $\hat{C} \leftarrow \emptyset$, $U \leftarrow \mathcal{Y}$;
2: **while** U is not empty **do**
3: Compute $\mathbf{L}^* = \left(\left[(\mathbf{L} + I_{\bar{\hat{C}}})^{-1}\right]_{\bar{\hat{C}}}\right)^{-1} - \mathbf{I}$;
4: $\mathbf{L} \leftarrow \mathbf{L}^*$
5: $U \leftarrow \{i | i \notin \hat{C}, \mathbf{L}_{ii} > 1\}$
6: **end while**
7: **Return:** $\hat{C}$

IV. DPP BASED MOTIF CLUSTERING

After the hashing-based similarity matching introduced in Section III, we obtain a number of pairwise repeated intervals, which could be represented as a set

$$\mathcal{Y} = \{A_1, \cdots, A_N\}. \tag{6}$$

As defined as in Eq. (1) and Eq. (2), $A_i \triangleq (\alpha(A_i), \beta(A_i))$ denotes a matched pair of intervals, $i = 1, \ldots, N$, indicating that the interval $\alpha(A_i)$ is hypothesized to repeat at a later time along the interval $\beta(A_i)$. This also means that the two intervals, $\alpha(A_i)$ and $\beta(A_i)$, are two hypothesized instances of a motif. Taking $\mathcal{Y}$ as the input, motif clustering is used to build clusters of intervals so that all hypothesized intervals belonging to a common motif are grouped together in one cluster. Moreover, we need to generate only one hypothesized interval for each instance of a motif.

As pointed out in the Introduction, the pairwise match results in $\mathcal{Y}$ are noisy and redundant. We propose applying subset selection to $\mathcal{Y}$ before clustering, which could be understood as a filtering step to filter out the noise and redundancy in $\mathcal{Y}$. As shown in our experiments, this filtering step with DPPs can significantly improve the precision of motif discovery, without sacrificing the recall performance.

A. DPP Background

The determinantal point processes (DPPs) [12] are elegant probabilistic models for subset selection problems where both quality and diversity are considered. Formally, given a set of

items $\mathcal{Y} = \{1, \cdots, N\}$, a DPP defines a probability measure $\mathcal{P}$ on $2^{\mathcal{Y}}$, the set of all subsets of $\mathcal{Y}$. For every subset $Y \subseteq \mathcal{Y}$, we have

$$\mathcal{P}(Y) \propto \det(\mathbf{L}_Y), \tag{7}$$

where $\mathbf{L}$, called the L-ensemble kernel, is an N by N positive semidefinite matrix and can be written as a Gram matrix $\mathbf{L} = \mathbf{\Gamma}^T\mathbf{\Gamma}$. Intuitively, $\det(\mathbf{L}_Y)$ could be viewed as the squared volume spanned by $\mathbf{\Gamma}_i, i \in \mathcal{Y}$, the column vectors of $\mathbf{\Gamma}$, representing items in $\mathcal{Y}$.

A popular decomposition of the kernel is to define $\mathbf{B}_i = q_i\boldsymbol{\phi}_i$, where $q_i \in \mathbb{R}^+$ measures the quality (magnitude) of item i in $\mathcal{Y}$, and $\boldsymbol{\phi}_i \in \mathbb{R}^k$, $\|\boldsymbol{\phi}_i\| = 1$, can be viewed as the angle vector of *diversity features* so that $\boldsymbol{\phi}_i^T\boldsymbol{\phi}_j$ measures the similarity between items i and j, $1 \leq i, j \leq N$. It can be shown that the probability of including i and j increases with the quality of i and j and diversity between i and j. As a result, a DPP assigns a high probability to subsets that are both of good quality and diverse. For subset selection, the inference of the mode (referred to as DPP-MODE)

$$\underset{Y \subseteq \mathcal{Y}}{argmax} \det(\mathbf{L}_Y) \tag{8}$$

is used to find the most probable subset. Although DPP-MODE inference is generally an NP-hard problem [23], the greedy DPP-MODE algorithm in Algorithm 1 is shown to perform well [24] and is used in our experiments.

In summary, the L-ensemble kernel $\mathbf{L}$ could be defined as

$$\mathbf{L} = diag(\mathbf{q}) * \mathbf{S} * diag(\mathbf{q}), \tag{9}$$

where $\mathbf{q}$ is the quality vector consisting of q_i, $diag(\mathbf{q})$ denotes the diagonal matrix that uses the vector $\mathbf{q}$ as its main diagonal, and $\mathbf{S}$ is the similarity matrix consisting of $S_{ij} = \boldsymbol{\phi}_i^T\boldsymbol{\phi}_j$. The quality-diversity decomposition allows us to construct $\mathbf{q}$ and $\mathbf{S}$ separately to address different concerns, which is utilized below to construct the DPP kernel for filtering the candidate matched pairs.

