Selection of Melody Lines for Music Databases

Michael TANG 葉志立 YIP Chi Lap Ben KAO
Department of Computer Science and Information Systems, The University of Hong Kong
{fmtang,clyip,kao}@csis.hku.hk

Abstract

One major approach to music retrieval is to model music as a sequence of features, after which traditional information retrieval techniques are applied on the sequence. Because of the temporal nature of music and the inexactness of user queries, most effort on music retrieval systems focus on issues such as indexing and approximation match. In contrast, the processing of music before feature extraction, such as the identification of melody track, were often considered easy or done. This may be the case in a controlled environment, such as one for musicology research, where the pieces are carefully analyzed by human beings before being submitted to the database. However, in an environment where large volumes of music is obtained from the Web, manual music analysis is impractical.

Since many well-known musical features often pertain to the melody of musical pieces, and users often remember the melody of a song, algorithms that select the melody tracks of a piece are important for Web-based content-based retrieval systems. In this paper, we describe a number of algorithms for automatic melody track selection in a music retrieval context. We will also study the performance of the algorithms by comparing their answers to those judged by human beings.

This work is supported by HK RGC under the grant HKU 7035/99E.

1. Introduction

In a content-based music retrieval system, music is processed in a number of stages. First, acquired computer representations of music are preprocessed. Only relevant parts of the representation, such as the melody line, accompaniment track or lyrics are retained for further analysis. Then, music is analyzed so that relevant features are extracted. Those features, such as the ups and downs of a melody line (called melodic contour or profile), are then seen to represent the original piece of music and are indexed in a database. User queries are preprocessed and analyzed in the same manner and the resulting features are matched with those in the database before results are presented to the user. In this approach, music is essentially converted to a sequence of features to which traditional information retrieval techniques can be applied.

Because of the temporal nature of music, musical features are often sequences. Representative ones include the interval sequence, melodic profile, or chord progressions. Because music is multidimensional and the same piece of music can be represented and interpreted in different ways, features from user queries often cannot be exactly the same as those of the matching pieces. Hence, much effort on music retrieval focus on issues such as indexing [3], approximation match [8], and the mapping of music retrieval to traditional retrieval problems [5]. The processing of music before feature extraction, such as time quantization of notes and the identification of melody tracks, were often considered easy or done. This may be the case in a controlled environment, such as one for musicology research, where the pieces have to be carefully analyzed by human beings before being submitted to the database. However, in an environment where large volumes of music is obtained from the Web, manual music analysis is impractical. Worse yet, since there is no standard way to make annotations on musical pieces in the many computer music representations available, the preprocessing of music for the extraction of features becomes a serious practical concern in the construction of music retrieval systems.

Among the various problems in music preprocessing, we consider the selection of melody tracks important for Web-based content-based music retrieval systems for a number of reasons. First, many well-known musical features, such as melodic profile, interval sequence, coarse interval sequence and note duration ratio sequence as proposed in [11], often pertain to the melody of musical pieces. Second, users often remember the melody of a song rather than, say, its chord progressions or harmonic patterns. Third, the retrieval accuracy of music retrieval systems can be increased if the melody lines are identified [1]. Thus, algorithms that select the melody tracks of a piece are important for Web-based content-based retrieval systems. In this paper, we describe

0-7695-0792-1/00 $10.00 © 2000 IEEE

a number of methods for automatic melody track selection in a music retrieval context. We will also study the performance of these methods by comparing their answers to those judged by human beings.

2. Melody line selection algorithms

The study on the recognition of melody and harmonic accompaniment has been an interest for music perception researchers for decades [4][10]. However, few works can be found on the topic of melody line selection from an information retrieval point of view. As discussed, preprocessing is often considered easy or done and researchers often use monophonic tunes for their experiments [7][9], or select features that do not require a melody line to be selected [3]. Literatures on music analysis do cover score following techniques that require the recognition of melody [2], but most of them are not suitable for computer analysis purposes because of their complexity or the lack of rigid rules for analysis.

