jSymbolic: A Software Application for Music Information Retrieval and Analysis

Cory McKay

Marianopolis College and CIRMMT

Topics

- Introduction to “features”
  - How they can be useful for musicologists and music theorists
- jSymbolic
- Overview of research performed with jSymbolic
- jMIR, SIMSSA and MIRAI
- Research collaborations

Personal context

- I was originally trained as a physicist and as a jazz guitarist before changing careers and focusing on music information retrieval
- As a former physicist, I am deeply attached to:
  - Overarching abstract theoretical models
  - Empirical validation of those models
- I think we do a great job at the first of these in music theory and musicology
  - But there is still room for improvement with respect to the second

Empiricism, software & statistics

- Empiricism, automated software tools and statistical analysis techniques allow us to:
  - Study huge quantities of music very quickly
    - More than any human could reasonably look at
  - Empirically validate (or repudiate) our theoretical suspicions
  - Do purely exploratory studies of music
  - See music from fresh perspectives
    - Can inspire new ways of looking at music

Human involvement is crucial

- Of course, computers certainly cannot replace the expertise and insight of musicologists and theorists
  - Computers instead serve as powerful tools and assistants that allow us to greatly expand the scope and reliability of our work
- Computers do not actually understand or experience music in ways at all similar to humans
  - We must pose the research questions for them to investigate
  - We must interpret the results they present us with
- Music is, after all, defined by human experience, not some “objective” externality

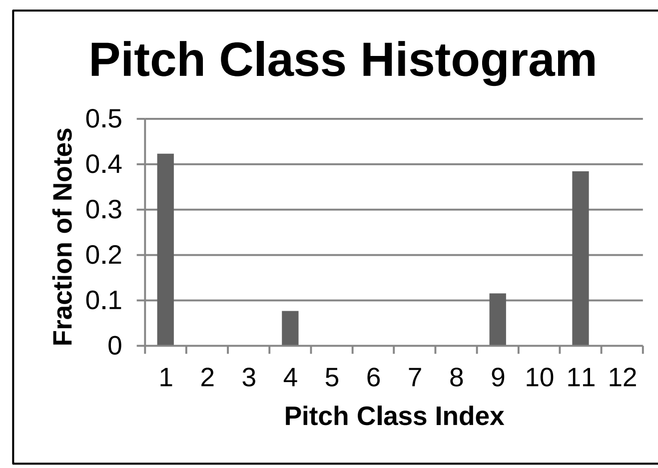
What are “features”?

- Pieces of information that can characterize something (e.g. a piece of music) in a simple way
- Usually numerical values
  - A feature can be a single value, or it can be a set of related values (e.g. a histogram)
- Can be extracted from pieces as a whole, or from segments of pieces

Example: Two basic features

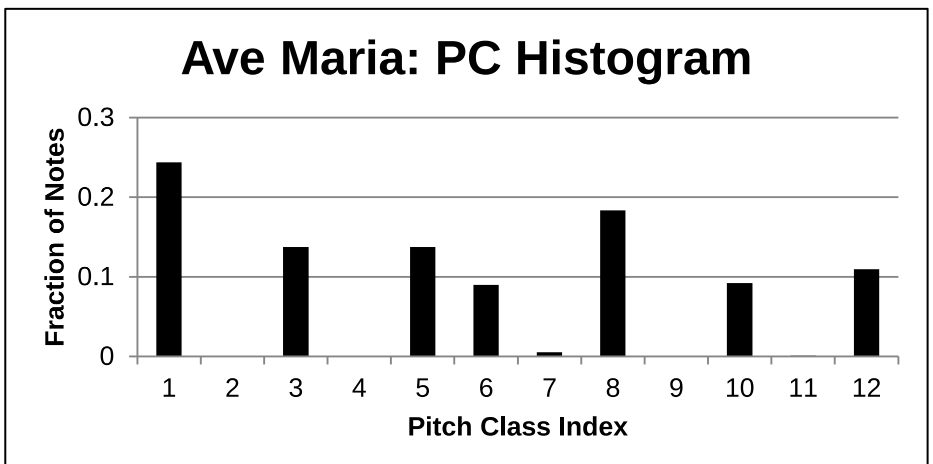
- Range (1-D): Difference in semitones between the highest and lowest pitches.
- Pitch Class Histogram (12-D): Each of its 12 values represents the fraction of notes of a particular pitch class. The first value corresponds to the most common pitch class, and each following value to a pitch class a semitone higher than the previous.

- Range = G - C = 7 semitones
- Pitch Class Histogram: see graph ->
    - Note counts: C: 3, D: 10, E: 11, G: 2
    - Most common note: E (11/26 notes)
        - Corresponding to 0.423 of the notes
    - E is thus pitch class 1, G is pitch class 4, C is pitch class 9, D is pitch class 11



Josquin's *Ave Maria... Virgo serena*

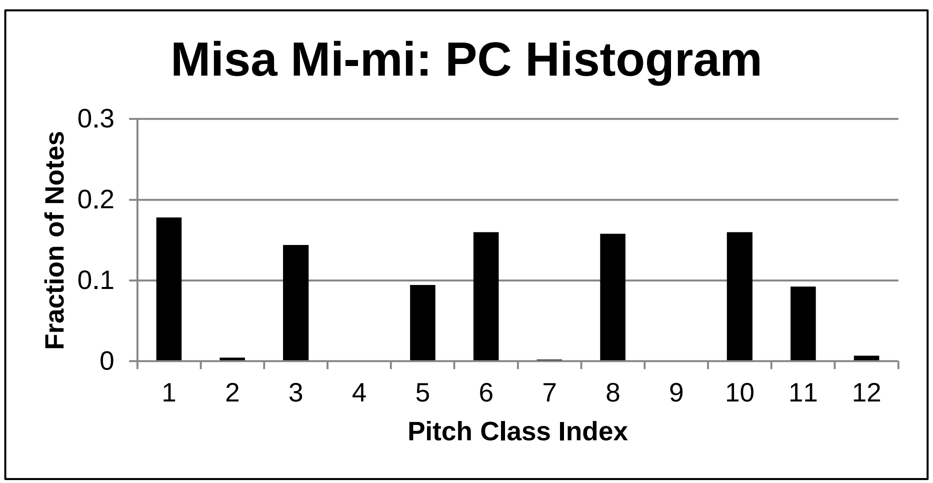
- Range: 34
- Repeated notes: 0.181
- Vertical perfect 4ths: 0.070
- Rhythmic variability: 0.032
- Parallel motion: 0.039



jMIR

Ockeghem’s *Missa Mi-mi* (Kyrie)

- Range: 26
- Repeated notes: 0.084
- Vertical perfect 4ths: 0.109
- Rhythmic variability: 0.042
- Parallel motion: 0.076



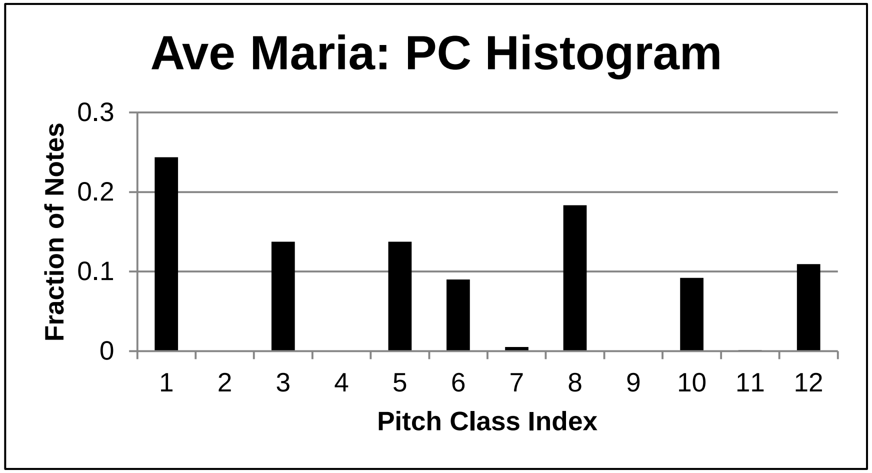
CIRMMT Centre for Interdisciplinary Research in Music Media and Technology

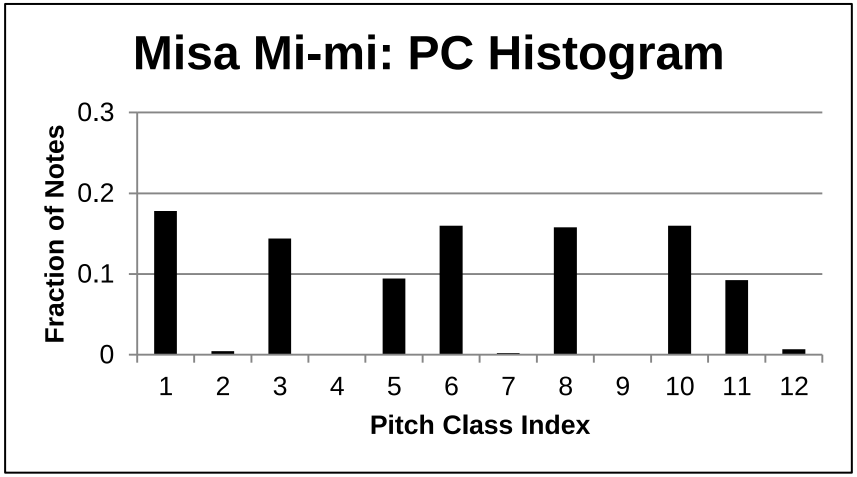
SIMSSA Single Interface for Music Score Searching and Analysis

MARIANOPOLIS COLLEGE

Feature value comparison

Feature	Ave Maria	Misa Mi-mi
Range	34	26
Repeated notes	0.181	0.084
Vertical perfect 4ths	0.070	0.109
Rhythmic variability	0.032	0.042
Parallel motion	0.039	0.076





How can we use features? (1/2)

