DAY 4

Intelligent Audio Systems:
A review of the foundations and applications of semantic audio analysis and music information retrieval
These lecture notes contain hyperlinks to the CCRMA Wiki.

On these pages, you can find supplemental material for lectures - providing extra tutorials, support, references for further reading, or demonstration code snippets for those interested in a given topic.

Click on the symbol on the lower-left corner of a slide to access additional resources.

WIKI REFERENCES...
Review from Day 3

• What are the 3 major components of a MIR system?
• Let’s hear your “compositions”... (“Tax Man”)
• True or false – it’s important to carefully chose meaningful features
• Follow-up questions on graphical models?
• Implementing?
• Friday grab-bag topics
• BBQ Today - 6PM
• Today’s overview:
  – Obtaining MIR data
  – GMMs
  – Wekainator
OBTAINING MIR DATA: SOCIAL MINING AND MIR GAMES, DATASETS
ANALYSIS AND DECISION MAKING: GMMS
\[ Q(\Theta, \Theta^g) = \sum_{y \in Y} \log (L(\Theta | \mathcal{X}, y)) p(y | \mathcal{X}, \Theta^g) \]

\[ = \sum_{y \in Y} \sum_{i=1}^{N} \log (\alpha_{y_i} p_{y_i}(x_i | \theta_{y_i})) \prod_{j=1}^{N} p(y_j | x_j, \Theta^g) \]

\[ = \sum_{y_1=1}^{M} \sum_{y_2=1}^{M} \cdots \sum_{y_N=1}^{M} \sum_{i=1}^{N} \log (\alpha_{y_i} p_{y_i}(x_i | \theta_{y_i})) \prod_{j=1}^{N} p(y_j | x_j, \Theta^g) \]

\[ = \sum_{y_1=1}^{M} \sum_{y_2=1}^{M} \cdots \sum_{y_N=1}^{M} \sum_{i=1}^{N} \sum_{\ell=1}^{M} \delta_{\ell, y_i} \log (\alpha_{\ell} p_{\ell}(x_i | \theta_{\ell})) \prod_{j=1}^{N} p(y_j | x_j, \Theta^g) \]

\[ = \sum_{\ell=1}^{M} \sum_{i=1}^{N} \log (\alpha_{\ell} p_{\ell}(x_i | \theta_{\ell})) \sum_{y_1=1}^{M} \sum_{y_2=1}^{M} \cdots \sum_{y_N=1}^{M} \delta_{\ell, y_i} \prod_{j=1}^{N} p(y_j | x_j, \Theta^g) \quad (3) \]
Mixture Models (GMM)

- K-means = hard clusters.
- GMM = soft clusters.
Fig. 3.1. Spherical covariance mixture model. Sampled data (*dots*), centres (*crosses*) and one standard deviation error bars (*lines*).
Mixture Models (GMM)

- GMM is good because:
  1. Can approximate any pdf with enough components
  2. EM makes it easy to find components parameters
     - EM - the means and variances adapt to fit the data as well as possible
  3. Compresses data considerably

- Can make softer decisions (decide further downstream given additional information)
GMM Parameters

Input
• Number of components (Gaussians)
  – e.g., 3
• Mixture coefficients (sum = 1)
  – e.g., [0.5 0.2 0.3]
  – “Priors” or “Prior probabilities”
  – Priors are “the original probability that each point came from a given mixture.”
  – “A prior is often the purely subjective assessment of an experienced expert.”
• Initialized centers, means, variances. (optional)

Output
• Component centers/means, variances, and mixture coeff.
• Posterior probabilities
  – “Posterior probabilities are the responsibilities which the Gaussian components have for each of the data points.”

Query
• Obtain similarity via Likelihood
Plot of data and mixture centres
Fig. 3.1. Spherical covariance mixture model. Sampled data (dots), centres (crosses) and one standard deviation error bars (lines).
4. Spherical covariance mixture model with six components fitted to the sampled from the full covariance two-component model in Fig. 3.3. Sampled points, centres (crosses) and one standard deviation error bars (lines).

From Netlab (p2-63)
Fig. 3.2. Diagonal covariance mixture model. Sampled data (dots), centre (crosses), covariance axes (thin lines) and one standard deviation error bars (thick lines).
3. Full covariance mixture model. Sampled data (dots), centres (crosses), axis axes (thin lines) and one standard deviation error bars (thick lines).
GMM

- “Pooled covariance" - using a single covariance to describe all clusters (saves on parameter computation)
GMM: Likelihood

1. Evaluate the probability of that mixture modeling your point.
   
   $\text{likelihood}_{gm1} = \text{gmmprob}(gm1, \text{testing\_features})$
   
   $\text{likelihood}_{gm2} = \text{gmmprob}(gm2, \text{testing\_features})$

   $\log\text{likelihood} = \log(\text{likelihood}\text{Kick} / \text{likelihood}\text{Snare})$

• Log-function is “order-preserving” – maximizing a function vs. maximizing its log gives same results
Minimization Problems

• EM is gradient-based – it does not find the global maximum in the general case, unless properly initialized in the general region of interest.

