SndsLike & BirdsEarApp: MIR in Action

Stephen T. Pope
stephen@FASTLabInc.com
MAT 240E
Feb 24, 2014

SndsLike Introduction
- Overview
  - Similarity measures
  - Genre/artist clustering
  - Recommender systems
- Background
  - Prior art - big field(s)
    - My previous Music DB work
      - HyperScore ToolKit, MODE (1980s)
      - Paleo, OperaBrowser, Siren (1990s)
      - FMAK 1 for Predixis (MusicIP, GraceNote, OFA)
      - FMAK 2 for Expert Mastering Assistant (2002-04)
      - FMAK 3 for Catalyst search engine (2006-08)
      - FMAK 4 for Imagine Research (2010-12, iZotope)

SndsLike Goals
- Similarity-based "recommender" system aimed at production music data sets (why?)
- Written from scratch (I sold the old code to iZotope)
- Simple, fast, portable, embeddable
  - C++, octave, java, python, (My)SQL DBMS
- Use the “latest features” (> 400 features)
- Use the “latest statistics” (sophisticated de-noising)
- Use the “latest distance metrics” (learned)
- Use existing noisy/partial labels to train clustering and distance metrics
The “Latest Features”
- Standard time- and freq-domain features
- HPF/LPF versions
- Many freq bands
- Chroma, harmonicity, MFCCs & spectral measures
  - Sp-slope, spread, bandwidth, variety, kurtosis, roll-off...
  - Spectral tracking and track birth/death stats (useful)
- Fluctuation pattern features (E Pampalk)
- Beat histograms (G Tzanetakis)
- Statistical Spectrum Descriptors (Lidy & Rauber)
- Several tempo estimates (+ stats)
- Several bass pitch estimates (+ stats) + tracking
- Several chord/key pitch estimates (+ stats)
- Musical segmentation and segment-related features

The Latest Statistics
- Lots of feature-dependent smoothing
  - Data mode: noisy, bi-modal, clicky, etc.
- Take Gaussian Mixture Models (GMM) of all features
- Save gmm-avg, main-lobe width/weight, bi-modality...
- Also save dev, del, del2

Feature Extractor Development
- The Latest Features
  - Standard time- and freq-domain features
  - HPF/LPF versions
  - Many freq bands
  - Chroma, harmonicity, MFCCs & spectral measures
    - Sp-slope, spread, bandwidth, variety, kurtosis, roll-off...
    - Spectral tracking and track birth/death stats (useful)
  - Fluctuation pattern features (E Pampalk)
  - Beat histograms (G Tzanetakis)
  - Statistical Spectrum Descriptors (Lidy & Rauber)
  - Several tempo estimates (+ stats)
  - Several bass pitch estimates (+ stats) + tracking
  - Several chord/key pitch estimates (+ stats)
  - Musical segmentation and segment-related features

- The Latest Statistics
  - Lots of feature-dependent smoothing
    - Data mode: noisy, bi-modal, clicky, etc.
  - Take Gaussian Mixture Models (GMM) of all features
    - Save gmm-avg, main-lobe width/weight, bi-modality...
    - Also save dev, del, del2

- The latest distance metrics
  - Using noisy labels
  - Dimensionality reduction vs clustering
  - PCA
  - FLDA
  - CURE
  - SVMs
  - FLDA training and clusterer app
  - Train on a couple dozen well-known genres
The third stage involves machine-learning or reduction based on numerical/statistical tempo or spectral tracks, followed by pruning data and derives higher-level features (e.g., auto-correlation of several of the distance measures shown in the first figure below shows five different feature vectors to identify the “typical” and “solo” segments, assuming a successful segmentation, we can use this kind of segmentation statistics and derived from segmentation, store these in the database, and use them to prune the base solo segment values for a selected set of features. Then we compute the ratios between the verse and the most average and the most different. Using weightings, we can carry out clustering and identify the boundaries and identify the phrases and verses. Our system can use this kind of segmentation statistics and derive higher-level features (e.g., excellent genre/style correlation).

Music Segmentation

- Detect onsets
- Find regular hierarchy of onsets
- Segment track into verses
- Detect intro/outro
- Detect “solo” verse or bridge
- Calculate segmentation-related features (excellent genre/style correlation)

Analyses core in C++: RMS & FFT features
Wrapper in Python
Call-outs to Java (SSD) and Octave (FP) code
Really painless!
Simple tests
Feature extraction
DB inserts
Higher-level features
Rhythm, key, bass line, SSDs, etc.
**Smalltalk Tools for SndsLike**

**DB insert from Analysis**

INSERT into FSongs (version, name, file, format, album, artist, title, genre, bit_rate, frame_rate, year, ...

**Testing/Demo GUIs**

- **Test GUI button panel**
- In Python & Qt
- Various data plots
- Several tools: gnuplot, XL, etc.
- MySQL tests
- Demo "Player" GUI
- C++/JUCE

**Calling C from Python**

# --- Load C API functions
# load analysis library and its C API functions
libanal = cdll.LoadLibrary(self.conf.analysisLib)
# unsigned convert_file(char * in_name, char * out_name, ...
self.convert_fcn = getattr(libanal, 'convert_file')

df.insert(self.conf.analysisLib)

**Inserting C API functions**

```python
# --- Load C API functions
# load analysis library and its C API functions
libanal = cdll.LoadLibrary(self.conf.analysisLib)
# unsigned convert_file(char * in_name, char * out_name, ...
self.convert_fcn = getattr(libanal, 'convert_file')
```

**Example C API function call**

```c
unsigned convert_file(char * in_name, char * out1_name, ...
```

**Example C API function call**

```c
unsigned convert_file(char * in_name, char * out1_name, ...
```

**Example C API function call**

```c
unsigned convert_file(char * in_name, char * out1_name, ...
```

**Example C API function call**

```c
unsigned convert_file(char * in_name, char * out1_name, ...
```

**Example C API function call**

```c
unsigned convert_file(char * in_name, char * out1_name, ...
```

**Example C API function call**

```c
unsigned convert_file(char * in_name, char * out1_name, ...
```

**Example C API function call**

```c
unsigned convert_file(char * in_name, char * out1_name, ...
```

**Example C API function call**

```c
unsigned convert_file(char * in_name, char * out1_name, ...
```

**Example C API function call**

```c
unsigned convert_file(char * in_name, char * out1_name, ...
```

**Example C API function call**

```c
unsigned convert_file(char * in_name, char * out1_name, ...
```
SndsLike Player

Data Sets

- FASTLab - 14 kSongs, very diverse, “high-quality,” well-encoded
- LikeZebra - 250 kSongs pop/rock
- MegaTrax - 160 kSongs + stems
- AudioNet - 200 kSongs + stems
- Others (not public)

Marketing

Production music houses "still don't get it"
- “Customers aren’t asking for this…”

Performance Problems
- Many versions of the same track with different instrumentations - “stems”
- Same track with/without vocals
- Cover songs

Lessons Learned

I still want it!
- …so please make me one that gets accepted by the on-line services…

They (production music houses, record labels, Gracenote, Apple, Adobe, …) still don't get it.

Q & A

Thank you!