Recognition and Summarization of Chord Progressions and Their Application to Music Information Retrieval

Yi Yu, Roger Zimmermann, Ye Wang
School of Computing
National University of Singapore
Singapore
Email: {yuy,rogerz,wangye}@comp.nus.edu.sg

Vincent Oria
Dept. of Computer Science
New Jersey Institute of Technology
Newark, USA
Email: vincent.oria@njit.edu

***Abstract*—Accurate and compact representation of music signals is a key component of large-scale content-based music applications such as music content management and near duplicate audio detection. This problem is not well solved yet despite many research efforts in this field. In this paper, we suggest mid-level summarization of music signals based on chord progressions. More specially, in our proposed algorithm, chord progressions are recognized from music signals based on a supervised learning model, and recognition accuracy is improved by locally probing n-best candidates. By investigating the properties of chord progressions, we further calculate a histogram from the probed chord progressions as a summary of the music signal. We show that the chord progression-based summarization is a powerful feature descriptor for representing harmonic progressions and tonal structures of music signals. The proposed algorithm is evaluated with content-based music retrieval as a typical application. The experimental results on a dataset with more than 70,000 songs confirm that our algorithm can effectively improve summarization accuracy of musical audio contents and retrieval performance, and enhance music retrieval applications on large-scale audio databases.**

***Keywords*-Chord progression-based summarization; locality sensitive hashing; audio representing and computing; music-IR**

I. Introduction

Music contents are now extremely popular on social web sites. Titles and tags, delivering semantic information of music songs, are used in retrieval and classification in community-generated audio repository. However, they are often noisy since users are likely to input incomplete, ambiguous or even irrelevant text information. It is important to provide the social web sites with a more robust retrieval method. Directly searching acoustic music contents, as a significant task, is able to effectively compensate for the vacancy of reliable annotations and help improve the quality of retrieving and mining multimedia music data when manually-labeled annotations are ambiguous or missing.

A crucial issue of content-based retrieval over a large music audio database is how to find accurate and compact representations for music signals so as to efficiently compare them. Some significant research progress has been made during the past few years. Music signals usually are described by sequences of low-level features such as short-time Fourier transform (STFT) [1], pitch [2], Mel-frequency cepstral coefficient (MFCC) [3], and chroma [4], [5]. Unfortunately, among most existing work, music audio content analysis and summarizations, by means of these low-level features, are inefficient and inflexible for a scalable music information retrieval (MIR) task since music knowledge is seldom considered. In comparison, mid-level features (chord [6], rhythm, instrumentation) represented as musical attributes are able to better extract music structures from complex audio signals and retain semantic similarity. Therefore, they are able to effectively and efficiently assist content-based music matching and retrieval. However, chord recognition accuracy is still relatively low in previous state-of-the-art algorithms [7]–[11], which affects the performance of chord-based music retrievals.

In this paper, we first discuss how to generate an accurate summary from a music signal based on chord progressions. To this end, we consider two aspects: improving accuracy of chord progressions and computing a compact summary from chord sequences. We have two significant contributions: (1) Recognition accuracy of chord progressions, under the SVM^{hmm} (support vector machine-hidden markov model) model [12], is greatly improved by exploiting multi-probing. Particularly, by a modified Viterbi algorithm, n-best chord progressions are locally probed and have different likelihoods. (ii) The probed chord progressions are used to create a histogram by the idea of locality sensitive hashing (LSH) [13], where different weights are assigned to chord progressions in terms of their likelihoods. Then, we investigate the recognition accuracy of chord progressions, the influence of multi-probing, and the properties of the generated summaries.

We further discuss how to apply this harmonically-related mid-level feature to the task of content-based music matching and retrieval. Our experiments, implemented on real-world large-scale datasets including more than 70,000 audio tracks, verify that the proposed approach achieves a nice balance between retrieval accuracy and efficiency and demonstrates the feasibility of using hashing-based

chord progression summarization for music content representation and scalable acoustic-based music matching and searching. Compared to previous schemes on music signal summarization, the proposed algorithm effectively improves summarization accuracy and retrieval performance.

The remainder of this paper is organized in the following: Section II reports a review of related work in this field. Section III explains our proposed approach, concentrating on how to extend Viterbi algorithm to probe multiple likely candidates of chord progressions, and how to summarize the probed chord progressions into a compact feature. Section IV gives our method to parameter tuning addressing how to build a training dataset and determine parameters for probing. Section V evaluates our proposed algorithm by the content-based music retrieval application on a large database. Finally, Section VI concludes this work.

II. Related Work

With multimedia music contents growing explosively on user-contributed social sharing websites, scalability of content-based MIR is becoming a challenging issue. Reliable summarization of musical audio signals, as an important component of MIR systems, can facilitate music content comparison and further accelerate the use of music retrieval. The majority of scalable music content-based retrieval algorithms are based on extracting low-level audio features from music signals. Their representations of music signals can be classified into four types according to their different levels of abstraction.

