Style-independent computer-assisted exploratory analysis of large music collections

Cory McKay
Schulich School of Music
McGill University
Montreal, Quebec, Canada
cory.mckay@mail.mcgill.ca

Ichiro Fujinaga
Schulich School of Music
McGill University
Montreal, Quebec, Canada
ich@music.mcgill.ca

Abstract

The first goal of this paper is to introduce musicologists and music theorists to the benefits offered by state-of-the-art pattern recognition techniques. The second goal is to provide them with a computer-based framework that can be used to study large and diverse collections of music for the purposes of empirically developing, exploring and validating theoretical models. The software presented in this paper implements techniques from the fields of machine learning, pattern recognition and data mining applied to and considered from the perspectives of music theory and musicology.

An important priority underpinning the software presented here is the ability to apply it to a much wider range of art, folk and popular musics of the world than is possible using the types of computer-based approaches traditionally used in music research. The tools and techniques presented here will thus enable exploratory research that can aid in the formation and validation of theoretical models for types of music for which such models have been elusive to date. These tools will also allow research on forming theoretical links spanning types of music that have traditionally been studied as distinct groups. A particular emphasis is placed on the importance of performing studies involving many pieces of music, rather than just a few compositions that may not in fact be truly representative of the overall corpus under consideration.

1. Introduction

Continuing advances in computer processing power and data analysis techniques are creating an environment offering great potential to the theoretical study of music. It is now possible to not only apply to music the same kinds of pattern recognition algorithms that have made tasks such as automated speech recognition and optical character recognition possible, but to do so using simple desktop or laptop computers.

Modern computer-based technology has been successfully adopted by many composers and performers for tasks such as sound synthesis, gestural control and automated or computer-assisted composition. Computer-based music analysis, however, has with only a few exceptions remained limited to traditional approaches, such as simple grammar-based techniques or string matching and searching. Although such techniques are still certainly useful and relevant, it is unfortunate that they have not been supplemented by more sophisticated approaches made available by recent advances in information science.

Researchers in the music information retrieval (MIR) research community, in contrast, have been making significant strides in applying modern pattern recognition techniques to music. This community has been rapidly developing in recent years, as demonstrated by the growth of the *International Conference on Music Information Retrieval* (ISMIR). This highly multi-disciplinary community benefits from the sharing of knowledge from fields as diverse as library sciences, electrical engineering, psychology and computer and information sciences. Unfortunately, only a few musicologists and almost no music theorists have become involved with ISMIR to date.

It is our hope that this trend will change in the future, as the musical insights of such researchers would be of great benefit to the MIR community, and a variety of MIR achievements and techniques would likewise be highly relevant to them. As convincingly argued by David Huron (1999), musicological insight and scientific empiricism can greatly complement one another. It is hoped that the technologies and software presented in this paper will help to bridge this gap by placing powerful computer-based tools for large-scale automated feature extraction and machine learning at the disposal of the music theory community, who can in turn apply and enrich these tools using their own musical expertise and experience.

An important factor contributing to the general relevance of computer-based tools to musicological inquiry is the increasing availability of source materials in digital form. Libraries and archives are continually digitizing both scores and audio recordings, and are increasingly making the results and their related metadata available online. As noted by Huron (1999), the discipline is going from a "data-poor" field to a "data-rich" field. This is making wide ranging empirical studies possible to an extent that was not previously feasible.

Although an expert human can certainly analyze one or a few pieces with far more insight and understanding than a computer, such experts are limited in the number of pieces that they can analyze in a reasonable amount of time and in the range of musics that fall within the scope of their expertise. A computer, in contrast, can process huge quantities of diverse musics hundreds of times faster than a human, and with perfect consistency.

Computer-assisted theoretical studies thus have the important advantage that they can each encompass many thousands of recordings. This breadth can reveal hidden musical insights that might not be apparent from studying just a few pieces, and can additionally allow one to empirically verify the validity of existing theoretical frameworks (e.g., Gingras & Knopke 2005). This can lead to important theoretical refinements and corrections, as well as inspire entirely new theoretical approaches and perspectives.

Although traditional manual musicological or theoretical studies involving only a few pieces of music are certainly worthwhile and of value in increasing the musical understanding of those specific pieces, the validity of generalized conclusions drawn from such research will always be questionable until verified using much larger collections of music. The practice of making generalizations based on only a few pieces was understandable in the past, given the relatively limited access to musical literature and the time constraints imposed by manual analysis. Computers and digitized music collections have now removed these constraints, however, making scientifically and statistically valid studies feasible. Increasing agreement with this perspective among music researchers in the arts and humanities has likely contributed to the popularity and effectiveness of automated music analysis tools such as Humdrum (Huron 2002).

As noted above, it is now possible to supplement such existing software-based music analysis tools with techniques drawn from the most recent developments in modern pattern recognition and data mining technology. Existing music analysis software packages have essentially served as aids allowing theorists and musicologists to automate the types of tasks that they have traditionally performed manually. This is in no way meant to diminish the worth of such tools and approaches as, indeed, they are of proven value, and offer a number of benefits that pattern recognition techniques do not, just as pattern recognition techniques offer advantages that traditional analysis techniques do not.

Research involving traditional analysis software has typically incorporated assumptions that, while appropriate for the limited studies for which they were intended, nonetheless ultimately limited the types of music to which they could be applied if one were of a mind to expand the scope of the studies. Although powerful tools such as Humdrum, for examples, can and have been applied to a wide variety of very different musics, each such application has involved adaptations that were specific to the type of music under consideration.

Ideally, one would like to have one single software system that could be applied to classical music, jazz and a wide variety of popular and traditional musics, including cross-disciplinary studies that span diverse types of music. Furthermore, one would like to be able to use this software without needing to make any manual adjustments or adaptations in order to deal with different types of music. The types of pattern recognition techniques used by the software presented in this paper make this possible.

Musical research involving modern machine learning algorithms also has the benefit of providing researchers with a fresh perspective on music, as models learned by such algorithms can avoid the potentially misleading ingrained assumptions and biases that humans invariably develop, despite their best efforts. Human researchers might unconsciously reject potentially valuable paths of inquiry because of such prejudices, whereas a computer would not.

Machine learning also enables computers to consider far more features (i.e., musical characteristics) at a time than humans, as well as more complex interrelationships between them. This can result in the discovery of sophisticated and theoretically valuable patterns and relationships that might not be apparent to human analysts.

It is, of course, recognized here that machine learning is not a substitute for human researchers, nor for traditional analysis software such as Humdrum. It is highly improbable that a computer will independently evolve any perfect musical model on its own. Machine learning and pattern recognition techniques can, however, reveal insights that might otherwise be obscured, and can perturb human researchers out of ideological ruts that they may have fallen into. This and the issues raised above are discussed in further detail in Section 6.

One final advantage of machine learning-based approaches is that they can be used to analyze features from audio directly, not just from symbolic representations such as scores or MIDI files. This is extremely useful not only in analyzing music for which no symbolic representation is available, including types of music with no written tradition, but also for analyzing performance practices that are not typically encapsulated by symbolic representations. There is not sufficient space to discuss audio-based MIR research in any depth here, other than to say that it is a very active field of inquiry at ISMIR, and that the jSymbolic feature extractor discussed in this paper has an analogue for processing audio recordings (McEnnis, McKay & Fujinaga 2006).

