jSymbolic: A Feature Extractor for MIDI Files

Cory McKay and Ichiro Fujinaga

Music Technology Area, Schulich School of Music, McGill University
cory.mckay@mail.mcgill.ca and ich@music.mcgill.ca

Abstract

A library of 160 high-level features is presented along with jSymbolic, a software package that extracts these features from MIDI files. jSymbolic is intended both as a platform for developing new features as well as a tool for providing features to data mining software that can be used to automatically classify music or evaluate musical similarity.

1 Introduction

Automatic music classification and similarity analysis are areas of research that have been receiving increasing attention in both the academic and commercial spheres. Computerized machine learning and data mining tools are used to measure the similarity of different pieces of music and/or classify them based on a variety of category types, such as genre, geographical/temporal/cultural origin, mood, and listening scenario. Related techniques can be used to perform tasks such as automatic playlist generation, music recommendation and performer or composer identification.

In order to conduct such tasks it is first necessary to extract characteristic pieces of information, known as "features," from music. Features typically fall into one of three categories:

- *Low-level features:* Spectral or time-domain information extracted directly from audio signals. Most features of this type do not provide information that seems intuitively musical, but they can have significant discriminating power when processed by computers. Examples include spectral flux, zero-crossing rate and RMS.
- *High-level features:* Information that consists of musical abstractions that are meaningful to musically trained individuals. Examples include instruments present, melodic contour, chord frequencies and rhythmic density.
- *Cultural features:* Sociocultural information outside the scope of musical content itself. These usually consist of statistics that can be automatically mined from the web. Examples include playlist co-occurrence, inter-performance correlation and purchase correlations.

The majority of research projects to date on music classification and similarity analysis have focused on just one of these feature types at a time. Of the three, low-level features have received by far the most attention. This emphasis is probably due to the relative difficulty of reliably extracting the other two types of features.

Cultural features can be difficult to quantify, and it can be difficult to experimentally test their validity. It is also necessary to have a certain amount of metadata about a recording, such as song title or performer name, in order to be able to collect cultural features.

Many high-level features cannot currently be reliably extracted from audio recordings. Although high-level features can be relatively easily extracted from music recorded in symbolic formats (e.g., MIDI), which store musical events and parameters rather than actual audio samples, MIR researchers have tended to concentrate more on audio formats because they are of more interest to the general public.

This neglect of high-level features is unfortunate, as it is reasonable to expect that advances in automatic transcription will eventually enable one to convert audio recordings into symbolic formats, from which high-level features can then be extracted. An existing body of research on high-level features could then be immediately taken advantage of.

Furthermore, the musical nature of research involving high-level features can have important musicological and music theoretical value that low-level features cannot provide. For example, research has shown that features relating to instrumentation are of particular importance when distinguishing between genres (McKay and Fujinaga 2005), a result that would be difficult to achieve if one were using only low-level features.

There is a great deal of music already encoded in symbolic formats such as MIDI or Humdrum's *kern*. A high-level feature extractor that could process such recordings would make it possible to use these resources in a wide range of automated musicological studies. In addition, optical music recognition techniques already make it possible to convert written scores for which audio recordings are not available to symbolic formats from which high-level features can be extracted.

jSymbolic, the focus of this paper, is a software package designed to extract high-level features from MIDI files. It is hoped that jSymbolic will be used to realize the research potential of high-level features and to capitalize on the numerous existing symbolic recordings.

Of course, one would ideally like to make use of all three types of features. The amount of work required to do so can be daunting, however, particularly considering the tendency of researchers to each develop their own specialized systems. Although a few successful software packages have gradually come to be used by a variety of researchers, most famously the MARSYAS audio feature extractor (Tzanetakis and Cook 2000), they are typically designed to deal only with audio recordings, and cannot process symbolic recordings or mine cultural features.

A particular barrier to inter-system research has been the lack of a widely accepted file format for storing feature values. Although the Weka ARFF format (Witten and Frank 2005) is becoming a de facto standard, it is a general purpose format, and as such has a number of important limitations with respect to music (McKay et al. 2005).

Research is being pursued at McGill University to overcome limitations on cooperative research by developing software frameworks that will allow diverse researchers to develop and share new features and data mining algorithms. The first step was the development of ACE, a meta-classifier that automatically experiments with a variety of algorithms in order to find a good approach for each particular problem (McKay et al. 2005). This included the development of the ACE XML file formats for storing feature values and recording metadata, which meet the particular needs of music and allow more flexibility and expressivity than existing formats such as ARFF.

