Applying Music Information Retrieval Techniques to Audio Production Education

Cory McKay
Marianopolis College
Montreal, Canada
Context: How I spend my time (1/2)

- Research on automatic music classification
  - Using machine learning to “teach” computers to classify music into various types of categories
    - Genre, mood, artist, performer, composer, etc.
    - Typically learn to classify unknown music by training on labelled known exemplars
  - Emphasis on multimodal approaches
    - Audio, symbolic, lyrics, “cultural”, etc.
- Development of open source music information retrieval (MIR) research software
  - Especially the jMIR framework
Context: How I spend my time (2/2)

- **Teaching**
  - Mainly sound recording, live computer music performance, psychoacoustics and basic music technology
  - Previously McGill music tech undergrads and sound recording Q-year students
  - Now I teach CEGEP students full-time
    - Both students in the music program and general students
  - **Not** automatic music classification or software development
Educational problems I have observed

- **Students** taking introductory courses in sound recording and production:
  - Are often very musical, but tend to short-change technical concerns
  - Often have a poor ear for detecting technical problems (at least initially)

- **Teachers and TAs** correcting assignments:
  - Spend a lot of time precisely annotating errors
Dare to dream . . .

Wouldn’t it be nice if there were some software that could _automatically proofread_ mixes prepared by students for _technical errors_?

- Like a _spell checker_ or (good) _grammar checker_ for audio engineering

**Benefits:**
- Would _save markers_ a lot of time by automating error annotations
- Would highlight technical errors to students _before_ they submit their work
  - Helping them to recognize and correct errors independently
- Could also be helpful to _amateurs_ making mashups, with home studios, etc.?
Dreams can come true!

- jProductionCritic is a software tool designed to do exactly these things
Well, maybe not *perfectly* (1/2)

- jProductionCritic only detects **technical** errors
  - It does not even attempt to comment on the *(essential)* **artistic** aspects of mixes
  - And even technical errors are sometimes detected **imperfectly** (although usually quite well)
Well, maybe not perfectly (2/2)

- So, **professorial intervention** is still needed when grading assignments
  - But **much less**, and mostly just the fun parts
  - It is unlikely an automated system could ever replace an expert human anyway

- Also, jProductionCritic is intended specifically for **markers, junior students, and amateurs**
  - **Pros and advanced students** are welcome to use it, but they may not need it
Which technical errors are looked for?

- **Dynamics**
  - Digital clipping
  - Insufficient variety in dynamics
  - Insufficient dynamic range
  - Insufficient dynamic range compression

- **Sustained noise and signal distortion**
  - Ground loop hum
  - Narrowband noise
  - Phasing
  - DC bias

- **Instantaneous noise**
  - Edit clicks
  - Other instantaneous noise

- **Channel problems**
  - Stereo channel balance
  - Stereo channel similarity
  - Is not stereo

- **Miscellaneous**
  - Long silences
  - Duration
  - Encoding format
But I want even more error detectors!

- Like all jMIR components, jProductionCritic is designed to be **highly extensible**
  - Fully **open source** and free Java implementation
  - Error detectors are added as **modular plug-ins**
    - The software automatically handles updates to the configuration file, etc. when new error detectors are added

- jProductionCritic is not just a tool
  - It is also a kind of **development framework** designed to encourage MIR researchers to look more at audio production (and it’s about time!)
But what about differences in style?

- Different styles of music can vary significantly
  - One style’s error is another style’s desirable aesthetic characteristic
  - e.g. Noise music vs. Baroque
- jProductionCritic is highly configurable
  - Each error checker can be turned on or off
    - e.g. dynamic compression vs. dynamic variety
  - Each error detector has its own settings controlling its sensitivity
  - So different sets of settings can be used for different styles
    - But good general default settings are available
What do I give jProductionCritic?

