MOBILE RINGTONE SEARCH THROUGH QUERY BY HUMMING

Lie Lu and Frank Seide

Microsoft Research Asia, Beijing, P.R. China

ABSTRACT

In the context of voice-based mobile search, this paper presents a new approach to mobile ringtone search through query by humming: A user can call a service, hum a part of melody through the mobile phone, and obtain the ringtones or songs he or she is looking for. Correspondingly, we propose a method of query by humming tailored to this scenario. A robust front-end processing is first presented to deal with the mobile phone recording, which is distorted due to GSM codec, environment and wireless transmission. Then, a systematic probabilistic model and matching procedure inspired by Hidden Markov Model (HMM) is presented, by considering the alignment and error tolerance in the matching between query and songs. A rescoring heuristic is finally employed to further improve matching accuracy. Moreover, our system is evaluated on realistic mobile recordings from the field. Experiments show our approach can achieve 83% accuracy on a database with 3000 songs in this realistic scenario.

Index Terms— mobile search, voice search, mobile ringtone search, query by humming

1. INTRODUCTION

Mobile search has attracted intensive research and commercialization in recent years to enable users to search on the move. Current mobile search engines, such as Live Mobile [8], are mostly shrunk versions of their desktop-based counterparts with modified screen layouts, using text-based query input. However, some information needs are not always suitable to be described by keywords, and even where this is possible, owing to the constrained input modalities, it is still inconvenient to use such search services on mobile devices. Instead, search by voice is a natural way for inputting search queries especially in the mobile scenario, since the users are very familiar with the basic function of mobile phones — voice communications.

Meanwhile, mobile ringtone downloads are, surprisingly, a billion-dollar market (4.9B dollars worldwide in 2005). In this paper, to facilitate ringtone search, we present our work on *mobile ringtone search through query by humming*, aiming at searching for a desired ringtone or song by singing, humming, or whistling its melody. Humming search is especially useful when a user does not know song's title or artist information. Moreover, humming and ringtone search is a perfect match in the mobile scenario: Voice is the natural means of input on a mobile phone, and significantly more convenient than text input; ringtones are usually available in MIDI format so that the melody extraction is no problem; and the obtained ringtones are intended for use on the mobile device itself, so that it is a one-step solution.

The core technology in this system, query by humming (QBH), has been a research topic since 1995 [1]. The key problem is to a match melody (pitch sequence) between a human-voice query and a ringtone/song database. This is essentially a problem of approximate *temporal sequence matching* with the special requirement of tolerance to errors, since humans seldom reproduce a tune exactly as the reference (key, speed, rhythmic/melodic deviation). Various approaches have been presented in the literature, regarding front-end processing (pitch extraction and note transcription), modeling and matching (approximate string matching and dynamic programming). For instance, [1] presents an approach by using approximate string matching of symbolic features: note up (U), note down (D), and note repetition (R). To be more robust with respect to insertion/deletion errors and timing deviation, Dynamic Time Warping (DTW) and its probabilistic pendant, Hidden Markov Model (HMM), are used for melody matching, either with continuous pitch as the melody representation [2][6], or with transcribed discrete notes (to speed up matching) [3][4][5]. The HMM, which has been successfully used in speech recognition, also has produced encouraging results in query by humming. However, previous work [4][6] has usually taken HMMs as a black box, without explicitly modeling the alignment between query and songs, nor addressing the error-tolerance in matching. [5] incorporates note deletion and insertion errors in HMM modeling. However, its complex model usually needs several training recordings for each song.

In this paper, we present a novel systematic probabilistic model and matching procedure, inspired by the HMM. It explicitly considers the alignment and error tolerance in the matching model and the related decoder, and can obtain satisfying accuracy with little training/development data. Moreover, compared to the microphone laboratory recordings in previous work, the mobile phone recordings of our scenario usually have low quality and are distorted by the GSM codec and wireless transmission. To address this, we present a robust approach to front-end processing. We also have a full evaluation on realistic query data collected from a realistic system. To our knowledge, this is the first work on QBH dealing with mobile phone field recordings.

The paper is organized as the following. Section 2 presents our humming search algorithms, Section 3 presents the evaluation results, and Section 4 concludes the paper.

2. HUMMING SEARCH

The core technology of query-by-humming mainly consists of three components: melody extraction from songs; melody extraction ("note transcription") from query recordings based on pitch tracking; and a matching model suitable to handle humming errors and variations of timing and key. Since the first component, mel-

1-4244-1484-9/08/$25.00 ©2008 IEEE

ody extraction from MIDI songs, is not a major problem, we will focus on the latter two components.

