Part B. The Fun Part

Applications of Topic Models to Signal Analysis
Overview of Topics

• Low-Rank Models of Signals
• Separation with Monophonic Mixtures
• Recognition in Mixtures
• Temporal demixing
• Pitch Tracking
• User Interfaces
• Multimodality
Learning Dictionaries

• Low-Rank Dictionaries

• Overcomplete Dictionaries

• Convolutive Dictionaries
Marginals of spectrograms

- The marginals of a 2-D distribution describe the distribution of the two involved variables:
  
  \[ P(f) = \int P(f,t) dt \]
  
  \[ P(t) = \int P(f,t) df \]

- In a “spectrogram distribution” these variables are time and frequency

- The marginals are the power spectrum and the signal envelope
  
  - Nothing special here

- Let’s move to PLCA now
Discovering Music

- Simple piano passage
  - Multiple notes
  - Variety in spectral and temporal distributions
  - Can’t be characterized by a single set of marginals
- Extracted marginals
  - Frequency marginals describe the spectra of the notes
  - Time marginals describe their corresponding energy in time
- Doesn’t require isolated notes
- Without supervision we have discovered musical structure!
Large scale version

• First six bars of Bach’s fugue XVI in Gm
Learning General Dictionaries

• Unlike music, not all sounds are cleanly composed out of discrete elements
  – But they are composed out of some elements!

• We can learn dictionaries for varying sounds
  – The frequency marginals of a PLCA decomposition of a sound’s spectrogram are the dictionary
  – Each dictionary will be unique to that sound
A Speech Dictionary

- Learned elements are characteristic spectra of speech
A Chime Dictionary

- For other sounds the learned elements are suitably adapted
The Geometry of the Dictionary

- Normalized magnitude spectra lie on a simplex
  - Each vertex is a frequency
  - Each point inside the simplex is a combination of frequencies
  - Each sound lies in its own space inside the simplex
Low-rank models

• Modeling the sound space

\[ x_t(f) \sim \sum_z P(f \mid z) P_t(z) \]

• Model defines a convex hull
  – \( P(f \mid z) \) are the vertices
Shift-Invariant Dictionaries

- Regular dictionaries don’t capture temporal information of elements of a sound
  - Using shift-invariant PLCA we can do that

Shift-invariant Dictionary of Speech

- Shift-Invariant dictionaries include temporal elements of sound
  - e.g., in speech we get phones and their inflection
Source Separation

- Using PLCA for sound separation
- Supervised separation
- Semi-supervised separation
- Denoising
Defining the problem

\[ x(t) \quad y(t) \]

\[ m(t) = x(t) + y(t) \]

• An ill-defined problem!
  – “Single-channel source separation”
A Fundamental Assumption

- There's approximate additivity in the time/freq space
Ignoring the Time Dimension

- There’s approximate additivity in the time/freq space

- Which holds for each column independently
Representation on a simplex

• Factoring out column gains this becomes:
  \[ \alpha x + \beta y = m \]
  – With constraints:
  \[
  x, y, m, \alpha, \beta \geq 0 \\
  \sum_{i} x_i = 1, \sum_{i} y_i = 1, \sum_{i} m_i = 1 \\
  \alpha + \beta = 1
  \]

• How do we estimate \( x, y \) given only \( m \)?
Direct search

- Each source lives on a source subspace
- We can try a direct search
  - For 10min training data
    - Each source is ~75,000 points
    - at ~2000 dimensions
  - For 10 sec mixture
    - About 1200 mixture points
    - 5,625,000,000 searches
      (per point!)
- Not very practical!
Low-rank models

• Modeling the source spaces

\[ x_t(f) \sim \sum_z P(f \mid z) P_t(z) \]

• Model defines a convex hull
  – Bases \( P(f \mid z) \) are the vertices

• Estimate with EM:

  **Posterior** \( P_t(z \mid f) \propto P_t(z)P(f \mid z) \)

  **Weights** \( P_t(z) \propto \sum_f P_t(z \mid f)x_t(f) \)

  **Bases** \( P(f \mid z) \propto \sum_t P_t(z \mid f)x_t(f) \)
Basis models

- Decompose \( \mathbf{m} \) using known bases:

\[
\mathbf{m}_t(f) = P_t(x) \sum_z P_x(f | z) P_t(z | x) + P_t(y) \sum_z P_y(f | z) P_t(z | y)
\]

- Known mix
- Unknown source priors
- Known source models
- Unknown source model weights

- Same model as before:

\[
\begin{align*}
P_t(s, z | f) & \propto P_t(s) P_t(z | s) P(s | z) \\
P_t(z | s) & \propto \sum_f P_t(s, z | f) \mathbf{m}_t(f) \\
P_t(s) & \propto \sum_f \mathbf{m}_t(f) \sum_z P_t(s, z | f)
\end{align*}
\]
Obtaining the source estimates

• Explains mixture in terms of the source models

\[ m_t \approx \hat{x}_t + \hat{y}_t \]

\[ \hat{x}_t = \sum_f P_x(f \mid z)P_t(z, x) \]

\[ \hat{y}_t = \sum_f P_y(f \mid z)P_t(z, y) \]

• Once we get \( x \) and \( y \) we can invert to waveforms
But ...

