25

Mining Digital Music Score Collections: Melody Extraction and Genre Recognition

Pedro J. Ponce de León, José M. Iñesta and David Rizo
Department of Software and Computing Systems
University of Alicante,
Spain

1. Introduction

In the field of computer music, pattern recognition algorithms are very relevant for music information retrieval (MIR) applications. Two challenging tasks in this area are the automatic recognition of musical genre and melody extraction, having a number of applications like indexing and selecting musical databases.

One of the main references for music is its melody. In a practical environment of digital music score collections the information can be found in standard MIDI file format. Music is structured as a number of tracks in this file format, usually one of them containing the melodic line, while other tracks contain the accompaniment. Finding that melody track is very useful for a number of applications, like speeding up melody matching when searching in MIDI databases, extracting motifs for musicological analysis, building music thumbnails or extracting melodic ringtones from MIDI files.

In the first part of this chapter, musical content information is modeled by computing global statistical descriptors from track content. 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 then selected as the one containing the melodic line of the MIDI file. The first part of this chapter ends with a discussion on results obtained from a number of databases of different music genres.

The second part of the chapter deals with the problem of classifying such melodies in a collection of music genres. A slightly different approach is used for this task, first dividing a melody track in segments of fixed length. Statistical features are extracted for each segment and used to classify them as one of several genres. The proposed methodology is presented, covering the feature extraction, feature selection, and genre classification stages. Different supervised classification methods, like Bayesian classifier and nearest neighbors are applied. As a proof of concept, the performance of such algorithms against different description models and parameters is analyzed for two particular musical genres, like jazz and classical music.

2. Symbolic music information retrieval

Music information retrieval (MIR) is a field of research devoted to the extraction of meaningful information from the content of music sources. In the application of pattern recognition techniques to MIR, two main folds can be found in the literature: audio information retrieval and symbolic music information retrieval.

Source: Pattern Recognition Techniques, Technology and Applications, Book edited by: Peng-Yeng Yin,
ISBN 978-953-7619-24-4, pp. 626, November 2008, I-Tech, Vienna, Austria

In the first case the raw digital audio signal is processed. Usually `wav` or `MP3` files are the input to these systems. No explicit information about notes, voices or any musical symbol or tag is encoded in the signal. On the other hand, symbolic `MIR` is based on processing symbols with direct musical meaning: notes with pitch and duration, lyrics, tags, etc. The most common formats used as input for these systems are ASCII text files like `kern`, `abc`, `MusicXML`, or binary files containing note control information like `MIDI` files. In these formats input data contain information about what and how is to be played, instead of the rendered music itself like in the audio signal. The semantics of both approaches is different, at least in the first stage of information retrieval algorithms. In the case of symbolic processing, as musical information use to be found as input, most existing music information theory can be applied to the task. On the other hand, the use as raw audio lacks from the basic music information as notes or voices notes or voices. Signal processing techniques must be used to extract this musical data, thus introducing noise to the actual musical material found in the audio. Currently, some of the most active tasks in audio information retrieval have as objective the extraction of that musical information like note onsets, timbre or voices. With this preprocessing of the raw audio, many of the work lines that can be found in the symbolic music information retrieval can also be tackled, but with the drawback of the possibly ill musical data extracted.

The goals of symbolic music information retrieval can be said to be more close to the actual music theory or musicological analysis that those of audio information retrieval. Some of the most active work areas in the symbolic approach nowadays is listed below:

