Musical Part Classification in Content Based Systems

Steven Blackburn and David De Roure

Department of Electronics and Computer Science, University of Southampton
sgb97r@ecs.soton.ac.uk

Abstract. In the Fourth Open Hypermedia Systems workshop, the authors presented scenarios and a set of prototype tools to explore the application of open hypermedia principles, including content-based navigation, to temporal media. The purpose of the current paper is to provide an update on this work, focussing in particular on the ongoing work on content based navigation of music. We describe the development of a system which classifies the musical parts in MIDI files. This classification is then used to enhance content based retrieval and navigation of music. The effectiveness of the various approaches to classification is evaluated with respect to both stand-alone performance and when used in conjunction with a contour database.

1 Introduction

In the paper "Amphion: Open Hypermedia Applied to Temporal Media" [2] in the Fourth Open Hypermedia Systems workshop, we presented scenarios and a set of prototype tools to explore the application of open hypermedia principles to temporal media (see also [1]). As well as hypermedia linking based on time positions and intervals, our interest continues to be in the navigation of a multimedia information space using content-based navigation; i.e. we move from multimedia content to features of that content to enhance our navigational hypermedia system. Our case study is musical information, which provides a particularly rich set of representations with which to work.

Of the many possible musical features, we have chosen melodic pitch contours as our case study. While other systems using pitch contours are aimed primarily at content based retrieval, we have addressed the particular requirements of an interactive content based navigation system. For example, the response time needs to be within acceptable bounds, so we have chosen to employ query expansion and constant-time lookup rather than fuzzy matching mechanisms.

The primary motivation for the study presented here is that informal empirical work has demonstrated a need to be able to refine our searches; e.g. we may have a query which is known to be a fragment of melody, and we wish to match this against other melodies and not all parts. This requirement is distinct from identifying instruments, which can sometimes be achieved from available meta-data anyway. Analogous situations occur in other content based navigation domains such as images, where we wish to compare images of a similar type.

The main part of this paper addresses the classification problem, why it is desirable and our methodology for investigating it, using MIDI as a case study. We have employed the MIDI format as a representation from which to extract the features which in turn are used to determine the classification. MIDI files store several independent but synchronised channels each typically representing one instrument and carrying note-on and note-off information; each channel may have many notes sounding at once. We can convert digital audio to MIDI by available pitch tracking techniques, and the representations can be aligned via time-stamps and position pointers. MIDI versions of material may be available directly from the production process (synchronized with digital audio) or they may be created independently.

A secondary motivation for this study has arisen in the course of the research. Initially we regarded multimedia content as containing locations and objects to which data in a link database might refer. We now propose another view: that multimedia content can itself be regarded as a linkbase. For example, given a fragment of music played on one instrument, a MIDI file enables us to find associated fragments played by other instruments, perhaps associated text, and indeed timing information which we can use to reference synchronised media such as video.

In order to enhance our prototype system and to explore 'content as linkbase', we have identified a need to classify channels of MIDI data in terms of the role they have in the music. This is an instance of a general approach where one could take compound multimedia entities and classify their sub-components in order to provide meaningful navigation in that space — we believe similar issues would arise with other entities that entwine various forms of information and a coordinate system, such as digital maps.

2 Classification

Classification can improve content based retrieval systems by decreasing the amount of noise in the database. Taking retrieval of music as an example, there is little point in indexing content which is not going to be the subject of subsequent searches. Removing all but the lead (melody) line should improve the matching ability of query by humming systems.

Classification has uses in both temporal and content based navigation, although one could argue that the temporal case is a form of the latter. The classification could be used as a parameter to link resolution. This enables temporal links to be authored on particular parts of the musi; e.g. from 20 seconds to 30 seconds of the bass line. Similarly, content based navigation would allow an anchor to take the form of 'lead lines that sound like this'.