(i) *Plain feature sequences without summarization*. Short-term feature sequences (STFT [1], pitch [2], chroma [4]) have the highest accuracy but with large redundancy. They are computationally inefficient because of high dimensional feature space. (ii) *Conventional global summarization*. Among the methods for audio sequence summarizations, a composite feature tree [14] (semantic features, such as timbre, rhythm, pitch, etc.) has been presented to facilitate kNN search. Statistics of most often-used spectral features (MFCC, chroma, pitch) are concatenated into a compact and semantic feature union [15], but assigned different weights in order to account for their different effects on human perception. These summaries are concise but inaccurate. Because an audio signal is not stationary, global summarization drops too many details and makes audio features less distinguishable. (iii) *Local summarization*. This method makes a tradeoff between accuracy and conciseness. The music signal is divided into multiple segments so that features within each segment are highly correlated and the statistics remain almost unchanged [16]. Then, a summary is computed for each segment. Log frequency cepstral coefficients (LFCCs) and pitch class profiles (PCPs) [17] of adjacent frames are concatenated to form audio shingles. Exact Euclidean locality sensitive hashing functions are used to compress the high-dimensional audio shingles to generate short local summaries. In this way, the music signal is represented by a sequence of local summaries. (iv) *Global summarization retaining harmonic progressions*. A multi-probe histogram [5] is calculated from the chroma feature sequences by heuristically probing the transition between major bins of adjacent chroma features, which is able to retain local spectral and temporal information to some degree. It is more concise compared with local summarization, and is more accurate than conventional global summarization by retaining the temporal information of music signals. But its performance is limited as a heuristic scheme without exploiting music knowledge.

Music signals differ from general audio signals in their harmonic structure, where a fundamental frequency (pitch) is usually accompanied by its harmonics. A chord in music is any combination of two or more notes (pitch) initiated simultaneously[1]. Chord, as a mid-level feature, is a concise representation of music signals. Chord progression represents harmonic content and the semantic structure of a music work, and is an inherent property of a music song. Hence, chord recognition has attracted great interest and many efforts have been devoted to transcribing chords from music signals [6]–[11]. The simplest way to chord recognition is to use the per-feature template matching [6], computing correlation between the chroma feature and a target chord template. This, however, does not always work well since unexpected components sometimes may dominate chroma energy [7]. A more effective policy is to consider chord progression and use sequence detection in chord recognition with the HMM model [9].

There exist a few works on exploiting spectral properties such as pitch histogram, concatenation of statistics of MFCC, chroma, etc., to summarize music signals [14], [15]. Melody information, however, is seldom retained in the generated summary, which therefore has limited capability in distinguishing music songs. Chord sequence can be extracted to represent music signals. Unfortunately, even the performance of state-of-the-art chord recognition algorithms is limited, which further affects the accuracy of chord-based music summarization. This inspires us to perform multi-probing to improve accuracy of chord progression recognition, as will be addressed in our algorithm.

The proposed algorithm belongs to the 4^{th} type of music presentation. Its advantages, in contrast to previous works, lie in the following key concepts: A supervised learning method is proposed to derive and generate probable chord progressions. Multi-probing is performed in the recognition so as to compensate for otherwise inaccurate chord progressions due to the low recognition accuracy. The computed summary is strongly associated with musical knowledge and captures the most-frequent chord progressions, where likelihood information of each probed chord progression is

[1]http://en.wikipedia.org/wiki/Chord (music)

associated with its own ranking.

III. Proposed Approach to Music Summarization

In this section, we present the music summarization algorithm. First, we give an overview of the proposed algorithm in Sec. III-A, briefly introducing the main steps. The model used for recognizing chord progressions from chroma sequences and the multi-probing procedure for improving recognition accuracy are discussed in Sec. III-B. To avoid directly comparing two chord sequences while retaining chord progressions, we further explain how to compute a LSH-based summary in Sec. III-C, focusing on the different effects of probing chords and probing chord progressions.

A. Overview of the Algorithm

Figure 1 shows the flowchart for summarizing music signals. It consists of four main parts: feature extraction, model training, chord progression recognition, and LSH-based summarization. Following these steps, music signals with variable lengths are summarized into fixed-length, compact digests.

The $D_O = 114$ dimensional CompFeat [9], computed from beat-synchronous chroma, is adopted as the feature. The sequence of CompFeat is to be transcribed to a chord sequence. Distinguishing all possible chords is quite complicated. For many applications, e.g., content-based similarity retrieval, it is enough to use a subset of chords as the vocabulary. Similar to previous works, we mainly consider the most frequent chords: 12 major triads $(C, C\#, D, \cdots, A\#, B)$ and 12 minor triads $(c, c\#, d, \cdots, a\#, b)$. All other types of chords are regarded as one type (O). Altogether there are $M = 25$ possible chords, where $O, C, C\#, \cdots, a\#, b$ are mapped to the numbers $1, 2, \cdots, M$ respectively, so as to uniquely identify each chord. As for the training part, we use the SVM^{hmm} model [12], which considers both the spectral structure in each feature and chord progressions embedded in adjacent features. Then, chord progressions are summarized into a compact histogram.

B. n-best Chord Progression Recognition

Each CompFeat corresponds to a chord. In addition, the composition rule of a song also places some constraints on adjacent chords, which determines chord progression and is reflected in adjacent CompFeats. We adopt the SVM^{hmm} model [12], SVM for per CompFeat chord recognition, and HMM for chord progression recognition.