- *Genre and mood classification:* the objective in those two tasks is to tell the mood or musical genre a given input belongs to (McKay & Fujinaga, 2004; Zhu et al., 2004; Cruz et al., 2003; Buzzanca, 2002; Pérez-Sancho et al., 2004; Dannenberg et al., 1997; van Kranenburg & Backer, 2004; Ponce de León et al., 2004)
- *Similarity and retrieval:* the final target in this work line is to be able to perform a search in a music data base to get the most similar pieces to an input query (Typke et al., 2003; Lemstrom & Tarhio, 2000)
- *Cover song identification:* the detection of plagiarisms and variations of the same song is the main goal in this case (Grachten et al., 2004; Li & Sleep, 2004; Rizo et al., 2008)
- *Key finding:* Guess the tonality and key changes of the score given the notes (Rizo et al., 2006a; Temperley, 2004)
- *Melody identification:* to identify the melody line among several MIDI tracks or music staffs, in opposite to those that contain accompaniment (Rizo et al., 2006b)
- *Motive extraction:* to find motives (short note sequences) in a score that are the most repeated ones acting as the main themes of the song (Serrano & Iñesta, 2006)
- *Meter detection:* given the input song reconstruct the meter from the flow of notes (Temperley, 2004)
- *Score segmentation:* split the song in parts like musical phrases (Spevak et al., 2002)
- *Music analysis:* perform musicological analysis for teaching, automatic or computed assisted composition, automatic expressive performance, and build a musical model for other MIR tasks (Illescas et al., 2007)

3. Melody characterization

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 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 or *melody*, specially in the case of modern popular music.

Melody is a somewhat elusive musical term that often refers to a central part of a music piece that catches most of the listener's attention, and which the rest of music parts are subordinated to. This is one of many definitions that can be found in many places, particularly music theory manuals. Most of these definitions share some melody traits, like 'sequential', 'monophonic', 'main reference', 'unity in diversity', 'lyrics related', 'culturally dependent', etc.

Our goal 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 (Uitdenbogerd & Zobel, 1999) or in audio format (Ghias et al., 1995), the process can be speeded up if the melody track is known or if there is 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.

3.1 Related works

To our best knowledge, the automatic description of a melody has not been tackled as a main objective in the literature. The most similar problem to the automatic melody definition is that of extracting a melody line from a polyphonic source. This problem has been approached from at least three different points of view with different understandings of what a melody is. The first approach is the extraction of melody from a polyphonic *audio* source. For this task it is important to describe the melody in order to leave out those notes that are not candidates to belong to the melody line (Eggink & Brown, 2004). In the second approach, a melody line (mainly monophonic) must be extracted from a *symbolic* polyphonic source where no notion of *track* is used (I.Karydis et al., 2007). With this approach, Uitdenbogerd and Zobel (Uitdenbogerd & Zobel, 1998) 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).

Other works on this line focus on how to split a polyphonic source into a number of monophonic sequences by partitioning it into a set of melodies (Marsden, 1992). In general, these works are called monophonic reduction techniques (Lemstrom & Tarhio, 2000).

The last approach to melody characterization is to select one track containing the melody from a list of input tracks of symbolic polyphonic music (e.g. MIDI). This is, by the way, our own approach. Other authors, like (Ghias et al., 1995), built a system to process MIDI files extracting a sort of melodic line using simple heuristics. (Tang et al., 2000) presented a work

where the aim was to propose candidate melody tracks, given a MIDI file. They take decisions based on single features derived from informal assumptions about what a melody track may be. (Madsen & Widmer, 2007) try to solve the problem by the use of several combination of the entropies of different melody properties like pitch classes, intervals, etc.

3.2 What's a melody?

Before focusing on the machine learning methodology to extract automatically the characterization of a *melody*, the musical concept ofmelody needs to be reviewed.

Melody is a concept that has been given many definitions, all of them complementary. The variability of the descriptions can give an idea on the difficulty of the task to extract a description automatically.

From the music theory point of view, Ernst Toch (Toch, 1997) defines it as "*a succession of different pitch sounds brighten up by the rhythm*". He also writes "*a melody is a sound sequence with different pitches, in opposition to its simultaneous audition that constitutes what is named as chord*". He distinguishes also the term 'melody' from the term 'theme'.

A music dictionary (Sadie & Grove, 1984) defines melody as: "*a combination of a pitch series and a rhythm having a clearly defined shape*".

The music theory literature lacks works about melody in favour of works about counterpoint, harmony, or "form" (Selfridge-Field, 1998). Besides, the concept of melody is dependant on the genre or the cultural convention. The most interesting studies about melody have appeared in recent years, mainly influenced by new emerging models like generative grammars (Baroni, 1978), artificial intelligence (Cope, 1996), and Gestalt and cognitive psychology (Narmour, 1990). All these works place effort on understand the melody in order to generate it automatically.