Section 2 of this paper discusses general approaches to feature extraction. Section 3 introduces jSymbolic, a software tool that the authours have developed for extracting features from MIDI files. Section 4 presents some basic ideas relating to music and machine learning, and Section 5 provides an overview of the Autonomous Classification Engine (ACE), a software system developed by the authours to make sophisticated pattern recognition techniques available to researchers in the arts and humanities. Section 6 outlines a variety of specific ways in which pattern recognition techniques are relevant to musicologists and theorists. Finally, Section 7 presents some overall conclusions.

2. Feature extraction

An essential part of realizing the goals discussed in Section 1 is the formalization and implementation of a large set of "features" that can be extracted from arbitrary types of music. The term "feature" refers to any characteristic or quality that may be measured and used to describe or characterize a piece of music. For example, measures of rubato or the amount of chromatic motion in a piece could both be features.

Features serve as the input to machine learning algorithms, and any such algorithm can only perform well if provided with features that capture a sufficient amount of relevant information.

There are three general types of features that can be automatically collected by computers and used as inputs to music-oriented pattern recognition systems:

- **Cultural features:** Sociocultural information outside the scope of musical content itself. These often consist of statistics that can be automatically mined from the web, such as cooccurrences of certain words.
- **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.
- **High-level features:** Information that consists of musical abstractions that are meaningful to musically trained humans.

This paper will focus on high-level features, as they are the most obviously relevant to music theory. The musicological importance of cultural features and the practical utility of low-level feature certainly make them very useful for other types of musical research, however.

An obvious source of inspiration when designing features is existing theoretical and musicological research. However, one must be careful when consulting such sources to avoid relying too heavily on features that are intrinsically linked to particular theoretical frameworks. Although some such features can be useful, too many will limit the types of music to which a system can be applied and will compromise the ability of the system to do objective exploratory research. For example, feature sets based too heavily on chord progressions could incorporate too many assumptions relating to tonal harmony to be applicable to types of music that do not operate on a tonal basis.

It is particularly important to avoid those features that are built upon highly sophisticated theoretical constructs. Although Schenkerian analysis could be used to produce a variety of features, for example, such features would each be limited by the applicability of the Schenkerian system to different types of music, as discussed above. Furthermore, sophisticated analytical methods often involve a high degree of subjectivity, as demonstrated by the fact that different experts often generate different analyses of a single piece. The subjectivity of features derived from such systems would undermine the consistency that should ideally be characteristic of automatic feature extraction. Automatic analysis based on sophisticated theoretical constructs can also be computationally expensive, and is in some cases an unsolved problem. Avoiding features based on sophisticated theoretical models is therefore a good general strategy.

It can also be useful to include features that encapsulate types of information that are traditionally given a relatively minor role in music analysis. So, while traditionally important features based on pitch class frequencies, melodic movement and chord progressions certainly should be included in the feature sets that are used, alternatives features that emphasize dynamics, rhythm and instrumentation, for example, should also play an important role.

Existing research by ethnomusicologists and popular musicologists can be particularly useful in designing features, as researchers in these fields have traditionally tended to take a more empirical approach that is less reliant on particular analytical models. The Cantometrics project (Lomax 1968), which compared several thousand songs from hundreds of different cultural groups, is a good example of this kind of research, although it did suffer from some methodological flaws and questionable anthropological assumptions. Research involving melodic contours (e.g., Adams 1976) can also be valuable, although it should not be relied on too heavily, as such an approach is not necessarily applicable to all types of music.

There are a number of additional musicological sources that can provide useful features (e.g., Tagg 1982; Cope 1991; LaRue 1992; Temperley 2001), although such work typically (but not always) emphasizes manual rather than automatic feature extraction. Those few musicological research projects that have actually implemented automatic feature extraction on computers have generally made assumptions that, while appropriate for their purposes, limit their general applicability. For example, both Aarden and Huron (2001) and Towsey et al. (2001) considered only melodic features.

A number of research projects from the field of MIR provide additional valuable sources of features, particularly with respect to automatic 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 features have been proposed in a number of miscellaneous studies (Eerola and Toiviainen 2004; Sapp, Liu, and Selfridge-Field 2004; Kirlin and Utgoff 2005).

As a general principle, it is best to concentrate primarily on features that can be represented by simple numbers or small vectors. Such features can be more easily processed by machine learning algorithms, and using them helps to avoid the temptation of developing excessively complex features that incorporate too many theoretical assumptions.

Simple statistical techniques such as calculations of means and standard deviations are useful in making it possible to capture overall characteristics of pieces, including how they change. This is important in helping to ensure that one does not miss the forest for the trees, as it were, although some features that consider specific local behaviour are also important. Histograms can serve as a particular useful statistical tool, as they permit an intermediate representation of music to which additional techniques such as peak picking can be applied in order to arrive at further features.

For example, researchers such as Brown (1993) have proposed “beat histograms” generated using autocorrelation of note onsets. Tzanetakis and Cook (2002) have successfully used both beat histograms and “pitch histograms” as sources of features.

Autocorrelation involves comparing a signal with versions of itself delayed by successive intervals. This yields the relative strength of different periodicities within the signal. In terms of musical data, autocorrelation allows one to find the relative strength of different rhythmic pulses. In the case of MIDI, this can be calculated based on *Note On* messages:

$$autocorrelation[lag] = \frac{1}{N} \sum_{n=0}^{N-1} x[n]x[n-lag]$$

where n is the input sample index (in MIDI ticks), N is the total number of MIDI ticks, x is the sequence of MIDI ticks and *lag* is the delay in MIDI ticks ($0 \leq lag < N$). The value of x[n] can be made proportional to the velocity of *Note Ons*. This ensures that beats are weighted based on the strength with which notes are played.

This autocorrelation function can be applied repeatedly to each MIDI sequence with different values of lag. These lag values corresponded to both rhythmic periodicities as well as bin labels in beat histograms, and the autocorrelation values can provide the magnitude value for each bin.

For example, consider the beat histograms extracted from MIDI representations of *I Wanna Be Sedated* by the punk band The Ramones and *'Round Midnight* by the jazz performer and composer Thelonious Monk, as shown in Figures 1 and 2 respectively. It is clear that *I Wanna Be Sedated* has significant rhythmic looseness, as demonstrated by the spread around each peak, each of which represent a strong beat periodicity. It also has several strong beats, including ones centred at 55, 66, 82, 111 (the tempo) and 164 beats per minute, the latter two of which are harmonics of 55 and 82 beats per minute. *'Round Midnight,* in contrast, has one very strong beat at 76 beats per minute, the tempo of the piece, and a wide range of much lower level beat strengths.

Histograms such as these can sometimes be used directly as features. Alternatively, a wide variety of features consisting of single values can be calculated from them, such as the number of strong peaks, the relative strengths of the highest peaks, the locations and harmonicity of peaks, the local spread around peaks and the relative contribution of bins not associated with peaks.

With such histograms in mind, 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 piece 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 a vector representation of the bin frequencies of a histogram portraying 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 (i.e., groups of models each developed using machine learning) that capitalize on the particular relatedness of the components of multi-dimensional features. For example, it was found experimentally that training a separate neural network on each multi-dimensional feature and a k-nearest neighbour classifier on all one-dimensional features improved results when performing automatic musical genre classification (McKay 2004).