The SVM^{hmm} model is described by Eq. (1) and explained as follows: $\boldsymbol{w}_C$ is a $M \times D_O$ matrix used to convert a $D_O \times 1$ CompFeat to a $M \times 1$ vector of chord scores which correspond to the likelihood of chords computed from the CompFeat (the effect of SVM). $\boldsymbol{w}_T$ is a $M \times M$ matrix describing the score of transiting from one chord to another between adjacent features (the effect of HMM). $\varphi_C(y_t)$ is a $1 \times M$ indicator vector that exactly has only one entry set to 1 corresponding to a chord y_t. $\varphi_T(y_{t-1}, y_t)$ is a $M \times M$ indicator matrix that only has one entry set to 1 corresponding to chord progression from y_{t-1} to y_t. With a CompFeat sequence $\{\boldsymbol{x}_t\}$ and a chord sequence $\{y_t\}, t = 1, 2, \cdots, l$, $\varphi_C(y_t) \cdot \boldsymbol{w}_C \cdot \boldsymbol{x}_t$ is the score (likelihood) that $\boldsymbol{x}_t$ is matched to chord y_t. $\varphi_T(y_{t-1}, y_t) \cdot \boldsymbol{w}_T$ is the score that the local chord sequence progresses from y_{t-1} to y_t. Consequently, the sum in Eq. (1) represents the total score that the CompFeat sequence $\{\boldsymbol{x}_t\}$ is matched to the chord sequence $\boldsymbol{y} = \{y_t\}$. In the end, the chord sequence with the maximal total score is found.

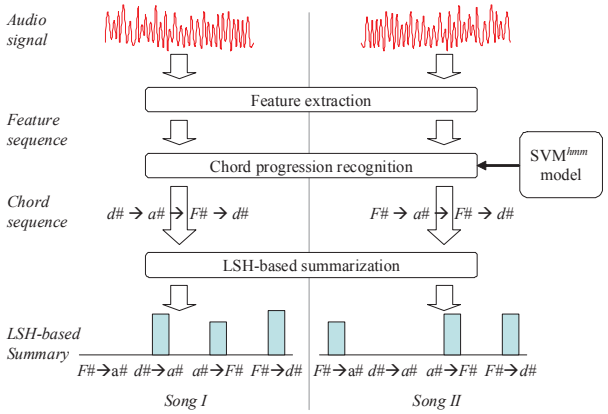

Figure 1. Comparing audio signals by LSH-based summarization.

$$\boldsymbol{y} = \arg\max_{\boldsymbol{y}} \left[\sum_{t=1,\cdots,l} \varphi_C(y_t) \cdot \boldsymbol{w}_C \cdot \boldsymbol{x}_t + \varphi_T(y_{t-1}, y_t) \cdot \boldsymbol{w}_T \right]. \quad (1)$$

Parameters $\boldsymbol{w}_C$ and $\boldsymbol{w}_T$ of the SVM^{hmm} model can be obtained by training, using the public dataset "Beatles" which has been manually annotated by Harte [18].

1) Chord Recognition with Multi-Probing: Chord recognition by Eq. (1) only returns the most likely chord sequence. However, even with state-of-the-art algorithms, the chord recognition accuracy is still relatively low, with a lower recognition accuracy of chord progressions. When the recognized chord sequence is used for retrieval, we argue that besides the most likely chord sequence, other chord progressions should also be probed as well, in order to improve the reliability. Although new features may be suggested for improving performance of chord recognition, the multi-probing method proposed here will still work well.

Chord recognition is to find a chord path across all features. Usually the optimal path is found by the well-known Viterbi algorithm [12]. We modified the Viterbi algorithm shown in Algorithm 1 to realize local multi-probing, not only probing chords but also probing chord progressions. Actually the latter is more important in this paper.

This modified Viterbi algorithm takes the CompFeat sequence $\{\boldsymbol{x}_t\}$ as input, and outputs chord progression set $\{\boldsymbol{z}_t\}$. The procedure is divided into two parts. The first part

Algorithm 1 Chord progression recognition

procedure CHORDPROCRECOG($\boldsymbol{x}_t, t = 1, 2, \cdots, l$)
  $\boldsymbol{r}_1 \leftarrow \boldsymbol{w}_C \cdot \boldsymbol{x}_1$ ▷ Initialization at $t = 1$
  $\boldsymbol{s}_1 \leftarrow \boldsymbol{r}_1$
  for $t = 2, 3, \cdots, l$ **do** ▷ Forward iteration
    $r_{t,j} \leftarrow \boldsymbol{w}_{C,j} \cdot \boldsymbol{x}_t, j = 1, \cdots M$
    $\boldsymbol{p}_{t,j} \leftarrow \boldsymbol{s}_{t-1} + \boldsymbol{w}_{T,j} + r_{t,j}, j = 1, \cdots M$
    $\boldsymbol{s}_t \leftarrow [\max(\boldsymbol{p}_{t,1}), \max(\boldsymbol{p}_{t,2}), \cdots, \max(\boldsymbol{p}_{t,M})]^T$
  end for
  $\boldsymbol{y}_l \leftarrow N_C$ top chords of $\boldsymbol{s}_l$ ▷ Initialization at $t = l$
  for $t = l-1, l-2, \cdots, 1$ **do** ▷ Reverse iteration
    $\boldsymbol{S}_t \leftarrow \sum_{j \in \boldsymbol{y}_{t+1}} \boldsymbol{p}_{t+1,j}$
    $\boldsymbol{y}_t \leftarrow N_C$ top chords of $\boldsymbol{S}_t$
    $\boldsymbol{P}_t \leftarrow \{(i, j, \boldsymbol{p}_{t+1,j,i}) \,|i \in [1, \cdots, M], j \in \boldsymbol{y}_{t+1}\}$
    $\boldsymbol{z}_t \leftarrow \{(i, j, rank_{i,j}) \,|\text{top } N_P \text{ of } \boldsymbol{P}_t\}$
  end for
  return $\boldsymbol{z}_t$, $t = 1, 2, \cdots, l-1$
end procedure

