Classifying Music with jMIR

Cory McKay
Marianopolis College and CIRMMT
Montreal, Canada
Lecture contents

- Introduction to music information retrieval
  - Automatic classification
- Overview of the jMIR software
- Multimodal classification experiments
  - Empirical results
- jSymbolic
- Other jMIR components
  - As time and interest permit
Goals of MIR

- Extract meaningful information from or about music
- Facilitate music analysis, organization and access
Main sources of information

- **Symbolic recordings**
  - e.g. MIDI

- **Audio recordings**
  - e.g. MP3

- **Cultural data**
  - e.g. web data, metadata tags, etc.

- **Lyrics**

- **Others**
  - Album art, videos, etc.
A (very partial) list of MIR tasks

- Automatic transcription
- Automatic music analysis
  - Harmonic analysis, structural segmentation, etc.
- Query by example
- Optical music recognition (OMR)
- Fingerprinting (song identification)
- Interfaces and visualizations
- Similarity
  - Recommendation, hit prediction, etc.
- Automatic classification
  - Genre, mood, artist, composer, instrument, etc.
Automatic music classification

- Typical procedure:
  - Collect annotated training / testing data
    - With appropriate ontologies
  - Extract features
  - Reduce feature dimensionality
  - Train a classification model
    - Typically supervised
  - Validate the model

- Most significant challenges:
  - Acquiring sufficiently large annotated datasets
  - Designing features that encapsulate relevant data
Overview of the jMIR software

- jMIR is software suite designed for performing research in automatic music classification

- Primary tasks performed:
  - Feature extraction
  - Machine learning
  - Data storage file formats
  - Dataset management
    - Acquiring, correcting and organizing metadata
Characteristics of jMIR

- Has a **separate software component** to address each important aspect of automatic music classification
  - Each component can be used independently
  - Can also be used as an integrated whole
- Free and **open source**
- Architectural emphasis on providing an **extensible platform** for iteratively developing new techniques and algorithms
- Interfaces designed for both **technical** and **non-technical** users
- Facilitates **multimodal research**
jMIR components

- **jAudio**: Audio feature extraction
- **jSymbolic**: Feature extraction from MIDI files
- **jWebMiner**: Cultural feature extraction
- **jLyric**: Extracts features from lyrical transcriptions
- **ACE**: Meta-learning classification engine
- **ACE XML**: File formats  
  - Features, feature metadata, instance metadata and ontologies
- **lyricFetcher**: Lyric mining
- **Codaich, Bodhidharma MIDI and SLAC**: datasets
- **jMusicMetaManager**: Metadata management
- **jSong Miner**: Metadata harvesting
- **jMIRUtilities**: Infrastructure for conducting experiments
Efficacy of multimodal approaches?

- Can combining features extracted from audio, symbolic, cultural and/or lyrical sources significantly improve automatic music classification performance?
  - Intuitively, they each seem to contain very different kinds of information

- Can this help us break the seeming music classification performance ceiling of 70% to 80% for reasonably-sized taxonomies?

- This was studied empirically (McKay et al. 2010)
  - A follow-up on a similar earlier study (McKay 2010)
Experimental methodology

- Extracted features from separate audio, symbolic, cultural and lyrical sources of data
  - Corresponding to the same musical pieces
  - Using the jMIR feature extractors
- Compared ACE-based genre classification performance of each of the 15 possible subsets of these 4 feature groups
  - Audio, Symbolic + Audio, Cultural, Symbolic + Cultural + etc.
  - Applied dimensionality reduction
  - 10-fold cross-validation
    - With reserved validation set
  - Wilcoxon signed-rank significance tests were used
Musical dataset used: SLAC

- The **SLAC Dataset** was assembled for this experiment
  - Symbolic Lyrical Audio Cultural
  - 250 recordings belonging to 10 genres
    - Collapsible to 5 genres
  - Audio and MIDI versions of each recording
    - Acquired separately
  - Accompanying **metadata** that could be used to extract cultural features from the web
  - Lyrics mined with lyricFetcher
Results: 5-genre taxonomy

- All feature groups involving cultural features achieved classification accuracies of 99% to 100%
- Symbolic features alone performed with a classification accuracy of 85%
Results: 10-genre taxonomy

- SAC achieved the best classification accuracy of 89%
- All feature groups that included cultural features achieved 81% or higher
- Symbolic features alone performed at 66%
Discussion: Combining feature types

- Combining features types tended to increase classification performance on average.