2.1. Note Transcription from Hummed Queries

In principle, the pitch sequence, which we will denote as o_t, can be directly matched against a song by HMM or DTW. However, this is prohibitively expensive for databases of thousands of songs [2]. The matching can be significantly sped up if we can assume constant pitch throughout a note and use note sequences instead of continuous pitch sequences in the matching procedure. Therefore, we first perform *note transcription* from hummed queries.

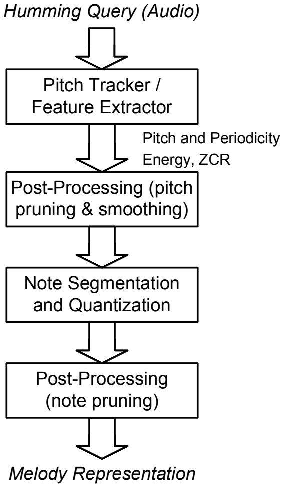
Humming Query (Audio)
Pitch Tracker / Feature Extractor
Pitch and Periodicity
Energy, ZCR
Post-Processing (pitch pruning & smoothing)
Note Segmentation and Quantization
Post-Processing (note pruning)
Melody Representation

Fig.1. Note transcription: transcribing a humming query into the melody representation.

The algorithm is illustrated in Fig 1. A number of acoustic features are first extracted from each audio frame (10 ms), including the pitch as well as three supporting features: energy, zero-crossing rate (ZCR), and periodicity (indicating pitch confidence). Then, the obtained pitch sequence is smoothed with a median filter and post-processed with several heuristics to remove abnormal pitch values or spurious pitched segments with low confidence caused by background noise.

This "cleaned" pitch sequence is now segmented into individual notes, i.e. dividing the pitch sequence into segments where each segment corresponds to one note. Segmentation is a three-step process:

- Energy contour based segmentation. Based on energy, the humming query is roughly divided into segments (energy transition such as energy drop/increase, and pause are strong indicators of note boundaries).
- Pitch change based segmentation. Where no obvious energy transition appears at a note boundary, a boundary is detected if the pitch change exceeds a threshold. Non-pitched segments are also boundaries.
- Finally, a heuristic post-processing step aims at removing spurious or abnormal notes caused by background noise, such as the notes that are too short, have too low confidence, or occur isolated.

With these steps, a note sequence is obtained. We now assume the pitch to be constant throughout one segment (one note), and consecutive segments with the same pitch value are merged, each note n_k is further represented by n_k = ($\Delta note_k$, $duration_k$), where $\Delta note$ stands for the pitch interval between consecutive two notes, in units of semitones ($\Delta note_k = 12 \cdot log_2 (f_k / f_{k-1})$), to handle key variants and conform to music theory; and *duration* represents the actual time a note is hummed or played.

Note that this way, both over-segmentation and under-segmentation errors can and do occur. These need to be compensated for in the following matching model and the related decoder.

2.2. Melody Matching

In this section, we present our probabilistic model and matching procedure, including the matching model, Viterbi alignment, pruning, and rescoring.

2.2.1. Matching Model

Given the note sequence for both query and song, the challenge here is to measure their similarity. Due to the inevitable variation in speed when a user reproduces a melody, the notes between query and song are not time-aligned.

Suppose Q is the observed query and D is the song (D for document). The song that most likely generated the query can be determined as $\hat{D} = \arg\max_D P(Q \mid D)$, where

$$P(Q \mid D) = \sum_A P(Q \mid A, D) P(A \mid D) \approx \max_A P(Q \mid A, D) P(A \mid D) \quad (1)$$

Here, A represents a monotonously increasing *time alignment* that assigns each frame/note in the query to a note in the document. The alignment is not known beforehand, and we therefore consider it a *hidden variable*. *P(Q|A, D)*, the pitch model, measures, given the song D is sung with time alignment A, the probability that the observed query Q is produced. Practically, it provides a measure of the pitch match between the aligned query and song segments. *P(A|D)* is the duration model that measures the probability that the song's actual timing gets distorted to A. In order to calculate *P(Q|D)*, we need to examine and sum up all possible alignments. As common in HMM decoding in speech recognition, we approximate the sum by the maximum, and introduce a flattening weight α over *P(Q|A, D)* to make the contribution of two models comparable (in Eq. (2)).

