A Pattern Recognition Approach for Melody Track Selection in MIDI Files

David Rizo, Pedro J. Ponce de León, Carlos Pérez-Sancho, Antonio Pertusa, José M. Iñesta
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante, Spain
inesta@dlsi.ua.es

Abstract

Standard MIDI files contain data that can be considered as a symbolic representation of music (a digital score), and most of them are structured as a number of tracks. One of them usually contains the melodic line of the piece, while the other tracks contain accompaniment music. The goal of this work is to identify the track that contains the melody using statistical properties of the musical content and pattern recognition techniques. Finding that track is very useful for a number of applications, like speeding up melody matching when searching in MIDI databases or motif extraction, among others. First, a set of descriptors from each track of the target file are extracted. These descriptors are the input to a random forest classifier that assigns the probability of being a melodic line to each track. The track with the highest probability is selected as the one containing the melodic line of that MIDI file. Promising results have been obtained testing a number of databases of different music styles.

Keywords: Melody finding, musical analysis, symbolic representation, music perception.

1. Introduction

A huge number of digital music score can be found on the Internet or in multimedia digital libraries. These scores are stored in files conforming to a proprietary or open format, like MIDI or the various XML music formats available. Most of these files contain music organized in a way such that the leading part of the music, the melody, is in some way stored separately from the rest of the musical content, which is often the accompaniment for the melody. In particular, a standard MIDI file is usually structured as a number of tracks, one for each voice in a music piece. One of them usually contains a melodic line, specially in the case for modern popular music. The goal of this work is to automatically find this melody track in a MIDI file using statistical properties of the musical content and pattern recognition techniques. The proposed methodology can be applied to other symbolic music file formats, because the information used to take decisions is based solely on how the notes are arranged within each voice of a digital score. Only the feature extraction front-end would need to be adapted for dealing with other formats.

The identification of the melody track is very useful for a number of applications. For example, in melody matching, when the query is either in symbolic format [1] or in audio format [2], the process can be sped up if the melody track is known or a way to know which tracks are most likely to contain the melody, because the query is almost always a melody fragment. Another useful application can be helping motif extraction systems to build music thumbnails of digital scores for music collection indexing.

The literature about melody voice identification is quite poor. In the digital sound domain, several papers aim to extract the melodic line from audio files [3, 4]. In the symbolic domain, Ghias et al. [2] built a system to process MIDI files extracting a sort of melodic line using simple heuristics. Tang et al. [5] present a work where the aim is to propose candidate melody tracks, given a MIDI file, much like in our work. They do so by taking decisions based on single features derived from informal assumptions about what a melody track may be. They conclude that using the track name is the best criteria for selecting a melody track from a MIDI file. In the work presented here, the use of track name (or similar metadata) is explicitly avoided both for labeling tracks and as part of the extracted features, as it has proven to be unreliable information.

In a different approach to melody identification, Uitdenbogerd and Zobel [6] developed four algorithms for detecting the melodic line in polyphonic MIDI files, assuming that a melodic line is a monophonic sequence of notes. These algorithms are based mainly on note pitches; for example, keeping at every time the note of highest pitch from those that sound at that time (skyline algorithm).

Another line of work focuses on how to split a polyphonic source into a number of monophonic sequences by partitioning it into a set of melodies [7] or selecting at most one note at every time step [1]. In general, these works are called monophonic reduction techniques [8]. Different approaches, like using voice information (when available), average pitch, and entropy measures have been proposed in these works.

Other approaches, related to motif extraction, focus on the development of techniques for identifying patterns as repetitions that are able to capture the most representative notes in a music piece [9, 10, 11].

Nevertheless, in this work the aim is not to extract a monophonic line from a polyphonic score, but to decide which of the tracks contains the main melody in a multitrack standard MIDI file. For this, we need to assume that the melody is indeed contained in a single track. This is often the case of popular music.

The features that should characterize melody and accompaniment voices must be defined in order to be able to select the melodic track. There are some features in a melody track that, at first sight, seem to be enough for identifying it, like the presence of higher pitches (see [6]) or being monophonic. Unfortunately, any empirical analysis will show that these hypotheses do not hold in general, and more sophisticated criteria need to be devised in order to take accurate decisions.