The second step is the development of specialized feature extractors for each of the three types of features. The jAudio package (McEnnis et al. 2005) extracts features from audio files, and jCultural is currently being developed to mine cultural features from the web. jSymbolic extracts high-level features from music stored in symbolic formats.

2 Designing High-Level Features

One must strike a careful balance when choosing which features to extract from recordings. From one perspective, maximizing the number of features helps to ensure that enough information is extracted to sufficiently segment different categories. A general-purpose system must be able to deal with arbitrary types of music, which means that a variety of features with wide applicability is needed. However, too many features can overwhelm classifiers. This problem is known as the "curse of dimensionality," which suggests that the number of labeled training and testing samples needed increases exponentially with the number of features. Although automated feature selection and weighting techniques can help, one must still be careful to avoid extracting too many features.

One compromise is to develop a large catalogue of features, with an emphasis on general features, and to give users the option of selecting which ones they want to extract. Users can then choose the ones that are best suited to each particular application, based on their own expertise.

High-level features have the important advantage that one can exploit existing musical research when designing them, particularly from the fields of music theory, ethnomusicology and popular musicology. The Cantometrics project (Lomax 1968), which compared several thousand songs from hundreds of different cultural groups, is an excellent example. Research on melodic contours, as discussed by scholars such as Charles Adams (1976), provides another particularly important resource.

Both Aarden and Huron (2001) and Towsey et al. (2001) have already used computers to extract high-level features for musicological studies. Additional high-level features have been developed in the course of research on symbolic genre classification (Gabura 1965; Dannenberg, Thom and Watson 1997; Chai and Vercoe 2001; Shan and Kuo 2003; Basili, Serafini, and Stellato 2004; Ponce de Leon and Inesta 2004). Further symbolic features have been proposed in a number of additional miscellaneous studies (Eerola and Toiviainen 2004; Sapp, Liu, and Selfridge-Field 2004; Kirlin and Utgoff 2005).

As a general principle when choosing features, it is generally best to concentrate on features that can be represented by relatively simple statistics, since extracted features will be processed by classification and clustering algorithms.

It can also be useful to construct intermediate representations from which further features can then be extracted. Histograms (represented as vectors) can be particularly fruitful in this respect, as useful information can be calculated from them which could be difficult to acquire directly. For example, researchers such as Brown (1993) and Tzanetakis and Cook (2002) have profitably used "beat histograms" generated using autocorrelation of note onsets to extract a variety of useful rhythmic features. Tzanetakis and Cook have also demonstrated the utility of "pitch histograms" based on both pitches and pitch classes.

In the course of developing jSymbolic, it was found that it is useful to consider two subclasses of high-level features, namely one-dimensional features and multi-dimensional features. The former each consist of a single number that represents an aspect of a recording in isolation. The latter each consist of a set of related values that have limited significance when considered individually, but together can reveal meaningful patterns. For example, the average duration of melodic arcs would be a one-dimensional feature and the bin frequencies of a histogram consisting of the relative frequency of different melodic intervals would be a multi-dimensional feature. This division into single and multi-dimensional features is useful because it makes it possible to use classifier ensembles that capitalize on the particular relatedness of the components of multi-dimensional features, such as an ensemble constructed by training a separate classifier (e.g., a neural net) on each multi-dimensional feature (McKay 2004).