Fortunately, in the information retrieval context, we can afford to use melody line extraction algorithms that are not always musically correct because multiple "melody lines" can be used for indexing. Indeed, given the number of music representations and the diversity of musical genres on the Web, the efficiency of the preprocessing and analysis modules are often more important than absolute musical correctness. Hence, we look for simple and efficient methods that rank or classify parts of musical representations as melody lines. The "quick and dirty filtering" approach is adopted: to process a large number of pieces, it is good enough to have an efficient and computationally inexpensive algorithm that selects a third of the tracks, half of which contain the melody, because more detailed music analysis and feature extraction processes could follow if required.

Here, we propose a number of methods for the selection of melody lines and compare their relative performances by applying them on a collection of MIDI music files. We assume that a melody line associates with a track in a MIDI file, so our goal is to find out the MIDI track containing the melody line. The methods we used are called AvgVel, PMRatio, SilenceRatio, Range, and TrackName. The first three rank tracks according to their likeliness as melody tracks and the last two classify a given track as melody or non-melody track.

2.1. AvgVel

Since a melody line is often the most prominent part in a piece, it is often the loudest part. Since the velocity of a MIDI message shows the loudness of a note, the average value of MIDI note velocity relates to the average loudness of all the notes in a track. Hence, AvgVel uses the average velocity of all non-zero MIDI NoteOn messages on the track for ranking. The MIDI tracks are ranked in descending order of average velocity, and tracks with zero average velocity are discarded. The rank so obtained is taken as the rank for the melody track.

2.2. PMRatio

As accompaniment tracks often contain chords, they are often polyphonic, that is, more than one note may sound simultaneously. Assuming that melodic lines are mostly monophonic, we define PMRatio as the polyphonic to monophonic time ratio of a MIDI track. For each track in the MIDI file, two numbers representing real time (as opposed to metric time) durations are recorded: one is the total duration when exactly one note is sounding (monophonic), and another is the total duration when more than one note sound together (polyphonic). The polyphonic to monophonic duration ratio is then calculated. Tracks are ranked in ascending order of PMRatio, which is taken as the rank for melody tracks. Tracks that are never monophonic are considered non-melody tracks and are given the lowest ranks.

2.3. SilenceRatio

Assuming that a melody tends to "fill the space" of a piece, we define SilenceRatio of a track as the ratio of silence time of the track to the total time of the piece. For every track, the total duration of silence, and the total duration of the piece, taken as the longest of all the track durations, both in real time, are found and their ratio calculated. The tracks are then ranked in ascending order of SilenceRatio, which is taken as the rank for melody track.

2.4. Range

The range of a melody is the pitch difference between its highest and lowest notes in number of semitones. Since melodies are often desiged to be sung and the human voice range is about two octaves, the range of a melody track cannot be too large. Also, tracks with little pitch variations or those containing too few notes are often non-melodies. Hence, for each track with no less than n distinct notes used, we classify it as a melody track if its range is between a lower bound m and an upper bound M in number of semitones. The n here is used to exclude tracks with too few notes from being a melody, m excludes tracks that are "too plain" and M excludes tracks that are too difficult to be sung.

2.5. TrackName

Each track of a MIDI file can have a name, and some MIDI file producers use the name to annotate the purpose of the track. Thus, case-insensitive substring match was used to match the track names. Tracks with the substring "melody", "vocal", "voice", or "solo" are classified as melody tracks.

3. Comparing the algorithms

A collection of 16012 MIDI files obtained from the Web were used to test the five algorithms. Music in these files include oldies, movie themes, national anthems, music for animation and video games, and pop songs of Hong Kong, Taiwan, Japan, Europe, and US. Tracks in all the pieces were ranked or classified by the five methods mentioned above. For the Range classification method, the note number threshold n is 4, the range lower bound m is 7 semitones (a perfect fifth), and four upper bounds M of 17, 19, 21 or 24 semitones were used in separate sets of experiments. These four upper bounds correspond to musical intervals of a compound perfect fourth (a perfect fourth above an octave), a compound perfect fifth, a compound major sixth, and two octaves respectively.

Since the number of pieces in the collection was large and it was impractical to find out the melody tracks by listening to each and every one of them, sampling was used to estimate the accuracy of the melody track selection methods. A random sample of 400 pieces was taken from the collection, their melody tracks identified by human and were taken as model answers. Tracks ranked or classified by the five methods were then matched with the model answers to find out their accuracy.