One must strike a careful balance when choosing which features to provide as input to machine learning algorithms. From one perspective, maximizing the number of available features helps to ensure that sufficient information is extracted to perform the tasks that one is interested in. However, too many features can overwhelm pattern recognition algorithms, a problem that is known in machine learning as "the curse of dimensionality." A good compromise is to develop a large catalogue of features, with an emphasis on general features. Researchers can then choose the features that are best suited to each particular application, either based on their own expertise or using automated dimensionality reduction techniques (see Section 4).

3. jSymbolic

jSymbolic is a Java-based software package that automatically extracts features from symbolic recordings. It is intended for users with a range of computer proficiency levels, and has an easy to use graphical interface (Figure 3).

jSymbolic can currently extract a total of 160 high-level features, far more than any other automated symbolic feature extraction system known to the authours. The term "symbolic" in general refers to abstract musical representations, such as scores, MIDI files or Humdrum kern files, but not audio files.

jSymbolic is open source, and is designed in such a way as to make it easy to add additional features in the future. Each feature is designed as a modular entity, but also has access to the values of other features, making it a simple matter to iteratively build related libraries of features if desired.

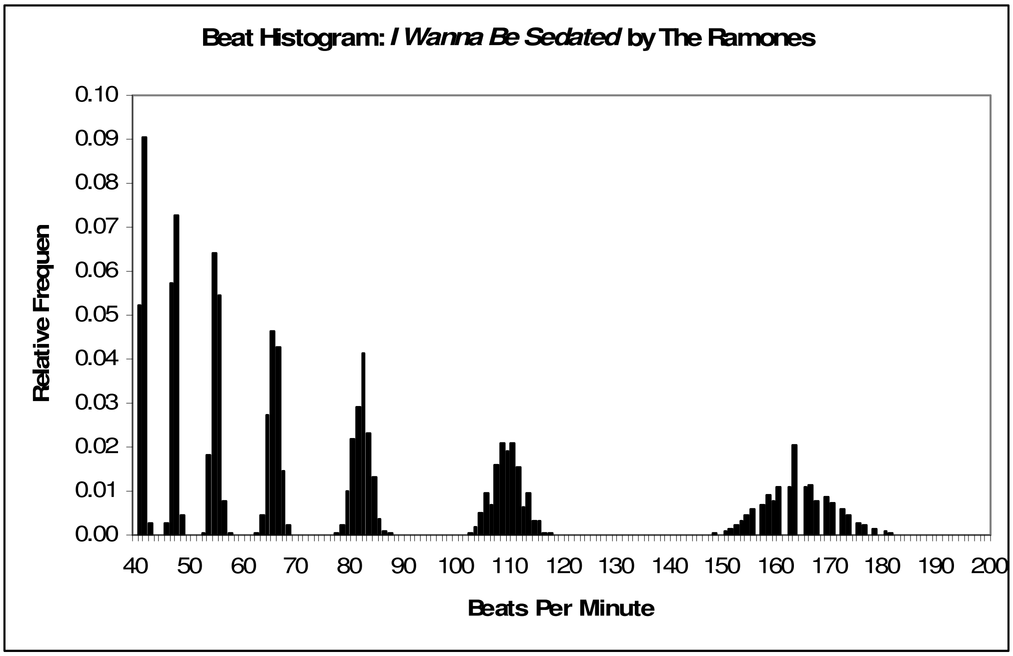


Figure 1: Beat histogram for *I Wanna Be Sedated* by the punk band The Ramones. Each bin corresponds to a rhythmic periodicity in the music. The vertical scale specifies the relative strength of each periodicity as calculated using autocorrelation of note onsets.

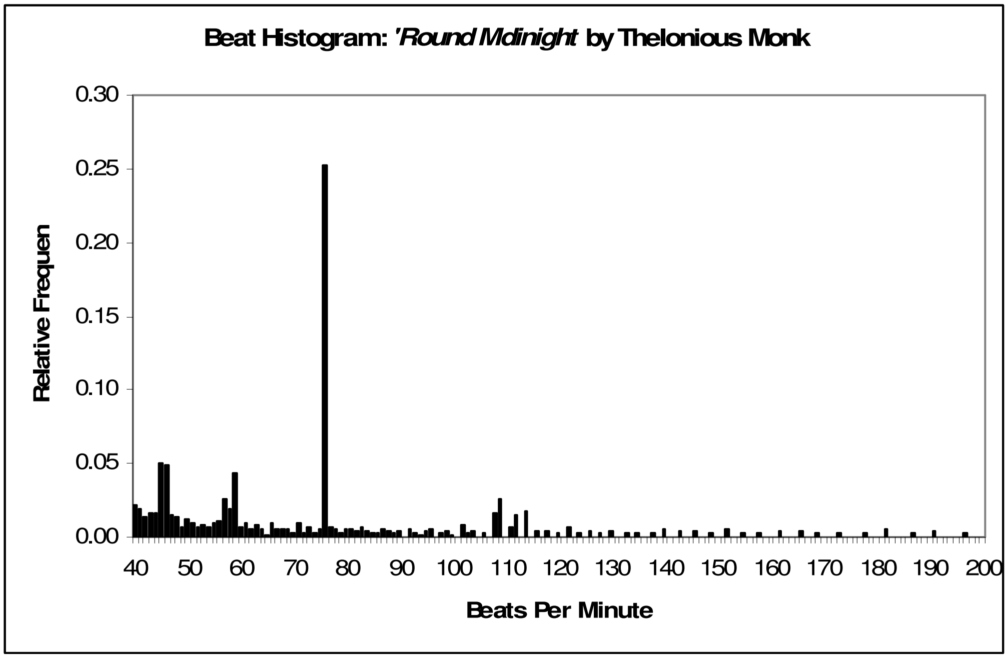
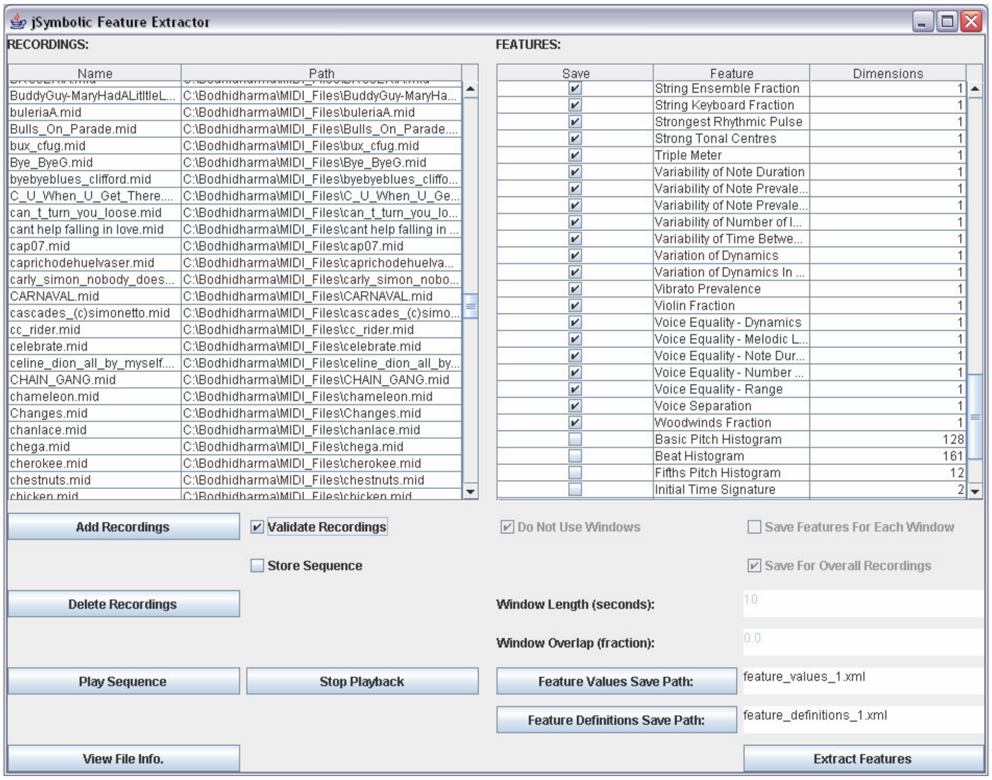


Figure 2: Beat histogram for *'Round Midnight* by Thelonious Monk. Each bin corresponds to a rhythmic periodicity in the music. The vertical scale specifies the relative strength of each periodicity as calculated using autocorrelation of note onsets.