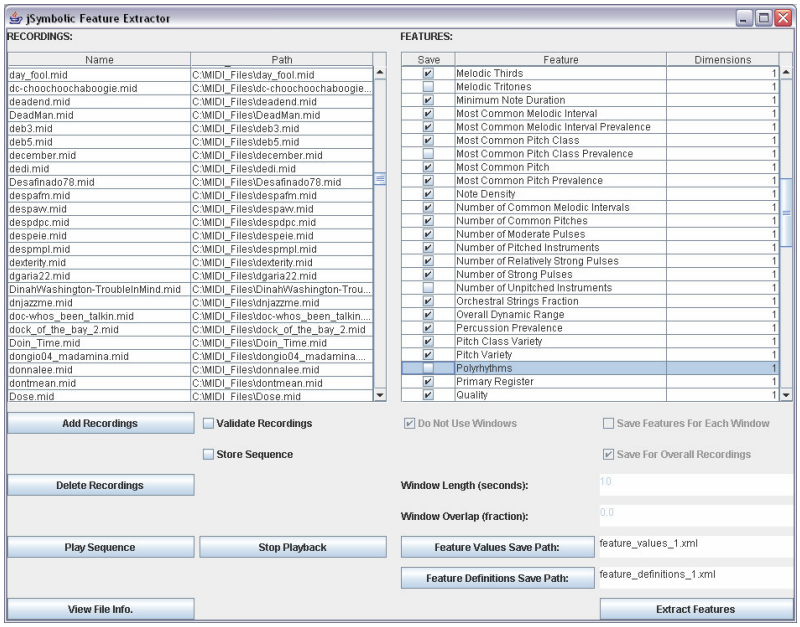

Figure 1. The jSymbolic interface

3 Features Implemented

A total of 160 features are implemented in jSymbolic, including both many original features and some features previously proposed in the sources described above. A number of intermediate representations are utilized, including beat and pitch histograms, histograms based on the instruments present, a "melodic interval histogram" that measures the frequency of various melodic intervals in each voice, a "vertical interval histogram" that measures the frequency of different vertical intervals and a "chord type histogram" that measures how often various chord types appear.

The 160 features implemented in jSymbolic were originally conceptualized during the design of the Bodhidharma genre classifier (McKay 2004). Bodhidharma placed first in all four categories of the MIREX 2005 Symbolic Genre Classification Contest (Downie 2005; Downie et al. 2005), even though Bodhidharma only includes implementations of 111 of the 160 proposed features.

The remaining 49 features are now implemented in jSymbolic, and the software is designed to make it easy to add more features in the future. Although too numerous to describe individually in this paper, jSymbolic's 160 features are each described elsewhere (McKay 2004). The features can be divided into the following seven categories:

- *Instrumentation:* What types of instruments are present and which are given particular importance relative to others? The importance of both pitched and non-pitched instruments and their interaction with each other is considered.
- *Texture:* How many independent voices are there and how do they interact (e.g., polyphonic, homophonic, etc.)? What is the relative importance of different voices?
- *Rhythm:* The time intervals between the attacks of different notes and the durations of each note are considered. What kinds of meters and rhythmic patterns are present? Is rubato used? How does rhythm vary from voice to voice?
- *Dynamics:* How loud are notes and what kinds of variations in dynamics occur?
- *Pitch Statistics:* What are the occurrence rates of different notes, in terms of both pitches and pitch classes? How tonal is the piece? What is its range? How much variety in pitch is there?
- *Melody:* What kinds of melodic intervals are present? How much melodic variation is there? What kinds of melodic contours are used? What types of phrases are used and how often are they repeated?
- *Chords:* What vertical intervals are encountered? What types of chords do they represent? How much harmonic movement is there, and how fast is it?

4 The jSymbolic Software

The jSymbolic software allows users to extract all or any subset of its 160 features from any Format 0 or 1 MIDI file. jSymbolic's Java implementation is open-source and platform-independent. jSymbolic can save features in Weka ARFF format as well as the more expressive ACE XML.

Special efforts have been made to make it easy for researchers to develop new features of their own and add

them to jSymbolic, a process that requires only a basic knowledge of Java and MIDI. Detailed knowledge of jSymbolic's implementation is unnecessary.

New features can also easily take advantage of previously implemented features, as jSymbolic automatically calculates feature dependencies when scheduling extraction. No user or developer intervention is needed. Each feature is implemented as a separate module, and new features can process MIDI data directly or make use of existing features.

jSymbolic is designed for users with varying levels of computer expertise. It therefore has a simple and easy-to-learn graphical interface (see Figure 1) and is well documented.

The jSymbolic software and documentation may be downloaded from http://sourceforge.net/projects/jmir.

5 Conclusions and Future Research

The jSymbolic software includes a library of 160 high-level features, likely the largest and most diverse high-level feature library for music currently available. This library includes many original features, and the use of a variety of intermediate representations has shown itself to be useful, as has the use of both one-dimensional and multi-dimensional features.

It is hoped that jSymbolic will be adopted not only as a tool for extracting features from music stored in symbolic music formats, but also as a platform for iteratively designing new features. Future research by the authors will therefore concentrate primarily on developing new features. There are also plans to expand jSymbolic's functionality so that it can process additional symbolic formats (e.g., kern, GUIDO, MusicXML, etc.) and extract features using overlapping and arbitrarily sized windows.