To overcome these problems, a classifier ensemble that is able to learn what is a melodic track, in a supervised manner based on note distribution statistics, has been used in this work. In order to setup and test the classifier, a number of data sets based on different music styles and consisting of multitrack standard MIDI files have been constructed. All tracks in such files are labeled either as melody or non-melody.

The rest of the paper is organized as follows: first the methodology is described, both the way a track is characterized and how the classifier is built. Then, the data used and the experiments designed to test the method are discussed, and finally, some conclusions about the results obtained are presented, and future work to be done is discussed.

2. Methodology

The usual methodology in the pattern recognition field has been used in this work. A vector of numeric descriptors is extracted from each track of a target midifile, and these descriptors are the input to a classifier that assigns to each track its probability of being a melody. The random forest classifier [12] –an ensemble of decision trees– was chosen as the pattern recognition tool for this task. The WEKA [13] toolkit was used to implement the system.

2.1. MIDI Track characterization

The content of each non-empty track [1] is characterized by a vector of descriptors based on descriptive statistics of note pitches and note durations that summarize track content information. This type of statistical description of musical content is sometimes referred to as *shallow structure description* [14, 15].

A set of descriptors has been defined, based on several categories of features that assess melodic and rhythmic properties of a music sequence, as well as track related properties. This set of descriptors is presented in Table 1. The left column indicates the category being analyzed, and the right one shows the statistics describing properties from that category.

Table 1. Extracted descriptors

Category	Descriptors
Track information	Normalized duration
	Number of notes
	Occupation rate
	Polyphony rate
Pitch	Highest
	Lowest
	Mean
	Standard deviation
Pitch intervals	Number of different intv.
	Largest
	Smallest
	Mean
	Mode
	Standard deviation
Note durations	Longest
	Shortest
	Mean
	Standard deviation
Syncopation	Number of Syncopated notes

Four features were designed to describe the track as a whole and fifteen to describe particular aspects of its content. For these fifteen descriptors, both normalized and non-normalized versions have been computed. The former were calculated using the formula $(value_i - min)/(max - min)$, where $value_i$ is the descriptor to be normalized corresponding to the i-th track, and min and max are, respectively, the minimum and maximum values for this descriptor for all the tracks of the target midifile. This makes it possible to know these properties proportionally to the other tracks in the same file, using non-dimensional values. This way, a total number of $4 + 15 \times 2 = 34$ descriptors were initially computed for each track.

The track information descriptors are its normalized duration (using the same scheme as above), number of notes, occupation rate (proportion of the track length occupied by notes), and the polyphony rate, defined as the ratio between the number of ticks in the track where two or more notes are active simultaneously and the track duration in ticks (the MIDI file resolution establishes how many ticks form a beat).

Pitch descriptors are measured using MIDI pitch values. The maximum possible MIDI pitch is 127 (note G_8) and the minimum is 0 (note C_{-2}).

The interval descriptors summarize information about the difference in pitch between consecutive notes. The absolute pitch interval values were computed.

Finally, note duration descriptors were computed in terms of beats, so they are independent from the MIDI file resolution.

A view to the graphs in Figure 1 provides some hints on

[1] tracks containing at least one note event. Empty tracks are discarded.

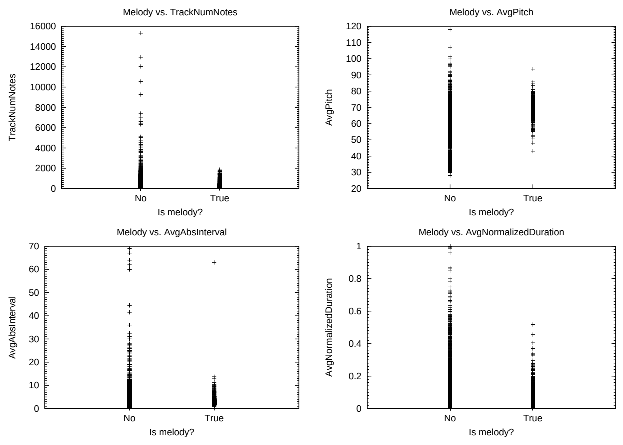

Figure 1. Distribution of values for some descriptors: (top-left) number of notes, (top-right) mean pitch, (bottom-left) mean absolute interval, and (bottom-right) mean relative duration.