is a forward process, where scores of all paths are computed. $\boldsymbol{r}_t = \boldsymbol{w}_C \cdot \boldsymbol{x}_t$ is a $M \times 1$ vector which contains scores of all chords when matched against $\boldsymbol{x}_t$. $\boldsymbol{s}_t$ is a $M \times 1$ vector, each of which corresponds to the optimal path from the beginning to a chord at t. At $t = 1$, $\boldsymbol{s}_1$ equals $\boldsymbol{r}_1$. When $t = 2, 3, \cdots, l$, scores of the paths from the beginning to chord j at t are composed of three parts: (1) $\boldsymbol{s}_{t-1}$, scores of the M optimal paths to all chords at $t-1$, (2) $\boldsymbol{w}_{T,j}$, scores of transiting from all chords at $t-1$ to chord j at t, and, (3) $r_{t,j}$, the score of chord j when matched against $\boldsymbol{x}_t$. Scores of these M paths leading to the same chord j at t are recorded in $\boldsymbol{p}_{t,j}$ and scores of the M optimal paths to M chords at t are stored in $\boldsymbol{s}_t$.

The second part is the reverse process, where potential chords and chord progressions are probed. At $t = l$, the N_C top chords of $\boldsymbol{s}_l$ are regarded as potential chords corresponding to the last CompFeat. When $t = l-1, l-2, \cdots, 1$, there is a path from each chord at t to each of the N_C chords in $\boldsymbol{y}_{t+1}$ at $t+1$. Scores of these N_C paths sharing the same chord at t are added together and saved in $\boldsymbol{S}_t$, from which the top N_C chords are found as $\boldsymbol{y}_t$. The $M \times N_C$ chord progressions from M chords at t to N_C chords in $\boldsymbol{y}_{t+1}$ at $t+1$ form a set $\boldsymbol{P}_t$, from which the top N_P are probed. These chord progressions, together with their ranks, are saved in $\boldsymbol{z}_t$.

C. Chord Progression-Based Summarization

The chord sequence recognized from the CompFeat sequence is a mid-level representation of an audio signal. Directly comparing two chord sequences is faster than comparing two chroma sequences. But it still requires the time-consuming dynamic programming (DP), in order to account for potential mis-alignment. To expedite the retrieval process, the chord sequences are further summarized into

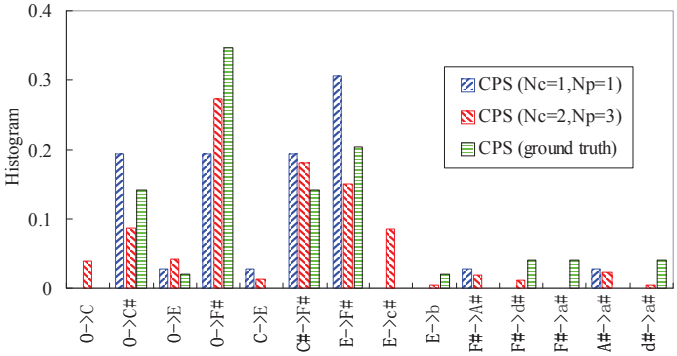

Figure 2. Chord progressions-based summarization ("A Hard Day's Night" of the album "A Hard Day's Night" by "The Beatles").

a compact feature—chord progression based summarization (CPS), computed from $\{\boldsymbol{z}_t\}$ as follows:

$$\begin{aligned} &\text{for } t = 1, 2, \cdots, l-1, k = 1, 2, \cdots, N_P, \\ &\quad \text{get } z_{t,k} = (i, j, rank_{i,j}) \text{ from } \boldsymbol{z}_t, \\ &\quad h = (i-1) \times M + j, w = N_P - rank_{i,j} + 1, \\ &\quad CPS(h) = CPS(h) + w. \end{aligned} \tag{2}$$

Each probed chord progression $\boldsymbol{z}_{t,k} = (i, j, rank_{i,j})$ is a triple. The chord progression $i \rightarrow j$ is hashed to a histogram bin h by the hash function $h = (i-1) \times M + j$, and the weight w is computed from the rank $rank_{i,j}$, a larger weight for a higher rank. Accordingly, the dimension of a CPS equals M^2.

Figure 2 shows an example of CPS. Top 4 chord progressions ($O \rightarrow F\#$, $E \rightarrow F\#$, $O \rightarrow C\#$, $C\# \rightarrow F\#$) can be detected without probing ($N_C = 1$, $N_P = 1$). Detecting less dominant chord progressions such as $F\# \rightarrow d\#$, $d\# \rightarrow a\#$ requires probing ($N_C = 2$, $N_P = 3$).

There are three dominant chord progressions (CPs) in Fig. 2. By further analysis, we find that each song usually has several dominant CPs. We computed the ratio of dominant CPs to all CPs for each song, and the cumulative distribution function (CDF) of this ratio for all 180 songs in the Beatles dataset. The CDF computed according to the ground truth of chord annotations is shown in Fig. 3. It is clear that CPs focus on several values. For example, in about 60% (CDF=0.4) percent of songs, the ratio of 5 dominant CPs is no less than 0.8. Similar results, based on recognized CPs, are shown in Fig. 4. Due to the effect of multi-probing, a single CP may be probed as multiple ones, and CPs are more distributed. At the same CDF=0.4, the ratio of 5 dominant CPs decreases to 0.6 in Fig. 4. But the trend that a few CPs are dominant remains the same as in Fig. 3.