Since multiple tracks can be of the same rank in the computed answers and the model answer may consist of more than one track, a top-n comparison method is used. Each algorithm is seen to propose sets of tracks as melody lines. A track was considered to match the top-n sets if it appears in both the model answer and any of the highest ranked n sets of tracks proposed by an algorithm. The number and percentage of tracks in the top-n match were found and reported, as well as the precision and recall values.

For each piece, a "track precision" value is calculated. It is defined as the ratio between the number of tracks the algorithm correctly proposes in any of the first n sets and the total number of tracks the algorithm proposed in the first n sets. An average of track precision over all the sampled pieces is then found. This value reflects the average accuracy precision of the proposed answers. Applications that require very accurate selection of the melody tracks, that is, those that tolerate few false positives, should use an algorithm that gives a high track precision value.

Similarly, for each piece, we can find a "track recall" value, defined as the ratio between the number of tracks the algorithm correctly proposes in any the first n sets, and the number of melody tracks in the model answer. The average track recall over all the pieces thus reflects the average proportion of melody tracks that can be proposed by the algorithms.

Besides track-based precision and recall values, file-based measures of the algorithms' capability and accuracy are also found. The "accuracy" for top-n match is the ratio between the number of pieces where at least one track in any of the top-n sets of proposed tracks are in the model answer, and the number of pieces the algorithm can propose any answer. A ranking method is accurate if it gives a high accuracy value for a small n, because fewer sets of tracks need to be proposed to cover all melody tracks. Similarly, the "capability" value of an algorithm is the ratio between the number of pieces the algorithm can give any answer and the total number of pieces in the database. It is essentially the proportion of pieces the algorithm can handle by giving nonempty answer sets. Note that the capability for ranking algorithms that does not give false drops is always 100 percent because they always give nonempty answer sets. In contrast, classification algorithms that reject all tracks of some pieces as melody tracks will have a capability value of less than 100 percent.

4. Results and observations

Table 1(a) shows the evaluation results of the melody track selection algorithms for the ranking methods, Figure 1 shows the corresponding graphs, and Table 1(b) shows the results for the classification methods. Although sampling was used to estimate the values shown in the table, analysis showed that at 95% confidence level, the interval sizes for the values shown are relatively small. Hence, they are not shown in the tables.

In our experiments, multiple tracks can be of the same rank. Hence, we record the average and standard deviation of the number and percentage of tracks proposed by the algorithm for each n. We found that although the variation on the number of track proposed is relatively large, each algorithm will in general propose 0.8 to 1 more track as the melody track for each increment of n. Ranking algorithms return from 11.3 to 19 percent of the tracks as their best match, and classification algorithms returns 12.9 to 33.7 percent.

Among all the algorithms, TrackName is the most promising one. Proposing only 12.9% of the MIDI tracks on average, it can achieve a track recall of 90.9%. This means that 90.9% of the real melody tracks can be found if the algorithm gives an answer. Also, the accuracy of 92.7% means that 92.7% percent of the time the algorithm pro-

	n	AvgVel		PMRatio		SilenceRatio		FillRatio	
		avg.	sd	avg.	sd	avg.	sd	avg.	sd
Number of tracks proposed	1	1.90	3.35	1.99	3.76	1.06	0.07	1.08	0.09
	2	2.89	2.95	2.98	3.79	2.08	0.22	2.09	0.32
	3	3.77	2.69	3.89	3.97	3.04	0.48	3.06	0.58
	4	4.49	2.66	4.66	4.29	3.90	0.86	3.93	0.98
	5	5.08	3.10	5.29	5.00	4.65	1.58	4.67	1.68
Percentage of tracks proposed	1	19.0	3.7	18.4	2.3	11.3	0.9	11.5	0.9
	2	28.2	3.8	27.9	3.0	21.2	1.7	21.2	1.8
	3	35.7	3.7	35.9	3.2	29.9	2.5	30.0	2.5
	4	41.4	3.4	42.1	3.2	37.0	2.8	37.2	2.8
	5	45.8	3.1	46.9	3.2	42.8	2.8	42.9	2.8
Average track precision	1	36.6		26.0		23.9		24.3	
	2	31.0		25.7		22.9		24.7	
	3	28.5		25.2		23.0		24.4	
	4	26.7		24.9		22.9		24.2	
	5	25.4		24.1		23.4		23.7	
Average track recall	1	47.9		39.7		23.1		23.2	
	2	67.2		58.4		39.2		42.9	
	3	80.5		72.9		55.0		59.3	
	4	88.9		84.3		67.2		72.7	
	5	93.6		89.9		78.8		80.6	
Accuracy	1	49.5		40.0		24.0		24.5	
	2	69.0		59.5		41.0		44.5	
	3	82.2		74.0		57.0		61.2	
	4	90.5		85.2		69.0		74.5	
	5	94.5		90.8		80.0		82.0	
Capability	1-5	100.0		100.0		100.0		100.0	