The types of tracks and descriptions of *melody* versus *accompaniment* is posed in (Selfridge-Field, 1998). The author distinguishes:

- *compound* melodies where there is only a melodic line where some notes are principal, and others tend to accompany, being this case the most frequent in unaccompanied string music.
- *self-accompanying* melodies, where some pitches pertain both to the thematic idea and to the harmonic (or rhythmic) support
- *submerged* melodies consigned to inner voices
- *roving* melodies, in which the theme migrates from part to part
- *distributed* melodies, in which the defining notes are divided between parts and the prototype cannot be isolated in a single part.

From the audio processing community, several definitions can be found about what a melody is. Maybe, the most general definition is found in (Kim et al., 2000): "*melody is an auditory object that emerges from a series of transformations along the six dimensions: pitch, tempo, timbre, loundness, spatial location, and reverberant environment*".

(Gomez et al., 2003) gave a list of mid and low-level features to describe melodies:

- Melodic attributes derived from numerical analysis of pitch information: number of notes, tessitura, interval distribution, melodic profile, melodic density.
- Melodic attributes derived from musical analysis of the pitch data: key information, scale type information, cadence information.
- Melodic attributes derived from a structural analysis: motive analysis, repetitions, patterns location, phrase segmentation.

Another attempt to describe a melody can be found in (Temperley, 2004). In that book, Temperley proposes a model of melody perception based on three principles:

- Melodies tend to remain within a narrow pitch range.
- Note-to-note intervals within a melody tend to be small.
- Notes tend to conform to a key profile (a distribution) that depends on the key.

All these different properties a melody should have can be used as a reference to build an automatic melody characterization system.

4. Melody track identification

As stated before, in this work the aim is 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 the case for today's world 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 (Uitdenbogerd & Zobel, 1998) 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 able to learn what is a melodic track from note distribution statistics has been used in this work. In order to setup and test the classifier, a number of data sets based on several music genres and consisting of multitrack standard MIDI files have been constructed. All tracks in such files are labeled either as melody or non-melody.

The classic 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. This is the big picture of the method. The random forest classifier (Breiman, 2001) –an ensemble of decision trees– was chosen as the pattern recognition tool for this task. The WEKA (Witten & Frank, 1999) toolkit was used to implement the system.

4.1 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 durations that summarize track content information. This kind of statistical description of musical content is sometimes referred to as *shallow structure description* (Pickens, 2001; Ponce de León et al., 2004b).

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 kind of statistics describing properties from that category.

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

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

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 allows to know these properties proportionally to the other tracks in the same file. This way, a total number of 4+15×2 = 34 descriptors were initially computed for each track.

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

Table 1. Track content descriptors

The track information descriptors are its normalized duration, 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.
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.

4.2 Feature selection

The descriptors listed above are a complete list of computed features, but any pattern recognition system needs to explore which are those features that actually are able to separate the target classes.
Some descriptors show evidence of statistically significant differences when comparing their distributions for melody and non-melody tracks, while other descriptors do not. This property is implicitly observed by the classification technique utilized (see Section 4.3), that performs a selection of features in order to take decisions.
A view to the graphs in Figure 1 provides some hints on how a melody track could look like. This way, a melody track seems to have less notes than other non-melody tracks, an average mean pitch, it contains small 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.