RECORDINGS:

Name	Path
BuddyGuy-MaryHadALitltleL...	C:\Bodhidharma\MIDI_Files\BuddyGuy-MaryHa...
buleriaA.mid	C:\Bodhidharma\MIDI_Files\buleriaA.mid
Bulls_On_Parade.mid	C:\Bodhidharma\MIDI_Files\Bulls_On_Parade....
bux_cfug.mid	C:\Bodhidharma\MIDI_Files\bux_cfug.mid
Bye_ByeG.mid	C:\Bodhidharma\MIDI_Files\Bye_ByeG.mid
byebyeblues_clifford.mid	C:\Bodhidharma\MIDI_Files\byebyeblues_cliffo...
C_U_When_U_Get_There....	C:\Bodhidharma\MIDI_Files\C_U_When_U_Ge...
can_t_turn_you_loose.mid	C:\Bodhidharma\MIDI_Files\can_t_turn_you_lo...
cant help falling in love.mid	C:\Bodhidharma\MIDI_Files\cant help falling in ...
cap07.mid	C:\Bodhidharma\MIDI_Files\cap07.mid
caprichodehuelvaser.mid	C:\Bodhidharma\MIDI_Files\caprichodehuelva...
carly_simon_nobody_does...	C:\Bodhidharma\MIDI_Files\carly_simon_nobo...
CARNAVAL.mid	C:\Bodhidharma\MIDI_Files\CARNAVAL.mid
cascades_(c)simonetto.mid	C:\Bodhidharma\MIDI_Files\cascades_(c)simo...
cc_rider.mid	C:\Bodhidharma\MIDI_Files\cc_rider.mid
celebrate.mid	C:\Bodhidharma\MIDI_Files\celebrate.mid
celine_dion_all_by_myself....	C:\Bodhidharma\MIDI_Files\celine_dion_all_by...
CHAIN_GANG.mid	C:\Bodhidharma\MIDI_Files\CHAIN_GANG.mid
chameleon.mid	C:\Bodhidharma\MIDI_Files\chameleon.mid
Changes.mid	C:\Bodhidharma\MIDI_Files\Changes.mid
chanlace.mid	C:\Bodhidharma\MIDI_Files\chanlace.mid
chega.mid	C:\Bodhidharma\MIDI_Files\chega.mid
cherokee.mid	C:\Bodhidharma\MIDI_Files\cherokee.mid
chestnuts.mid	C:\Bodhidharma\MIDI_Files\chestnuts.mid
chicken.mid	C:\Bodhidharma\MIDI_Files\chicken.mid

FEATURES:

Save	Feature	Dimensions
☑	String Ensemble Fraction	1
☑	String Keyboard Fraction	1
☑	Strongest Rhythmic Pulse	1
☑	Strong Tonal Centres	1
☑	Triple Meter	1
☑	Variability of Note Duration	1
☑	Variability of Note Prevale...	1
☑	Variability of Note Prevale...	1
☑	Variability of Number of I...	1
☑	Variability of Time Betwe...	1
☑	Variation of Dynamics	1
☑	Variation of Dynamics In ...	1
☑	Vibrato Prevalence	1
☑	Violin Fraction	1
☑	Voice Equality - Dynamics	1
☑	Voice Equality - Melodic L...	1
☑	Voice Equality - Note Dur...	1
☑	Voice Equality - Number ...	1
☑	Voice Equality - Range	1
☑	Voice Separation	1
☑	Woodwinds Fraction	1
□	Basic Pitch Histogram	128
□	Beat Histogram	161
□	Fifths Pitch Histogram	12
□	Initial Time Signature	2

☑ Validate Recordings

□ Store Sequence

☑ Do Not Use Windows

□ Save Features For Each Window

☑ Save For Overall Recordings

Window Length (seconds): 1.0

Window Overlap (fraction): 0.0

Feature Values Save Path: feature_values_1.xml

Feature Definitions Save Path: feature_definitions_1.xml

Figure 3: The jSymbolic feature extractor interface.

The jSymbolic software currently only extracts features from MIDI files. Although the shortcomings of the MIDI format are well documented, a much wider range of styles and genres of popular, art and folk musics are available in MIDI than in any other format. This made it possible to collect a diverse experimental training and testing library consisting of 950 MIDI recordings spanning 38 different types of music (McKay 2004).

This is in no way meant to minimize the importance of alternative symbolic formats such as MusicXML, GUIDO or Humdrum's variants. Such formats have a number of advantages over MIDI, and there is a large body of work in certain particular styles that has been encoded using them. It is therefore a priority in future development to incorporate functionality into jSymbolic for parsing additional symbolic formats. In the meantime, converters can be used to translate music in alternative formats into MIDI.

jSymbolic can save extracted features as both ACE XML and Weka ARFF files (see Section 5). These file formats are standards that can be parsed by a variety of machine learning systems. Tools also exist for converting ARFF files to a variety of simple delimited text file formats.

Although too numerous to describe individually in this paper, each of jSymbolic's features are described in detail elsewhere (McKay 2004). Some of these features are original and some are derived from the sources described in Section 2. In general, jSymbolic's 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 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 metrical structures 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 present? What types of chords do they represent? How much harmonic movement is there and how fast is it?

As discussed in Section 2, these features consist of both one-dimensional and multi-dimensional features, and make use of a variety of intermediate representations. These include beat histograms, absolute pitch histograms, pitch class histograms, wrapped pitch class histograms, several histograms based on the instruments present and the relative roles that they play, "melodic interval histograms" that measure the frequency of various melodic intervals in each voice, "vertical interval histograms" that measures the frequency of different vertical intervals and "chord type histograms" that measure how often various chord types appear. These intermediate representations are well documented elsewhere (McKay 2004).

Figures 4 and 5 show the values of twenty sample features extracted from two measures each of a Chopin nocturne and a Mendelssohn piano trio. A comparison of the two examples and their features makes it apparent how such features can be useful. For example, the *Average Note To Note Dynamic Change, Overall Dynamic Range* and *Variation of Dynamics* features demonstrates the greater range in dynamics of the nocturne, the *Note Density* feature demonstrates the greater number of notes per second of the trio, the *Orchestral Strings Fraction* feature indicates that strings play roughly half the notes in the trio but are absent in the nocturne and the *Variability of Note Duration* feature shows that this portion of the nocturne has more rhythmic variety than the trio. More traditional features are also present, such as the *Chromatic Motion* feature, which demonstrates that this portion of the trio has more chromatic motion, or the *Range* feature, which shows that the lowest and highest notes of the nocturne span a greater interval. Although not necessarily significant when considered individually, pattern recognition systems can simultaneously examine many such features in order to find meaningful patterns.