• Error wants to be $-\infty$, which occurs when Gaussian is fit for each data point. (mean = data point and variance = 0)

• “There are often a large number of local minima which correspond to poor models. Solution is to build models from many different initialization points and take the best model.”
GMM

• Application:
  – State-of-the-art speech recognition systems
  – estimate up to 30,000 separate GMMs, each with about 32 components. This means that these systems can have up to a million Gaussian components!! All the parameters are estimated from (a lot of) data by the EM algorithm.

Application:
Speaker Recognition
GMM System

- **Separate models for** $p(x|\text{sing})$, $p(x|\text{no sing})$
  - combined via likelihood ratio test

- **How many Gaussians for each?**
  - say 20; depends on data & complexity

- **What kind of covariance?**
  - diagonal (spherical?)
Combining Vocal Source and MFCC Features for Enhanced Speaker Recognition Performance Using GMMs

- 623 Speakers
- MFCC and delta-MFCC
- 30 ms frames, 15ms overlap
- GMM
- 24 components
- K-means+EM
- Log-likelihood
- Training set: 20 seconds
- Test set: 7 seconds
GENRE
“Because feature vectors are computed from short segments of audio, an entire song induces a cloud of points in feature space.”

“The cloud can be thought of as samples from a distribution that characterizes the song, and we can model that distribution using statistical techniques. Extending this idea, we can conceive of a distribution in feature space that characterizes the entire repertoire of each artist.”

• Genre Classification:
  • Manual: 72% (Perrot/Gjerdigen)
  • Automated (2002) 60% (Tzanetakis)
  • Automated (2005) 82% (Bergstra/Casagrande/Eck)
  • Automated (2007) 76%

From ISMIR 2007 Music Recommender Tutorial (Lamere & Celma)
How?

• Version 1 - One feature vector per song
  – High-level features extracted from data
    • Timbral (MFCCs, etc), Rhythmic content (beat histogram, autocor, tempos), Pitch info
    • Sampling of the frames in the song
  – Statistics of features extracted from a piece (includes means, weights, etc)
  – Representative of MFCC spectral shape
  – Could further use “Anchor space” where classifiers are training to represent musically meaningful classifiers. (Euclidean distance between anchor space)

• Version 2 - Cloud of points
  – Extract audio every $N$ frames
  – K-Means or GMM representing a “cloud of points” for song
    • Clusters: mean, covariance and weight of each cluster = signature for song/artist/genre
• Automatic annotation
  ❖ Similarity based on classification

From ISMIR 2007 Music Recommender Tutorial (Lamere & Celma)
MORE REAL-WORLD APPLICATIONS
Music Recommendation and Discovery Systems

Today

Tomorrow

All music will be on line
Billions of tracks
Millions more arriving every week
Finding new, relevant music is hard!

If *everything* is online, how do we find it?
“A wealth of content creates a poverty of attention”  
Herbert A. Simon, Nobel Prize Winner

“iPod whiplash”

The Long Tail

Study of 5,000 iPod users:  
80% of plays in 23% of songs  
64% of songs never played
So much feature extraction...

- Features extracted on your host then piped to a server.
- Features only taken on select waveform areas
## Tag breakdown

- **Social tags**
  
  - **Distribution of Tags**

<table>
<thead>
<tr>
<th>Type</th>
<th>Freq</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Genre</td>
<td>68%</td>
<td>Heavy metal, punk</td>
</tr>
<tr>
<td>Locale</td>
<td>12%</td>
<td>French, Seattle</td>
</tr>
<tr>
<td>Mood</td>
<td>5%</td>
<td>Chill, party</td>
</tr>
<tr>
<td>Opinion</td>
<td>4%</td>
<td>Love, favorite</td>
</tr>
<tr>
<td>Instrumentation</td>
<td>4%</td>
<td>Piano, female vocal</td>
</tr>
<tr>
<td>Style</td>
<td>3%</td>
<td>Political, humor</td>
</tr>
<tr>
<td>Misc</td>
<td>3%</td>
<td>Coldplay, composers</td>
</tr>
<tr>
<td>Personal</td>
<td>1%</td>
<td>Seen live, I own it</td>
</tr>
</tbody>
</table>

Courtesy: ISMIR 2007 Recommender Tutorial
Much of last.fm data is currently available via web services, such as:

- User Profile Data
- Artist Data
- Album Data
- Track Data
- Tag Data

http://www.audioscrobbler.net/data/webservices/
Music Recommendation

• Cloud of points from frames of song
  – High-level features extracted from data
  – Classifier: Weighted attribute nearest neighbors or fast distance measures.
  – k-Means or GMM used to create clusters.
  – The mean, covariance and weight of each cluster = signature for the song.

  – Compare distance between other songs (signature) using various techniques to measure distance between probability distributions. (Most similar = closest distance)
• Mahalanobis
  – Normalize the distance between the test point(s) and the existing cluster set

\[ \frac{x - \mu}{\sigma} \]
Distance measures between clusters

- The distances between these clusters are computed using the
  - “Centroid distance”
  - Mahalanobis distance
  - Kullback-Leibler Divergence
  - Earth Movers Distance
GMM

• Sampling