IV. PARAMETER TUNING

In this section, we first investigate how to select the training set for chord recognition. Then, with a small database,

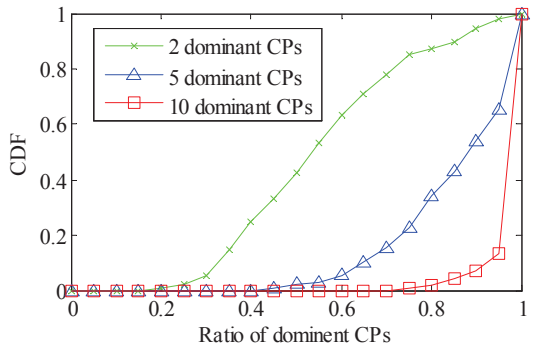

Figure 3. Distribution of ratio of dominant chord progressions in the 180 songs of the Beatles dataset, based on the ground-truth of chord annotations.

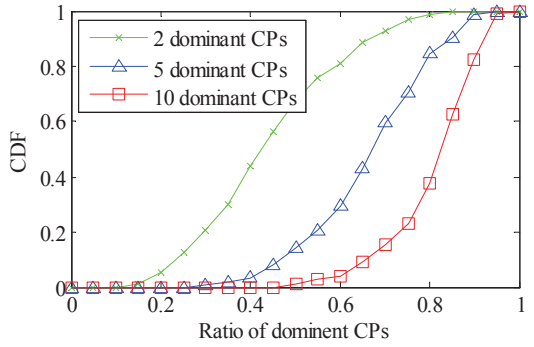

Figure 4. Distribution of ratio of dominant chord progressions in the 180 songs of the Beatles dataset, based on recognized chord progressions.

we examine the effect of probing on both chord recognition and retrieval, and determine the optimal parameters.

A. Selecting a Training Set

State-of-the-art chord recognition algorithms, evaluated in MIREX[2], all are trained and tested on the Beatles sets [18]. About 3/4 of the 180 songs are used for training and the other 1/4 is for testing. With such a policy, the trained model may be over fitted to the training set and does not generalize well to other databases.

Different from a Gaussian model which heavily depends on the size of the training set, the SVM model is decided by the number of support vectors. The training set would work well if all typical support vectors are included. Instead of chords, we are more interested in chord progressions. We use the MRR1 metric to measure the performance of chord progression recognition. MRR1 is defined as the mean reciprocal rank of the correct CP in the probed CP list, which identifies both the recognition accuracy and the quality of chord progression in times of probing. To avoid over-fitting and remove features specific to training songs, we select a small training set from Beatles and use others as the testing set. We wish to find a training set that contains most typical

[2]http://www.music-ir.org/mirex/wiki/2011:Audio Chord Estimation

Algorithm 2 Find the optimal training set

procedure FINDTRAINSET($\boldsymbol{G}$) ▷ Annotated songs
  Equally divide $\boldsymbol{G}$ into N_1 groups $\boldsymbol{G}'_i$, $i = 1, 2, \cdots, N_1$, each with N_2 songs
  for $i = 1, 2, \cdots, N_1$ **do**
    Use $\boldsymbol{G}'_i$ as the training set and train a model
    Test the model with $\boldsymbol{G} - \boldsymbol{G}'_i$, compute $MRR1'_i$
  end for
  Sort $MRR1'_i, i = 1, 2, \cdots, N_1$, in the decreasing order, accordingly $\{\boldsymbol{G}'_i\}$ becomes $\{\boldsymbol{G}_i\}$
  Use last N_3 groups as the common testing set $\boldsymbol{T}_T$.
  $\boldsymbol{T}_R \leftarrow \boldsymbol{G}_1$, train a model with $\boldsymbol{T}_R$
  Test model with $\boldsymbol{T}_T$ and set its $MRR1$ to $MRR1_{best}$
  for $i = 2, \cdots, N_4$ **do**
    Use $(\boldsymbol{T}_R \bigcup \boldsymbol{G}_i)$ to train a model
    Test the model with $\boldsymbol{T}_T$ and compute $MRR1_i$
    if $MRR1_i > MRR1_{best}$ **then**
      $\boldsymbol{T}_R \leftarrow \boldsymbol{T}_R \bigcup \boldsymbol{G}_i$ ▷ Update the training set
      $MRR1_{best} \leftarrow MRR1_i$
    end if
  end for
  return $\boldsymbol{T}_R$ as the selected training set.
end procedure

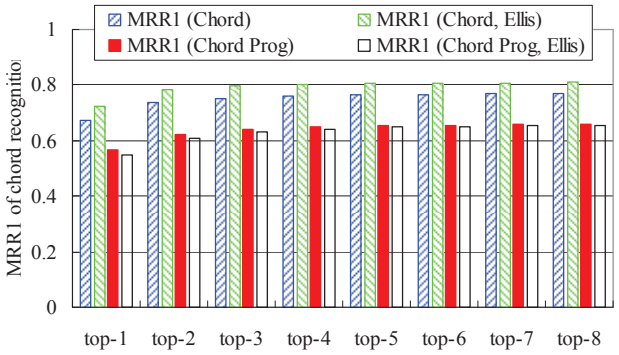

Figure 5. Effect of multi-probing in chord-progression recognition.

support vectors and maximizes MRR1 on the testing set so that the trained model can be well generalized to other datasets.