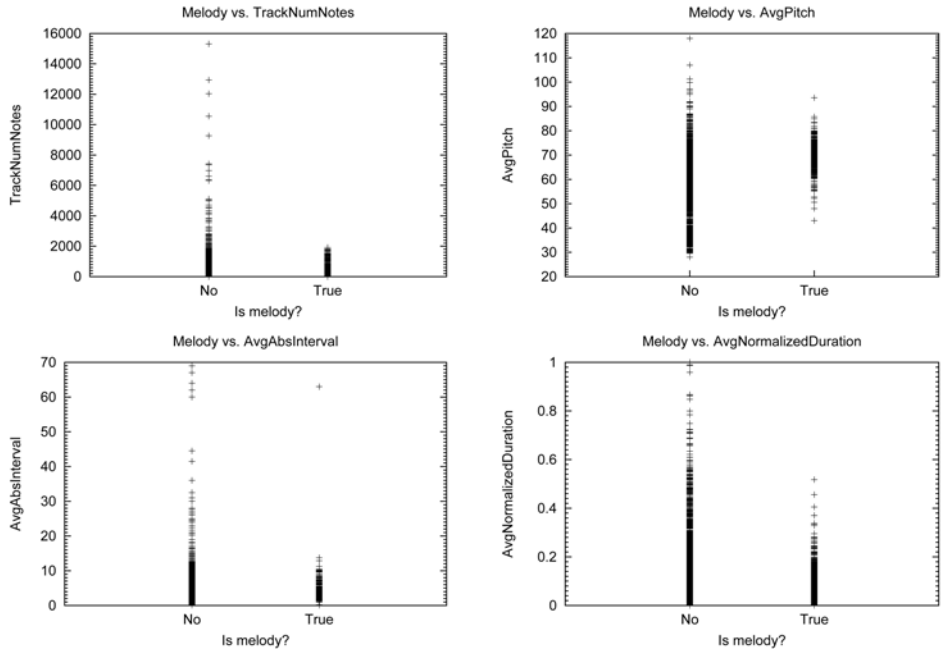

Fig. 1. Distribution of values for some descriptors: (top-left) number of notes, (top-right) mean pitch, (bottom-left) mean absolute interval, and (bottom-right) mean relative duration.

4.3 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 (Breiman, 2001) 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 with a high robustness with respect to noise. One forest consists of *K* trees. Each tree is built to maximum size using CART (Duda et al., 2000) 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" is interpreted as the probability that a track will contain a melodic line.

4.4 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 (Uitdenbogerd & Zobel, 1998) 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 non-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 a near-zero probability threshold.

4.5 Experiments

4.5.1 Datasets and tools

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 a 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.

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

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

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 tracks simultaneously. For each file, tracks 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*.
Some songs had no tracks tagged as melody because either it was absent, or the song contained some kind of melody-less accompaniment, or it had a canon-like structure, where the melody moves constantly 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*.

The WEKA package was used to carry out the experiments described here. It was extended to compute the proposed track descriptors directly from MIDI files.
Four experiments have been carried out, as listed below:

- Melody vs. non-melody classification
- Melody track selection
- Genre specificity
- Training set specificity

The first one tries to assess the capability of random forests to classify melodic and non-melody tracks properly. In the second experiment, the aim is to evaluate how accurate the system is 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.

4.5.2 Melody versus non-melody classification

As described before, our aim is to assess the capability of the classifier to discriminate melody from non-melody tracks. Therefore, given a set of tracks, this experiment classifies them either as melody or non-melody. 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.
As a proof of concept, 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 ten-fold cross-validation scheme was used to estimate the accuracy of the method. The success results are shown in Table 3 and figure 2, along with the baseline ratio when considering a dumb classifier that always output the most frequent class (*non-melody* for all datasets). 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. This shows that content statistics in combination with decision tree based learning can produce good results on the task at hand.
Also, precision (P), recall (R) and the F-measure (F) 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:

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

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

$$F = \frac{2 \times R \times P}{R+P}$$

These results have been obtained using $K = 10$ trees and $F = 5$ randomly selected features for the random forest trees. The same classifier structure was used in the rest of experiments presented in the next sections.

Corpus	Success	Std. dev.	Baseline	P	R	F
CL200	99.6%	0.7	71.3%	0.996	0.998	0.997
JZ200	98.3%	1.4	74.4%	0.981	0.996	0.989
KR200	96.8%	1.8	87.0%	0.971	0.994	0.982

Table 3. Melody versus non-melody classification results.