6 Acknowledgments

The generous financial support of the *Social Sciences and Humanities Research Council of Canada,* the *Canada Foundation for Innovation* and the *Fonds Québécois de la recherche sur la société et la culture* has helped to make this research possible.

References

Aarden, B., and D. Huron. 2001. Mapping European folksong: Geographical localization of musical features. *Computing in Musicology* 12: 169–83.

Adams, C. 1976. Melodic contour typology. *Ethnomusicology* 20 (2): 179–215.

Basili, R., A. Serafini, and A. Stellato. 2004. Classification of musical genre: A machine learning approach. *Proceedings of the International Conference on Music Information Retrieval.* 505–8.

Brown, J. C. 1993. Determination of meter of musical scores by autocorrelation. *Journal of the Acoustical Society of America* 94 (4): 1953–7.

Chai, W. and B. Vercoe. 2001. Folk music classification using hidden Markov models. *Proceedings of the International Conference on Artificial Intelligence*.

Cope, D. 1991. *Computers and musical style*. Madison, WI: A-R Editions.

Dannenberg, R. B., B. Thom, and D. Watson. 1997. A machine learning approach to musical style recognition. *Proceedings of the International Computer Music Conference*. 344–7.

Downie, J. S. 2005. *MIREX 2005 contest results.* Available on-line at http://www.music-ir.org/evaluation/mirex-results. Retrieved 23 February 2006.

Downie, J. S., K. West, A. Ehmann, and E. Vincent. 2005. The 2005 music information retrieval evaluation eXchange (MIREX 2005): Preliminary overview. Proceedings of the International Conference on Music Information Retrieval. 320–3.

Eerola, T., and P. Toiviainen. 2004. MIR in Matlab: The MIDI Toolbox. *Proceedings of the International Conference on Music Information Retrieval.* 22–7.

Gabura, A. J. 1965. Computer analysis of musical style. *Proceedings of the ACM National Conference*. 303–14.

Kirlin, P. B., and P. E. Utgoff. 2005. VoiSe: Learning to segregate voices in explicit and implicit polyphony. *Proceedings of the International Conference on Music Information Retrieval.* 552–7.

Lomax, A. 1968. *Folk song style and culture*. Washington, DC: American Association for the Advancement of Science.

McEnnis, D., C. McKay, I. Fujinaga, and P. Depalle. 2005. jAudio: A feature extraction library. *Proceedings of the International Conference on Music Information Retrieval*. 600–3.

McKay, C., R. Fiebrink, D. McEnnis, B. Li, and I. Fujinaga. 2005. ACE: A framework for optimizing music classification. *Proceedings of the International Conference on Music Information Retrieval*. 42–9.

McKay, C., and I. Fujinaga. 2005. Automatic music classification and the importance of instrument identification. *Proceedings of the Conference on Interdisciplinary Musicology*. CD-ROM.

McKay, C. 2004. Automatic genre classification of MIDI recordings. *M.A. Thesis*. McGill University, Canada.

Ponce de Leon, P. J., and J. M. Inesta. 2004. Statistical description models for melody analysis and characterization.. *Proceedings of the International Computer Music Conference.* 149–56.

Sapp, C. S., Y. W. Liu, and E. Selfridge-Field. 2004. Search-effectiveness measures for symbolic music queries in very large databases. *Proceedings of the International Conference on Music Information Retrieval.* 266–73.

Shan, M. K., and F. F. Kuo. 2003. Music style mining and classification by melody. *IEICE Transactions on Information and Systems* E86-D (3): 655–9.

Tagg, P. 1982. Analysing popular music: Theory, method and practice. *Popular Music* 2: 37–67.

Towsey, M., A. Brown, S. Wright, and J. Diederich. 2001. Towards melodic extension using genetic algorithms. *Educational Technology & Society* 4 (2): 54–65.

Tzanetakis, G., and P. Cook. 2002. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing* 10 (5): 293–302.

Tzanetakis, G., and P. Cook. 2000. MARSYAS: A framework for audio analysis. *Organized Sound* 4 (3): 293–302.

Witten, I. H., and E. Frank. 2005. *Data mining: Practical machine learning tools and techniques*. New York, NY: Morgan Kaufman.