- However, there were exceptions:
  - e.g. LC performed significantly less well than C in the 10-genre experiment.

![Average Classification Performance Based on the Number of Feature Types](chart.png)
Discussion: Feature type dominance

- Cultural features significantly outperformed other feature types.
- For the 10-genre taxonomy, all groups including cultural features outperformed all groups of the same size that did not include cultural features.
- Symbolic features were useful in general:
  - Symbolic groups all performed at 70% or above.
  - SAC was the best group overall, at 89%.

![Graph showing the comparison of average classification accuracy with and without cultural features](image)
Experimental conclusions

- Excellent overall genre classification results were obtained
  - 89% on 10 genres, compared to the best MIREX audio-only result to date of 80% on 10 genres
  - As a side note, jMIR holds the MIREX record (2005) for symbolic-only genre classification in a separate experiment
    - 84% on a 9-class taxonomy
    - 46% on a 38-class taxonomy
- Combining feature types tended to improve results
- Cultural features dominated
Important research question

Should research efforts be focused on fingerprinting and cultural feature extraction rather than bothering with extracting content-based features?

- Assuming reliable fingerprinting, this could result in very high classification results

- However, this marginalizes the *musicological* and *music theoretical* insights about musical categories that can be achieved from content-based analysis

- Cultural features are also of no or limited utility for *brand new music*
Introduction to jSymbolic

- Extracts features from MIDI files
- 111 implemented features
  - By far the largest existing symbolic feature catalogue
  - Many are original
- An additional 49 features are proposed but not yet implemented
- Features saved to ACE XML
Instrumentation:
- What types of instruments are present and which are given particular importance relative to others?
- Found experimentally to be the most effective symbolic feature type (McKay & Fujinaga 2005)

Texture:
- How many independent voices are there and how do they interact (e.g., polyphonic, homophonic, etc.)?

Rhythm:
- Time intervals between the attacks of different notes
- Duration of notes
- What kinds of meters and rhythmic patterns are present?
- Rubato?

Dynamics:
- How loud are notes and what kinds of dynamic variations occur?
jSymbolic feature types (2/2)

- Pitch Statistics:
  - What are the occurrence rates of different pitches and pitch classes?
  - How tonal is the piece?
  - 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?

- Chords (planned):
  - What vertical intervals are present?
  - What types of chords do they represent?
  - How much harmonic movement is there?
More on jSymbolic

- Easy to add new features
  - Modular plug-in design
  - Automatic provision of all other feature values to each new feature
  - Dynamic feature extraction scheduling that automatically resolves feature dependencies

- A variety of histogram aggregators are used
  - Beat histograms
  - Pitch and pitch class histograms (including wrapped)
  - Instrumentation histograms
  - Melodic interval histograms
  - Vertical interval histograms and chord type histograms
**Beat histogram example**

- Beat histograms use autocorrelation to calculate the relative strengths of different beat periodicities within a signal.

- *I Wanna Be Sedated* by The Ramones (top)
  - Several harmonic peaks with large spreads around them.

- *'Round Midnight* by Thelonious Monk (bottom)
  - Only one strong peak, with a large low-level spread.