- Manual analysis to look for patterns
- Use supervised machine learning to classify music
  - Done by training on pre-labelled data
  - Can study music using whatever categories (“classes”) one likes
    - e.g. style, genre, period, culture, geographical location, composer, etc.
  - Many possible applications
    - e.g. identify the composers of unattributed musical pieces
- Use unsupervised machine learning to cluster music
  - Done by training on unlabelled data
  - i.e. see how the system groups pieces based on statistical similarity
    - And then see if we can find meaning in these groups

How can we use features? (2/2)

- Perform sophisticated searches of large musical databases
  - e.g. find all pieces with less than X amount of chromaticism and more than Y amount of contrary motion
- Apply statistical analysis and visualization tools to study features extracted from large collections of music
  - Look for patterns
  - Study the relative musical importance of various features
  - Learn new things about the music from the features

Ways to examine features

- Manually
  - Text editors
  - Spreadsheets
- With automatic assistance
  - Statistical analysis software
    - e.g. SPSS, SAS, etc.
  - Machine learning and data mining software
    - e.g. Weka, Orange, etc.
- Many of these tools can produce helpful visualizations

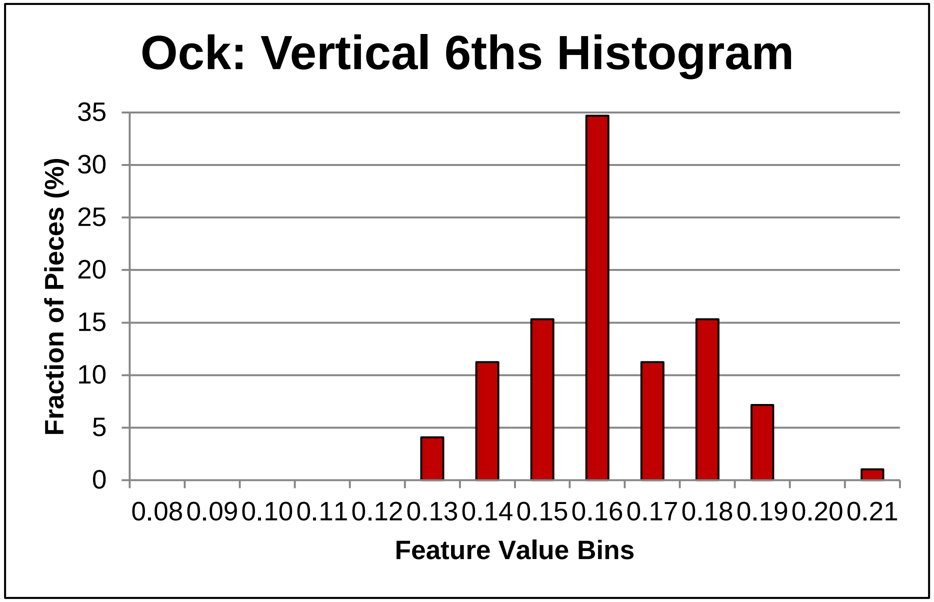
Feature visualization: Histograms (1/6)

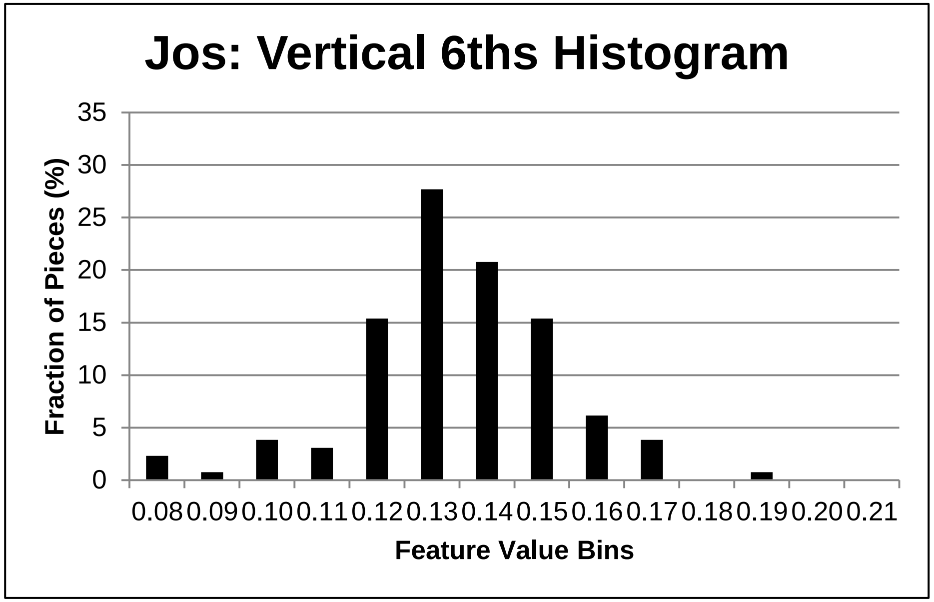
- Histograms are one good way to visualize how the values of a feature are distributed across a corpus as a whole
  - As opposed to focusing on individual pieces
- The x-axis corresponds to a series of bins, with each corresponding to a range of values for a given feature
  - e.g. the first bin could correspond to Parallel Motion feature values between 0 and 0.1, the next bin to Parallel Motion values between 0.1 and 0.2, etc.
- The y-axis indicates the fraction of all pieces that have a feature value within the range of each given bin
  - e.g. if 30% of pieces in the corpus have Parallel Motion values between 0.1 and 0.2, then this bin (0.1 to 0.2) will have a y-coordinate of 30% (or, equivalently, 0.3)

Feature visualization: Histograms (2/6)

- In other words:
  - Each bar on a histogram represents the fraction of pieces in a corpus with a feature value falling in that bar’s range of feature values
- Clarification: I am speaking here about a way to visualize a 1-dimensional feature as it is distributed across a corpus of interest
  - This is distinct from the multi-dimensional histogram features discussed earlier
    - e.g. Pitch Class Histograms
  - Although both are equally histograms, of course

Feature visualization: Histograms (3/6)

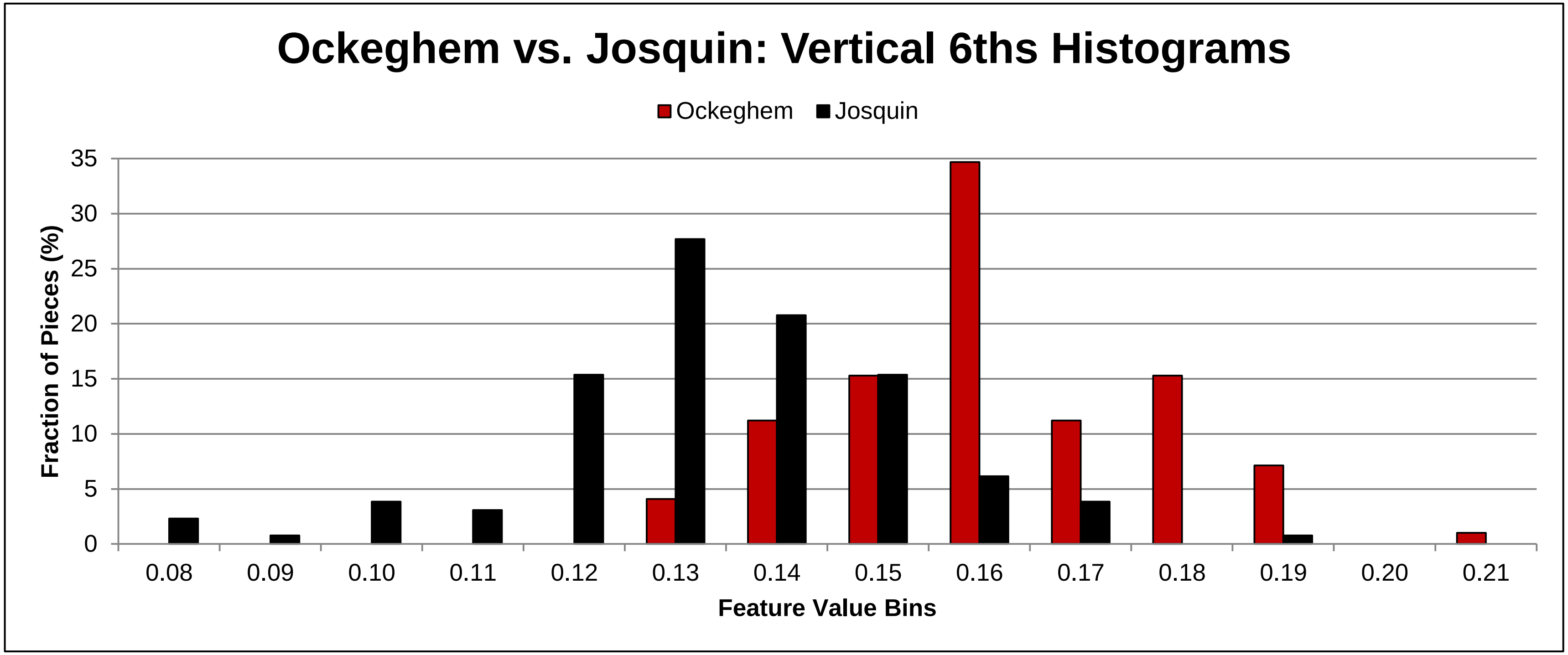




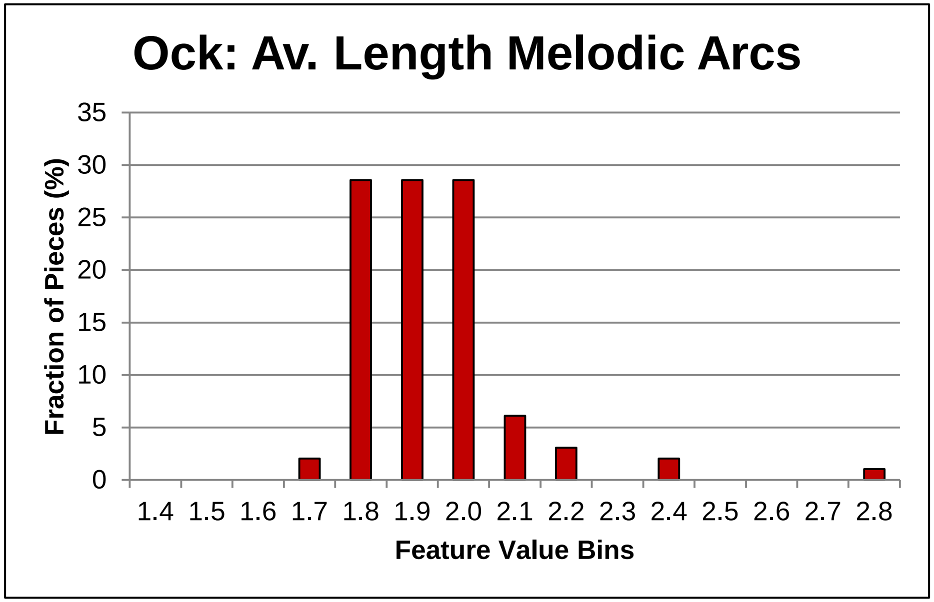
- These histograms show that Ockeghem tends to have more vertical 6^{ths} (between all pairs of voices) than Josquin
  - Ockeghem peaks in the 0.16 to 0.17 bin
  - Josquin peaks in the 0.13 to 0.14 bin
- Of course, there are also clearly many exceptions
  - This feature is helpful, but is limited if only considered alone