4. Machine learning

Once features have been extracted from music, machine learning and pattern recognition algorithms can then be used to process them.

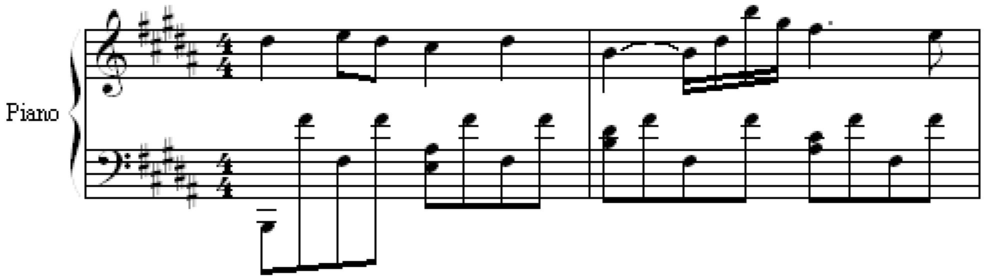
Machine learning refers to techniques that allow a computer to automatically build ("learn") internal models that map particular stimuli (i.e., feature sets) corresponding to individual examples ("instances") to particular outputs. With respect to music, these outputs are typically musically meaningful categories or ontological structures, and a recording or score might represent an instance.

The notion of classification into categories (or "classes") is important in machine learning. A class can have a wide spectrum of possible meanings dependant on the subject of interest, and can be anything from a pitch or rhythmic duration to a historical period or compositional style.

One of the key advantages of machine learning is that there is no need to manually specify the details of the model to be learned, or even necessarily to have any a priori knowledge of the model at all. This is because the learning algorithms construct models automatically. This means that machine learning is useful not only in the mappings performed by the learned models, but also in terms of the insights that the models themselves can provide.

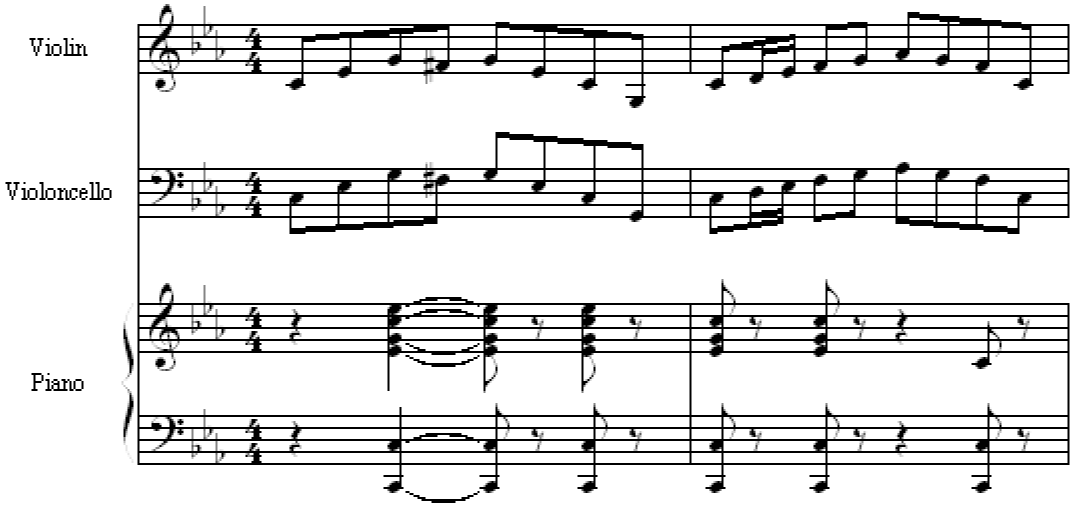
There are three general pattern recognition paradigms, each with its strengths and weaknesses:

- **Expert Systems:** These systems use predefined rules to process features and arrive at classifications. These rules are typically specified manually by humans, and do not usually utilize machine learning.
- **Supervised Learning:** These systems attempt to formulate their own classification rules by using machine learning techniques to train on model labelled instances. Previously unseen instances can then be classified into one or more of the candidate classes using the rules automatically generated during training.
- **Unsupervised Learning:** These systems cluster unlabelled instances based on similarities and differences that they themselves perceive.



```
Average Note To Note Dynamics Change: 6.03              Orchestral Strings Fraction: 0
Chromatic Motion: 0.0769                                Overall Dynamic Range: 62
Dominant Spread: 3                                      Pitch Class Variety: 7
Harmonicity of Two Strongest Rhythmic Pulses: 1         Range: 48
Importance of Bass Register: 0.2                        Relative Strength of Most Common Intervals: 0.5
Interval Between Strongest Pitch Classes: 3             Size of Melodic Arcs: 11
Most Common Pitch Class Prevalence: 0.433               Stepwise Motion: 0.231
Note Density: 3.75                                      Strength of Strongest Rhythmic Pulse: 0.321
Number of Common Melodic Intervals: 3                   Variability of Note Duration: 0.293
Number of Strong Pulses: 5                              Variation of Dynamics: 16.4
```

Figure 4: Twenty sample features extracted from the first two measures of Fryderyk Chopin's *Nocturne in B, Op. 32, No. 1*. The features and their units are each defined elsewhere (McKay 2004), and would typically be extracted over the entire piece, not just two measures. Performance information relating to dynamics and rubato beyond the contents of the score is often available in MIDI files.



```
Average Note To Note Dynamics Change: 1.46              Orchestral Strings Fraction: 0.56
Chromatic Motion: 0.244                                 Overall Dynamic Range: 22
Dominant Spread: 2                                      Pitch Class Variety: 7
Harmonicity of Two Strongest Rhythmic Pulses: 1         Range: 39
Importance of Bass Register: 0.373                      Relative Strength of Most Common Intervals: 0.8
Interval Between Strongest Pitch Classes: 7             Size of Melodic Arcs: 7.27
Most Common Pitch Class Prevalence: 0.39                Stepwise Motion: 0.439
Note Density: 29.5                                      Strength of Strongest Rhythmic Pulse: 0.173
Number of Common Melodic Intervals: 6                   Variability of Note Duration: 0.104
Number of Strong Pulses: 6                              Variation of Dynamics: 5.98
```

Figure 5: Twenty sample features extracted from measures 10 and 11 of the first movement of Felix Mendelssohn's *Piano Trio No. 2 in C minor, Op. 66*. The features and their units are each defined elsewhere (McKay 2004), and would typically be extracted over the entire piece, not just two measures. Performance information relating to dynamics and rubato beyond the contents of the score is often available in MIDI files.

```
if ( parallel_fifths == 0 &&

     landini_cadences == 0 )
       then composer → Palestrina
else composer → Machaut
```

Figure 6: Pseudocode for a simple expert system designed to distinguish between the music of Guillaume de Machaut and Giovanni Pierluigi da Palestrina. If a piece has neither parallel fifths nor Landini cadences, then the system concludes that it is by Palestrina. If there are either parallel fifths or Landini cadences, then the system concludes that the piece is by Machaut. Although unrealistically simple systems such as this are easy to implement, both the necessary pre-existing knowledge and the necessary sophistication of expert systems rapidly grows to unmanageable proportions when one must deal with more realistic problems involving composers with more subtle distinctions between them.