(a) Ranking methods

	TrackName		Range							
			7–17		7–19		7–21		7–24	
	avg.	sd	avg.	sd	avg.	sd	avg.	sd	avg.	sd
no. of tracks	1.60	0.89	2.73	2.57	3.10	3.36	3.55	4.47	4.21	6.37
% of tracks	12.9	0.6	22.3	1.9	25.1	2.0	28.5	2.5	33.7	2.8
Avg. trk. prec.	72.9		24.2		26.9		28.1		26.0	
Avg. trk. recall	90.9		51.4		60.9		70.5		77.3	
Accuracy	92.7		51.7		61.0		70.9		77.8	
Capability	44.3		79.3		86.0		88.5		92.3	

(b) Classification methods

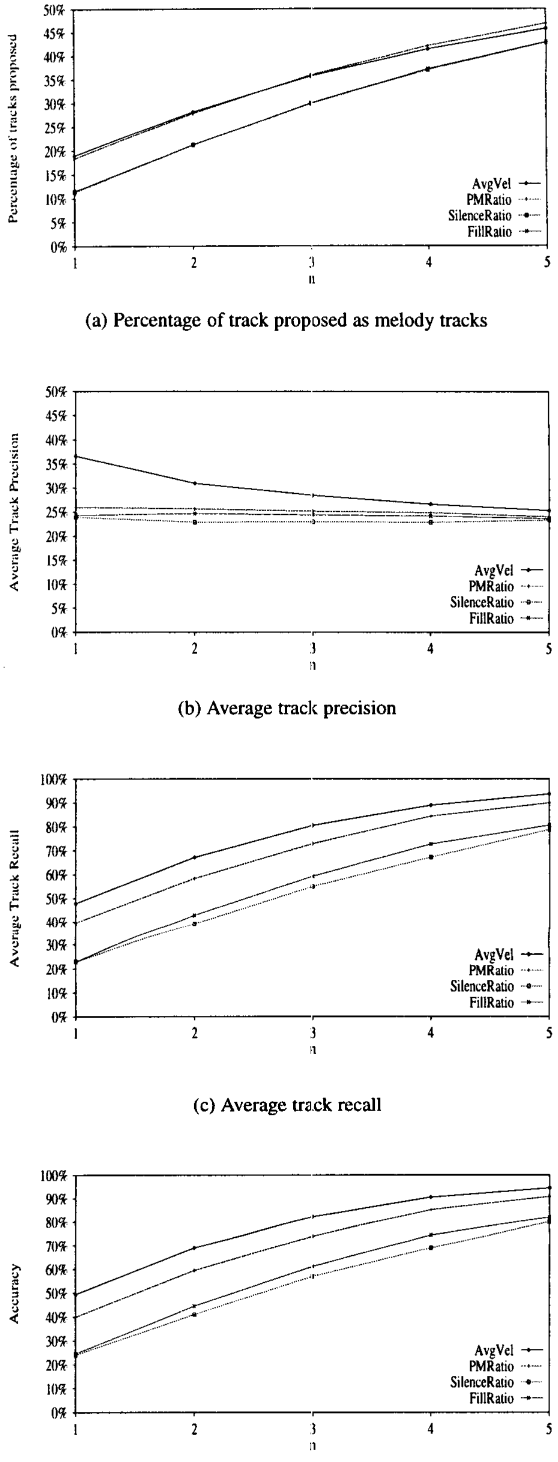
For the whole collection,

average number of tracks = 12.8

standard deviation of number of tracks = 6.47

(c) Some statistics

Table 1. Results for the whole collection



(a) Percentage of track proposed as melody tracks

(b) Average track precision

(c) Average track recall

(d) Accuracy

Figure 1. Graphs for the ranking methods

poses an answer, a correct answer is in the proposed set of tracks. Music retrieval systems can take advantage of this by indexing every track proposed by the TrackName algorithm. The downside of the TrackName algorithm is that its capability is only 44.3%, that is, only 44.3% of the sampled collection contains MIDI tracks with names that the algorithm can work on. This is rather low compared with other classification algorithms. Yet, because of its high track precision, accuracy, and run-time efficiency, we recommend the use of TrackName for selection of melody tracks from music files even if other methods are used concurrently.

Also, from the tables, we found that the Range methods propose more tracks as melody track than others. Range classifies tracks by the musical range of notes in the track. Since accompaniment tracks, as well as melody tracks, may have the specified ranges, Range tends to create false positives by treating non-melody tracks as melody tracks. Hence, the algorithm propose a relatively large number of tracks, resulting in a relatively low precision values. Since these results are comparable with those of AvgVel and PMRatio for $n = 2$, the latter two methods are recommended since they give a larger capability values.

Regarding the number of tracks recommended by the algorithms, AvgVel seconds only to the Range methods. On average, 19.0% of the tracks of a file is proposed as the melody tracks for the topmost match. Yet, its track recall, track precision and accuracy are the best among the three ranking algorithms. With a 80.5% average track recall and a 28.5% average track precision when $n = 3$, the top three or four sets of tracks proposed by AvgVel can be used with confidence that one of the proposed tracks contain a true melody line. A reason that AvgVel proposes a relatively large number of tracks as melody tracks is that many MIDI music were normalized by computers during sequencing. This causes many tracks to have the same average velocity and thus the relatively large number of proposed melody tracks.