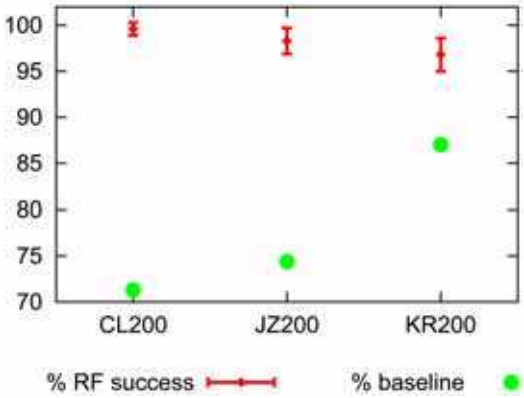

Fig. 2. Melody vs. non-melody classification success and baseline

4.5.3 Melodic track selection experiment

In this second experiment, the goal is to test wether the method selects the proper melody track from a MIDI file. For this experiment, the system was trained the same way as in the previous 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. The classifier assigns a class membership probability to each track in a test file. For each file, the system outputs the track number that gets highest membership probability for class *melody*, except when all these probabilities are near-zero, in which case the system considers the file has no melody track.

The classifier answer for a givenMIDI file is considered correct if

1. At least one track is tagged as *melody* and the selected track is one of them.
2. There are no melody tracks and the classifier outputs no melody track number.

Results are shown in Table 4.

Corpus	Success
CL200	100.0%
JZ200	96.5%
KR200	72.3%

Table 4. Melody track selection results.

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.

4.5.4 Genre specificity

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

Train.	Test	Success
CL200	CLA	60.6%
JZ200	JAZ	96.5%
KR200	KAR	73.9%

Table 5. Melody track selection within genre.

Train.	Test	Success
KAR+JAZ	CLA	71.7%
CLA+KAR	JAZ	92.6%
CLA+JAZ	KAR	64.9%

Table 6. Melody track selection across genres.

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 CLAS corpus, causing less quality in the statistics that also degrades the classification results.

As both the training and test corpora 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, despite that 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 that direction.

Most errors for the CLA test set were produce because a non-melody track was selected as melody (a kind of false positive). Same type of errors can be found for the KAR corpus,

along with errors due to the classifier not finding any melody tracks in files with a melody tagged track (a kind of false-negative).
Results from the second sub-experiment show that performance is poorer (with respect to the first one) when no data from the test genre were used for training. This does not happen in classical music, probably due to effects related to the problems expressed above.

4.5.5 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 problem to solve is again the one discussed in section 4.5.3: selecting the proper melody track from a MIDI file. The results are detailed in Table 7.
This shows that, when using a multi-genre dataset, the performance of the system is somewhat improved (now the training set contains samples from the same genre 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 Section 4.5.4.

Training	Test	Success
ALL200	CLA	73.8%
ALL200	JAZ	97.0%
ALL200	KAR	70.2%

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

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 %.
The method proposed here can be used as a tool for extracting the melody track in conjunction with a system for music genre recognition presented in the next section. This system is a melody based genre recognition system. It extracts information from melody tracks in order to recognize the melody genre. This way MIDI files need not to be preprocessed by an expert in order to identify the melody track.

5. Music genre recognition

One of the problems to solve in MIR is the modelization of music genre. The computer could be trained to recognise the main features that characterise music genres in order to look for that kind of music over large musical databases. The same scheme is suitable to learn stylistic features of composers or even model a musical taste for users. Other application of such a system can be its use in cooperation with automatic composition algorithms to guide this process according to a given stylistic profile.
A number of papers explore the capabilities of machine learning methods to recognise music genre. Pampalk et al. (Pampalk et al., 2003) use self-organising maps (SOM) to pose the problem of organising music digital libraries according to sound features of musical themes, in such a way that similar themes are clustered, performing a content-based classification of the sounds. (Whitman et al., 2001) present a system based on neural networks and support vector machines able to classify an audio fragment into a given list of sources or artists. Also, in (Soltau et al., 1998) a neural system to recognise music types from sound inputs is described. An *emergent* approach to genre classification is used in (Pachet et

al., 2001), where a classification emerges from the data without any *a priori* given set of genres. The authors use co-ocurrence techniques to automatically extract musical similarity between titles or artists. The sources used for classification are radio programs and databases of compilation CDs.