Expert systems have the advantage that the existing knowledge of musicologists and theorists can be incorporated directly. Unfortunately, such systems typically only work well when applied to problems that can be easily formalized using only a few simple heuristics. Aside from limited and specialized applications, music is in general too sophisticated and the theory surrounding it too sparse, inconsistent and contradictory for expert systems to be viably applied. This is readily apparent when one considers the range of popular, art and folk musics of the world and the variety of theoretical structures or lack thereof relating to each corpus.

Supervised and unsupervised approaches are more promising in general with respect to music. As discussed previously, insights gained form exploratory research utilizing such technologies can help lead to the iterative construction of theoretical frameworks. The resultant theory can then potentially be formalized into expert systems, making them increasingly viable in specialized areas of musical inquiry. For example, expert systems can and have been successfully applied to baroque counterpoint, but are not yet as applicable to musics that are less formally understood.

Supervised learning is most useful when one has a set of existing labels of any kind that are known to be associated with particular musical instances. Supervised learning then allows one to train a model using these pre-labelled instances so that the system learns how to assign correct labels to unlabelled instances.

Unsupervised learning is more appropriate when one is unable or unwilling to impose particular pre-existing labels on instances, and prefers to have a system autonomously cluster the instances into groups that it finds to be similar based on the dimensions specified by the extracted features.

The example of an imagined musicological research project involving the attribution of a set of historical pieces whose composers are unknown can be used to illustrate the relative merits of the three pattern recognition paradigms. An expert system would be appropriate if the stylistic practices of each candidate composer are well understood and can be easily formalized into heuristics (e.g., Figure 6). A supervised learning approach would be suitable if one has a number of pieces known to be by each of the candidate composers, as these could be used to train the system so that it could learn to automatically recognize the characteristics of each composer (e.g., Figure 7). Finally, an unsupervised approach would be suitable if one does not have a set of candidate composers, but would like to segment the music into groups that are likely to each correspond to a different composer (e.g., Figure 8).

There are a wide variety of different algorithms that can be used to implement supervised or unsupervised learning, and each of these also has its own strengths and weaknesses. For example, Bayesian classifiers can perform extremely well, but require significant statistical knowledge of the data that one is dealing with in order to operate at their best. Nearest neighbour classifiers are simple and fast, but cannot infer logical relationships. Artificial neural networks can learn sophisticated relationships, but can take a long time to train. Tree induction algorithms are not always as effective as other methods, but the inferences that they learn are more easily interpretable by humans than the models provided by most other algorithms. Hidden Markov models are good at modelling how instances change with time, but are less appropriate when dealing with independent feature sets.

There are many further algorithms that can be used as well and, to further complicate matters, classifiers can also be combined into ensembles, each of which also have their own strengths and weaknesses. How one can go about choosing the best algorithms to use in particular music research projects is addressed in Section 5.

Feature selection and weighting, which are examples of “dimensionality reduction” techniques,

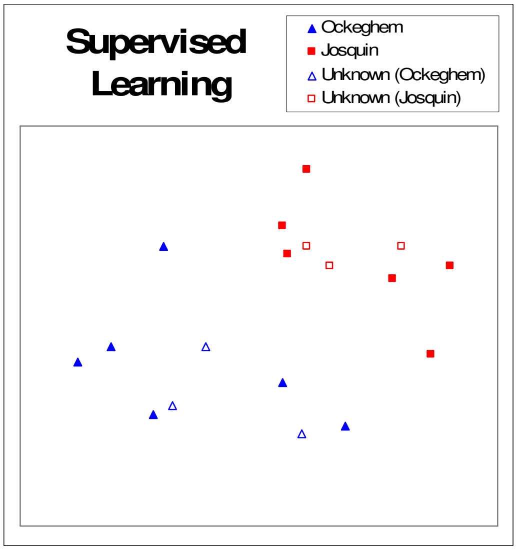


Figure 7: An example of supervised learning. In this case, many features are projected into two dimensions so that they can be more easily displayed. The problem is to teach the system to distinguish between compositions by Johannes Ockeghem (triangles) and Josquin Desprez (squares). The system is first trained by providing it with labelled compositions that are known to be by each composer (the filled in triangles and squares). The system is then given six unlabelled compositions (the empty triangles and squares). Based on the examples that the system was trained on, the system identifies three compositions as being by Ockeghem and three as being by Josquin. Note that, unlike the expert system in Figure 6, it is not necessary to explicitly specify any of the characteristics of the composers themselves when training the system, since the system learns these characteristics itself from the examples.

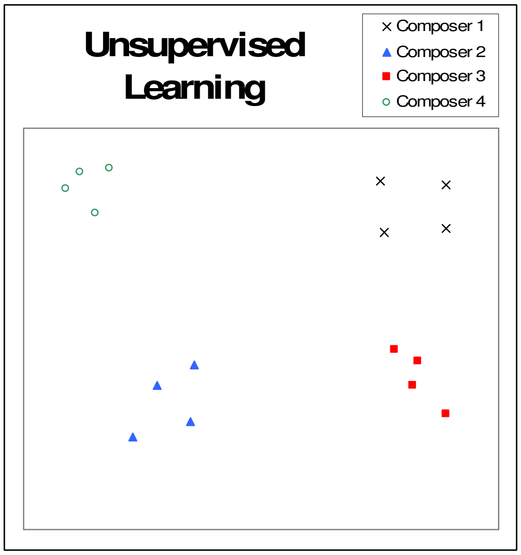


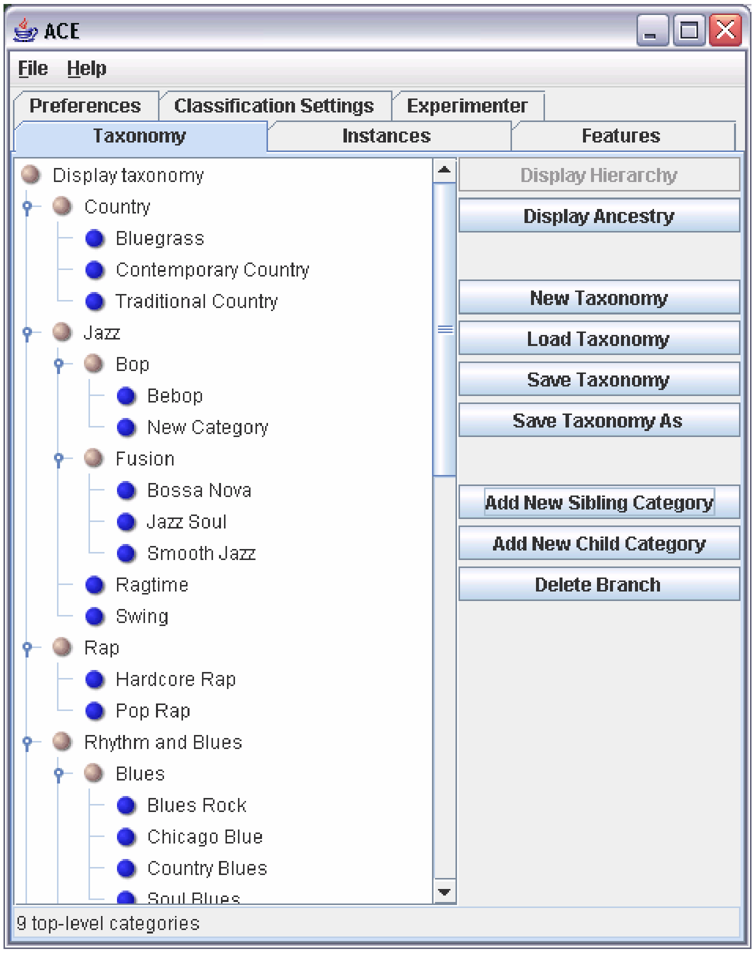
Figure 8: An example of unsupervised learning. As in Figure 7, many features are once again projected into two dimensions so that they can be more easily displayed. The problem is to teach the system to separate a body of 16 anonymous pieces, for which one has no reliable information at all as to their composers. This means that supervised learning is not an option, because no labelled training samples are available. The unsupervised learning algorithm examines the 16 pieces, and separates them into four groups based on their relative differences and similarities, with each group corresponding to a different composer.