what a melody track could look like using these descriptors. This way, a melody track seems to have less notes than other non-melody tracks, an average mean pitch, it contains smaller intervals, and has not too long notes. When this sort of hints are combined by the classifier, a decision about the track "melodicity" is taken. This type of reasoning has been used by other authors, like in [5], in order to build a series of rules able to take a decision on which is the melody track in a symbolic music file.

2.2. The random forest classifier

A number of classifiers were tested in an initial stage of this research and the random forest classifier yielded the best results among them, so it was chosen for the experiments presented in the next section.

Random forests [12] are weighed combinations of decision trees that use a random selection of features to build the decision taken at each node. This classifier has shown good performance compared to other classifier ensembles and it is robust with respect to noise. One forest consists of K trees. Each tree is built to maximum size using CART [16] methodology without pruning. Therefore, each leaf on the tree corresponds to a single class. The number F of randomly selected features to split on the training set at each node is fixed for all the trees. After the trees have grown, new samples are classified by each tree and their results are combined, giving as a result a membership probability for each class.

In our case, the membership for class "melody" can be interpreted as the probability that a track will contain a melodic line.

2.3. Track selection procedure

There are MIDI files that contain more than one track which is suitable to be classified as melody: singing voice, instrument solos, melodic introductions, etc. On the other hand, as usually happens in classical music, some songs do not have an obvious melody, like in complex symphonies or single-track piano sequences. The algorithm proposed here can deal with the first case. For the second case, there are more suitable methods [6] that perform melody extraction from polyphonic data.

In some of the experiments in the next section, at most one melody track per MIDI file is selected. However, a file can contain more than one melody track. Therefore, given a file, all its non-empty tracks are classified and their probabilities of being a melody are obtained. Then the track with the highest probability is selected as the melody track. If all tracks have near-zero probability (actually less than 0.01), no melody track is selected –that is, all tracks are considered as not melody tracks.

In the first stages of this work, a probability threshold around 0.5 was established in order to discard tracks whose probability of being a melody was below that value. This resulted in some files in our test datasets being tagged as melody-less. However most of those files actually have a melody. In general, this produced systems with lower estimated accuracy than systems with no probability threshold.

3. Results

3.1. Datasets

Six corpora (see Table 2) were created, due to the lack of existing databases for this task. The files were downloaded from a number of freely accessible Internet sites. First, three corpora (named JZ200, CL200, and KR200) were created to set up the system and to tune the parameter values. JZ200 contains jazz music files, CL200 has classical music pieces where there was an evident melody track, and KR200 contains popular music songs with a part to be sung (karaoke (.kar) format). All of them are made up of 200 files. Then, three other corpora (named JAZ, CLA, and KAR) from the same music genres were compiled from a number of different sources to validate our method. This dataset is available for research purposes on request to the authors.

Table 2. Corpora used in the experiments, with identifier, music genre, number of files, total number of tracks, and total number of melody tracks.

Corpus ID	Genre	Files	Tracks	Melody tracks
CL200	Classical	200	687	197
JZ200	Jazz	200	769	197
KR200	Popular	200	1370	179
CLA	Classical	131	581	131
JAZ	Jazz	1023	4208	1037
KAR	Popular	1360	9253	1288

The main difficulty for building the data sets was to label the tracks in the MIDI files. Text tagging of MIDI tracks based on metadata such as the track name, is unreliable. Thus, a manual labeling approach was carried out. A musician listened to each one of the MIDI files playing all the tracks simultaneously with a sequencer. For each file, the track(s) containing the perceived melody were identified and tagged as *melody*. The rest of tracks in the same file were tagged as *non-melody*. In particular, introduction passages, second voices or instrumental solo parts were tagged as *non-melody*, aiming to characterise what is clearly a lead melody part.