There has been little work on classifying music to assist with search, with Uitdenbogerd and Zobel [5] being notable exceptions. They tested several simple algorithms for producing a monophonic melody from a polyphonic MIDI file. A small user trial suggested that always using the top note gave acceptable results for many cases. However, this approach is obviously flawed if the melody is not the highest pitch being played.

The MIDI databases developed by Ghias et al [3], McNab [4] and Blackburn [1] make use of the General MIDI standard which defines MIDI channel 10 to be dedicated to drums. Assuming that all MIDI files adhere to the standard, which the majority do, rhythm tracks can be identified with a high degree of certainty and not added to the database.

Classification involves extracting a set of features, referred to as a feature vector, from each channel. Two feature vectors can be compared by determining the Euclidean distance between them. The Euclidean distance is found by summing the squares of the difference between each element in a vector and then taking the square root. Another function evaluates the so-called 'city block' distance, which simply uses the mean difference. It is referred to as the city block metric due to the shape of the boundaries in the feature space, a diamond.

3 Part Types

For our experiments we use four musical part classes, according to how a layman listener is likely to identify each part. This approach avoids the use of any genre specific terminology and so is more generally applicable. The classifications used are: accompaniment; bass; drums / rhythm and lead.

The use of a classification to describe irrelevant parts was considered but it was decided that it would simply introduce noise into the classifiers, making identification more difficult. Having a separate 'interesting / confidence' measure would be more relevant.

4 Compiling the Training Data

Classifiers require a set of files by which to evaluate their performance, and many require the same data to implement the classifier.

We selected 100 files at random from a collection of 7500 MIDI files, covering a variety of genres, found on the Internet. If it consisted solely of popular music then certain rules may be adhered to. The variable quality and variety of musical styles makes classification that much harder, but the result should be a more generally applicable classifier. The part classification for each channel was marked up manually and then extracted to individual MIDI files (so there was only one channel per MIDI file).

This resulted in a total of 535 classified musical parts: 152 accompaniment; 103 bass; 144 drums and 136 lead.

5 Musical Features

A variety of features are extracted from each training channel. These are:

- The number of notes per second

- The number of chords per second
- The pitch of notes (lowest, highest and the mean average)
- The number of pitch classes used in the channel (ie: C, C#, D, E, etc)
- Pitch classes used
- Pitch class entropy
- The length of notes (lowest, highest and mean average)
- Number of semitones between notes (lowest, highest and mean average)
- The polyphony of the channel (lowest, highest and mean average)
- How repetitive the channel is
- The number of the MIDI channel
- The instrument patches used

Repetitiveness is a number based on how many distinct n-grams there are in the primary pitch contour [1] of the channel. This is the number of distinct n-grams, represented as a percentage of the total number of pitch directions in the channel.

All of these features are normalised to a value between 0 and 1, as this is improves the performance of neural networks for classification. It also improves the effectiveness of other classification systems and makes calculation of a probability straightforward.

6 Approaches to Classification

This section discusses the three methods of classification that we have investigated.

6.1 A Simplistic Statistical Approach

We first take the simple approach of averaging each of the features for all of the channels of a common part classification. The result is four feature vectors which describe the features common to each class type.

A feature vector to be classified is compared with each of these class templates. The closest match is determined to be the classification. Although not particularly scientific, this technique can be implemented quickly and provides a benchmark by which to evaluate the more complex approaches.

6.2 Ad-hoc Classification

The ad-hoc classifier is based on directly coding the knowledge of a musician. The classifier notes properties in a channel that would make it more or less likely to be a particular part. A probability can then be assigned to each part and the most likely chosen.

As previously mentioned, General MIDI (GM) files assign channel 10 to drums. In addition to this, there are some GM instrument numbers that are usually used for rhythm channels (i.e. timpani, taiko drum, melodic tom, synth drum, reverse cymbal and wood block). A channel exhibiting these properties is assumed to be drums. The probability of other tracks being drums is taken to be 70% of the repetitiveness measure.