Finally, note that in practice, instead of explicitly designing normalized feature vectors $\boldsymbol{\phi}_i$, it is sometimes more convenient to directly build the positive semidefinite matrix $\mathbf{S}$, which implies the feature vectors. The diversity behavior of DPPs is encoded in the functional form of the similarity matrix $\mathbf{S}$. This is analogous to the fact that the behavior of a function drawn from a Gaussian process with zero mean is encoded in its covariance kernel function. Many classes of positive semidefinite kernel functions have been proposed [25]. A common choice is the squared exponential kernel, also known as the Gaussian kernel, which is used in our work as described below.

B. Quality-Diversity Decomposition of the DPP Kernel

1) Quality: To measure the quality of a matched pair A_i, without loss of generality, we use the duration of its first interval $\alpha(A_i)$. The longer the duration, the higher the confidence we retain in the matched pair in the subset, rather than filtering it out. Specifically, we define

$$q_i = p_{r(A_i),y} - p_{l(A_i),y} \tag{10}$$

2) Diversity: The similarity matrix $\mathbf{S}$ is defined to reflect the diversity among the candidate matched pairs, so that we can reduce redundancy and noise by subset selection.

With no loss of generality, we check the first intervals of each candidate matched pair. Due to the imprecise nature of hashing-based similarity matching, there are overlaps between the first intervals. The overlapping can happen when the two first intervals are detected with only slightly different beginning and ending locations, or when there are false alarm short intervals. Therefore, the overlap between the first intervals from two candidate matched pairs indicates some redundancy or noise. In order to encourage the model not to choose overlapping first intervals, our diversity model is designed to reflect the difference between the first interval locations.

Specifically, we build the similarity matrix as an addition kernel:

$$S_{ij} = \frac{S_{ij}^b + S_{ij}^e}{2}, \tag{11}$$

where

$$S_{ij}^b = exp\left(-\frac{1}{2\sigma_{loc}^2}\left(p_{l(A_i),y} - p_{l(A_j),y}\right)^2\right) \tag{12}$$

$$S_{ij}^e = exp\left(-\frac{1}{2\sigma_{loc}^2}\left(p_{r(A_i),y} - p_{r(A_j),y}\right)^2\right) \tag{13}$$

where σ_{loc} is the variance parameter representing the location diversity level and is set to 3 second in our experiment. Fig. 8 shows an example of the similarity matrix $\mathbf{S}$ constructed according to Eq. (11). As a result, candidate match pairs with close beginning or end locations of the first intervals appear similar, and the model prefers to select non-overlapping first intervals.

3) Balance: Note that from the above definition, the quality q_i is the duration of the first interval from the candidate matched pair, which takes values ranging from tens of seconds to hundreds of seconds, while the similarity S_{ij} takes much smaller values (< 1). The resulting DPP model will thus show a biased preference to quality, regardless of the diversity. Therefore, to balance quality and diversity, we use the logarithm and square root function to compress the dynamic range of the durations and re-define

$$q_i = \sqrt{log\left(p_{r(A_i),y} - p_{l(A_i),y}\right)} \tag{14}$$

Fig. 7 shows an example result of DPP-based subset selection, where the selection order of the items (i.e., matched pairs) are labeled. The selection result is as desired to filter out the redundancy and noise. Longer items (i.e., high quality) are preferentially selected first, while avoiding selection of overlapping items (i.e., being diverse).

C. From Pairwise Matched Intervals to Clusters of Repeated Intervals via Graph Clustering

After DPP-based subset selection, we obtain a selected subset of pairwise matched intervals $Y \subseteq \mathcal{Y}$, denoted as

$$Y = \{B_1, \cdots, B_M\}. \tag{15}$$

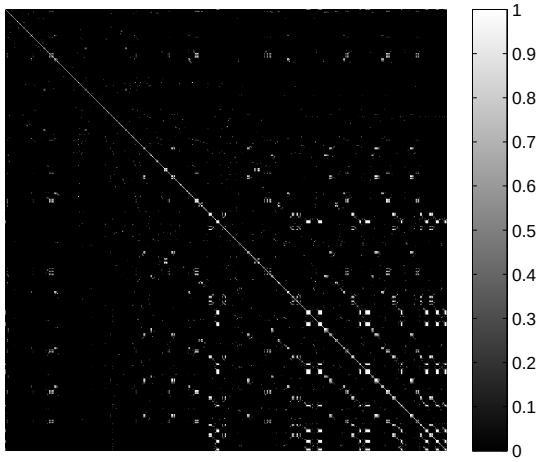


Fig. 8. The similarity matrix $\mathbf{S}$ of size 920×920 constructed from the 9-hour real-world audio according to Equ. (11).

Starting from these filtered pairwise matched intervals, this stage involves building clusters of repeated intervals. Each cluster will be summarized as a motif with a number of motif instances. Graph clustering is an approach used to address this problem.

A wide range of graph clustering techniques can be applied at this stage. We use a classic technique. Consider the matched pairs as the nodes, and link two nodes with an edge if the similarity between the two nodes exceeds a threshold δ_{edge}. Subsequently, we find connected components to generate the clusters of the nodes, which then translate into clusters of intervals.