Some songs had no tracks tagged as melody because either it was absent, or the song contained just a melody-less accompaniment, or the melody constantly moves from one track to another. Other songs contained more than one melody track (e.g. duplicates, often with a different timbre) and all those tracks were tagged as *melody*.

3.2. Experiments

The WEKA package was used to carry out the experiments described here, and it was extended to compute the proposed track descriptors directly from MIDI files.

Four experiments were carried out. The first one tried to assess the capability of random forests to classify melodic and non-melody tracks properly. In the second experiment, the aim was to evaluate how accurate the system was for identifying the melody track in a MIDI file. Finally, the specificity of the system with respect to both the music genre and the corpora utilized were tested.

3.2.1. Melody versus non-melody classification

The random forest classifier assigns a class membership probability to each test sample, so in this experiment a test track is assigned to the class with the highest membership probability.

Three independent sub-experiments were carried out, using the three 200-file corpora (CL200, JZ200, and KR200). This way, 2826 tracks provided by these files were classified in two classes: *melody* / *non-melody*. A 10-folded cross-validation scheme was used to estimate the accuracy of the method. The results are shown in Table 3. The remarkable success percentages obtained are due to the fact that the classifier was able to successfully map the input feature vector space to the class space.

Also, precision, recall and the F-measure are shown for melody tracks. These standard information retrieval measures are based on the so-called *true-positive* (TP), *false-positive* (FP) and *false-negative* (FN) counts. For this experiment, TP is the number of melody tracks successfully classified, FP is the number of misclassified non-melody tracks, and finally, FN is the number of misclassified melody tracks. The precision, recall and F-measure are calculated as follows:

$$Precision = \frac{TP}{TP + FP}$$

Table 3. Melody versus non-melody classification results.

Corpus	Success	Precision	Recall	F-measure
CL200	98.9%	0.99	0.98	0.98
JZ200	96.8%	0.95	0.92	0.94
KR200	96.8%	0.92	0.80	0.86

$$Recall = \frac{TP}{TP + FN}$$

$$Fmeasure = \frac{2 \times Recall \times Precision}{Recall + Precision}$$

There was a somewhat low recall of melody tracks (0.8) for the KR200 corpus. This indicates that most errors are due to FNs, i.e., the classifier missed to detect a significant number of melody tracks. The causes of this confusion are rather obscure, but the reason seems to be the heterogeneous way in which music is organized within karaoke MIDI files.

These results have been obtained using $K = 10$ trees and $F = 5$ randomly selected features for the random forest trees. Due to the relatively good results obtained in the three sub-experiments, the same classifier structure was used in the rest of experiments presented in the next sections.

3.2.2. Melodic track selection experiment

Now, the goal is to know how many times does the method select as melody track the proper track from a file. For this experiment, the system was trained the same way as in the latter one, but now a test sample is not a single track but a MIDI file. Due to the limited number of samples available (200 per corpus), this experiment was performed using a leave-one-out scheme at the MIDI file level to estimate the classification accuracy. All the tracks from a file have a class membership probability. For each file, the system outputs the track number that gets a higher membership probability for the class *melody*, except when all these probabilities are near-zero, in which case the system concludes that the file has no melody track.

The classifier answer is considered as a success if

1. the file has at least one track tagged as *melody* and the selected track is one of them.

2. The file has no melody tracks and the classifier outputs no melody track number.

The obtained results are shown in Table 4. The precision, recall, and F-measure values have been calculated. In order to obtain these values, songs without melody tracks were not considered. The TP value is computed as the number of times that the algorithm returns an actual melody track. Note that this value differs from the success rate because here the songs without any melody track are not included.

The FP value is calculated as the number of times that the system selects a track that is not tagged as melody. The FN are those cases in which the classifier does not select any track, but the song actually has one or more melody tagged tracks.

Table 4. Melody track selection results.

Corpus	Success	Precision	Recall	F-measure
CL200	100.0%	1	1	1
JZ200	96.5%	0.97	0.99	0.98
KR200	72.3%	0.72	0.99	0.84

Note the high quality of the results for CL200 and JZ200. However, a lower success rate has been obtained for the karaoke files. This is due to the fact that 31 out of 200 files in this corpus were tagged by a human expert as having no actual melody track, but they have some portions of tracks that could be considered as melody (like short instrument solo parts), thus confusing the classifier as FP hits, therefore lowering the classifier precision for this corpus.