The algorithm for selecting a training set is shown in Algorithm 2, which takes as input all 180 Beatles songs ($\boldsymbol{G}$) with chord annotations, and outputs a refined training set $\boldsymbol{T}_R$. At first, $\boldsymbol{G}$ is divided into N_1 groups, $\boldsymbol{G}'_i, i = 1, 2, \cdots, N_1$, each with N_2 songs. N_2 should be small enough so that there will be some groups that do not contain support vectors that are only specific to the training set. N_2 should also be large enough so that a SVM^{hmm} model can be trained. Using each group $\boldsymbol{G}'_i$ as the training set and the other songs in $\boldsymbol{G} - \boldsymbol{G}'_i$ as the testing set, $MRR1'_i$ is computed. The obtained $MRR1'_i, i = 1, 2, \cdots, N_1$, is

sorted in decreasing order, and accordingly $\{\boldsymbol{G}_i'\}$ is re-arranged to $\{\boldsymbol{G}_i\}$. Then, starting with $\boldsymbol{T}_R = \boldsymbol{G}_1$ and $MRR1_{best} = MRR1_1$, a new set of songs $(\boldsymbol{T}_R \bigcup \boldsymbol{G}_i)$ is used as a temporary training set. Its $MRR1$ is evaluated on the common testing set $\boldsymbol{T}_T$, and computed as $MRR1_i$. The set $(\boldsymbol{T}_R \bigcup \boldsymbol{G}_i)$ will be used as the new training set if $MRR1_i$ is greater than $MRR1_{best}$. For this process, we used $N_2 = 5, N_1 = 36, N_3 = 26, N_4 = 10$, and the final training set only contains 45 songs, or 1/4 of the total annotated songs.

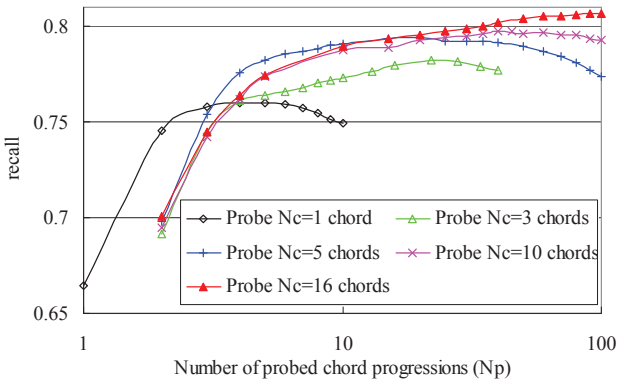

Figure 6. Effect of probing in retrieval.

B. Parameters for Probing

We investigated MRR1 of chord and chord progression over the testing set $\boldsymbol{T}_T$. Besides the proposed algorithm, we also applied the multi-probing procedure to a model pre-trained by Ellis [9] [3]. As shown in Fig. 5, the latter does have a higher MRR1 for chords, but it has a lower MRR1 for chord progressions, especially at the most important 1^{st} rank and 2^{nd} rank. This justifies the necessity of refining the training set.

We tried different probing policies (N_C and N_P) in computing CPSs and tested them on a small database by the kNN (k-nearest neighbor) retrieval. The result of the often-used recall metric is shown in Fig. 6. This figure reveals three points. (i) The effect of probing chord progressions. Under a fixed N_C, recall first increases with N_P and then decreases, which indicates that a suitable N_P can lead to a local maximal recall. (ii) The effect of probing chords. Increasing N_C usually leads to a higher peak recall. (iii) The effect of probing is large when N_C and N_P are small. When there is no probing, N_C = 1 and N_P = 1, recall is only 0.665. Simply probing one more chord progression by using N_P = 2, the recall increases to 0.746. When probing N_C = 16 chords, the max recall reaches 0.806 at N_P = 90. This figure confirms that probing is necessary in order to improve the summarization accuracy to achieve a high recall in the retrieval. Hereafter, N_C = 16 and N_P = 90 are used.

[3]As for the pre-trained model, part of the testing sets $\boldsymbol{T}_T$ might overlap with its training set.

Table I
DATASET DESCRIPTION.

Datasets	Name	# Audio tracks
I	Covers79 (Q993,D79)	1,072
II	Background1	10,041
III	Background2	62,942

V. EVALUATION

Content-based similarity retrieval, as a typical application, is used to evaluate the performance of the proposed chord progression based summarization algorithm over a large music audio database. Since there are no large databases publicly available for simulating scalability of audio content matching and retrieval, we collected audio tracks from MIREX, lastfm.com, and the music channel of Youtube. We use the three datasets shown in Table I, with a total of 74,055 audio tracks. Dataset I, Covers79, is the same as in [5] and consists of 1072 cover versions of 79 songs. Datasets II and III are used as background music.

In the experiments, each track is 30s long in mono-channel mp3 format and the sampling rate is 22.05 KHz. From these mp3 files, the CompFeat features [9] are calculated. Then, chord progressions are recognized through the trained model. To evaluate the performance of multi-probing, the summary without probing is named as CPS and the one with multi-probing and refined parameters is named as CPS+.

We compare the proposed CPS+ scheme to another music summarization method MPH [5] using kNN as the retrieval method. Our retrieval task is to run a batch of multi-version querying against the original one. Multi-version means different cover versions of the same song produced by different people. To this end, the Cover79 dataset is split into two parts: D79 containing original versions of the 79 songs and Q993 containing the rest 1072-79=993 songs in Cover79. Unless stated otherwise, in the evaluation, we use the following default setting: each audio track in the dataset Q993 is used as a query to retrieve its relevant audio track from the datasets D79 plus Dataset II, which have 10,120 audio tracks. The exception is in the second experiment where Dataset III is also used for evaluating the effect of the database size. We use recall, precision and MRR1 as the main metrics to evaluate large-scale retrieval performance.

1) Precision-Recall Curves: A retrieval algorithm should make a good tradeoff between recall and precision. In this subsection we investigate this tradeoff by the classical precision-recall curves.