An important question at this stage is the definition of the similarity measure (i.e., the edge weight) between two matched pairs of intervals. Specifically, we propose defining the similarity between two matched pairs B_k and B_m in Y, $1 \leq k, m \leq M$, as a product of two terms:

$$W(k,m) = W_1(B_k, B_m) W_2(B_k, B_m) \tag{16}$$

where

$$W_1(B_k, B_m) = max\Bigg\{ \frac{|\alpha(B_k) \cap \alpha(B_m)|}{min\left(|\alpha(B_k)|, |\alpha(B_m)|\right)}, \frac{|\alpha(B_k) \cap \beta(B_m)|}{min\left(|\alpha(B_k)|, |\beta(B_m)|\right)}, \frac{|\beta(B_k) \cap \alpha(B_m)|}{min\left(|\beta(B_k)|, |\alpha(B_m)|\right)}, \frac{|\beta(B_k) \cap \beta(B_m)|}{min\left(|\beta(B_k)|, |\beta(B_m)|\right)} \Bigg\} \tag{17}$$

$$W_2(B_k, B_m) = exp\left(- \frac{\left[\left(p_{r(B_k),y} - p_{l(B_k),y}\right) - \left(p_{r(B_m),y} - p_{l(B_m),y}\right)\right]^2}{2\sigma_{dur}^2} \right) \tag{18}$$

where $|.|$ denotes the length of an interval and $\cap$ the intersection of two intervals. $W_1(B_k, B_m)$ calculates the overlapping ratios between either interval from the first matched pair B_k and either interval from the second matched pair B_m, and the maximum is chosen. Therefore, it measures how much the two matched pairs overlap. $W_2(B_k, B_m)$ calculates the difference between the durations of the first intervals (without loss in generality) from the two matched pairs and measures how similar the two matched pairs are in terms of durations. σ_{dur} is the variance parameter representing the duration difference level and is set to 3 second in our experiments. The combination of $W_1(B_k, B_m)$ and $W_2(B_k, B_m)$ yields a good similarity measure. This is in contrast to the previously used similarity measures in other motif clustering studies, e.g., [5][10][18], most of which only measure the degree of overlap.

With the above definition of edge weights and a weight threshold δ_{edge}, a depth-first-search algorithm is used in our experiments to find the connected components. All the intervals appearing in a connected component are assigned to a cluster, corresponding to a motif. If any two candidate intervals in a cluster overlap in time, they are merged and replaced by their union. Finally, the discovered motifs are

$$\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_D\}, \tag{19}$$

where each discovered motif $\mathcal{D}_d$, $1 \leq d \leq D$, is a set of non-overlapping intervals, and each element in $\mathcal{D}_d$ is a discovered motif instance.

D. Discussion of the Clustering Methods

One can roughly distinguish between two different classes of clustering methods in motif discovery or similar tasks. Connectivity-based clustering, also known as hierarchical clustering, connects objects to form clusters based on their distances. Loosely speaking, the graph clustering used in this paper and in [5] and clustering by transitive closure [10][18] are connectivity-based clustering methods. In a centroid-based clustering method, clusters are represented by a central vector in the original or transformed feature space. K-means clustering and extensions (kernel k-means, mixture models) and spectral clustering [13][19] are centroid-based clustering methods.

Motif clustering has not received much attention in the existing literature. This is partly because in most previous studies, the pairwise match results are easy for grouping, after an expensive and elaborate similarity matching step. However, in our task of processing long broadcast streams, the similarity matching is super fast and thus coarse, and false alarms are generated in the large number of pairwise matches, as illustrated in Fig. 1. Direct clustering, no matter what clustering method is used (connectivity based or centroid based), cannot filter the false alarms. The main point here is the necessity of applying subset selection before clustering in our case, not the superiority of a particular clustering method. Theoretically, after DPP-based subset selection, any clustering method can be applied to the selected subset of pairwise matched intervals Y. The graph clustering technique designed in Section IV-C is one choice (with some novelty), which is found to perform well in

the experiments. Therefore, in the following experiments, we are more interested in comparing the performances obtained with and without DPP-based filtering.

For clustering without DPP-based filtering, the tested methods include graph clustering, agglomerative clustering and kernel k-means clustering. The first two are connectivity based, and the third is centroid based. It is important to note that the different clustering methods appearing in other motif discovery or similar studies are covered as much as possible with reference to the adaptation to our task in this paper. Please refer to Section V-B for more details on the comparison experiments.

V. Experiments

As shown in Fig. 2, our fingerprint motif discovery system consists of two stages, hashing-based similarity matching and DPP-based motif clustering, which also correspond to the two contributions of this paper and are introduced in Sections III and IV, respectively. For evaluation, we conduct two parts of the experiments. We list in Table I the parameters and their values, which are empirically optimized and fixed for the two stages in the following experiments.