In contrast to AvgVel, SilenceRatio proposes very few tracks as melody tracks. However, it has poor track recall and accuracy values. This reflects the assumption that melodies "fill the space" of the music is not valid. Indeed, one might guess that the reverse might be true; accompaniments are often designed to fill the space of musical pieces. Hence, we designed an algorithm FillRatio, one that reverses the rank of SilenceRatio and used it to rank melody lines. The result is shown in the rightmost columns of Table 1(a). We found that the performance of FillRatio is not much different than that of SilenceRatio. Hence, we can only conclude that the time ratio of sounding and non-sounding notes are not useful in determining melody tracks. It is also interesting to note that for SilenceRatio, the average track precision value remain relatively unchanged as n increases.

Another method, the PMRatio, seems to be more promising than SilenceRatio. With a bit more proposed tracks for each n, PMRatio can give a 72.9% average track recall for $n = 3$. However, like SilenceRatio, the average track precision is rather low. To achieve a better track recall value, systems using the algorithm should take not only the first but also the second and later sets of proposed tracks.

As expected, the values for average track recall and accuracy increases as n increases. In general, to achieve a track recall of 80%, the top three to four highest-ranked answer sets proposed by the algorithms can be used for feature extraction and indexing. That would mean that about three tracks need to be indexed for every piece. Considering that the average number of tracks is 12.8, indexing three tracks for every piece would be acceptable.

5. Discussions

In our evaluation of melody track extraction algorithms, we made an implicit assumption that there is a melody line for the music. This is the case for many pieces in our music collection, which consists mostly of pop songs or music with a theme. However, this assumption may not always hold. Sometimes there is no melody, and sometimes every track is a melody. For example, a file can only contain the accompaniement of a song, and classical music often makes use of a musical device called counterpoint: the simultaneous combination of two or more melodies that makes musical sense. The musical genre affects whether a melody line can be found, and thus whether systems based on melody-related features are effective.