A channel is more likely to be accompaniment if it only uses notes which are no more than one octave either side of middle C and is likely to be polyphonic. Bass parts tend to use the two octaves below accompaniment and is quite repetitive (a measure greater than 0.6). Bass is very likely to be monophonic and is heavily penalised if not. Lead parts often use the two octaves above middle C and is also likely to be monophonic. It is penalised for being repetitive.

The particular technique we employ in determining the probability has similar properties to comparing a feature vector, so the different distance metrics may produce different results. However, the values used in the evaluation are very coarse, so are unlikely to differ.

6.3 K-Nearest Neighbour

The k-nearest neighbour (K-NN) classification method is, theoretically at least, the most accurate one considered here. The entire training set of feature vectors is stored, along with their classification. Classification finds the example vector that best matches the unclassified one and returns the classification of that example vector.

Due to the relatively small size of the training set (hundreds instead of thousands) it was feasible to keep the entire set in memory. Searching against the set should result in a noticeable performance hit (compared to the previous two methods). No improvements to performance were considered here as accuracy was our primary concern.

7 Evaluation

We evaluate the various approaches. Primarily for accuracy, as the algorithms are not optimised for speed, although speed is interesting to note.

7.1 Comparison of Accuracy

The manual classification was compared against that given by each classification method (see Table 1). The results are broken down to identify which method performs best for each part type. The mismatches were then broken down, to show where each method gets confused — when a classification is wrong, what is it mistaken for. Both Euclidean and ‘city block’ distance metrics were evaluated.

The K-NN classifier used 90% of the training set, so that the remaining 10% of the set could be used to evaluate its performance, allowing a more direct comparison with the other methods.

Simple Template The statistical templates were tested against the set of channels used to calculate the templates. The results are shown in Table 2.

Table 1. Overall classifier accuracy

Classifier	Euclidean	City block
Statistical	58.88%	66.17%
Ad-hoc	71.21%	71.21%
K-NN	67.27%	72.73%

Table 2. Breakdown of statistical template classification

	Correct		Class when wrong	
Part	Euclidean	City Block	Euclidean	City Block
Acc	40.70%	53.29%	22.18%	27.62%
Bass	76.70%	80.58%	32.73%	34.81%
Drums	60.42%	62.50%	4.55%	3.87%
Lead	63.97%	73.53%	34.55%	33.70%
Overall	58.88%	66.17%		

Ad-hoc Classification The ad-hoc classification was tested against the marked up set of channels. The results are shown in Table 3.

Table 3. Breakdown of ad-hoc classification

	Correct		Class when wrong	
Part	Euclidean	City Block	Euclidean	City Block
Acc	59.21%	59.21%	40.91%	40.91%
Bass	80.58%	80.58%	24.03%	24.03%
Drums	94.44%	94.44%	1.30%	1.30%
Lead	52.94%	52.94%	33.77%	33.77%
Overall	71.21%	71.21%		

K-Nearest Neighbour The k-nearest neighbour method was only tested with examples not used in the feature vector table. This allows a more direct comparison with the simple template approach, although the test set is only a subset of that used for evaluating the other methods. The results are shown in Table 4.

7.2 Comparison of Speed

Timing analysis was performed by reading the set of MIDI files, used to train the classifiers, into memory. These files were then classified as many times as

Table 4. Breakdown of k-nearest neighbour classification

	Correct		Class when wrong	
Part	Eucl.	City	Eucl.	City
Acc	40.00%	53.33%	33.33%	46.67%
Bass	84.62%	84.62%	11.11%	6.67%
Drums	90.00%	90.00%	5.56%	6.67%
Lead	64.71%	70.73%	50.00%	40.00%
Overall	67.27%	72.73%		

possible in ten minutes. Tests were completed on a single user 500 MHz Pentium III running Windows NT 4. Note that K-nearest classification is not using an optimised algorithm. The results, given in Table 5, show that the k-nearest neighbour classifier is not noticeably slower than the statistical approach.