In the first part, we use synthetic data to evaluate the second stage (motif clustering with DPP-based subset selection), based on the following two considerations. First, note that given a synthetic $\mathcal{Y} = \{A_1, \cdots, A_N\}$, the second stage can be evaluated in isolation using the same metric (recall and precision) as described below. Second, motif occurrences in real-world audio are rather sparse, and we can generate synthetic data with more dense occurrences of more motifs. Therefore, the synthetic data can be used for a more intense test of the second stage.

In the second part, we evaluate the whole motif discovery system on 9-hour real-world audio, which is continuously recorded from a public broadcast radio in Beijing and manually annotated with motif labels. A motif instance is a labeled interval that repeats in the audio. A motif has at least two motif instances. There are a total of 398 motif instances from 139 motifs. In manual annotation, repetitive segments that are meaningful (e.g., commercials, signature tunes and even entire programs and songs) and of longer durations are identified as motifs with priority.

The evaluation metrics used in both parts of the experiments are the recall and precision. The recall is the percentage of the labeled motif instances that are correctly discovered. The precision is the percentage of hypothesized intervals that correctly correspond to the labeled motif instances. Specifically, the motif discovery result is denoted using $\{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_D\}$ as defined in Eq. (19). The ground truth is denoted using

$$\{\mathcal{G}_1, \mathcal{G}_2, \ldots, \mathcal{G}_G\}, \tag{20}$$

where each labeled motif $\mathcal{G}_g$, $1 \leq g \leq G$, is a set of non-overlapping intervals, and each element of $\mathcal{G}_g$ is a labeled motif instance. Then, the recall and precision are defined as follows:

$$R = \frac{\sum_{g=1}^{G} \sum_{u \in \mathcal{G}_g} I\left(\max_{d=1,\ldots,D, v \in \mathcal{D}_d} \left(\frac{|u \cap v|}{min(|u|,|v|)} \right) > \delta_{eval} \right)}{\sum_{g=1}^{G} \sum_{u \in \mathcal{G}_g} 1} \tag{21}$$

$$P = \frac{\sum_{d=1}^{D} \sum_{v \in \mathcal{D}_d} I\left(\max_{g=1,\ldots,G, u \in \mathcal{G}_g} \left(\frac{|u \cap v|}{min(|u|,|v|)} \right) > \delta_{eval} \right)}{\sum_{d=1}^{D} \sum_{v \in \mathcal{D}_d} 1} \tag{22}$$

where $|.|$ denotes the length of an interval and $|u \cap v|$ is the overlap time between the two intervals u and v. $I(.)$ is the indicator function, which is one when its argument is satisfied and is zero otherwise. δ_{eval} is the threshold parameter for correctness judgment and is set to 50% in our experiments. The above evaluation metrics are similar to the Occurrence Recall/Precision metrics defined in the recent MIREX campaign [3], which also uses intersection and thresholding.

A. Evaluation of Motif Clustering with DPP-based Subset Selection

1) Synthetic Data Generation: Similar to [26], we use Markov chains to generate synthetic label sequences. The Markov chain state space consists of the 193 labeled names from the 9-hour real-world audio[4]. Both the initial state probability distribution and the state transition matrix (193×193) are uniform. The Markov chain is simulated to randomly generate a number of label sequences.

For each synthetic label sequence, we treat the generated labels as pseudo labeled intervals. A motif instance is a labeled interval that repeats (i.e., has the same label as that of other labeled intervals). From these motif instances, we construct a synthetic set of matched pairs $\mathcal{Y} = \{A_1, \cdots, A_N\}$ for each synthetic label sequence.

2) Tests over Synthetic Data: For each synthetic set $\mathcal{Y}$, we run the second stage to obtain the discovered motifs as defined in Eq. (19) and calculate the recall and precision metrics. The statistics of the synthetic data and the results from the second stage are shown in Table II. Specifically, we randomly generate four synthetic data sets (i.e., four synthetic label sequences). N_{len} denotes the length of a synthetic label sequence (i.e., total number of labels), among which there are $N_{instance}$ motif instances from N_{motif} motifs. N_{motif} is denoted by G, as defined in Eq. (20) . $Duration$ is the total time duration of the labeled intervals. N denotes the size of the synthetic set $\mathcal{Y}$ constructed from the synthetic data. M is the selected matched pairs, as defined in Eq. (15). D is the number of discovered motifs, as defined in Eq. (19).

Based on Table II, we make several observations. First, we note that if a motif has n motif instances, then the minimum

[3] http://www.music-ir.org/mirex/wiki/2015:Discovery_of_Repeated_Themes_%26_Sections

[4] In addition to the 139 labeled names, each of which occurs at least two times in the 9-hour audio and hence is called a motif, there are 54 other named labels in our annotation, each of which appears only once in the audio. Therefore, the total number of labeled names is 193.

TABLE I
SYSTEM PARAMETERS AND THE VALUES USED IN OUR EXPERIMENTS.