We represent Q by the observed pitch sequence o_t with $t=1...T$; song document D by a note (*state*) sequence $\{s_j, d_j\}$, $j=1...N$, indicating pitch and duration; alignment A by a set of aligned time points $\{t_n\}$, $n=1...M$, with start note j_0; and M denotes the number of notes actually matched ($M<N$). Here, the beginning of the query can be aligned to any state s_{j0} in the song, and the matched note sequence is commonly a sub-sequence of the entire song. With these notations, the Eq. (1) can be further developed into,

$$\begin{aligned} P(Q \mid D) &\approx \max_A P^{\alpha}(Q \mid A, D) P(A \mid D) \\ &= \max_A \prod_j P^{\alpha}(o_{t_{j-1}+1} ... o_{t_j} \mid s_{j_0+j}, o_{t_{j-2}+1} ... o_{t_{j-1}}, s_{j_0+j-1}) \cdot P(t_j - t_{j-1} \mid d_{j_0+j}) \\ &= \max_{\{t_j\}, j_0} \prod_{j=1}^{M} \left\{ \left(\prod_{t=t_{j-1}+1}^{t_j} C_\sigma \exp(-\alpha \frac{\mid \Delta o_t - \Delta s_{j_0+j} \mid}{\sigma^2}) \right) \cdot C_v \exp(-\frac{\mid (t_j - t_{j-1}) - d_{j_0+j} \mid}{v^2}) \right\} \end{aligned} \quad (2)$$

Δo_t and Δs_j is used to represent the pitch interval ($\Delta s_j = s_j - s_{j-1}$; Δo_t requires further explanation, see below); C_σ, C_v, σ and v are the constants for normalization and deviations of the *Laplacian* densities used for pitch and duration deviation, whose impacts on final ranking is simply integrated into the flattening weight *a,* which is set to 1/9 in our experiments. It is noted that Laplacian is used since it shows to outperform Gaussian densities in terms of accuracy. To find the optimal alignment producing the maximum *P(Q|D)*, the Viterbi algorithm [7] is adopted to search through all possible alignments.

2.2.2. Alignment in Viterbi Decoding

In principle, any arbitrary time point t in the query can be aligned to a note state s_j in the document. To reduce the search space, we chose to impose a major constraint: We constrain the alignment points to the segment boundaries of the query, based on the note transcription step described in the previous section. This has an important implication: Since for the note-transcription process we have assumed pitch to be constant throughout a segment (a note), Δo_t in Eq. (2) is simply $\Delta o_t = o_{t_j} - o_{t_{j-1}}$.

However, the above constraint is too strong in the presence of transcription errors such as over-segmentation and under-segmentation, or user errors such as note insertion and deletion. Therefore, we relax the constraint to allow multiple-on-1 or 1-on-multiple matches, that is, we allow multiple (two or three) query segments to match one note in the document, and vice versa, as illustrated in Fig.2. The merged multiple segments or notes are considered as one note or one state (with timing retained), so that Eq. (2) can be easily applied.

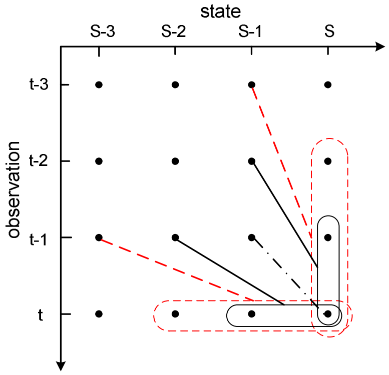

Fig.2. Multiple-on-1 matches in Viterbi alignment, with various paths going to observation-state pair (s, t). Solid line: with a 2-on-1 match; dashed line: with a 3-on-1 match

2.2.3. Pruning

In order to improve the matching speed, *beam pruning*, another common technique in speech recognition, is further employed in the Viterbi decoding: A partial alignment path ending at (o_t, s_j) is pruned (excluded from further examination) if the partial path probability up to this point is significantly less than the best probability at t, assuming all of the future expansions of the alignment path have only negligibly small chance to become the final globally optimal alignment. Paths are pruned if

$$\frac{Q(o_t, s_j)}{\max_i Q(o_t, s_i)} < f_{pr} \qquad (f_{pr} < 1) \tag{3}$$

where $Q(o_t, s_j)$ stands for the partial path probability up to this point, which can also be calculated from Eq.(2), and f_{pr} is the pruning parameter (*beam width*) determined experimentally.