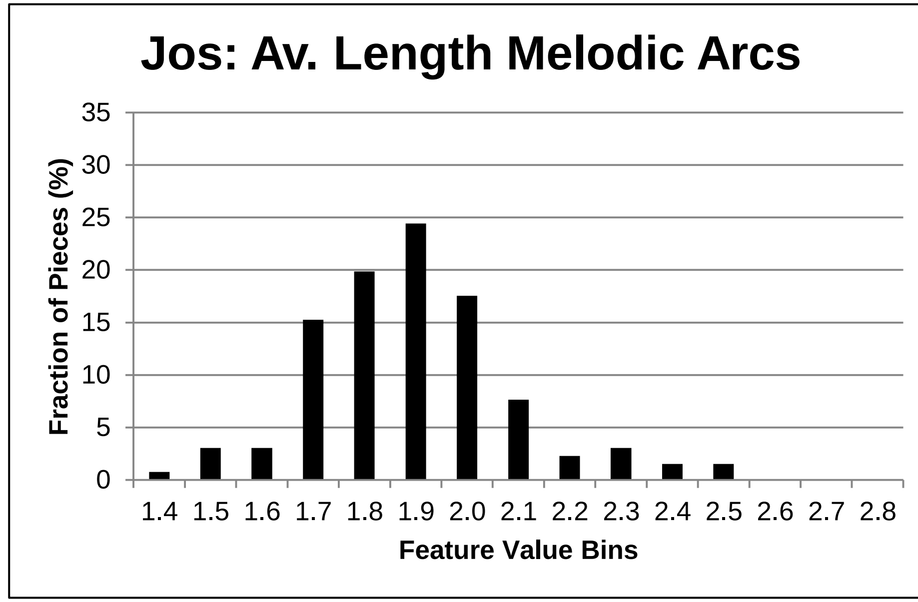
Feature visualization: Histograms (4/6)

- The histograms for both composers can also be superimposed onto a single chart:



Feature visualization: Histograms (5/6)

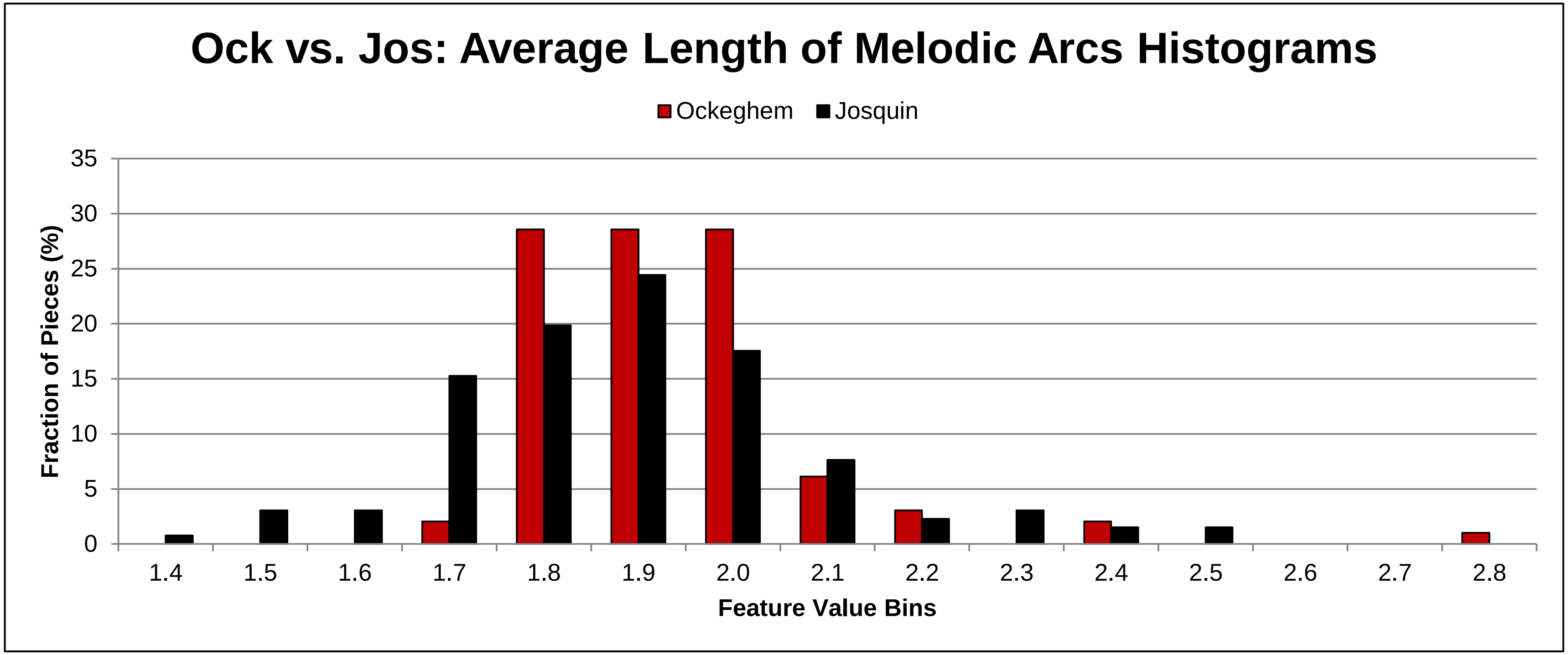




- These histograms show that Ockeghem tends to have longer melodic arcs (average number of notes separating peaks & troughs)
  - Both peak in the 1.9 to 2.0 bin
  - However, Josquin's histogram is (slightly) more skewed to the far left
- Of course, there are once again clearly many exceptions
  - This feature is also helpful, but also limited if considered alone

Feature visualization: Histograms (6/6)

- The histograms for both composers can also be superimposed onto a single chart:



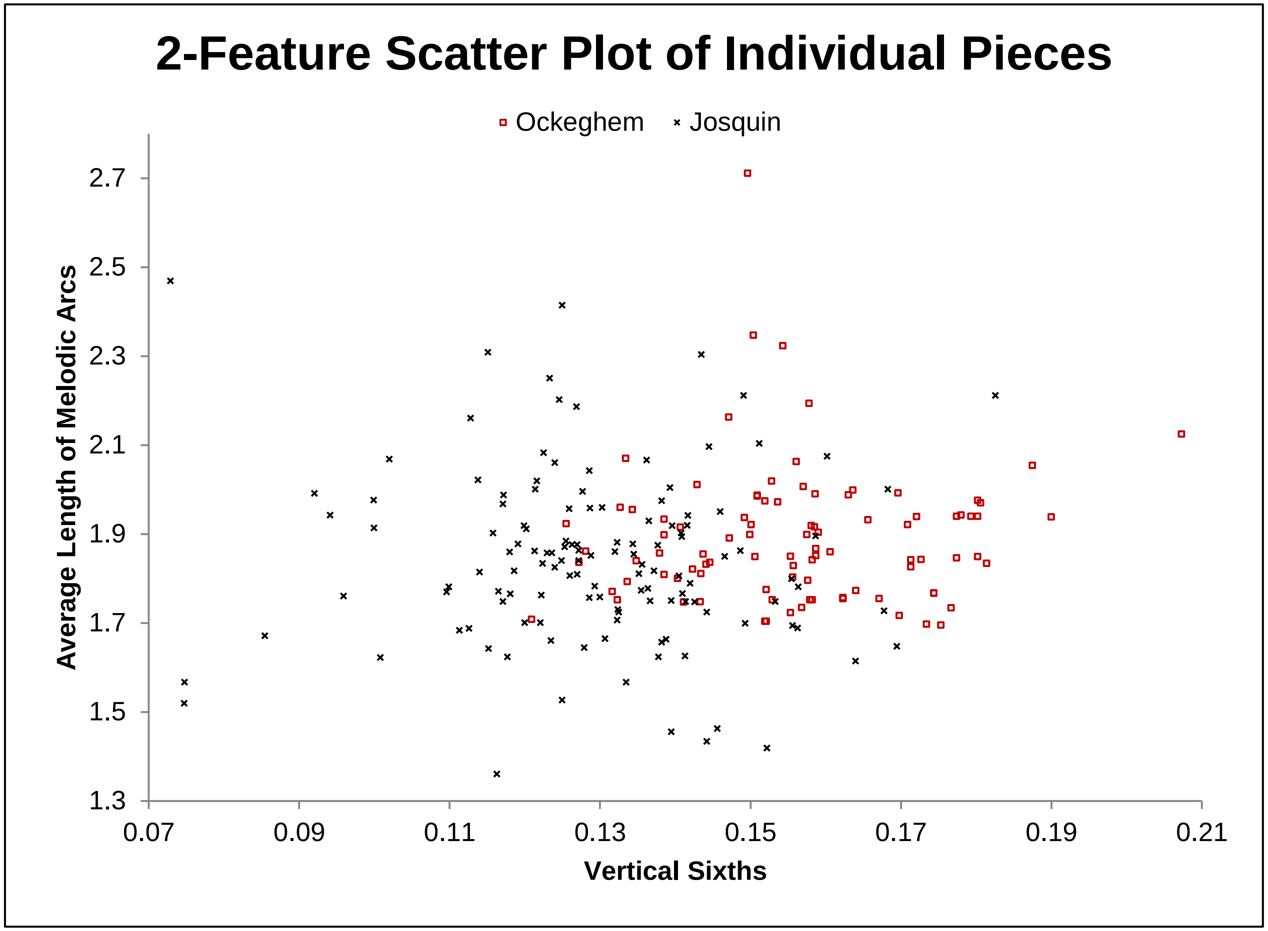
jMIR

Feature visualization: Scatter plots (1/6)