For hashing-based similarity matching	
Minimum histogram bin height δ_{height}	4
Minimum temporal difference in the repetition δ_{bin}	1s
Minimum distance for two adjacent points to be clustered δ_{gap}	5s
Minimum interval width δ_{width}	1s
Minimum number of points in a cluster δ_{num}	4
For DPP-based motif clustering	
The variance parameter used in defining the DPP similarity matrix σ_{loc}	3s
The variance parameter used for graph clustering σ_{dur}	3s
Edge weight threshold in graph clustering δ_{edge}	0.75

TABLE II
THE STATISTICS OF THE SYNTHETIC DATA AND THE RESULTS OF THE SECOND STAGE. SEE SECTION V-A FOR DETAILS.

Data	N_{len}	$N_{instance}$	N_{motif}, i.e. G	$Duration$	N, i.e. $\|\mathcal{Y}\|$	M, i.e. $\|Y\|$	D	P	R
1	31	20	7	$285s$	19	13	7	100%	100%
2	501	475	154	$5734s$	600	320	154	100%	99.8%
3	601	572	155	$6625s$	957	417	155	100%	100%
4	1001	993	184	$10919s$	2628	804	185	100%	99.5%

number of matched pairs of intervals that can fully represent the motif is $n-1$. Thus, $N_{instance}-N_{motif}$ gives the minimum size of the subset from $\mathcal{Y}$ that can be used to discover all motifs. It can be seen from Table II that the size of the DPP-based selected subset Y, which is automatically determined through the DPP-MODE algorithm, is very close to $N_{instance}-N_{motif}$. With DPP-based subset selection as a filtering step, we can reduce the original set to a compact subset as much as possible while not losing any necessary information. Second, the final number of discovered motifs, D, nearly equals to the ground truth number of motifs G. The recall and precision are nearly perfect. This demonstrates the effectiveness of the second stage of our motif discovery method. Finally, noting that the synthetic set $\mathcal{Y}$ constructed above is redundant but noise-free, we will evaluate the whole motif discovery system on real-world audios.

B. Evaluation of the Whole Motif Discovery System

1) Reference methods: For comparison, we test several alternative methods for motif clustering to determine the effect of using DPP-based subset selection as a filtering step in our scheme.

First, we can apply the graph clustering technique described in Section IV-C directly to the pairwise matched intervals $\mathcal{Y}$, without DPP-based subset selection, to obtain the discovered motifs as defined in Eq. (19). The difference is that the graph clustering operates on Y in our method (denoted as "DPP+GC") but directly on $\mathcal{Y}$ in this reference method (denoted as "direct GC"). This compares the effects of using or not using DPP-based filtering.

Second, note that the positive semidefinite matrix $\mathbf{L}$ as defined by Eq. (9)(11)(14) can be used not only as the DPP kernel in our method ("DPP+GC") but also as the similarity matrix for other kinds of clustering methods, for example, the kernel k-means [27] and the agglomerative clustering methods [28]. This enable us to compare the effects of using the same kernel information but with different manners in motif clustering.

By using $\mathbf{L}$ as the similarity matrix, two matched pairs, A_i and A_j, are similar if their first intervals differ only slightly from each other. Thus, the kernel k-means or agglomerative clustering of the items in $\mathcal{Y}$ with respect to $\mathbf{L}$ could be viewed as obtaining "equivalent classes of matched pairs". Matched pairs in an equivalent class have slightly different first intervals and thus are aligned horizontally in the scatterplot. In contrast, the DPP-based subset selection with respect to $\mathbf{L}$ could be viewed as obtaining representative matched pairs. A representative matched pair is good in quality (longer duration) and represents other matched pairs that are aligned horizontally with the representative matched pair in the scatterplot.

Third, as indicated above, after we apply the kernel k-means and agglomerative clustering to $\mathcal{Y}$, we obtain "equivalent classes of matched pairs". Then, the same graph clustering technique described in Section IV-C can be adapted for the "equivalent classes of matched pairs" to obtain the discovered motifs as defined in Eq. (19). The modification is that the similarity value between two "equivalent classes of matched pairs" is calculated as the maximum of the pairwise similarity values between the matched pairs from the two equivalent classes.

In summary, in addition to the reference method "direct GC", there are three more reference methods, all based on $\mathbf{L}$. The reference method of applying kernel k-means followed by graph clustering is denoted by "kmeans+GC"[5]. Note that the agglomerative clustering could be maximum linkage or minimum linkage[6]. Accordingly, the reference methods for applying agglomerative clustering with different linkage criteria followed by graph clustering are denoted using "maxAgglo+GC" and "minAgglo+GC", respectively.

[5] For running kernel k-means, the initial centroids are randomly selected from $\mathcal{Y}$, and the number of the centroids is set to be equal to the size of the DPP-based selected subset Y. Kernel k-means [27] then runs exactly the same as regular k-means except that the inner-products are substituted with the positive semi-definite kernel $\mathbf{L}$.

[6] In both cases, the stopping criterion for merging clusters is that the cluster-to-cluster distance is below the threshold 1.0, which is tuned by some pilot experiments.