---

**Beat Histogram: *I Wanna Be Sedated* by The Ramones**

- **Beats Per Minute:** 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200
- **Relative Frequency**

**Beat Histogram: *'Round Midnight* by Thelonious Monk**

- **Beats Per Minute:** 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200
- **Relative Frequency**
Chopin’s *Nocturne in B, Op. 32, No. 1*

- Average Note To Note Dynamics Change: 6.03
- Chromatic Motion: 0.0769
- Dominant Spread: 3
- Harmonicity of Two Strongest Rhythmic Pulses: 1
- Importance of Bass Register: 0.2
- Interval Between Strongest Pitch Classes: 3
- Most Common Pitch Class Prevalence: 0.433
- Note Density: 3.75
- Number of Common Melodic Intervals: 3
- Number of Strong Pulses: 5

- Orchestral Strings Fraction: 0
- Overall Dynamic Range: 62
- Pitch Class Variety: 7
- Range: 48
- Relative Strength of Most Common Intervals: 0.5
- Size of Melodic Arcs: 11
- Stepwise Motion: 0.231
- Strength of Strongest Rhythmic Pulse: 0.321
- Variability of Note Duration: 0.293
- Variation of Dynamics: 16.4
Mendelssohn’s *Piano Trio No. 2*

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

<table>
<thead>
<tr>
<th>Feature</th>
<th>Nocturne</th>
<th>Trio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Note To Note Dynamic Change</td>
<td>6.03</td>
<td>1.46</td>
</tr>
<tr>
<td>Overall Dynamic Range</td>
<td>62</td>
<td>22</td>
</tr>
<tr>
<td>Variation of Dynamics</td>
<td>16.40</td>
<td>5.98</td>
</tr>
<tr>
<td>Note Density</td>
<td>3.75</td>
<td>29.50</td>
</tr>
<tr>
<td>Orchestral Strings Fraction</td>
<td>0.00</td>
<td>0.56</td>
</tr>
<tr>
<td>Variability of Note Duration</td>
<td>0.293</td>
<td>0.104</td>
</tr>
<tr>
<td>Chromatic Motion</td>
<td>0.077</td>
<td>0.244</td>
</tr>
<tr>
<td>Range</td>
<td>48</td>
<td>39</td>
</tr>
</tbody>
</table>
Work to be done on jSymbolic

- Implement more features
  - 49 proposed
  - Many others possible

- Windowed feature extraction

- Parsers for more symbolic formats
  - Humdrum, OSC, MusicXML, etc.

- Output feature values using additional file formats
  - Especially Weka ARFF
More details?

- **jAudio**: Audio feature extraction
- **jWebMiner**: Cultural feature extraction
- **lyricFetcher** and **jLyric**: Lyric harvesting and feature extraction
- **ACE**: Meta-learning classification engine
- **ACE XML**: File formats
  - Features, feature metadata, instance metadata, ontologies
- **Codaich, Bodhidharma MIDI** and **SLAC**: datasets
- **jMusicMetaManager** and **jSongMiner**: Metadata management and harvesting

- **General questions?**
jAudio: An audio feature extractor

- Implemented jointly with Daniel McEnnis
- Extracts features from audio files
  - MP3, WAV, AIFF, AU, SND
- 28 bundled core features
  - Mainly low-level, some high-level
- Can automatically generate new features using metafeatures and aggregators
  - e.g. the change in a feature value from window to window
- Includes tools for testing new features being developed
  - Synthesize audio, record audio, sonify MIDI, display audio, etc.
jWebMiner: A cultural feature extractor

- Extracts cultural features from the web using search engine web services
- Calculates how often particular strings co-occur on the same web pages
  - e.g. how often does “J. S. Bach” co-occur on a web page with “Baroque”, compared to “Prokofiev”?
  - Results are processed to remove noise
- Additional options:
  - Can assign weights to particular sites
  - Can enforce filter words
  - Permits synonyms
- Also calculates features based on Last.FM user tags frequencies
lyricFetcher

- lyricFetcher automatically harvests lyrics from online lyrics repositories
  - LyricWiki and LyricsFly
  - Queries based on lists of song titles and artist names
- Post-processing is applied to the lyrics in order to make them sufficiently consistent for feature extraction
  - Deals with situations where sections of lyrics are abridged using keywords such as "chorus", "bridge", "verse", etc.
  - Filters out keywords that could contaminate the lyrics
- Ruby implementation
jLyrics