- Scatter plots are another good way to visualize feature data
  - The x-axis represents one feature
  - The y-axis represents some other feature
  - Each point represents the values of these two features for a single piece
- Scatter plots let you see pieces individually, rather than aggregating them into bins like histograms
  - Scatter plots also let you see more clearly how the two features divide the different composers
- To make them easier to read, scatter plots typically have just 2 dimensions
  - Computer classifiers, in contrast, work with n-dimensional scatterplots (one dimension per feature)

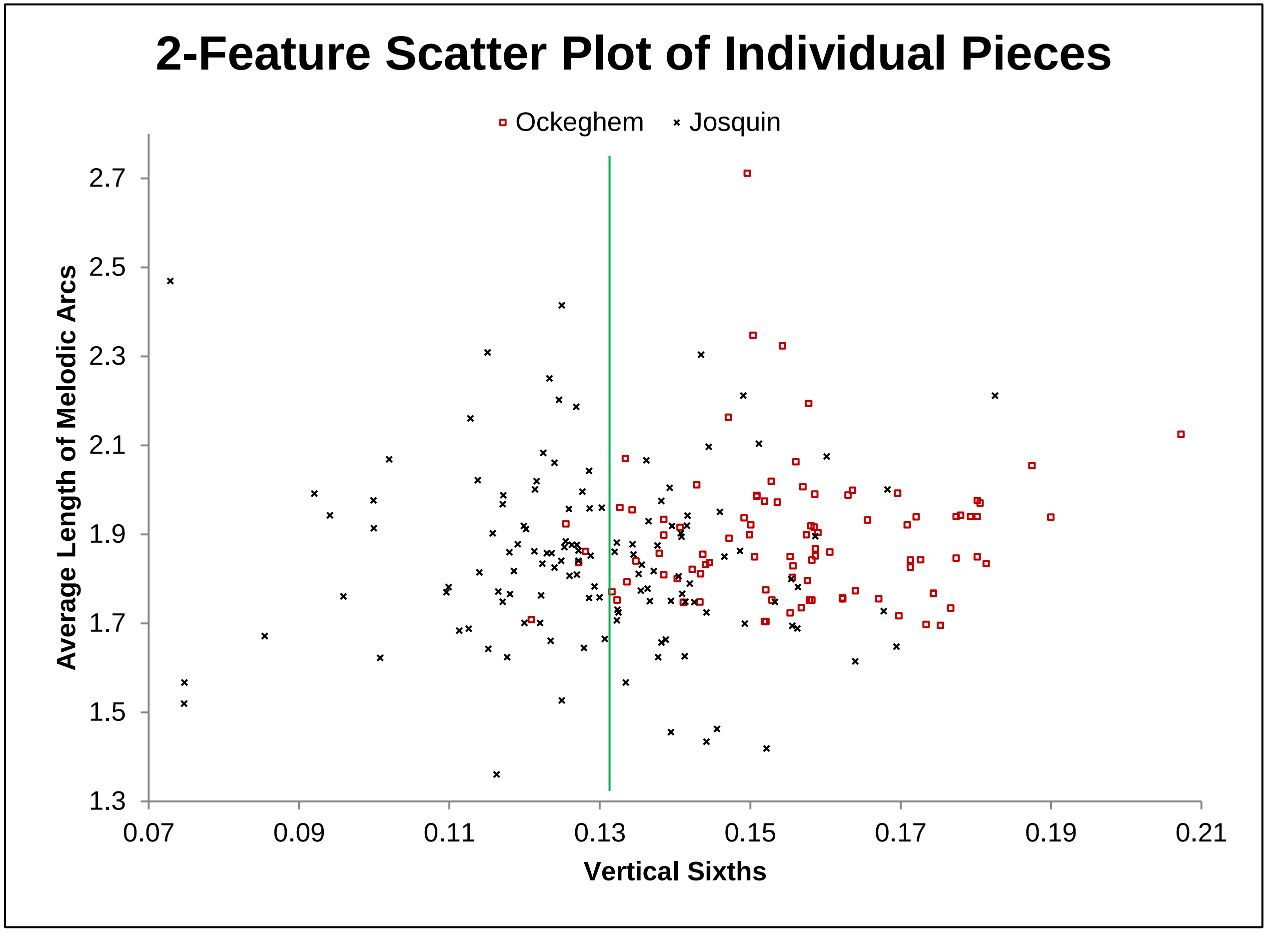
Feature visualization: Scatter plots (2/6)

- Josquin pieces tend to be left and low on this graph



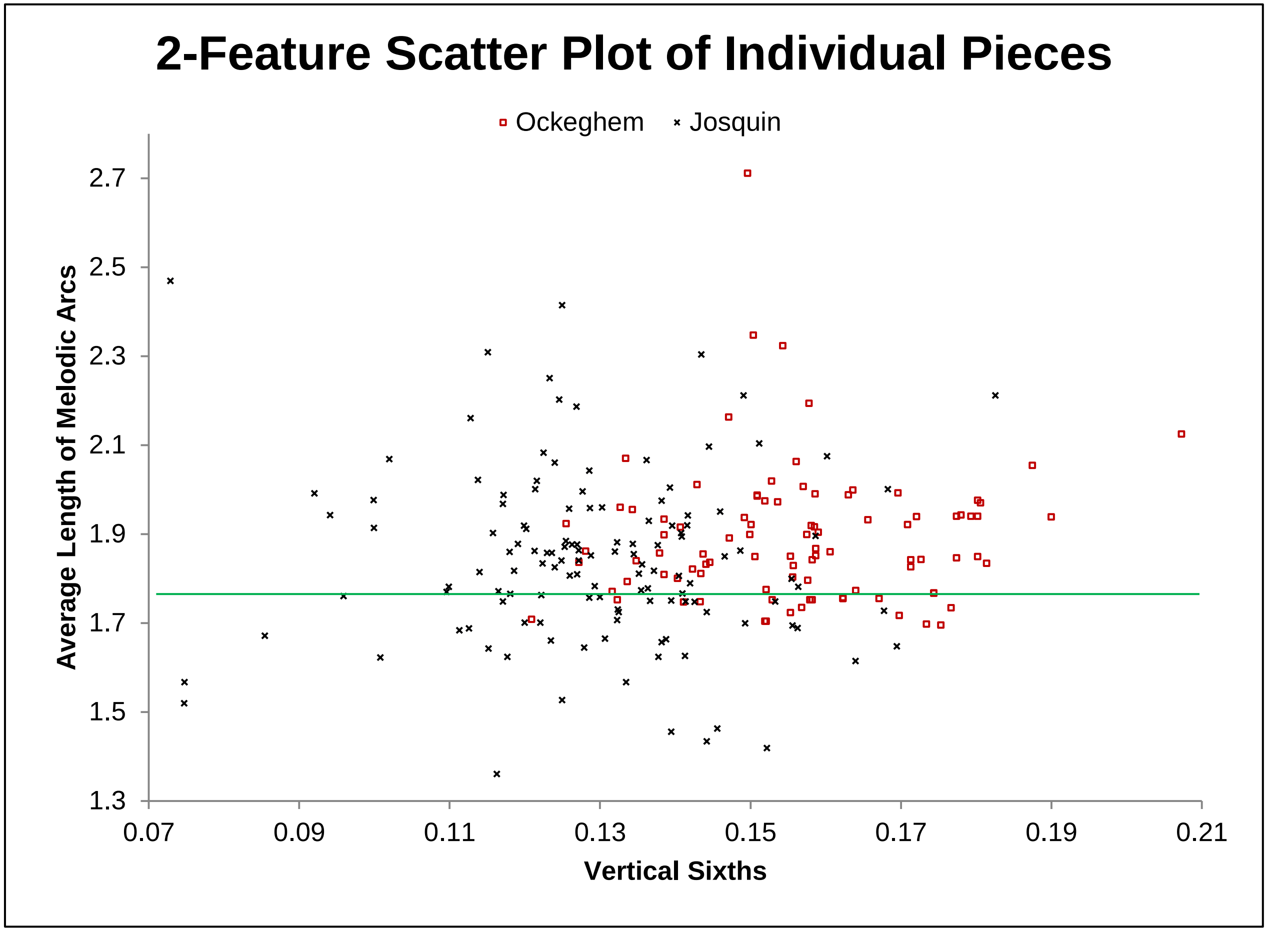
Feature visualization: Scatter plots (3/6)

- Simply drawing a single 1-D dividing line ("discriminant") results in a not entirely terrible classifier based only on Vertical Sixths!
  - But many pieces would still be misclassified
  - Get 62% classification accuracy using an SVM and just this one feature



Feature visualization: Scatter plots (4/6)