TABLE III
THE RESULTS OF VARIOUS METHODS ON THE 9-HOUR REAL-WORLD AUDIO. M REPRESENTS THE SIZE OF THE ORIGINAL SET OF MATCHED PAIRS FOR "DIRECT GC" (I.E., $|\mathcal{Y}|$), THE SIZE OF THE SELECTED SUBSET FOR "DPP+GC" (I.E., $|Y|$), OR THE SIZE OF THE SET OF "EQUIVALENT CLASSES OF MATCHED PAIRS" RESULTING FROM KERNEL K-MEANS OR AGGLOMERATIVE CLUSTERING. SEE SECTION V-B FOR DETAILS.

Method	$N_{instance}$	N_{motif}, i.e. G	N, i.e. $\|\mathcal{Y}\|$	M	D	P	R	F
DPP+GC	398	139	920	364	153	92.3%	98.2%	95.21%
direct GC				920	158	88.2%	98.7%	93.16%
maxAgglo+GC				314	117	88.3%	98.7%	93.16%
minAgglo+GC				228	92	88.2%	98.7%	93.16%
kmeans+GC				364	95	88.3%	98.7%	93.16%

2) Tests on Real-world Audio: The results of various methods on the 9-hour real-world audio are shown in Table III. Comparing "DPP+GC" with "direct GC", we can see that adding a filtering step using DPP-based subset selection is crucial, significantly improving the precision performance of motif discovery, without sacrificing the recall performance.

For comparing different ways of using the same kernel information, the "DPP+GC" method (i.e., the clustering with DPP approach) performs better than "kmeans+GC", "maxAgglo+GC" and "minAgglo+GC" (i.e., the direct clustering without DPP approach). Remarkably, the number of discovered motifs D by using the "DPP+GC" method is the closest to the number of the labeled motifs G. In reducing the noise in $\mathcal{Y}$, the clustering with the DPP approach has a clear advantage over the direct clustering without DPP approach. In fact, the direct clustering approach is not helpful for reducing the false alarm noise in $\mathcal{Y}$, as the false alarm pairwise matches are still retained in the "equivalent classes of matched pairs".

From Table III, careful readers may find that (i)[7] the recall metrics are close in value for different methods, and (ii) the "DPP+GC" method is mainly superior in terms of precision. For (i), this presumably occurs because true motifs are highly likely to be covered in the pairwise match results generated from similarity matching. Therefore, the performances of different methods in recall do not differ much. The challenge is to discard false alarm motifs, which also explains (ii). Direct clustering, no matter what clustering method is used (agglomerative or kernel kmeans clustering), cannot filter the false alarm pairwise matches. The main benefit of applying subset selection before clustering is to reduce the false alarm pairwise matches, thus improving the precision.

For more tests of different methods, we use the bootstrap technique to create more data sets from the 9-hour real-world audio. The idea behind bootstrapping is to create replications of a statistic (e.g., the recall and precision) by random sampling from the data set with replacement. It has been used in [29] for significance analysis in evaluating speech recognition performance. In this way, a number of bootstrap data sets are created. Specifically, from the 9-hour real-world audio, we create 200 different audios by randomly remove about 100 labels. Each resulting bootstrap audio is about 7 hours. Then, we repeat running the "DPP+GC", "direct GC", "kmeans+GC", "maxAgglo+GC" and "minAgglo+GC" methods separately on each bootstrap audio and obtain the Monte Carlo mean and standard deviation of the recall and precision for each method. The results are shown in Fig. 9, from which we can observe similar differences between the five methods as in Table III. These results further confirm the performance superiority of our fingerprint motif discovery method ("DPP+GC").

[7]We thank one of the reviewers for pointing this out.

TABLE IV
THE RUNTIME STATISTICS OF THE TWO STAGES FOR PROCESSING THE 9-HOUR REAL-WORLD AUDIO.

First Stage	
Hash extraction	191s
Histogram-based repetition detection	1s
Formation of repeated intervals	2s
Second Stage	
DPP+GC	586s
direct GC	4s
kmeans+GC	570s
maxAgglo+GC	7s
minAgglo+GC	6s

The runtime statistics of the different methods are shown in Table IV. All the methods are run in a single thread on a computer with a 3 GHz core and 48 GB of memory. First, it is remarkable that the first stage (hashing-based similarity matching) is computationally very efficient (above 100 times faster than real time), as it has linear complexity. Second, it can be seen that the total runtime of our system is dominated by the second stage ("DPP+GC") because the greedy DPP-MODE algorithm needs to calculate matrix inversions iteratively. Our current implementation does not consider the sparsity of the kernel $\mathbf{L}$ and uses naive matrix inversions. As future work, one could speed up the DPP-MODE inference by applying some special algorithm, e.g., [30], to take advantage of the sparsity of the kernel $\mathbf{L}$. Third, the total time for processing the 9-hour audio by "DPP+GC" takes around 10 minutes. This is a rough estimate of processing efficiency, as the computational time cost of "DPP+GC" depends on how frequently the motif instances occur in the audio.