3.2.3. Style specificity

This experiment was designed in order to evaluate the system robustness against different corpora. In other words, it is interesting to know how specific the classifier's inferred rules are with respect to the music genre of the files considered for training. For it, two melody track selection sub-experiments were performed: in the first one, the classifier was trained with a 200-file corpus of a given style, and tested with a different corpus of the same style (see Table 5). For the second sub-experiment, the classifier was trained using the data of two styles and then tested with the files of another style (see Table 6).

Table 5. Melody track selection within style.

Train.	Test	Success	Precision	Recall	F
CL200	CLA	60.6%	0.62	0.89	0.73
JZ200	JAZ	96.5%	0.97	0.99	0.98
KR200	KAR	73.9%	0.86	0.80	0.83

The results in Table 5 show that the performance of the system degrades when more complex files are tested. The 200-file corpora are datasets that include MIDI files that were selected among many others for having an 'easily' (for a human) identifiable melody track. This holds also for the JAZ corpus, as most jazz music MIDI files have a lead voice (or instrument) track plus some accompaniment tracks like piano, bass and drums. However, it does not hold in general for the other two corpora. Classical music MIDI files (CLA corpus) come in very different structural layouts, due to both the way that the original score is organized and the idiosyncrasy of the MIDI file authors. This is also mostly true for the KAR corpus. Moreover, karaoke files tend to make intensive use of duplicate voices and dense pop arrangements with lots of tracks containing many ornamentation motifs. In addition, we have verified the presence of very short sequences for the CLA corpus, causing less quality in the statistics that also degrades the classification results.

As both the training and test corpus contain samples of the same music genre, better results were expected. However, the CLA and KAR corpora are definitively harder to deal with, as it became clear in the second experiment presented in this section. So, it can be said that the difficulty of the task resides more on the particular internal organization of tracks in the MIDI files than on the file music genre, although the results in Table 5 seem to point out that genre makes a difference. The second experiment presented in Table 6 showed some evidence in this direction.

An interesting point is that, despite low success and precision, recall can be considered high for the CLA corpus. This is due to the fact that most errors are produced because a non-melody track is selected as melody (false-positives). The KAR corpus suffers from both somewhat low precision and low recall: errors come from both false-positives and false-negatives (the classifier does not get a melody from melody tagged tracks).

Table 6. Melody track selection across styles.

Train.	Test	Success	Prec.	Recall	F
KAR+JAZ	CLA	71.7%	0.73	0.92	0.81
CLA+KAR	JAZ	92.6%	0.96	0.97	0.96
CLA+JAZ	KAR	64.9%	0.77	0.78	0.78

The results in Table 6 show that the performance is poorer (with respect to values in Table 5) when no data from the style tested were used for training. This does not happen in classical music, probably due to effects related to the problems expressed above.

3.2.4. Training set specificity

To see how conditioned are these results by the particular training sets utilized, a generalization study was carried out building a new training set merging the three 200-files corpora (named *ALL200*), and then using the other corpora for test. The results are detailed in Table 7.

This shows that, when using a multi-style dataset, the performance of the system is somewhat improved (now the training set contains samples from the same style as the test dataset). Note that the results are better despite that the size of the training set is smaller than the size of those used in the second experiment of Section 3.2.3 (Table 6).

When combining all the success results, taking into account the different cardinalities of the test sets, the average successful melody track identification percentage is 81.2 %.

Table 7. Melody track selection by styles when training with data from all the styles.

Training	Test	Success	Precision	Recall	F
ALL200	CLA	73.8%	0.75	0.92	0.82
ALL200	JAZ	97.0%	0.97	0.99	0.98
ALL200	KAR	70.2%	0.84	0.79	0.82

4. Conclusions and future work

A method to identify the voice containing the melody in a multitrack digital score has been proposed here. It has been applied to standard MIDI files in which music is stored in several tracks, so the system determines whether a track is a melodic line or not. The track with the highest probability among the melodic tracks is finally labeled as the one containing the melody of that song.