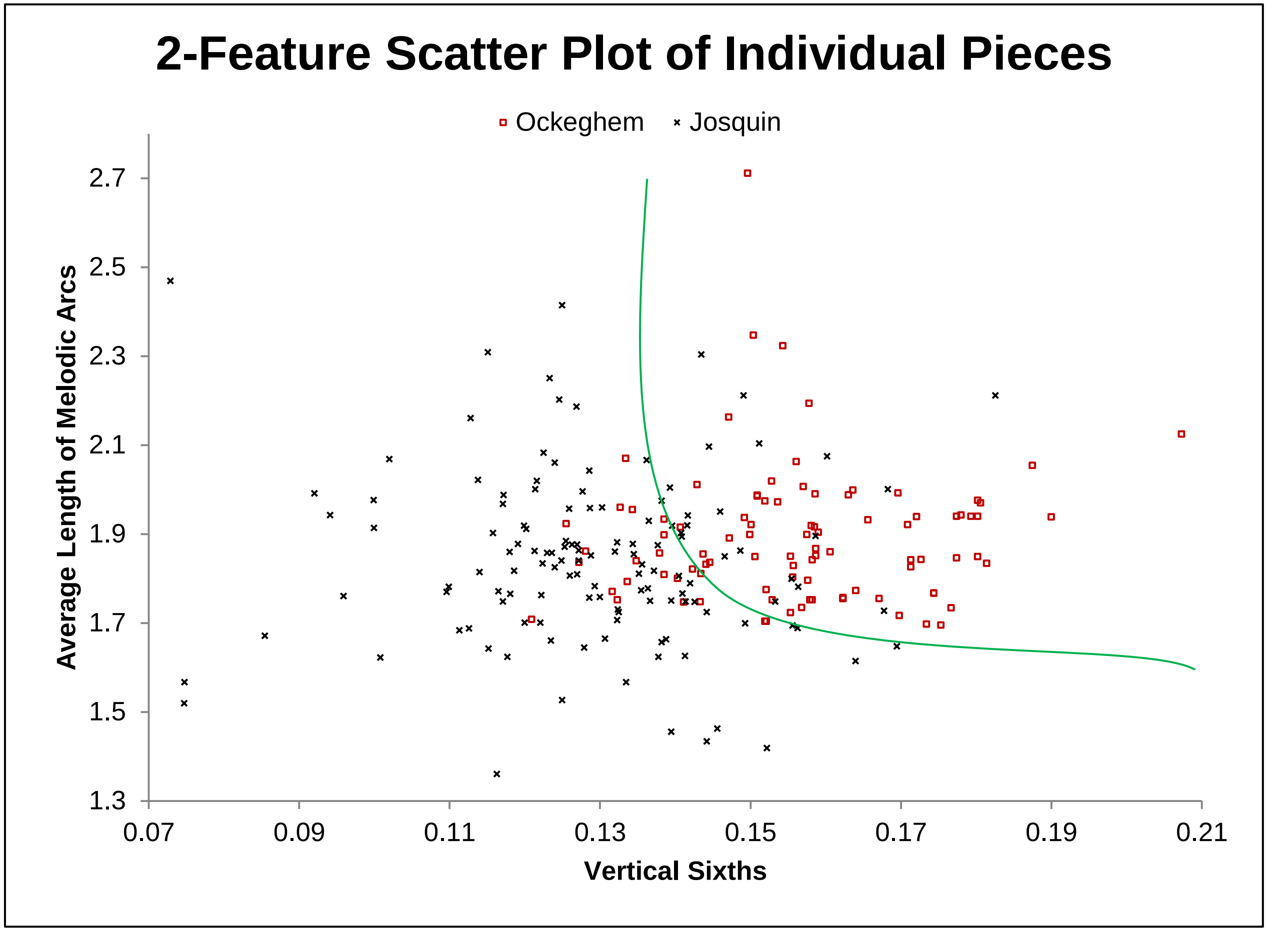
- Could alternatively draw a 1-D discriminant dividing the pieces based only on the Average Length of Melodic Arcs
  - Get 57% classification accuracy using an SVM and just this one feature
  - Not as good as the Vertical Sixths discriminant (62%)



jMIR

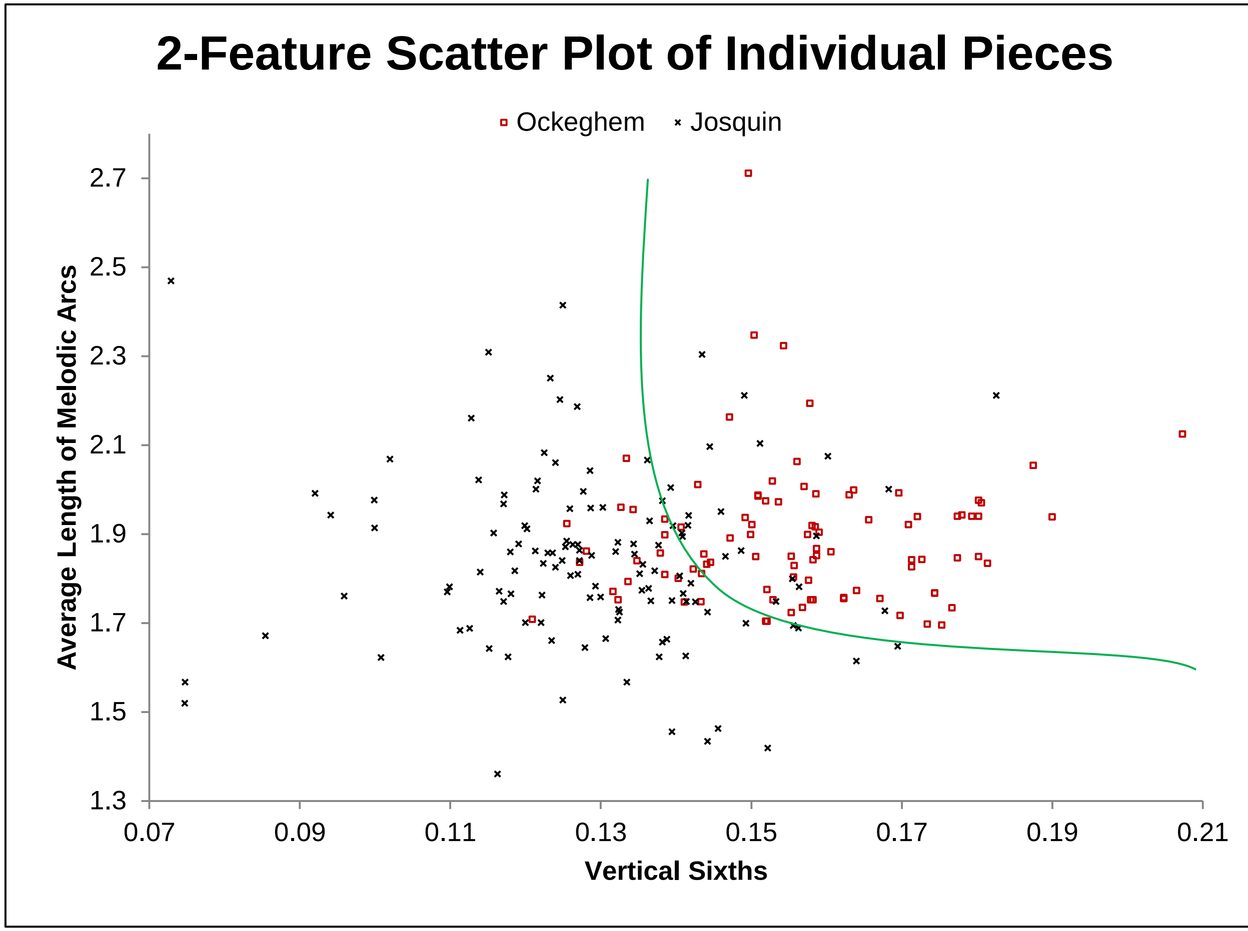
Feature visualization: Scatter plots (5/6)

- Drawing a curve (another kind of discriminant) divides the composers still better that either of the previous discriminants
  - Get 80% accuracy using an SVM and just these 2 features!
- More than 2 features are clearly needed to improve performance



Feature visualization: Scatter plots (6/6)

- In fact, many (but not all) types of machine learning in effect simply learn where to place these kinds of discriminants as they train
- But typically with many more then just two features, of course



Benefits of features

- Can quickly perform consistent empirical studies involving huge quantities of music
- Help to avoid potentially incorrect ingrained assumptions and biases
- Can be applied to diverse types of music in consistent ways
- Permit simultaneous consideration of thousands of features and their interrelationships
  - And can statistically condense many features into low-dimensional spaces when needed
- No need to formally specify any heuristics or queries before beginning analyses
  - Unless you want to, of course

jSymbolic 2.1: Introduction

- jSymbolic 2.1 (soon to be 3.0) is a software platform I have implemented for extracting features from symbolic music
  - Part of our much larger jMIR package
- Compatible with Macs, PCs and Linux computers
- Free and open-source

What does jSymbolic 2.1 do?

- Extracts 246 unique features
- Some of these are multi-dimensional histograms, including:
  - Pitch and pitch class histograms
  - Melodic interval histograms
  - Vertical interval histograms
  - Chord types histograms
  - Rhythmic value histograms
  - Beat histograms
  - Instrument histograms
- In all, extracts a total of 1497 separate values

jMIR

jSymbolic 2.1: Feature types (1/2)

- 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 / horizontal intervals:
  - 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 / vertical intervals:
  - What vertical intervals are present?
  - What types of chords do they represent?
  - How much harmonic movement is there?

jSymbolic 2.1: Feature types (2/2)