Other works use music data in symbolic form (most MIDI data) to perform genre recognition. (Dannenberg et al., 1997) use a naive Bayes classifier, a linear classifier and neural networks to recognize up to eight moods (genres) of music, such as lyrical, frantic, etc. Thirteen statistical features derived from MIDI data are used for this genre discrimination. In (Tzanetakis et al., 2003), pitch features are extracted both from MIDI data and audio data and used separately to classify music within five genres. Pitch histograms regarding to the tonal pitch are used in (Thom, 2000) to describe blues fragments of the saxophonist Charlie Parker. Also pitch histograms and SOM are used in (Toiviainen & Eerola, 2001) for musicological analysis of folk songs. Other researchers use sequence processing techniques like Hidden Markov Models (Chai & Vercoe, 2001) and universal compression algorithms (Dubnov & Assayag, 2002) to classify musical sequences.

(Stamatatos & Widmer, 2002) use stylistic performance features and the discriminant analysis technique to obtain an ensemble of simple classifiers that work together to recognize the most likely music performer of a piece given a set of skilled candidate pianists. The input data are obtained from a computer-monitored piano, capable of measuring every key and pedal movement with high precision.

Compositions from five well known eighteenth-century composers are classified in (van Kranenburg & Backer, 2004) using several supervised learning methods and twenty genre features, most of them being counterpoint characteristics. This work offers some conclusions about the differences between composers discovered by the different learning methods.

In other work (Cruz et al., 2003), the authors show the ability of grammatical inference methods for modeling musical genre. A stochastic grammar for each musical genre is inferred from examples, and those grammars are used to parse and classify new melodies. The authors also discuss about the encoding schemes that can be used to achieve the best recognition result. Other approaches like multi-layer feed-forward neural networks (Buzzanca, 2002) have been used to classify musical genre from symbolic sources.

(McKay & Fujinaga, 2004, 2007a) use low and mid-level statistics of MIDI file content to perform music genre recognition by means of genetic algorithms and pattern recognition techniques. They have developed several tools for feature extraction from music symbolic sources (particularly MIDI files) or web sites (McKay & Fujinaga, 2006a, 2007b). In (McKay & Fujinaga, 2006b), the authors provide some insight on why is it worth continuing research in automatic music genre recognition, despite the fact that the ground-truth information available for research is often not too reliable, being subject to subjective tagging, market forces or being culture-dependent. Most of the classification problems detected seem to be related to the lack of reliable ground-truth, from the definition of realistic and diverse genre labels, to the need of combining features of different nature, like cultural, high- and low-level features. They also identify, in particular, the need for being able to label different sections of a music piece with different tags.

The system presented in this section share some features with the one developed by McKay, as the use of low level statistics and pattern recognition techniques but, while McKay extract features from the MIDI file as a whole, our system focus on melody tracks, using a *sliding*

window technique to obtain melody segments that become instances to feed the pattern recognition tools. This allows to obtain partial decisions for a melody track that can offer the users sensible information for different parts of a music work. Also, this decisions can be combined to output a classification decision for a music piece.

5.1 An experimental framework for automatic music genre recognition

In this section a framework for experimenting on automatic music genre recognition from symbolic representation of melodies (digital scores) is presented. It is based on shallow structural features of melodic content, like melodic, harmonic, and rhythmic statistical descriptors. This framework involves all the usual stages in a pattern recognition system, like feature extraction, feature selection, and classification stages, in such a way that new features and corpora from different musical genres can be easily incorporated and tested.

Our working hypothesis is that melodies from a same musical genre may share some common low-level features, permitting a suitable pattern recognition system, based on statistical descriptors, to assign the proper musical genre to them.

Two well-defined music genres, like jazz and classical, have been chosen as a workbench for this research. The initial results have been encouraging (Ponce de León & Iñesta, 2003) but the method performance for different classification algorithms, descriptor models, and parameter values needed to be thoroughly tested. This way, a framework for musical genre recognition can be set up, where new features and new musical genres can be easily incorporated and tested.

This section presents the proposed methodology, describing the musical data, the descriptors, and the classifiers used. The initial set of descriptors will be analyzed to test their contribution to the musical genre separability. These procedures will permit us to build reduced models, discarding not useful descriptors. Then, the classification results obtained with each classifier and an analysis of them with respect to the different description parameters will be presented. Finally, conclusions and possible lines of further work are discussed.