In our sets of experiments, we made a simplifying assumption that each melody line resides in one MIDI track. Although it is true for many MIDI arrangements, it is possible for the melody line to be distributed among different MIDI tracks. One method to handle such files is to use moving window analysis. For each track, a moving window of, say, eight measures in length can be analyzed and marked as melody or non-melody. Further analysis on the marked windows can then be used to determine the true melody segments.

6. Summary and future work

In this paper, we have proposed five methods for the selection of melody lines from music files on the Web. Three of them, namely AvgVel, PMRatio, and SilenceRatio, rank tracks according to their odds of being a melody track, and two of them, namely TrackName and Range, classify tracks as melody tracks or non-melody tracks. A collection of 16012 MIDI files obtained from the Web were used to test the algorithms, and it was found that TrackName, which looks for keywords in textual descriptions of MIDI tracks, is the most effective method for selecting melody tracks. It gives a rather accurate proposal of melody track though it can only handle about 44.3% of the files. Since TrackName is simple and efficient, we recommend its use for selection of melody tracks from music files even if other methods are also used. On the other hand, experiments on SilenceRatio and FillRatio show that the time ratio of sounding and non-sounding notes is not useful in determining melody tracks.

Among the three track-ranking methods, AvgVel returns the largest proportion of tracks on average but can give acceptable values of track precision and recall. Since its average track recall is quite high when $n = 3$, tracks in its first few proposed sets can be treated as melody lines with confidence that at least one of them is a true melody line.

Besides the methods introduced here, we have also implemented more sophisticated methods such as the method proposed in [6] for chord analysis. Experiments to evaluate their performances as melody line selectors are under way. Also, data mining techniques for finding effective features for melody line extraction are being investigated.

References

[1] D. Bainbridge, C. G. Nevill-Manning, I. H. Witten, L. A. Smith, and R. J. McNab. Towards a digital library of popular music. In E. A. Fox and N. Rowe, editors, *Proceedings of the fourth ACM conference on Digital Libraries 1999*, pages 161–169, Berkeley, CA, USA., 11–14 Aug. 1999. The Association for Computing Machinery. ISBN 1-58113-145-3.

[2] R. Bennett. *Score-reading*. Cambridge University Press, 1986.

[3] T.-C. Chou, A. L. Chen, and C.-C. Lu. Music databases: Indexing techniques and implementation. In *Proceedings IEEE International Workshop on Multimedia Data Base Management Systems 1996*, 1996.

[4] D. Deutsch. Octave generalization and tune recognition. *Perception and Psychophysics*, 11(6):411–412, 1972.

[5] S. J. Downie. Music retrieval as text retrieval: Simple yet effective. In *Proceedings of the 22th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99)*.

[6] H. J. Maxwell. An expert system for harmonizing analysis of tonal music. In M. Balaban, K. Ebcioğlu, and O. Laske, editors, *Understanding Music with AI: Perspectives on Music Cognition*, pages 335–353. The AAAI Press and The MIT Press, 1992. ISBN 0-262-52170-9.

[7] R. J. McNab, L. A. Smith, I. H. Witten, C. L. Henderson, and S. J. Cunningham. Towards the digital music library: Tune retrieval from acoustic input. In *Proceedings of the 1st ACM International Conference on Digital Libraries*, pages 11–18. The Association for Computing Machinery, 1996.

[8] M. Mongeau and D. Sankoff. Comparison of musical sequences. *Computers and the Humanities*, 24(3):161–175, June 1990.

[9] H. Schaffrath. The Essen associative code: A code for folksong analysis. In E. Selfridge-Field, editor, *Beyond MIDI: The Handbook of Musical Codes*, chapter 24. The MIT Press, 1997. ISBN 0-262-19394-9.

[10] R. S. Wolpert. Recognition of melody, harmonic accompaniment, and instrumentation: Musicians vs. nonmusicians. *Music Perception*, 8(1):95–106, Fall 1990.

[11] C. L. Yip and B. Kao. A study on musical features for melody databases. In *Proceedings of the 10th International Conference and Workshop on Database and Expert Systems Applications (DEXA 1999)*, number 1677 in LNCS, pages 724–733. Springer-Verlag, 1999.