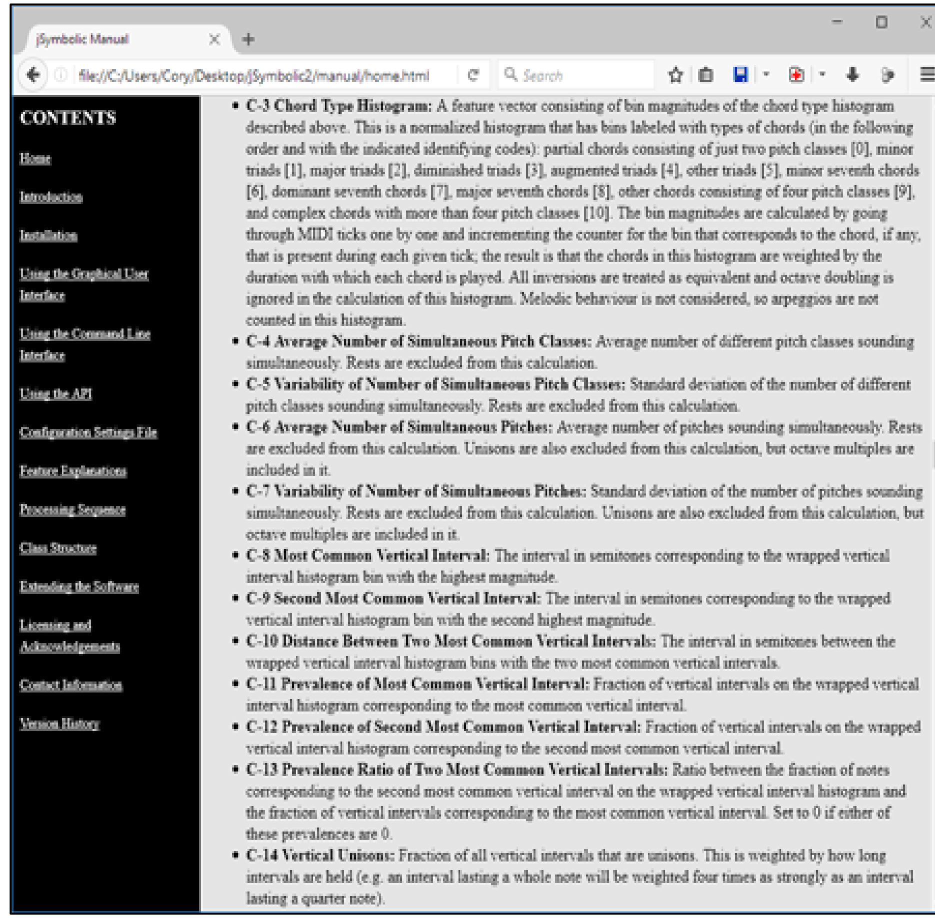
- Instrumentation:
  - What types of instruments are present and which are given particular importance relative to others?
- Texture:
  - How many independent voices are there and how do they interact (e.g., polyphonic, homophonic, etc.)?
- Rhythm:
  - Rhythmic values of notes
  - Durations of notes
  - Time intervals between the attacks of different notes
  - What kinds of meters and rhythmic patterns are present?
  - Rubato?
- Dynamics:
  - How loud are notes and what kinds of dynamic variations occur?

How does jSymbolic differ from other software?

- jSymbolic is intrinsically different from other software often used in empirical symbolic music research
  - e.g. music21 (includes a port of jSymbolic1's features)
  - e.g. Humdrum
  - e.g. Elvis
- This other software is excellent for finding exactly where specific things one is searching for happen
  - Perfect for very targeted research based on specific searches
- jSymbolic, in contrast, allows one to acquire large amounts of summary information about music without knowing a priori what one is looking for
  - Good for general annotation of symbolic databaes
  - Good for statistical analysis and machine learning
  - Good for exploratory research
  - Good for large-scale validations

jSymbolic 2.1: Manual

- Extensive manual includes:
  - Detailed feature descriptions
  - Detailed instructions on installation and use



jSymbolic Manual

file:///C:/Users/Cory/Desktop/jSymbolic2/manual/home.html

CONTENTS

Home

Introduction

Installation

Using the Graphical User Interface

Using the Command Line Interface

Using the API

Configuration Settings File

Feature Explanations

Processing Sequence

Class Structure

Extending the Software

Licensing and Acknowledgements

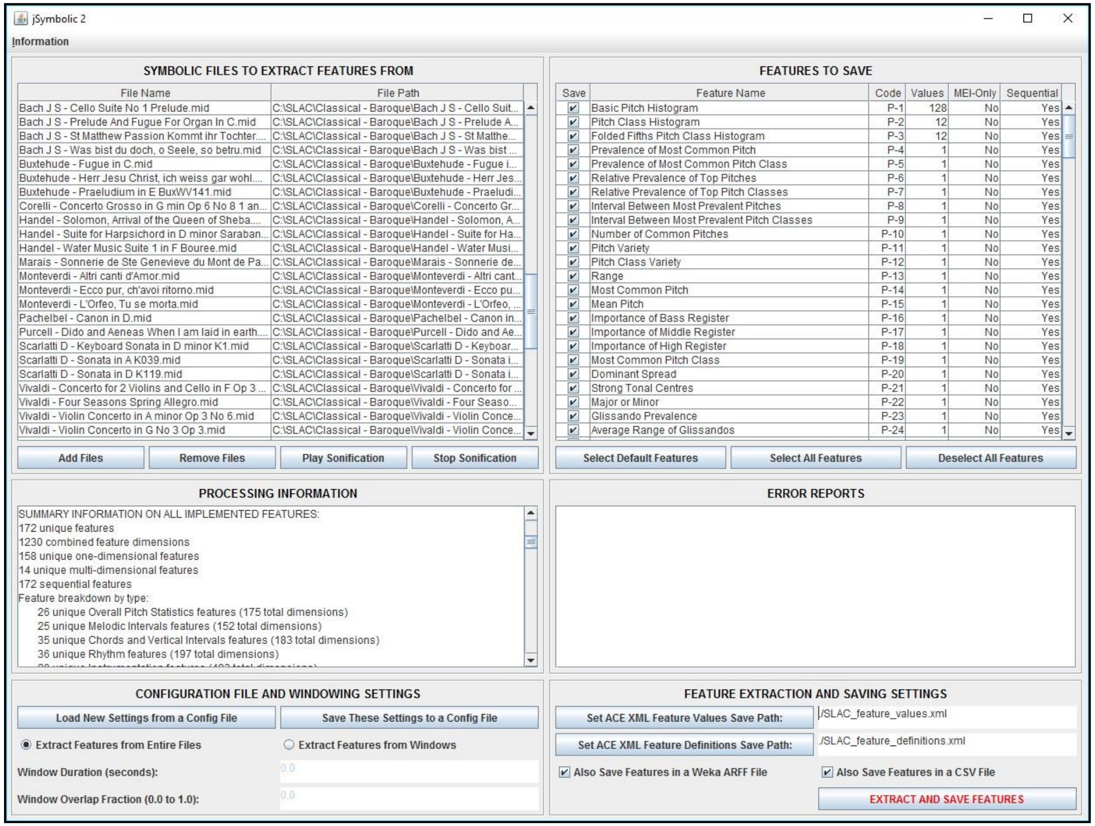
Contact Information

Version History

- **C-3 Chord Type Histogram:** A feature vector consisting of bin magnitudes of the chord type histogram described above. This is a normalized histogram that has bins labeled with types of chords (in the following order and with the indicated identifying codes): partial chords consisting of just two pitch classes [0], minor triads [1], major triads [2], diminished triads [3], augmented triads [4], other triads [5], minor seventh chords [6], dominant seventh chords [7], major seventh chords [8], other chords consisting of four pitch classes [9], and complex chords with more than four pitch classes [10]. The bin magnitudes are calculated by going through MIDI ticks one by one and incrementing the counter for the bin that corresponds to the chord, if any, that is present during each given tick; the result is that the chords in this histogram are weighted by the duration with which each chord is played. All inversions are treated as equivalent and octave doubling is ignored in the calculation of this histogram. Melodic behaviour is not considered, so arpeggios are not counted in this histogram.
- **C-4 Average Number of Simultaneous Pitch Classes:** Average number of different pitch classes sounding simultaneously. Rests are excluded from this calculation.
- **C-5 Variability of Number of Simultaneous Pitch Classes:** Standard deviation of the number of different pitch classes sounding simultaneously. Rests are excluded from this calculation.
- **C-6 Average Number of Simultaneous Pitches:** Average number of pitches sounding simultaneously. Rests are excluded from this calculation. Unisons are also excluded from this calculation, but octave multiples are included in it.
- **C-7 Variability of Number of Simultaneous Pitches:** Standard deviation of the number of pitches sounding simultaneously. Rests are excluded from this calculation. Unisons are also excluded from this calculation, but octave multiples are included in it.
- **C-8 Most Common Vertical Interval:** The interval in semitones corresponding to the wrapped vertical interval histogram bin with the highest magnitude.
- **C-9 Second Most Common Vertical Interval:** The interval in semitones corresponding to the wrapped vertical interval histogram bin with the second highest magnitude.
- **C-10 Distance Between Two Most Common Vertical Intervals:** The interval in semitones between the wrapped vertical interval histogram bins with the two most common vertical intervals.
- **C-11 Prevalence of Most Common Vertical Interval:** Fraction of vertical intervals on the wrapped vertical interval histogram corresponding to the most common vertical interval.
- **C-12 Prevalence of Second Most Common Vertical Interval:** Fraction of vertical intervals on the wrapped vertical interval histogram corresponding to the second most common vertical interval.
- **C-13 Prevalence Ratio of Two Most Common Vertical Intervals:** Ratio between the fraction of notes corresponding to the second most common vertical interval on the wrapped vertical interval histogram and the fraction of vertical intervals corresponding to the most common vertical interval. Set to 0 if either of these prevalences are 0.
- **C-14 Vertical Unisons:** Fraction of all vertical intervals that are unisons. This is weighted by how long intervals are held (e.g. an interval lasting a whole note will be weighted four times as strongly as an interval lasting a quarter note).

jSymbolic 2.1: User interfaces

- Graphical user interface
- Command line interface
- Java API
- Rodan workflow



jSymbolic 2.1: File formats

- Input:
  - MIDI
  - MEI
- Output:
  - CSV
  - ACE XML
  - Weka ARFF

jSymbolic 2.1: Miscellany

- Windowed feature extraction
  - Including overlapping windows
- Configuration files
  - Pre-set feature choices
  - Pre-set input and output choices
  - More
- Can combine jSymbolic with other jMIR components to perform multimodal research
  - i.e. combine symbolic features with other features extracted from audio, lyrics and cultural data
  - This improves results substantially! (McKay et al. 2010)

What can you do with jSymbolic 2.1's features?