Table 5. Comparison of classifier execution time

Classifier	Euclidean	City block
Statistical	50.14 ms	50.65 ms
Ad-hoc	49.75 ms	49.83 ms
K-NN	54.90 ms	53.42 ms

7.3 Evaluation of Data Filtering on the Database

One of the reasons for classifying the channels was to improve the quality of data in a musical database. We evaluated this by building a contour database consisting of a collection of 800 MIDI files and the training set. The database was of the type described in [1] and used an atomic unit of 14 pitch directions. Secondary contours were used to both build and query the database.

Only accompaniment and lead parts were added. Ideally, only lead parts would be added, but the classification systems easily confuse lead with accompaniment, so both lead and accompaniment parts are used. The city block metric gave the best performance, so this was the one used for all classifiers. The resulting performance, considering three part classifications, is given in the Table 6.

Database queries were simulated by extracting a segment of a lead part from the training set, as the manual classification represents a realistic query. The set queries used was 500 MIDI files, each 10 seconds long, although 57 of the files had pitch contours less than 12 pitch directions and so ignored, to represent a sensible query. The rank of the file from which the query (the target file) is

Table 6. Comparison of classifer accuracy when considering three classes

Classifier	Performance
Statistical	82.24%
Ad-hoc	88.97%
K-NN	90.91%

taken is noted, along with the total number of matches found. The test was run with varying numbers of error, d, accounted for in the query. The errors are in addition to those that the pitch contour representation does not allow for (ie: more than changes in the exact pitch). Note that errors were not simulated in the query and so assumes exact pitch contour recall on behalf of the user.

The mode ranking found in all the tests was 1. This does not tell the whole story as, in the case of $d = 2$, this accounts for less than 35% of the queries. The graphs (Fig. 7.3, Fig. 7.3 and Fig. 7.3) show that, for $d = 0$, 90% of queries will return the target file in the top 8 files, whether a classifier is used or not. Classification does improve matching although this tends to reduce the number of false positives, rather than having a large impact on the ranking of common queries.

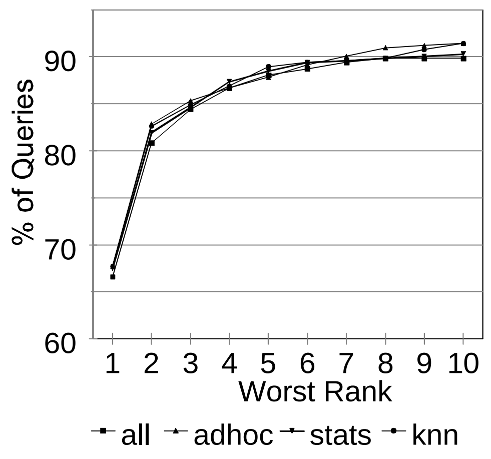

Fig. 1. Response when using exact matching ($d = 0$)

Tables 7, 8 and 9 show the average (mean) recall accuracy, taking into account any extreme results. It can be seen that the ad-hoc classifier gives consistently good average results, although the benefits of filtering only become apparent when matching allowing for larger errors.

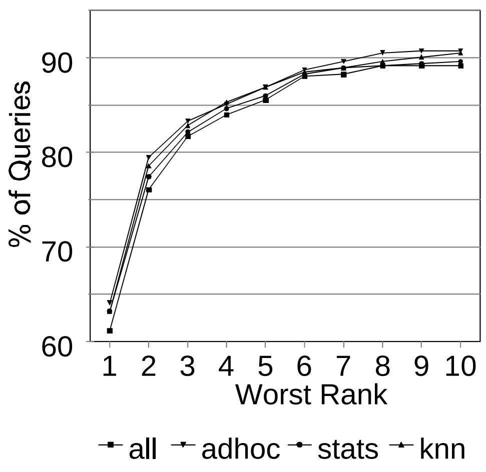

Fig. 2. Response when allowing for one error ($d = 1$)