are other areas of research that are important in musical machine learning, as they provide a means for emphasizing to classifiers those features that are most salient. It is important to note that feature weighting has value beyond simply improving classification performance, as examinations of those features that are most significant in particular situations can have important musicological relevance. A variety of techniques are available for performing feature selection and weighting, including classical forward-backward selection as well as genetic algorithms that use principles of survival of the fittest to "evolve" high-quality feature sets.

There are a variety of complementary resources that can be consulted for further information on machine learning, pattern recognition and data mining (Duda, Hart & Stork 2001; Hastie, Tibshirani, & Friedman 2001; Alpaydin 2004; Kuncheva 2004; Witten & Frank 2005).

5. Autonomous classification engine

Machine learning and pattern recognition are subtle and sophisticated areas that require a high level of technical knowledge and experience in order to be exploited to their full potential. Choosing the best algorithm(s) to use for a particular application and effectively parameterizing



9 top-level categories

Figure 9: The ACE classification framework's graphical interface.

them are not tasks that can be optimally performed by inexperienced researchers.

The Autonomous Classification Engine (ACE) was developed as a solution to this problem (McKay et al. 2005). Given a set of feature values from a feature extractor such as jSymbolic, ACE automatically performs experiments with a variety of classifiers, classifier parameters, classifier ensembles and dimensionality reduction techniques in order to arrive at an effective configuration for the particular problem at hand.

ACE may also be used directly as a classifier. Once appropriate classifier(s) have been chosen, whether through automatic ACE optimization or using pre-existing knowledge, users need only provide ACE with feature vectors and model classifications (in the case of supervised learning). ACE then trains itself and presents users with trained classifier(s).

ACE is designed to facilitate classification for those new to pattern recognition as well as provide flexibility for those with more experience. Even those researchers with a great deal of experience in pattern recognition must often resort to experimentation, and the meta-learning approach used by ACE automates this process for them.

Most significantly for the musicological and music theoretical communities, ACE makes sophisticated machine learning techniques available without requiring any understanding of how the underlying algorithms work. ACE has a simple interface to make it easily usable by those with even the most limited technical background, and the final touches are currently being put on a graphical user interface for the software (Figure 9).

An important advantage of ACE is that, like jSymbolic, it is open source and freely distributable. This means that individual researchers are free to modify and customize it as they see fit. ACE is also implemented in Java, which means that the framework is portable among operating systems and is easy to install.

In all, ten supervised classification and dimensionality reduction algorithms are currently implemented in ACE, and there are plans to incorporate many more, including a variety of unsupervised algorithms. ACE is built upon the Weka framework (Witten & Frank 2005), which means that algorithms developed for Weka can easily be added to ACE. ACE can read both Weka ARFF files and files in custom designed XML-based formats that allow far more expressivity with respect to ground truth (model classifications) and features than traditional machine learning file formats.

In terms of performance validation, ACE has outperformed existing systems at classifying standard UCI test datasets, percussion instruments and beatboxing samples in benchmarking tests (McKay et al. 2005).

6. Applications

As emphasized in Section 1, the feature extraction and pattern recognition approaches discussed in this paper are intended to facilitate musicological and music theoretical research involving diverse types of music and very large bodies of musical data. A particular stress has been placed on the potential for performing exploratory research and verification of existing theoretical models. This section introduces specific examples of such research. Of course, these applications only represent a small subset of the full range of possible research topics.

To begin with a very simple example, a feature could be implemented that describes the prevalence of parallel fifths in a piece. This feature could be input to an unsupervised clustering algorithm, which would organize compositions into groups based on how often parallel fifths occur. One would expect the music of J. S. Bach, for example, to be in a group with little or no parallel fifths, and the music of Green Day (a punk band) to be in a group with many parallel fifths. If the algorithm segments musical examples along these lines, then this would confirm certain theoretical models concerning Baroque counterpoint and the types of chord voicings and progressions found in punk music. If not, then one would be led to question these same models. Additional types of music, such as jungle electronic dance music or Irish jigs, for example, could also be input to the model to see how they compare to other types of music in the spectrum of parallel fifths.

This example is, of course, highly simplistic, and commonly used tools such as Humdrum could just as easily be used to perform equivalent tasks. However, consider now a case where one wishes to consider hundreds of features and how they interrelate to one another, rather than just a single feature.

One would no longer have such clear expectations of how different types of music would be clustered or classified, nor would it likely be practically feasible to manually formalize relationships between features and groups of music. This makes the lack of applicability of expert system or manual query-based approaches to such problems readily apparent, particularly considering the potentially highly convoluted feature dependencies that would likely exist.

The additional problem arises of how one could represent the results of analyses of feature spaces involving hundreds of dimensions using either manual or traditional computer-based methods. Even statistical tools such as cooccurrence and correlation analyses are limited in how well they can represent such results in ways that are musically meaningful.

Of course, traditional manual and computer-based analysis techniques have not typically involved hundreds of features such as this, perhaps specifically because of these problems. Simplifications have therefore been unavoidably necessary in the past. However, this does not mean that such simplifications should be propagated now that more powerful alternatives are available.

It is obvious that considerations relating to harmony, rhythm, melody, dynamics, instrumentation and other factors are all essentially intertwined. A certain melodic progression might be appropriate only when played softly, for example, or perhaps only when certain notes fall on weak beats. A certain chord might sound better using a particular instrumentation than another. Any attempt to isolate one set of musical parameters from all others, while traditionally an unavoidable simplification in order to make any analysis at all possible, unavoidably results in at least some level of corruption of results due to the failure to fully consider music in a fully holistic sense.

One of the important advantages of the machine learning approach is that it enables computers to consider hundreds of features at a time, as well as the interrelationships between them. There is no requirement to formalize the possible relationships between features, as these are automatically learned by machine learning algorithms, nor to incorporate assumptions into systems that would contaminate the objectivity of a model. Furthermore, pattern recognition and dimensionality reduction algorithms allow results to be meaningfully represented in low dimensional space as self-organized clusters or specifically labelled categories. Some techniques, such as decision tree algorithms, also allow empirically learned rules and dependencies to be output directly.

Approaches based on machine learning thus have the important advantages over traditional computer-based analysis of being able to consider many variables at once and the dependencies between them, of avoiding the necessity of explicitly specifying the types of relationships that one wishes to compare and of representing the results of sophisticated processing in relatively simple and easy to understand ways.

Although these advantages also apply to expert human analysts, machine learning algorithms can also avoid the biases and assumptions that humans inevitably develop, despite their best efforts to remain objective. Such algorithms can also be used to analyze music of many kinds hundreds of times faster than humans, and with much greater consistency.