The decisions are taken by a pattern recognition algorithm based on statistical descriptors (pitches, intervals, durations and lengths), extracted from each track of the target file. The classifier used for the experiments was a decision tree ensemble classifier named random forest. It was trained using MIDI tracks with the melody track previously labeled by a human expert.

The experiments yielded promising results using databases from different music styles, like jazz, classical, and popular music. Unfortunately, the results could not be compared to other systems because of the lack of similar works.

The results show that enough training data of each style are needed in order to successfully characterize the melody track, due to the specificities of melody and accompaniment in each style. Classical music is particularly hard for this task, because of the lack of a single track that corresponds to the melodic line. Instead, the melody is constantly changing among different tracks along the piece. To overcome this problem, more sophisticated schemes oriented to melodic segmentation are needed.

Also, the use of information about the layout of the tracks within a MIDI file is being investigated. We hope this would help to improve the performance of the system when dealing with particularly hard instances like popular music files. Finally, the extraction of human-readable rules from the trees in the random forest that help characterize melody tracks is another topic under research now.

5. Acknowledgments

This work was supported by the projects: Generalitat Valenciana GV06/166 and Spanish CICyT TIC2003–08496–C04, partially supported by EU ERDF.

References

[1] A. Uitdenbogerd and J. Zobel, “Melodic matching techniques for large music databases,” in *Proceedings of the seventh ACM International Multimedia Conference (Part 1)*. ACM Press, 1999, pp. 57–66.

[2] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith, “Query by humming: Musical information retrieval in an audio database,” in *Proc. of 3rd ACM Int. Conf. Multimedia*, 1995, pp. 231–236.

[3] A. Berenzweig and D. Ellis, “Locating singing voice segments within music signals,” in *Proceedings of the IEEE workshop on Applications on Signal Processing to Audio and Acoustics (WASPAA)*, October 2001.

[4] J. Eggink and G. J. Brown, “Extracting melody lines from complex audio,” in *5th International Conference on Music Information Retrieval (ISMIR)*, 2004, pp. 84–91.

[5] M. Tang, C. L. Yip, and B. Kao, “Selection of melody lines for music databases.” in *Proceedings of Annual Int. Computer Software and Applications Conf. COMPSAC*, 2000, pp. 243–248.

[6] A. L. Uitdenbogerd and J. Zobel, “Manipulation of music for melody matching,” in *Proceedings of the sixth ACM International Multimedia Conference*. ACM Press, 1998, pp. 235–240.

[7] A. Marsden, “Modelling the perception of musical voices: a case study in rule-based systems,” in *Computer Representations and Models in Music*. Academic Press, 1992, pp. 239–263.

[8] K. Lemstrom and J. Tarhio, “Searching monophonic patterns within polyphonic sources,” in *Proceedings of the RIAO Conference, volume 2*, 2000, pp. 1261–1278. [Online]. Available: citeseer.ist.psu.edu/lemstrom00searching.html

[9] E. Cambouropoulos, “Towards a general computational theory of musical structure,” Ph.D. dissertation, Faculty of music and Department of Artificial Intelligence, University of Edinburgh, 1998.

[10] O. Lartillot, “Perception-based advanced description of abstract musical content,” in *Digital Media Processing for Multimedia Interactive Services*, E. Izquierdo, Ed., 2003, pp. 320–323.

[11] B. Meudic, “Automatic pattern extraction from polyphonic MIDI files,” in *Proc. of Computer Music Modeling and Retrieval Conf.*, 2003, pp. 124–142.

[12] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, October 2001.

[13] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques*, ser. Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, 1999.

[14] J. Pickens, “A survey of feature selection techniques for music information retrieval,” Center for Intelligent Information Retrieval, Departament of Computer Science, University of Massachussetts, Tech. Rep., 2001.

[15] P. J. Ponce de León, J. M. Iñesta, and C. Pérez-Sancho, “A shallow description framework for musical style recognition,” *Lecture Notes in Computer Science*, vol. 3138, pp. 871–879, 2004.

[16] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. John Wiley and Sons, 2000.