5.2 Musical data

MIDI files from jazz and classical music, were collected. These genres were chosen due to the general agreement in the musicology community about their definition and limits. Classical melody samples were taken from works by Mozart, Bach, Schubert, Chopin, Grieg, Vivaldi, Schumann, Brahms, Beethoven, Dvorak, Haendel, Paganini and Mendelssohn. Jazz music samples were standard tunes from a variety of well known jazz authors including Charlie Parker, Duke Ellington, Bill Evans, Miles Davis, etc. The MIDI files are composed of several tracks, one of them being the melody track from which the input data are extracted[2]. The corpus is made up of a total of 110 MIDI files, 45 of them being classical music and 65 being jazz music. The length of the corpus is around 10000 bars (more than 6 hours of music). Table 8 summarizes the distribution of bars from each genre. This dataset is available for research purposes on request to the authors.

[2] All the melodies are written in 4/4 meter. Anyway, any other meter could be used because the measure structure is not use in any descriptor computation. All the melodies are monophonic sequences (at most one note is playing at any given time).

	Min.	Max.	Avg.	Total	% of total
JAZZ	16	203	73	4734	47.5%
CLAS	44	297	116	5227	52.5%

Table 8. Distribution of melody length in bars

This corpus has been manually checked for the presence and correctness of key, tempo and meter meta-events, as well as the presence of a monophonic melody track. The original conditions under which the MIDI files were created are unknown; They may be human performed tracks or sequenced tracks (i.e. generated fromscores) or even something of both worlds. Nevertheless, most of the MIDI files seem to fit a rather common scheme: a human-performed melody track with several sequenced accompaniment tracks.
The monophonic melodies consist of a sequence of musical events that can be either notes or silences. The pitch of each note can take a value from 0 to 127, encoded together with the MIDI note onset event. Each of these events at time t has a corresponding note off event at time $t+d$, being d the note duration measured in ticks[3]. Time gaps between a note off event and the next note onset event are silences.

5.3 Description scheme

A description scheme has been designed based on descriptive statistics that summaris the content of the melody in terms of pitches, intervals, durations, silences, harmonicity, rhythm, etc.
Each sample is a vector of musical descriptors computed from each melody segment available (See section 5.4 for a discussion about how these segments are obtained). Each vector is labeled with the genre of the melody which the segment belongs to. We have defined an initial set of descriptors based on a number of feature categories that assess the melodic, harmonic and rhythmic properties of a musical segment, respectively.
This initial model is made up of 28 descriptors summarized in table 9, and described next:

- Overall descriptors:
  - *Number of notes*, *number of significant silences, and number of not significant silences*. The adjective *significant* stands for silences explicitly written in the underlying score of the melody. In MIDI files, short gaps between consecutive notes may appear due to interpretation nuances like *stacatto*. These gaps (interpretation silences) are not considered significant silences since they should not appear in the score. To make a distinction between kinds of silence is not possible from the MIDI file and it has been made based on the definition of a silence duration threshold. This value has been empirically set to a duration of a sixteenth note. All silences with longer or equal duration than this threshold are considered significant.
- Pitch descriptors:
  - *Pitch range* (the difference in semitones between the highest and the lowest note in the melody segment), *average pitch* relative to the lowest pitch, and *standard deviation of pitches* (provides information about how the notes are distributed in the score).

[3] A *tick* is the basic unit of time in a MIDI file and is defined by the resolution of the file, measured in ticks per beat.