- Empirically study and analyze huge collections of music in new ways
  - Search music databases based on feature values
  - Use machine learning
  - Analyze and visualize music based on feature values

jSymbolic 2.1: Types of users

- Musicologists and music theorists
  - Can use features to study music in many research domains
  - Not just composer identification and genre classification, which I have focused on so far
- MIR researchers
  - Especially as optical music recognition (OMR) and audio transcription technology improve
  - Can work even if the output retains some noise
- Music librarians
  - Permits sophisticated content-based search queries

jSymbolic 2.1: Extensibility

- jSymbolic is specifically designed such that music scholars can design their own features and work with programmers to then very easily add these features to the jSymbolic infrastructure
  - Fully open source
  - Modular plug-in feature design
  - Automatically handles feature dependencies and scheduling
  - Very well-documented code

Important software principles

- As Frans Wiering wisely pointed out at IMS 2017, those of us who produce research software must be careful to give musicologists what they want and need
  - Rather than trying to impose choices on them
- This emphasizes the importance of establishing on on-going dialog
  - Software designers should find out from musicologists what will be valuable to them
  - Software designers can also present musicologists with the possibility of options that they would not necessarily have though of, or thought possible
- So, please let me know what you need or want!

Research involving jSymbolic

- I will now discuss some previous research that I have published based on jSymbolic features
  - To give you an idea of what is possible
- I will spend some time discussing a study comparing Renaissance composers, as it is particularly relevant here
- I will then present brief hightights of certain other studies

Composer attribution study

- We used jSymbolic 2.0 features to automatically classify pieces of Renaissance music by composer
  - As an example of the kinds of things that can be done with jSymbolic
  - As a meaningful research project in its own right

RenComp7 dataset

- Began by constructing our "RenComp7" dataset:
  - 1584 MIDI pieces
  - By 7 Renaissance composers
- Combines:
  - Top right: Music drawn from the Josquin Research Project (Rodin, Sapp and Bokulich)
  - Bottom right: Music by Palestrina (Miller 2004) and Victoria (Sigler, Wild and Handelman 2015)

Composer	Pieces
Busnoys	69
Josquin *(only includes the 2 most secure Jesse Rodin groups)*	131
La Rue	197
Martini	123
Ockeghem	98

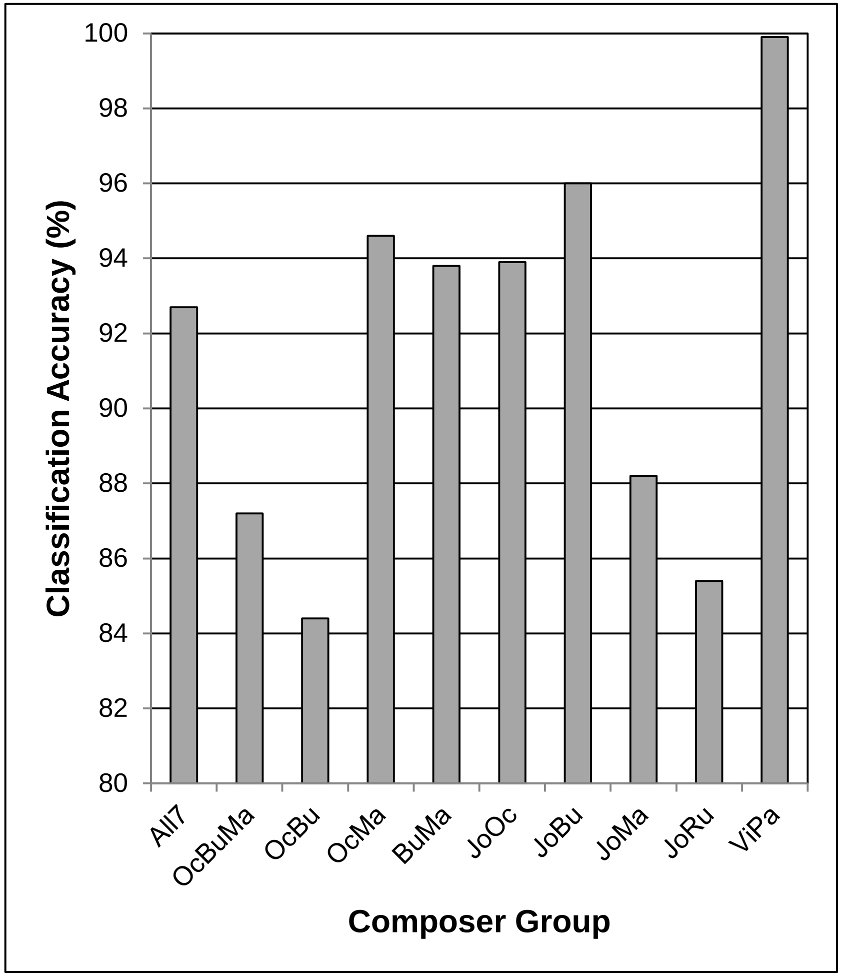
Composer	Pieces
Palestrina	705
Victoria	261

Methodology

- Extracted 721 feature values from each of the 1584 RenComp7 pieces using jSymbolic
- Used machine learning to teach a classifier to automatically distinguish the music of the composers
  - Based on the jSymbolic features
- Used statistical analysis to gain insight into relative compositional styles
- Performed several versions of this study
  - Classifying amongst all 7 composers
  - Focusing only on smaller subsets of composers
    - Some more similar, some less similar

Classification results

Composer Group	Classification Accuracy
All 7	92.7%
Ockeghem / Busnoys / Martini	87.2%
Ockeghem / Busnoys	84.4%
Ockeghem / Martini	94.6%
Busnoys / Martini	93.8%
Josquin / Ockeghem	93.9%
Josquin / Busnoys	96.0%
Josquin / Martini	88.2%
Josquin / La Rue	85.4%
Victoria / Palestrina	99.9%



Direct applications of such work

- Validating existing suspected but uncertain attributions
- Helping to resolve conflicting attributions
- Suggesting possible attributions of currently unattributed scores

Comparison with other work

- Brinkman, Shanahan and Sapp (2016) used 53 features to classify amongst 6 composers (J. S. Bach and five Renaissance composers)
  - Obtained success rates of roughly 63% on average
  - Did very well in distinguishing Bach from the Renaissance composers (97% on average)
  - This highlights both the high quality of their approach and the particular difficulty of differentiating the music of Renaissance composers
    - Which, in turn, makes the success of the jSymbolic 2.0 features on exclusively Renaissance (92.7% amongst 7 composers) music all the more encouraging
    - Of course, non-identical datasets make direct comparisons problematic

How do the composers differ?

- Some very interesting questions:
  - What musical insights can we learn from the jSymbolic feature data itself?
  - In particular, what can we learn about how the music of different composers differs?
- Chose to focus on two particular cases:
  - Josquin vs. Ockeghem: Relatively different
  - Josquin vs. La Rue: Relatively similar

A priori expectations (1/3)

- What might an expert musicologist expect to differentiate the composers?
  - Before actually examining the feature values
- Once formulating these expectations, we can then see if the feature data confirms or repudiates these expectations
  - Both are useful!
- We can also then see if the feature data reveals unexpected insights

A priori expectations (2/3)

- What do you think might distinguish the composers?
  - Josquin vs. Ockeghem?
  - Josquin vs. La Rue?
- I consulted one musicologist (Julie Cumming) and one theorist (Peter Schubert), both experts in the period . . .

A priori expectations (3/3)

- Josquin vs. Ockeghem: Ockeghem may have . . .
  - Slightly more large leaps (larger than a 5^{th})
  - Less stepwise motion in some voices
  - More notes at the bottom of the range
  - Slightly more chords (or simultaneities) without a third
  - Slightly more dissonance
  - A lot more triple meter
  - More varied rhythmic note values
  - More 3-voice music
  - Less music for more than 4 voices
- Josquin vs. La Rue: La Rue may have . . . Hard to say!
  - Maybe more varied repetition (melodic and contrapuntal, including rhythm)?
  - Maybe more compressed ranges?

Were our expectations correct?

- Josquin vs. Ockeghem: Ockeghem may have . . .
  - OPPOSITE: Slightly more large leaps (larger than a 5^{th})
  - SAME: Less stepwise motion in some voices
  - SAME: More notes at the bottom of the range
  - SAME: Slightly more chords (or simultaneities) without a third
  - OPPOSITE: Slightly more dissonance
  - YES: A lot more triple meter
  - SAME: More varied rhythmic note values
  - YES: More 3-voice music
  - YES: Less music for more than 4 voices
- Josquin vs. La Rue: La Rue may have . . .
  - UNKNOWN: Maybe more varied repetition (melodic and contrapuntal, including rhythm)?
  - SAME: Maybe more compressed ranges?