For efficiency comparison with other motif discovery methods, referring to evaluations in related studies already provides a clear contrast. Most previous schemes operate around real time, e.g., [1][2]. It is reported in [16] that it takes 6 minutes to process 1-hour audio, and the run-time grows quadratically with the length of the input audio. The efficiency of the scheme in [10] is close to ours, as both use hashing and histogram techniques. The advantage of our scheme compared to [10] lies in repeated interval formation and motif clustering, as we explain in detail in Section III-D and IV-D.

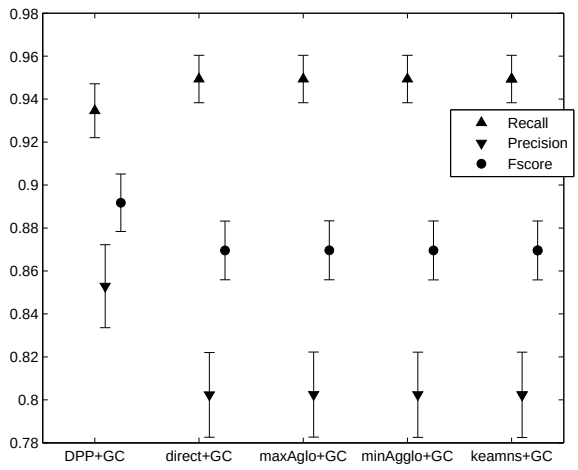


Fig. 9. The Monte Carlo means and standard deviations of the recall, precision and F-score for different methods, computed over the 200 bootstrap audio.

VI. Conclusion and Future Work

In this paper, we investigated scalable discovery of audio fingerprint motifs in long broadcast streams, and we make two contributions. First, as we are confined to handle limited variability in this task, we can adapt the Wang hashing technique [8], originally proposed for searching a given music clip in music tracks, to successfully devise a linear complexity similarity matching method with a new step of repeated interval formation. Second, as the similarity matching is super fast and thus coarse, there are false alarms in the large number of pairwise matches generated, which are a major source of noise. Regarding this, we propose applying a DPP-based subset selection to the original set of pairwise matches to reduce the noise and successfully design the DPP kernel according to the quality-diversity decomposition. The proposed method is thoroughly evaluated on 9-hour real-world audio. It is shown that the similarity matching is computationally very efficient, and the filtering step with DPPs can significantly improve the precision of motif discovery, without sacrificing the recall performance.

We believe that there are some worthwhile future directions. First, we notice the recall and precision performance gap between testing with the synthetic set $\mathcal{Y}$ and testing over real-world audio. This gap informs us that a significant source of errors is due to the inaccuracy of audio hashing. The design of hashing that can be tolerant to more variability is an important future exploration. If we had sufficiently good hashing, we can apply the method presented in this paper to discover a wider range of motifs in audio, e.g., spoken terms and music parts. Second, the potential utility of DPPs has not been fully explored. It is interesting to explore new DPP kernel constructions using new definitions of quality and diversity features. Moreover, it is convenient to combine several kernels to jointly exploit various features, as attempted in [31]. Third, we could speed up the DPP-MODE inference by applying some advanced algorithms, e.g., [30], to take advantage of the sparsity of the kernel. Using GPU to accelerate the computation of matrix inversions should also be of great benefit.

References

[1] C. Herley, "Argos: automatically extracting repeating objects from multimedia streams," *Multimedia, IEEE Transactions on*, vol. 8, no. 1, pp. 115–129, 2006.

[2] Y. Hao, M. Shokoohi-Yekta, G. Papageorgiou, and E. Keogh, "Parameter-free audio motif discovery in large data archives," in *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. IEEE, 2013, pp. 261–270.

[3] J. Paulus, M. Müller, and A. Klapuri, "Audio-based music structure analysis." in *ISMIR*, 2010, pp. 625–636.

[4] J. Serra, M. Müller, P. Grosche, and J. L. Arcos, "Unsupervised detection of music boundaries by time series structure features," in *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[5] A. S. Park and J. R. Glass, "Unsupervised pattern discovery in speech," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 16, no. 1, pp. 186–197, 2008.

[6] P. Cano, E. Batle, T. Kalker, and J. Haitsma, "A review of algorithms for audio fingerprinting," in *Multimedia Signal Processing, 2002 IEEE Workshop on*. IEEE, 2002, pp. 169–173.

[7] J. Haitsma and T. Kalker, "A highly robust audio fingerprinting system." in *ISMIR*, vol. 2002, 2002, pp. 107–115.

[8] A. Wang *et al.*, "An industrial strength audio search algorithm." in *ISMIR*, 2003, pp. 7–13.

[9] H. Lin, Z. Ou, and X. Xiao, "Generalized time-series active search with kullback-leibler distance for audio fingerprinting," *IEEE Signal Processing Letters*, vol. 13, no. 8, pp. 465–468, 2006.

[10] X. Wu and S. Satoh, "Temporal recurrence hashing algorithm for mining commercials from multimedia streams," in *ICASSP*, 2011.

[11] J. Chen, L. Zhu, B. Feng, P. Ding, and B. Xu, "A robust approach to mining repeated sequence in audio stream." in *INTERSPEECH*, 2011, pp. 2277–2280.