Category	Descriptors
Overall	Number of notes
	Number of significant silences
	Number of non-significant silences
Pitch	Pitch range
	Average pitch
	Dev. pitch
Note duration	Note duration range
	Avg. note duration
	Dev. note duration
Silence duration	Silence duration range
	Avg. silence duration
	Dev. silence duration
Inter Onset Interval	IOI range
	Avg. IOI
	Dev. IOI
Pitch interval	Interval range
	Avg. interval
	Dev. interval
Non-diatonic notes	Num. non-diatonic notes
	Avg. non-diatonic degrees
	Dev. non-diatonic degrees
Syncopation	Number of syncopes
Normality	Pitch distrib. normality
	Note duration distrib. normality
	Silence duration distrib. normality
	IOI distrib. normality
	Interval distrib. normality
	Non-diatonic degree distrib. normality

Table 9. Musical descriptors

- Note duration, silence duration and IOI[4] descriptors are measured in ticks and computed using a time resolution of $Q = 48$ ticks per bar [5]. Interval descriptors are computed as the difference in absolute value between the pitches of two consecutive notes.
- Harmonic descriptors:
  - *Number of non diatonic notes.* An indication of frequent excursions outside the song key (extracted from the MIDI file) or modulations.
  - *Average degree of non diatonic notes.* Describes the kind of excursions. This degree is a number between 0 and 4 that indexes the non diatonic notes of the diatonic scale of the tune key, that can be major or minor key[6]
  - *Standard deviation of degrees of non diatonic notes.* Indicates a higher variety in the non diatonic notes.

[4] An IOI is the distance, in ticks, between the onsets of two consecutive notes. Two notes are considered consecutive even in the presence of a silence between them.

[5] This is call quantisation. $Q = 48$ means that when a bar is composed of 4 beats, each beat can be divided, at most, into 12 ticks.

[6] Non diatonic degrees are: 0: ♭ II, 1: ♭ III (♮III for minor key), 2: ♭ V, 3: ♭ VI, 4: ♭ VII. The key is encoded at the beginning of the melody track. It has been manually checked for correctness in our data.

- Rhythmic descriptor:
  - *Number of syncopations*: notes that do not begin at measure beats but in some places between them (usually in the middle) and that extend across beats.
- Normality descriptors. They are computed using the D'Agostino statistic for assessing the distribution normality of the n values v_i in the segment for pitches, durations, intervals, etc. The test is performed using this equation:

$$D = \frac{\sum_i (i - \frac{n+1}{2}) v_i}{\sqrt{n^3 (\sum_i v_i^2 - \frac{1}{n} (\sum_i v_i)^2)}} \quad (1)$$

For pitch and interval properties, the range descriptors are computed as maximum minus minimum values, and the average-relative descriptors are computed as the average value minus the minimum value (only considering the notes in the segment). For durations (note duration, silence duration, and IOI descriptors) the range descriptors are computed as the ratio between the maximum and minimum values, and the average-relative descriptors are computed as the ratio between the average value and the minimum value.
This descriptive statistics is similar to histogram-based descriptions used by other authors (Thom, 2000; Toiviainen & Eerola, 2001) that also try to model the distribution of musical events in a music fragment. Computing the range, mean, and standard deviation from the distribution of musical properties, we reduce the number of features needed (each histogram may be made up of tens of features). Other authors have also used this sort of descriptors to classify music (Tzanetakis et al., 2003; Blackburn, 2000), mainly focusing on pitches.

5.4 Free parameter space

Given a melody track, the statistical descriptors presented above are computed from equal length segments extracted from the track, by defining a window of size ω measures. Once the descriptors of a segment have been extracted, the window is shifted δ measures forward to obtain the next segment to be described. Given a melody with $m > 0$ measures, the number of segments s of size $\omega > 0$ obtained from that melody is

$$s = \begin{cases} 1 & \text{if } \omega \geq m \\ 1 + \lceil \frac{m-\omega}{\delta} \rceil & \text{otherwise} \end{cases} \quad (2)$$

showing that at least one segment is extracted in any case (ω and s are positive integers; m and δ may be positive fractional numbers).
Taking ω and δ as free parameters in our methodology, different datasets of segments have been derived from a number of values for those parameters. The goal is to investigate how the combination of these parameters influences the segment classification results. The exploration space for this parameters will be referred to as $\omega\delta$-space. A point in this space is denoted as $\langle \omega, \delta \rangle$.
ω is the most important parameter in this framework, as it determines the amount of information available for the descriptor computations. Small values for ω would produce windows containing few notes, providing little reliable statistical descriptors. Large values for ω would lead to merge –probably different– parts of a melody into a single window and they also produce datasets with fewer samples for training the classifiers (see Eq. 2). The