The number of output is changed and the pairs of recall and precision achieved by all schemes are obtained and plotted in Fig. 7. With more outputs, recall of all schemes increases because the chance that relevant audio tracks

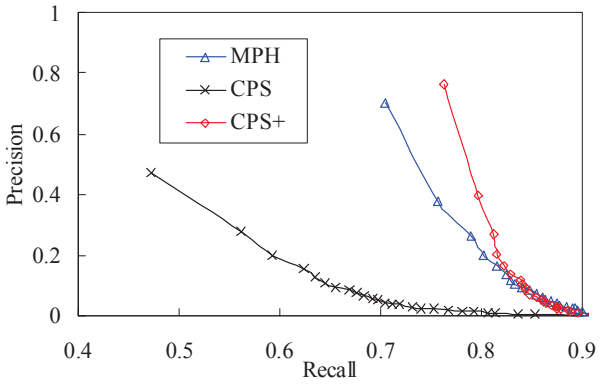

Figure 7. Recall-precision curves of different schemes.

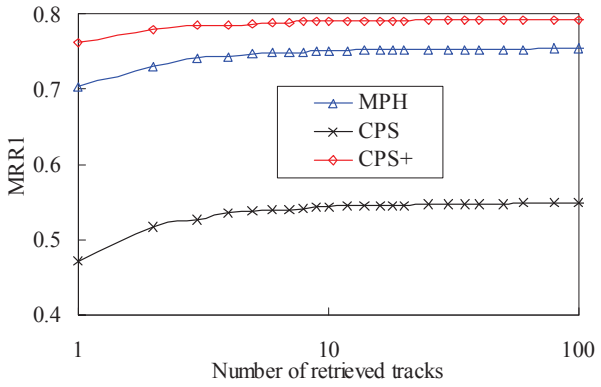

Figure 8. MRR1 under different numbers of output audio tracks.

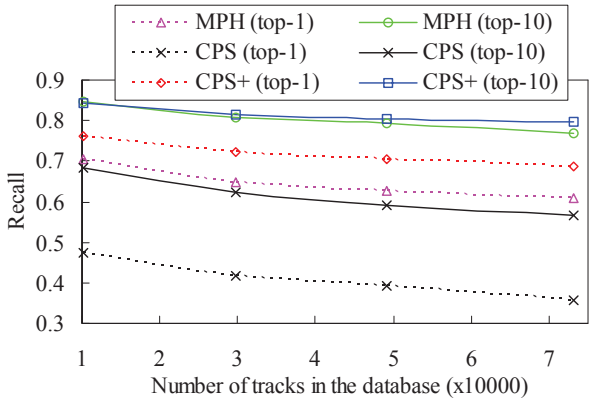

Figure 9. Recall under different database sizes.

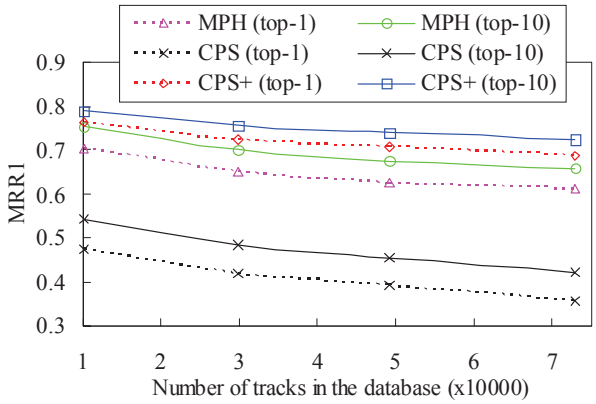

Figure 10. MRR1 under different database sizes.

appear in the ranked list gets larger. The maximal precision is achieved when the number of output is set to 1, the actual number of relevant audio tracks in the database. At this point, precision equals to recall. Increasing the number of outputs leads to a decrease in precision. At the same precision, CPS+ achieves a much higher recall than MPH. But the performance of CPS without exploiting probing in the recognition and summarization is usually very poor.

When the number of relevant audio tracks equals to 1, the tradeoff between precision and recall is better reflected by the MRR1 metric, which reflects both recall and the rank of the retrieved audio tracks. As shown in Fig. 8, MRR1 first increases with the number of output audio tracks and then approaches a constant value. This indicates most relevant tracks that can be found usually appear in the top-10 list and increasing the number of output audio tracks has little effect. This figure again confirms that CPS+ is much superior to MPH.

2) Effect of the Database Size: Content-based MIR usually applies to large online music databases. By varying the database size from 10,120 to 73,062, we evaluate the effect of database size on the performance of CPS+.

Recall curves of three schemes are shown in Fig. 9, where the number of output is set to 1 or 10. Recall decreases in all schemes with the increase of the database size, but it is less obvious in CPS+. One observation is that the recall difference between top-1 results of CPS+ and MPH is very large, but almost disappears in the top-10 results. This can be explained as follows: Some of the relevant audio tracks found by MPH have a low rank, and as a result, the recall of MPH depends more on the number of outputs than that of CPS+.

To better compare these schemes, we computed the MRR1 metric as well. Top-1 and top-10 MRR1 results of three schemes are shown in Fig. 10, all decreasing as the database size increases. Under all cases, the result of CPS without probing is the worst, followed by top-1 result of MPH. Increasing the number of outputs from 1 to 10 effectively increases MRR1 of MPH. But even the top-10 MRR1 result of MPH is still lower than that of top-1 MRR1 of CPS+. This again confirms that CPS+ is more distinguishable than MPH: the relevant audio tracks retrieved by CPS+ have higher ranks than those by MPH.