Of course, the ultimate goal of any analysis is to represent musical truth as interpreted by humans, so any computer analysis can never be more than an approximation of what humans perceive. It is therefore impossible for a computer to ever perform analyses better than an expert human, and computers are certainly not being proposed as replacements for human music researchers.

It is important to realize that this does not in any way negate the value of computer-based analyses, however. Although computers can only approximate human perception, this does not mean that they can not approximate it well, and the ability to model more diverse types of music than a human could feasibly become expert in makes it possible for a computer to consider far more pieces than a human ever could, and therefore arrive at results with more universal meaning and scope. The fresh perspective offered by computers can also provide human analysts with valuable ideas and insights that they can then build on.

The potential and advantages of machine learning when applied to validating existing theoretical models and performing exploratory research on how different features are distributed and interrelated with respect to different types of music should now be clear. There are also many additional possible musical applications for machine learning, however.

Pattern recognition has most widely been applied to music in the MIR field. Those wishing a more complete survey of MIR research than the brief review presented in the following paragraphs should consult the work of Byrd and Crawford (2001) and of Downie (2003), or the proceedings of the various ISMIR conferences.

Optical music recognition makes it possible to extract symbolic representations (e.g., GUIDO, LilyPond, Finale, etc.) from scores or microfilms of scores. Research in automatic transcription is working towards transforming audio performances into symbolic formats like MIDI, something that is particularly useful when applied to types of music with no written tradition, as well as in capturing performance characteristics that are not specified in scores.

Watermarking and fingerprinting allow one to automatically identify particular pieces of music. Performer identification and composer attribution make it possible to automatically determine probable authourship of anonymous recordings and scores.

Research in database structuring and metadata is highly relevant to the archiving and retrieval of valuable primary sources. Query by humming enables searching of databases using queries entered sonically rather than symbolically.

Automatic genre, mood, style, temporal and geographical classification systems can be used to properly label pieces along a variety of dimensions, and can help researchers to understand precisely what it is that separates various categories. Finally, automated similarity measurement, in addition to many practical uses such as play list generation, recommendation and hit prediction, can also help researchers determine what it is that makes specific pieces and collections of music similar in various ways. Computer-based research in music classification and similarity analysis can also be useful in a context beyond a musicological and music theoretical research by helping to understand the psychological processes involved in human music classification and similarity perception.

Before concluding, it is appropriate to provide an example of experimental evidence supporting the effectiveness of jSymbolic and ACE. These software packages both grew directly out of the Bodhidharma MIDI genre/style classification system (McKay 2004), and are essentially expansions and generalizations of the features and machine learning algorithms implemented in Bodhidharma.

Bodhidharma operates by classifying MIDI files into one or more of 38 candidate genres,[1] ranging from bluegrass to baroque to hardcore rap to bebop. The effectiveness of Bodhidharma at performing this task was demonstrated by the fact that it placed first in all four categories of the 2005 MIREX Symbolic Genre Classification Contest (Downie 2005).

The models learned by Bodhidharma were examined in order to extract information that might help to understand what it is that distinguishes different genres of music from each other (McKay & Fujinaga 2005). Although there is insufficient room to review the results in detail here, it was found that features based on instrumentation were in general consistently and significantly more effective in distinguishing between genres than other types of features. Detailed analysis of surprising results such as this could lead to interesting empirical musicological and theoretical developments.

7. Conclusions

It is hoped that a convincing case has been made for the adoption of sophisticated pattern recognition and machine learning approaches in musicological and music theoretical research. These methodologies have advantages relating to the ability to automatically form models that consider a large number of features and the interrelationships between them, the lack of a need to formally specify any heuristics or queries before beginning analyses, the ability to present results of sophisticated processing in low-dimensional spaces and the lack of built in biases and assumptions in analyses.

Machine learning algorithms are particularly well suited to processing very large sets of music, which makes it possible to perform large-scale theoretical exploratory analysis and empirical validation of existing theories. Machine learning techniques can be applied to many diverse types of music, including a variety of art, popular and folk musics of the world, many of which do not yet have established theoretical frameworks. Finally, and perhaps most importantly, the results of processing using machine learning can cause human researchers to see music from new perspectives and can inspire them to pursue promising research directions that might not otherwise have been obvious to them.

It is also hoped that the features implemented by jSymbolic will be of research value, and that the jSymbolic and ACE software packages themselves will not only be used by musicologists and theorists in their research, but that researchers will also develop further features and contribute them to these systems so that they can be used by others.

jSymbolic and ACE are both parts of the jMIR research project. Software and documentation may be downloaded from http://sourceforge.net/projects/jmir.

8. Acknowledgements

The generous financial support of the *Social Sciences and Humanities Research Council of Canada*, the *Centre for Interdisciplinary Research in Music, Media and Technology (CIRMMT)* and the *Canada Foundation for Innovation* is greatly appreciated.

9. Bibliography

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. E*thnomusicology* 20 (2): 179–215.

Alpaydin, E. 2004. *Introduction to machine learning.* Cambridge, MA: MIT Press.

Basili, R., A. Serafini, and A. Stellato. 2004. Classification of musical genre: A machine learning approach. *Proceedings of the Interna-*

[1] Understood here to be broad cultural and stylistic groups of music of any kind, as discussed by Fabbri (1999).

tional 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.

Byrd, D., and T. Crawford. 2001. Problems of music information retrieval in the real world. *Information Processing & Management* 38 (2): 249–72.

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, S. 2005. *MIREX 2005 Contest Results*. Available on-line at http://www.music-ir.org/evaluation/mirex-results. Retrieved January 9, 2006.

Downie, S. 2003. Music information retrieval. In *Annual Review of Information Science and Technology 37*, B. Cronin ed. Medford, NJ: Information Today.

Duda, R. O., P. E. Hart, and D. G. Stork. 2001. *Pattern classification*. New York: John Wiley & Sons Inc.

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

Fabbri, F. Browsing music spaces: Categories and the musical mind. *Proceedings of the IASPM Conference*.

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

Gingras, B. and I. Knopke. 2005. Evaluation of voice-leading and harmonic rules of J. S. Bach's chorales. *Proceedings of the Conference on Interdisciplinary Musicology*.

Hastie, T., R. Tibshirani, and J. Friedman. 2001. *The elements of statistical learning*. New York: Springer.

Huron, D. 2002. Music information processing using the Humdrum toolkit: Concepts, examples, and lessons. *Computer Music Journal* 26 (2): 11–26.

Huron, D. 1999. The new empiricism: Systematic musicology in a postmodern age. *1999 Ernst Bloch Lecture*. University of California, Berkeley.

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.

Kuncheva, L. 2004. *Combining pattern classifiers*. Hoboken, NJ: Wiley.

LaRue, J. 1992. *Guidelines for style analysis*. Warren, MI: Harmonie Park Press.

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

McEnnis, D., C. McKay, and I. Fujinaga. 2006. jAudio: Additions and improvements. *Proceedings of the International Conference on Music Information Retrieval.* 385–6.

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. and I. Fujinaga. 2004. Automatic genre classification using large high-level musical feature sets. *Proceedings of the International Conference on Music Information Retrieval*. 525–30.

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.

Temperley, D. 2001. *The cognition of basic musical structures*. Cambridge, MA: MIT Press.

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.

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