- **Extracts features** from lyrics stored in text files
  - Automated Readability Index
  - Average Syllable Count Per Word
  - Contains Words
  - Flesh-Kincaid Grade Level
  - Flesh Reading Ease
  - Function Word Frequencies
  - Letter-Bigram Components
  - Letter Frequencies
  - Letters Per Word Average
  - Letters Per Word Variance
  - Lines Per Segment Average
  - Lines Per Segment Variance
  - Number of Lines
  - Number of Segments
  - Number of Words
  - Part-of-Speech Frequencies
  - Punctuation Frequencies
  - Rate of Misspelling
  - Sentence Count
  - Sentence Length Average
  - Topic Membership Probabilities
  - Vocabulary Richness
  - Vocabulary Size
  - Word Profile Match
  - Words Per Line Average
  - Words Per Line Variance

- Can also automatically generate **word frequency profiles** for particular classes if training data is provided

- **Central framework** implemented in Java
  - Other technologies used by third-party components
ACE: A meta-learning engine

- Evaluates the relative suitability of different dimensionality reduction and classification algorithms for a given problem
  - Can also train and classify with manually selected algorithms
- Evaluates algorithms in terms of
  - Classification accuracy
  - Consistency
  - Time complexity
- Based on the Weka framework, so new algorithms can be added easily
ACE XML: MIR research file formats

- Standardized file formats that can represent:
  - Feature values extracted from instances
  - Abstract feature descriptions and parameterizations
  - Instance labels and annotations
  - Class ontologies
- Designed to be flexible and extensible
  - Able to express types of information that are particularly pertinent to music
- Allow jMIR components to communicate with each other
  - Can also be adopted for independent use by other software
- ACE XML 2.0 provides even more expressivity
  - e.g. potential for integration into RDF ontologies
jMIR datasets

- Codaich is a MP3 research set
  - Carefully cleaned and labelled
  - The published 2006 version has 26,420 recordings
    - Belonging to 55 genres
    - Is constantly growing: currently 35,363 MP3s
- Bodhidharma MIDI has 950 MIDI recordings
  - 38 genres of music
- SLAC consists of 250 matched audio recordings, MIDI recordings, lyrical transcriptions and metadata that can be used to extract cultural features
  - Useful for experiments on combining features from different types of data
  - 10 genres of music (in 5 pairs of similar genres)
jMusicMetaManager: A dataset manager

- Detects metadata errors/inconsistencies and redundant copies of recordings
- Detects differing metadata values that should in fact be the same
  - e.g. “Charlie Mingus” vs. “Mingus, Charles”
- Generates HTML inventory and profile reports (39 reports in all)
- Parses metadata from ID3 tags and iTunes XML
jSongMiner

- Software for automatically acquiring formatted metadata about songs, artists and albums
- Designed for use with the Greenstone digital library software
  - May also be used for other purposes, such as cultural feature extraction
- Identifies music files
  - Uses Echo Nest fingerprinting functionality and embedded metadata
- Mines a wide range of metadata tags from the Internet and collates them in a standardized way
  - Data extracted from The Echo Nest, Last.FM, MusicBrainz, etc.
  - Over 100 different fields are extracted
  - Data may be formatted into unqualified and/or qualified Dublin Core fields if desired
- Saves the results in ACE XML or text
  - Can also be integrated automatically into a Greenstone collection
Acknowledgements

- Collaborators at McGill and Waikato Universities
- Funding, past and present:
  - CIRMMT
  - Centre for Open Software Innovation
  - Andrew W. Mellon Foundation
  - Social Sciences and Humanities Research Council of Canada
  - Canadian Foundation for Innovation
- Contact information:
  - jmir.sourceforge.net
  - cory.mckay@mail.mcgill.ca