- **A final stereo master mix**
  - Makes sure no unchecked errors are introduced during the final mixdown
  - Exports of individual tracks can also be processed if one really wants to, however

- **Any standard audio file format parsable by Java**
  - WAV, AIFF, AU, MP3 😞, etc.
What do I get back?

- Basic text reports
  - Text files and/or at the command line
- HTML reports
  - Can be published to a course page
- Audacity label tracks
  - So that errors can be seen synched to the waveform
- ACE XML and Weka ARFF files
  - Shhh, don’t tell anyone, but jProductionCritic’s output can also be used for machine learning!
But isn’t this functionality already available?

- Pro Tools, Nuendo, etc. do include some basic technical error detection functionality
  - And there are VST and other plug-ins that add more
- **BUT**
  - No single rival offers anywhere near this number of error detectors (16) in one place
  - No rival offers jProductionCritic’s *integrated* reporting or *batch* functionality
  - Many rivals are proprietary *closed source* black boxes
  - Most rivals are quite *expensive*
  - Rivals often focus on *correction* rather than detection
  - No rivals are designed with *education* in mind
  - Many of the competing algorithms seem surprisingly *naïve* . . .
Example: Digital Clipping (1/2)

- What is typically done:
  - Detect clipping if samples are at the representational maximum
    - But normalized signals will be falsely noted as clipped!
  - Some systems therefore only detect clipping if more than a minimum number of consecutive samples are at the maximum
Example: Digital Clipping (2/2)

- Problem: Students are sneaky
  - If a recording clips, they may just attenuate the signal a little to trick the clipping detector

- Simple and relatively effective jProductionCritic solution:
  - Detect clipping if consecutive samples beyond a threshold at any signal level have the same value
Are the detection algorithms awesome?

- Short answer:
  - They are pretty good, but not necessarily super duper

- Most of them work quite well
  - But not perfectly

- They are almost all original
  - But many are based on improvements to existing ideas

- They are designed with the special needs of education in mind
  - e.g. as demonstrated with digital clipping just now

- They can all be improved
  - There are lots of people who know more than me about DSP, production and education
Yay community development!

- So, please feel free to propose improvements to the algorithms!
  - And please invent new ones!

- A primary goal of jProductionCritic is to encourage community involvement

- Gooooooooooopen source!
But does jProductionCritic work?

- An evaluation was done based on 110 mixes from CEGEP student assignments
  - Live and studio recordings (classical and jazz)
  - Mashups (many genres)
- 44 of these were randomly selected and used to tune the error detectors
- The remaining 66 were used to test the tuned detectors
  - Compared results to those produced earlier via manual marking (and, later, remarking)
- Caveat: These results are certainly biased, as there was only one human corrector (me)
Experimental results

- jProductionCritic found 89% of the true errors
  - And 92% of the errors it detected were true errors
- The human corrector found 98%
  - Although the human was (seemingly) infinitely better at avoiding false positives

<table>
<thead>
<tr>
<th></th>
<th>True Positives</th>
<th>False Positives</th>
<th>False Negatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Human</td>
<td>499</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>jPC</td>
<td>452</td>
<td>38</td>
<td>55</td>
</tr>
</tbody>
</table>
Discussion of results

- jProductionCritic is not as good as an expert human
  - But it is much better than inexpert students!
- It is also more than good enough to save an expert corrector a lot of time annotating errors
  - And even find a few that an expert corrector missed (8 in these experiments)
- Three error detectors were responsible for most of jProductionCritic’s problems (73% of them):
  - Phasing (very very bad)
  - General background noise (very bad)
  - Non edit click instantaneous noise (badish)
- The other error detectors performed very well
More information

- Read the upcoming ISMIR paper:
  - November 4 to 8 in Curitiba, Brazil
  - www.ppgia.pucpr.br/ismir2013/

- Try jProductionCritic (and/or mod it):
  - jmir.sourceforge.net
  - There is a nice on-line manual
  - But it won’t be posted until late October

- Let me know what you think:
  - cory.mckay@mail.mcgill.ca