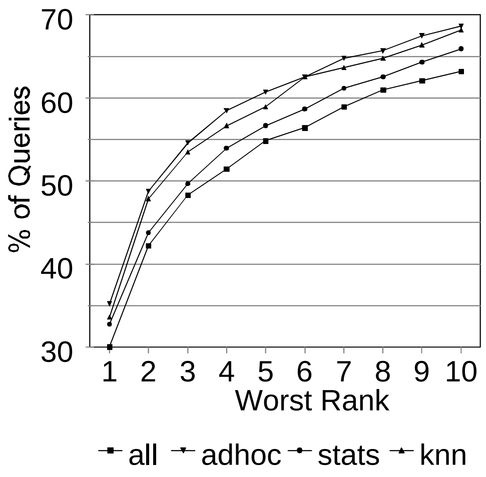

Fig. 3. Response when allowing for two error ($d = 2$)

Table 7. Ranking of targets using exact matching

Classifier	Mean rank	Mean set size
None	13.8059	46.9097
Statistical	11.3747	39.2460
Ad-hoc	10.1041	33.5056
K-NN	10.4244	35.7246

Table 8. Ranking of targets allowing for one error in the query

Classifier	Mean rank	Mean set size
None	16.5598	68.0316
Statistical	13.8623	59.8126
Ad-hoc	12.1174	50.8440
K-NN	12.5102	53.9052

Table 9. Ranking of targets allowing for two errors in the query

Classifier	Mean rank	Mean set size
None	46.605	334.461
Statistical	39.8442	319.779
Ad-hoc	32.6343	288.009
K-NN	33.7607	295.641

8 Conclusions and Future Work

We have shown the application of alternative classification techniques to improve our content-based navigation system. Some techniques are purely statistical while others attempt to bring to bear some knowledge of the nature of the content. It can be seen that classification of musical part does improve matching, more notably when performing approximate queries. The ad-hoc method appears to work best and is also the fastest.

Reliable classification can be used to enhance a content based navigation system by allowing more precise (but still general) links to be authored. For example: 'bass lines like this one are linked to some information'. We have yet to evaluate the recall accuracy of a link database when used in conjunction with part filtering. It should prove to be at least as useful as shown here, due to the fact that only a subset of the music is stored. The accuracy of the classifiers is shown to be extremely good when considering three types of musical part, as much as 90%. Other classifiers may perform better on the difficult problem of differentiating between accompaniment and lead. Neural networks are one method we plan to investigate, along with a hybrid classifier that uses a combination of the most accurate features of the other methods.

We have exercised the notion of 'content as linkbase' in an experiment using MIDI files which contain textual representation of lyrics, enabling us to map from contour to text (and thence tools for linking with text). Future work includes building on the classification system to map from one part to another; e.g. given a chord sequence, what do we know about the corresponding bass lines within a given context? A metric for one of the characteristics of musical parts, repetitiveness, was obtained as a side effect of the content-processing implemented to compute the contour database, an aspect we hope to explore in future work.

This demonstrates a general point: content-based navigation involves features; once we have features, we have the basis for classification of components of content and of the content itself (e.g. classification of musical genre). Classification improves the effectiveness of searches; furthermore, by classifying in advance, we reduce computation at query time.

9 Acknowledgements

We would like to thank Pete Elsdon for his input into the ad-hoc classifier. This work was partially funded by EPSRC QuIC.

References

[1] S. Blackburn and D. DeRoure. A tool for content based navigation of music. In *Proc. ACM Multimedia '98*, pages 361–368, 1998.

[2] D. DeRoure and S. Blackburn. Amphion: Open hypermedia applied to temporal media. In *Proc. 4th Open Hypermedia Workshop*, pages 27–32, 1998.

[3] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith. Query by humming - musical information retrieval in an audio database. In *Proc. Multimedia '95*, pages 231–236, San Francisco, California, November 1995.

[4] 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 *Proc. DL'96*, pages 11–18, 1996.

[5] A.L. Uitdenbogerd and Zobel J. Manipulation of music for melody matching. In *Proc. ACM Multimedia '98*, pages 235–240, 1998.