2.2.4. Rescoring

It is noted that one can imagine that the duration model *P(A|D)* could be improved by considering longer-span speed deviations. For example, we could include the duration of previous note and build a "*bigram*" model, as,

$$\begin{aligned} P_{bi}(A \mid D) &= \prod_j P(t_{j+1} - t_j \mid d_{j_0+j}, t_j - t_{j-1}, d_{j_0+j-1}) \\ &= \prod_j \exp(-\frac{d_{j_0+j} \mid (t_{j+1} - t_j)/d_{j_0+j} - (t_j - t_{j-1})/d_{j_0+j-1} \mid}{\nu^2}) \end{aligned} \tag{4}$$

However, experiments have shown that no performance improvement can be achieved. This "bigram" approach is still insufficient to model the longer-span rhythm deviations. Instead, we found that a rescoring heuristic is able to compensate for this effect which utilizes the global correlation of the aligned duration sequence between query and document. A similar global correlation heuristic is also applied to the aligned pitch sequence after global normalization. These heuristics lead to significant accuracy improvements. Besides this global correlation, some other factors are also used for final rescoring or reranking, based on the aligned melody sequence as well,

1. *Prior probability of entry point*: Queries are not hummed with a random entry point in a song, but mostly start from the first note of a song, or the beginning of the refrain and chorus. Due to no label data on refrain and chorus, only the first note is given higher entry prior. It is more probable that a query is hummed from the song that their first notes are matched.
2. *Emphasize local melody extrema*: We find improved matching accuracy from emphasizing local melody extrema. That is, the change points from the melody going up/down to going down/up (local peak or valley in the melody line) are given higher weights in probability measure.
3. *Note repetition information*: From note transcription, we can obtain some partial information on note repetition. Higher weight is given to one segment if the number of note repetition is matched.

3. EVALUATIONS

In the experiments, we built an Interactive Voice Response (IVR) system to collect data in realistic scenarios, where the user were requested to call the service number by a mobile phone, pick up a song from a list, and then hum for 15 seconds, without further instructions on environment, humming style, etc. The goal was to collect data that is consistent with the real scenario as much as possible, including realistic background noise, as well as possible signal distortion through GSM codec and transmission. Overall, we collected around 500 humming queries through this prototype system. The test dataset is composed of 3000 MIDI files.

In addition to top-1 accuracy, we evaluate top-N accuracy, that is, the probability that the correct hit is in the top N results, for the scenario that a user can select a song from a returned list. We also evaluate the Mean Reciprocal Rank (MRR), a common metric used in information retrieval, which indicates the average (reciprocal) rank position of the correct hit:

$$MRR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{n_i} \tag{5}$$

where N is the number of queries and n_i is the rank position of the correct hit of query i.

Table 1. Accuracy comparison between symbolic-based approaches and hmm-inspired approaches, with/without rescoring and various factors (config: 2-on-1 matching, -log f_{pr} = *2500*).

Config:	*Symbolic*		*HMM-inspired*		
Metric	*w/o pitch*	*w/ pitch*	*w/o rescoring*	*base rescoring*	*rescore w/ all factors*
Acc. @Top1:	36.3	46.8	67.8	***78.1***	***82.6***
MRR:	43.5	54.0	72.9	***81.7***	***85.2***
Acc.@Top5:	53.3	60.4	78.5	86.1	88.5
Acc.@Top20:	65.6	68.0	85.5	89.6	90.0

Table 2. The effect of various rescoring factors (config: 2-on-1 matching, -log f_{pr} = *2500*).

Config: Metric	*base rescoring*	*w/ entry prior*	*w/ melody extrema*	*w/ repetition*	*w/ all*
Acc.@Top1:	78.1	80.9	78.9	80.5	82.6
MRR:	81.7	84.2	82.4	83.1	85.2
Acc.@Top5:	86.1	87.9	86.5	86.3	88.2
Acc.@Top20:	89.6	90.2	89.8	89.8	90.0

Table 3. Comparison on 2-on-1 and 3-on-1 matching in hmm-inspired approach with/without rescoring (config: -log f_{pr} = *2500*).

Config:	*hmm*		*hmm w/ base rescoring*	
Metric	*2-on-1*	*3-on-1*	*2-on-1*	*3-on-1*
Acc.@Top1:	67.8	69.9	78.1	80.1
MRR:	72.9	75.4	81.7	84.1
Acc.@Top5:	78.4	82.0	86.1	88.7
Acc.@Top20:	85.5	89.1	89.6	93.8
Time [s/query]	1.6	1.9	1.6	1.9

Table 4. Comparison on pruning parameters vs. matching speed (config: *hmm,* 2-on-1 matching).