[12] A. Kulesza and B. Taskar, "Determinantal point processes for machine learning," *arXiv preprint arXiv:1207.6083*, 2012.

[13] M. Cooper and J. Foote, "Summarizing popular music via structural similarity analysis," in *Applications of Signal Processing to Audio and Acoustics, 2003 IEEE Workshop on*. IEEE, 2003, pp. 127–130.

[14] M. Muller, N. Jiang, and P. Grosche, "A robust fitness measure for capturing repetitions in music recordings with applications to audio thumbnailing," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 21, no. 3, pp. 531–543, 2013.

[15] A. Muscariello, G. Gravier, and F. Bimbot, "Unsupervised motif acquisition in speech via seeded discovery and template matching combination," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 20, no. 7, pp. 2031–2044, 2012.

[16] A. Jansen, K. Church, and H. Hermansky, "Towards spoken term discovery at scale with zero resources." in *INTERSPEECH*, 2010, pp. 1676–1679.

[17] J. P. Ogle and D. P. Ellis, "Fingerprinting to identify repeated sound events in long-duration personal audio recordings," in *ICASSP*, vol. 1, 2007, pp. I–233.

[18] R. B. Dannenberg and N. Hu, "Pattern discovery techniques for music audio," *Journal of New Music Research*, vol. 32, no. 2, pp. 153–163, 2003.

[19] R. Flamary, X. Anguera, and N. Oliver, "Spoken wordcloud: Clustering recurrent patterns in speech," in *Content-Based Multimedia Indexing (CBMI), 2011 9th International Workshop on*. IEEE, 2011, pp. 133–138.

[20] V. Chandrasekhar, M. Sharifi, and D. A. Ross, "Survey and evaluation of audio fingerprinting schemes for mobile query-by-example applications." in *ISMIR*, vol. 20, 2011, pp. 801–806.

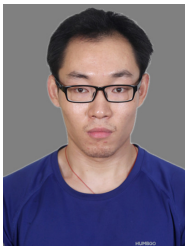
[21] M. Müller, "Content-based audio retrieval," in *Fundamentals of Music Processing*. Springer, 2015, pp. 355–413.

[22] N. Q. Duong and F. Thudor, "Movie synchronization by audio landmark matching," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3632–3636.

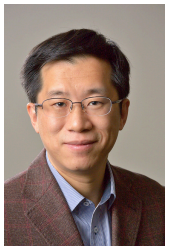
[23] C.-W. Ko, J. Lee, and M. Queyranne, "An exact algorithm for maximum entropy sampling," *Operations Research*, vol. 43, no. 4, pp. 684–691, 1995.

[24] J. Gillenwater, A. Kulesza, and B. Taskar, "Near-optimal map inference for determinantal point processes," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2012, pp. 2735–2743.

[25] C. K. Williams and C. E. Rasmussen, "Gaussian processes for machine learning," *MIT Press*, 2006.
[26] R. Radhakrishnan, Z. Xiong, A. Divakaran, and T. Kan, "Time series analysis and segmentation using eigenvectors for mining semantic audio label sequences," in *Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on*, vol. 1. IEEE, 2004, pp. 611–614.
[27] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means, spectral clustering and normalized cuts," in *Proc. ACM SIGKDD*, 2004, pp. 551–556.
[28] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*. John Wiley & Sons, 2012, chapter 10.9.
[29] M. Bisani and H. Ney, "Bootstrap estimates for confidence intervals in asr performance evaluation," in *ICASSP*, 2004.
[30] J. Zhang and Z. Ou, "Block-wise map inference for determinantal point processes with application to change-point detection," *arXiv preprint arXiv:1503.06239*, 2015.
[31] R. Reichart and A. Korhonen, "Improved lexical acquisition through dpp-based verb clustering." in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2013, pp. 862–872.



Haotian Xu received the B.S. degree in electronic engineering from University of Electronic Science and Technology of China in 2013. He is currently pursuing a master degree at the Department of Electronic Engineering of Tsinghua University under the supervision of Zhijian Ou. His recent research interests focus on unsupervised and semi-supervised learning with graphical models, speech and audio processing.



Zhijian Ou (Senior Member, IEEE) received the B.S. degree with the highest honor in electronic engineering from Shanghai Jiao Tong University in 1998 and the Ph.D. degree in electronic engineering from Tsinghua University in 2003. Since 2003, he has been with the Department of Electronic Engineering of Tsinghua University and is currently an associate professor. From August 2014 to July 2015, he was a visiting scholar at Beckman Institute, University of Illinois at Urbana-Champaign. He has actively led research projects from National Science Foundation China (NSFC), China 863 High-tech Research and Development Program, and China Ministry of Information Industry, as well as joint-research projects with Intel, Panasonic, IBM, and Toshiba. His recent research interests include speech processing (speech recognition and understanding, speaker recognition, natural language processing), and statistical machine intelligence (particularly with graphical models).