Diving into the feature values

- There are a variety of statistical techniques for attempting to evaluate which features are likely to be effective in distinguishing between types of music
- We used seven of these statistical techniques to find:
  - The features and feature subsets most consistently statistically predicted to be effective at distinguishing composers
- We then manually examined these feature subsets to find the features likely to be the most musicologically meaningful

Novel insights revealed (1/2)

- Josquin vs. Ockeghem (93.9%):
  - Rhythm-related features are particularly important
    - Josquin tends to have greater rhythmic variety
      - Especially in terms of both especially short and long notes
    - Ockeghem tends to have more triple meter
      - As expected
    - Features derived from beat histograms also have good discriminatory power
  - Ockeghem tends to have more vertical sixths
  - Ockeghem tends to have more diminished triads
  - Ockeghems tends to have longer melodic arcs

Novel insights revealed (2/2)

- Josquin vs. La Rue (85.4%):
  - Pitch-related features are particularly important
    - Josquin tends to have more vertical unisons and thirds
    - La Rue tends to have more vertical fourths and octaves
    - Josquin tends to have more melodic octaves

Research potential (1/2)

- The results above are the product of an initial accurate but relatively simple analysis
- There is substantial potential to expand this study
  - Apply more sophisticated and detailed statistical analysis techniques
  - Perform a more detailed manual exploration of the feature data
  - Implement new specialized features
  - Look at more and different composer groups

Research potential (2/2)

- Composer attribution is just one small example of the many musicological and theoretical research domains to which features and jSymbolic2 can be applied

Tools used

- All machine learning and feature selection/weighting was performed using the Weka machine learning framework
  - Free and open source
  - Surprisingly easy to use for such technical software

Excluded features

- Only 721 of the available 1230 jSymbolic 2.0 features were used in order to avoid bias
  - Some excluded features were irrelevant to the data under consideration
  - Some excluded features were correlated with the source of the data
- In practice, this primarily meant removing features linked to instrumentation, dynamics and tempo

Sidebar: Avoiding encoding bias

- Including information in a study that is biased based on the source of the data will artificially inflate results
  - e.g. if one set of files has precise tempo markings but another is arbitrarily annotated at 120 BPM
- One must be careful to avoid this problem when using features
  - Ideally, use data that was consistently generated using precisely the same methodology
  - If this is not possible, exclude all features that will perceive the bias

Josquin attribution study (1/2)

- I also did a second study using the JRP data
  - This one focusing on proper attribution of pieces by Josquin
- Jesse Rodin has broken Josquin's music into 6 levels of attribution certainty
- I used the jSymbolic 2.0 features to train a 2-class SVM classifier
  - First class: Josquin
    - The Josquin music in the 2 most secure Rodin levels
  - Second class: NotJosquin
    - All the JRP music available from 21 other Renaissance composers similar to Josquin
- This model was then used to classify the Josquin music in the remaining 4 Jesse Rodin levels:

Josquin attribution study (2/2)

- It turns out that, the more insecure a piece according to Rodin's classification, the less likely it was to be classified as being by Josquin by our classifier
- This demonstrates some empirical support for Rodin's categorizations

Rodin Certainty Level	% Classified as Josquin
Level 3 "Tricky"	48.6%
Level 4 "Questionable"	17.2%
Level 5 "Doubtful"	14.0%
Level 6 "Very doubtful"	5.5%

Current work: Origins of the madrigal

- This is a project Julie Cumming and I will present at MedRen 2018
- Where did the madrigal come from?
  - The frottola (Einstein 1949)?
  - The chanson and motet in Florence (Fenlon and Haar 1988) ?
  - The Florentine carnival song, villotta, and improvised solo song (A. Cummings 2004)?
- How can we decide?
  - Do an analysis with jSymbolic features
- Corpus: All the pieces in Florence BNC 164-167 (c. 1520)
  - Madrigals (27)
  - Motets (12)
  - Frottole and Villotte (24)
  - Chansons (24)

jMIR

Current work: Building digital symbolic music research corpora

- This is a paper Julie Cumming and I will submit to ISMIR 2018
- Presents techniques and workflows for building a corpus optimized for statistically valid empirical music
- Presents several corpora designed this way

Genre classification study (1/4)

- One of the first experiments I performed with the jSybmolic 2.0 features was to classify music according to a variety of genres
  - Including popular music
- I used my SLAC dataset to do this
  - Composed of 250 pieces
- Each piece in SLAC has a matching:
  - MIDI transcription
  - Text file containing lyrics (if any)
  - Audio recording
  - Metadata mined from search engines
    - Containing “cultural” information

Genre classification study (2/4)

- SLAC is divided amongst 10 genres
  - 25 pieces of music per genre
- These 10 genres can be grouped into 5 pairs of similar genre
  - This permits both 5-genre and 10-genre experiments
- The genres are:
  - Blues: Modern Blues and Traditional Blues
  - Classical: Baroque and Romantic
  - Jazz: Bop and Swing
  - Rap: Hardcore Rap and Pop Rap
  - Rock: Alternative Rock and Metal

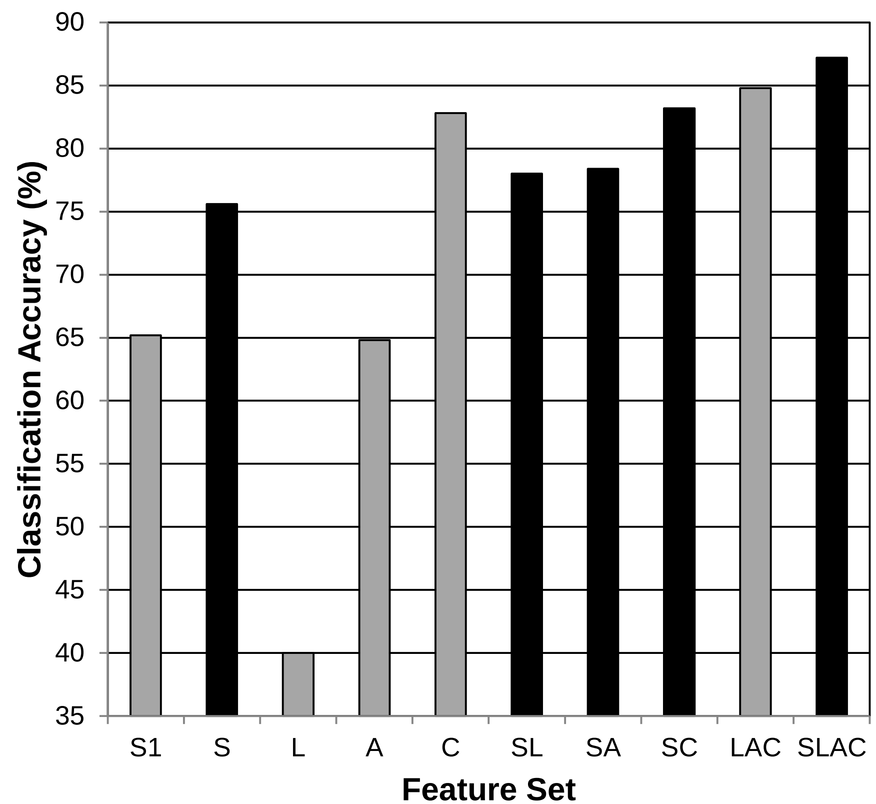
Genre classification study (3/4)

- Using just the MIDI files, the jSymbolic 2.0 features were able to classify the music correctly
  - 10 genres: 75.6% of the time
  - 5 genres: 76% of the time
- Experiments were also performed with other types of features, alone and in various combinations . . .

Genre classification study (4/4)

- S1 = jSymbolic 1.0
- S = jSymbolic 2.0
- L = jLyrics
- A = jAudio
- C = jWebMiner

- Combining different feature groups substantially improved performance:
  - 87.2% amongst 10 classes
- This offers support for multimodal research
  - i.e. research involving different types of data



Overview of jMIR

- jSymbolic is actually part of my larger jMIR toolset
  - Designed specifically for multimodal music research
- 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
  - http://jmir.sourceforge.net
- 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

- jSymbolic: Feature extraction from MIDI files
- jAudio: Audio feature extraction
- jWebMiner: Cultural feature extraction
- jLyrics: 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
- jSongMiner: Metadata harvesting
- jProductionCritic: Detecting mixing and editing errors
- jMIRUtilities: Infrastructure for conducting experiments

SIMSSA and MIRAI context

- The work I have presented is all part of the much larger, multi-institutional SIMSSA and MIRAI projects
- We are also very happy to be able to assist in hosting various other projects
  - Like the wonderful Portuguese Early Music Database
- This project aims to provide public access to as much as possible of the huge number of digitized scores held at libraries around the world, and storing the results in easily accessible and searchable databases
  - A particular focus on using OMR to transform images of scores into digital symbolic formats
  - The goal is also to annotate all this music with jSymbolic features

SIMSSA and MIRAI context

- Not only will this allow music researchers to query scores in relatively traditional ways (e.g. using textual metadata or melodic segments); it will also allow content-based searches based on feature values and ranges
  - A researcher could thus filter results based on the amount of chromaticism in a piece, for example, or the amount of parallel motion between voices
- Can use statistical analysis to build multidimensional combinations of features that allow sophisticated searches
  - e.g. the level of tonality of a piece, where this is estimated based on the values of several existing features
- Can use features to train classification models for directly assisting research by music scholars
  - e.g. identifying composers of Renaissance pieces with unknown attribution

SIMSSA and MIRAI context

- All of this functionality will be accessible via the SIMSSA user interfaces
- The technical work will be done in the background in a distributed and efficient manner using SIMSSA's Rodan workflow management system
- Work is currently underway to implement automatic jSymbolic annotation of pieces as they are added to SIMSSA's ELVIS database, with later expansion to the Musiclibs database
  - https://database.elvisproject.ca
  - https://musiclibs.net

Research collaborations (1/2)

- I enthusiastically welcome research collaborations with other musicologists and theorists
- In particular, I am always looking for ideas for interesting for new features to implement
  - jSymbolic makes it relatively easy to add bespoke features
  - Can iteratively build increasingly complex features based on existing features

Research collaborations (2/2)

- Please do not hesitate to speak to me if you would like more information on:
  - Using jSymbolic
  - How one can apply statistical analysis or machine learning to extracted features
  - How feature values can be visualized and explored manually
- I am also more than happy to show you any of our data or code
  - jSymbolic is open-source and free

Next up: Interactive workshop (1/2)

- 14:30 to 17:30: Interactive workshop on using jSymbolic and Weka
  - NOVA FCSH, R&D Building, Room 1.05
- Please bring your computers
  - If possible, have Java, a text editor and a spreadsheet pre-installed

Next up: Interactive workshop (2/2)

- www.music.mcgill.ca/~cmckay/jSymbolic Workshop/index.html
  - Presentation slides
  - Workshop instructions
  - Workshop data and software

Thanks for your attention!

- **jSymbolic:** http://jmir.sourceforge.net
- **E-mail:** cory.mckay@mail.mcgill.ca

McGill

Schulich School of Music
École de musique Schulich

CIRMMT
Centre for Interdisciplinary Research in Music Media and Technology

Social Sciences and Humanities Research Council of Canada
Conseil de recherches en sciences humaines du Canada

Canada

MARIANOPOLIS COLLEGE

Fonds de recherche sur la société et la culture
Québec

DDMAL
DISTRIBUTED DIGITAL MUSIC ARCHIVES & LIBRARIES LAB

SIMSSA
Single Interface for Music Score Searching and Analysis

jMIR