A simple comparison among MPH, CPS and CPS+ is summarized in Table II, where the database size is $N =$

Table II
COMPARISON AMONG MPH, CPS AND CPS+.

	Comp. cost	recall(1)	recall(10)	MRR1(1)	MRR1(10)
MPH	$144 \cdot N$	0.610	0.767	0.610	0.658
CPS	$625 \cdot N$	0.356	0.565	0.356	0.420
CPS+	$625 \cdot N$	0.687	0.796	0.687	0.724

$73,062$. With the kNN retrieval, the average computation cost is proportional to the database size N and the dimension of the summary feature. CPS+ has higher dimension than MPH and requires more retrieval time. On the other hand, CPS+ outperforms MPH in recall and MRR1. Therefore, CPS+ achieves much higher retrieval performance at an acceptable computation cost. This computation cost of CPS+ can be further reduced by locality sensitive hashing, as will be a part of our future work. This comparison also shows the necessity of multi-probing: CPS without probing has a very poor retrieval performance.

VI. CONCLUSION

In this paper, we proposed a novel mid-level summarization approach for music audio signals based on studying the properties of chord progressions. The proposed algorithm consists of two key points: recognizing chord progressions from a music audio track based on a supervised learning model related to musical knowledge and computing a summary of the audio track from the recognized chord progressions by locality sensitive hashing. In particular, we exploited multi-probing in chord progression recognition via the modified Viterbi algorithm, which outputs multiple likely chord progressions and increases the probability of finding the correct one. A histogram based on chord progressions is put forward to summarize the probed chord progressions in a concise form, which is efficient and retains local chord progressions. The proposed approach improves accuracy of music summarization, which has a wide range of applications such as web audio spam detection and music copyright enforcement. The evaluation results of large-scale content-based similarity retrieval systems confirmed the effectiveness of the proposed approach.

ACKNOWLEDGMENT

This research has been supported by the Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

REFERENCES

[1] C. Yang, "Efficient acoustic index for music retrieval with various degrees of similarity," in *Proc. ACM MM'02*, 2002, pp. 584–591.

[2] W. H. Tsai, H. M. Yu, and H. M. Wang, "A query-by-example technique for retrieving cover versions of popular songs with similar melodies," in *Proc. ISMIR'05*, 2005, pp. 183–190.

[3] T. Pohle, P. Knees, M. Schedl, and G. Widmer, "Automatically adapting the structure of audio similarity spaces," in *Proc. 1st Workshop on Learning the Semantics of Audio Signals (LSAS)*, 2006, pp. 66–75.

[4] D. Ellis and G. Poliner, "Identifying cover songs with chroma features and dynamic programming beat tracking," in *Proc. IEEE ICASSP'07*, vol. 4, 2007, pp. 1429–1432.

[5] Y. Yu, M. Crucianu, V. Oria, and E. Damiani, "Combing multi-probing histogram and order-statistics based LSH for scalable audio content retrieval," in *Proc. ACM MM'10*, 2010, pp. 381–390.

[6] T. Fujishima, "Realtime chord recognition of musical sound: a system using common lisp music," in *Proc. ICMC'99*, 1999, pp. 464–467.

[7] K. Lee, "Automatic chord recognition from audio using enhanced pitch class profile," in *Proc. ICMC'06*, 2006, pp. 306–313.

[8] H.-T. Cheng, Y.-H. Yang, Y.-C. Lin, I.-B. Liao, and H. H. Chen, "Automatic chord recognition for music classification and retrieval," in *Proc. IEEE ICME'08*, 2008, pp. 1505–1508.

[9] D. Ellis and A. Weller, "The 2010 LABROSA chord recognition system," in *Proc. MIREX2010*, 2010.

[10] T. Cho, R. J. Weiss, and J. P. Bello, "Exploring common variations in state of the art chord recognition systems," in *Proc. Sound and Music Computing Conference (SMC)*, 2010, pp. 1–8.

[11] M. McVicar, Y. Ni, T. D. Bie, and R. S. Rodriguez, "Leveraging noisy online databases for use in chord recognition," in *Proc. ISMIR'11*, 2011, pp. 639–644.

[12] T. Joachims, T. Finley, and C.-N. Yu, "Cutting-plane training of structural SVMs," *Machine Learning*, vol. 77, pp. 27–59, Oct. 2009.

[13] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proc. 30th ACM STOC*, 1998, pp. 604–613.

[14] B. Cui, J. Shen, G. Cong, H. Shen, and C. Yu, "Exploring composite acoustic features for efficient music similarity query," in *Proc. ACM MM'06*, 2006, pp. 412–420.

[15] Y. Yu, K. Joe, V. Oria, F. Moerchen, J. S. Downie, and L. Chen, "Multi-version music search using acoustic feature union and exact soft mapping," *International Journal of Semantic Computing*, vol. 3, pp. 209–234, Jun. 2009.

[16] Y. Yu, M. Crucianu, V. Oria, and L. Chen, "Local summarization and multi-level LSH for retrieving multi-variant audio tracks," in *Proc. ACM MM'09*, 2009, pp. 341–350.

[17] M. Casey, C. Rhodes, and M. Slaney, "Analysis of minimum distances in high-dimensional musical spaces," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 16, pp. 1015–1028, Jul. 2008.

[18] C. Harte and M. Sandler, "Automatic chord identification using a quantized chromagram," in *Proc. Proc. 118th Convention. Audio Engineering Society*, 2005.