-log f_{pr}:	*1500*	*2000*	*2500*	*3000*	*no prune*
Acc.@Top1:	67.0	67.4	67.8	67.8	67.8
MRR:	71.9	72.5	72.9	72.9	72.9
Acc.@Top5:	77.3	78.1	78.5	78.5	78.5
Acc.@Top20:	83.8	85.0	85.5	85.5	85.7
Time[s/query]	1.4	1.5	1.6	1.7	2.7

For comparison, in addition to our HMM-inspired algorithm, we have also implemented two variants of a common symbolic-based matching technique proposed in previous literature [1][3]:

- *Without pitch information***:** symbolic matching based on *UD* (up/down) information and dynamic time warping (DTW);
- *With pitch information***:** consider both *UD* sequence and pitch interval information at DTW matching and similarity score calculation.

For all experiments, if not further specified, the default settings used are 2-on-1 matching and pruning parameter -log f_{pr} = 2500.

Table 1 shows comparative results for the five algorithms: the symbolic approach without or with pitch information, our HMM-inspired matching without rescoring, with *base rescoring* based on the global correlation between aligned duration and pitch sequence, and with rescoring based on all the factors listed in Section 2.2.4. It can be seen that HMM matching significantly improves accuracy over the symbolic methods: 31% and 21% absolute accuracy improvement compared with the symbolic approach without or with pitch information. With the rescoring heuristic, accuracy is further improved about 10% and 15% points, by the rescoring scheme based on the global correlation, and all the rescoring factors, respectively

Table 2 show the individual effect of each rescoring factor added on the base rescoring scheme (including *entry point prior, melody extrema*, and *repetition*). Each factor helps about 1-2% improvement on accuracy, compared with the base rescoring (i.e., global correlation).

In the second experiment, we compare the effect of 2-on-1 matching and 3-on-1 matching, under the configuration without or with rescoring. 3-on-1 matching is more error-tolerant since it can handle the insertion or deletion of two subsequent segments. However, with 3-on-1 matching, the possible alignment paths are significantly increased, leading to increased runtime. Table 3 shows the result. Compared with 2-on-1 matching, 3-on-1 matching has an around 2%-point improvement on accuracy and 2.5% on MRR, whether with or without rescoring. Runtime, however, is increased by nearly 20%.

The third experiment evaluates pruning. We compared several different f_{pr} values, as well as no pruning. Choosing the pruning parameter involves a tradeoff between the accuracy and speed. Table 4 shows results for five pruning configurations. At -log f_{pr} = 2500, the searching process is sped up by 40% at no loss of accuracy. It is still a nice improvement although it is not as significantly effective as the pruning in speech recognition.

4. CONCLUSION

This paper has presented an approach to query by humming for mobile ringtone search. We first presented an approach to robust front-end processing to deal with the mobile phone recording, which is usually distorted by GSM codec and wireless transmission. Then we presented a systematic matching procedure, including the probabilistic modeling, alignment at Viterbi decoding, pruning, and rescoring. Experiments showed our approach can achieve 83% top-1 accuracy and 85% MRR on a database with 3000 songs.

Room to improve the proposed approach is, for example, database indexing. Currently, near linear search is performed through the database, which is unaffordable when the database size increases large. Machine learning approaches can also be used to learn entry point priors of each song, from a set of training data.

REFERENCE

[1] A. Ghias et al, “Query By Humming - Musical Information Retrieval in an Audio Database,” *Proc. ACM Multimedia 95*, 231-236, 1995.

[2] J.S.R. Jang, H.R. Lee. “Hierarchical filtering method for content-based music retrieval via acoustic input,” *Proc. ACM Multimedia 2001*, 401-410, 2001.

[3] L. Lu et al., “A new approach to query by humming in music retrieval,” *ICME 2001*, 776-779, 2001.

[4] J. Shifrin, and W. Birmingham, “Effectiveness of hmm-based retrieval on large databases,” *ISMIR 2003* 33-39.

[5] B. Liu et al. “A Linear Hidden Markov Model for Music Information Retrieval Based on Humming,” *Proc. ICASSP 2003*, Vol. V, 533–536.

[6] J.S.R Jang et al. "Continuous HMM and Its Enhancement for Singing/Humming Query Retrieval," *Proc ISMIR 2005*

[7] L. Rabiner, B.H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1993.

[8] Live Search Mobile, http://mobile.live.com/search