• The learned hulls can be too big
  – We risk source-hull overlaps which are bad

**Bad case**
Source hulls are practically identical

**Desired case**
Source hulls are tight and minimally overlapping
Tightening the source hulls

• Using sparse weights to minimize hull area

\[ x_t(f) \sim \sum_z P(f \mid z) P_t(z) \]

• Shrinking \( P_t(z) \) forces bases to be closer to the training data

• How do we enforce sparsity?
  – We use the entropic prior
    • \( P(q) = e^{-H(q)} \)

Supervised separation

• A mixture will contain the frequency bases of the present sound classes appropriately mixed
  – Since we have a linear model we assume that spectra add linearly
  – Not too far from truth

• If we roughly know the frequency bases of the sounds in a mixture we can separate these sounds

• The catch: the sounds must have dissimilar spectral composition
  – Not too much though
In more detail

- Learn frequency bases of the sound classes in mixture
  - From examples outside the mixture data
- Estimate mixture weights for the known bases from the available mixture spectrogram
  - Assume that the mixture is made up from the known bases
- Do selective resyntheses of the mixture spectrogram using only the frequency bases of one sound type at a time
  - Use phase from the original mixture to back to time domain

Semi-supervised separation

- We usually don’t know the frequency marginals for all sounds in a mixture
  - We might only know some
- Complementary learning
  - Explain input with known marginals of some sounds
  - Explain mystery parts with a new set of learned marginals
  - Learn mixture weights as well to model the mixture
- Unknown sounds are treated as one new class we learn online
- Invert to time as before
Special case - denoising

- Very similar to a Wiener filter
  - Instead of a single and rigid noise model we have a dictionary describing the interfering sounds
- Can be done in two ways
  - Have model of noise, extract extras
  - Have model of target, remove extras
- Quality of results depends on how similar the noise is to the target
  - Superior performance for non-stationary noise removal
- Can also use additional temporal and co-occurrence constrains
  - Markovian structure, etc ...
Recognition in Mixtures

• Regular classification

• The Markov Selection Model
Classification and Sounds

• Using classifiers on mixtures is a shaky idea
  – Most classifiers return a winner-takes-all answer
  – Mixtures are not resolved properly
    \[ P(x + y) \neq P(x) + P(y) \]

• We can use PLCA to alleviate this problem
  – Because our probabilities are additive!
Measuring the presence of marginals

- In PLCA there is a 1-to-1 mapping between frequency and time marginals
  - Spectrum $P(f \mid z_i)$ is modulated by $P(z_i) P(t \mid z_i)$
- The time marginals indicate amount of presence of frequency marginals across the input’s timeline
- The likelihood of $P(f \mid z_i)$ at time $t$ is:

$$\sum \sum P(z_i) P(t \mid z_i)^{P(f,t)}$$
Sound recognition in mixtures

- Use known dictionaries to estimate presence of these sounds in a mixture

\[
P(f,t) = \sum_z P(z) \begin{bmatrix}
P_{shaker}(f | z) \\
P_{cymbals}(f | z) \\
P_{jingles}(f | z) \\
P_{pig}(f | z)
\end{bmatrix} \cdot \begin{bmatrix}
P_{shaker}(t | z) \\
P_{cymbals}(t | z) \\
P_{jingles}(t | z) \\
P_{pig}(t | z)
\end{bmatrix}
\]
Indexing Media with Mixtures

- Example of classifiers running on a movie
  - Notice how mixtures are properly resolved
A Unique Property

- With the PLCA model we have additivity
  \[ P(x + y) = P(x) + P(y) \]

- This is a very powerful model for sounds
  - We can modify existing models to use it instead
The Markov Selection Model

- Use PLCA as a state model for an HMM
Decoding a speech mixture

- Decoding process
  - Consolidate dictionaries of sources
  - Analyze mixture using that dictionary
  - Use each source’s sub-dictionary to compute Markov model likelihoods

- Results in fast search
  - $O(KN^2)$ vs $O(N^{2K})$
  - $N$ parallel Viterbi searches
Superimposed Digit Recognition

- Train on digits \{1,2,3,4,5\}
  - Obtain digit Markov selection models
- Run on mixtures of two mixed digits at a time
  - Pick the two models with highest likelihood
Speaker Separation Challenge Case

Correct classification %

Mixture type: 0dB, -3dB, -6dB, -9dB, Clean

Baseline: Green
Speaker 2: Red
Speaker 1: Blue

Missing Data Imputation

• Spectral holes happen!
  – User editing
  – Poor compression
  – Bandlimited sounds
  – Binary masking based algorithms
  – Aggressive background subtraction

• This results in audible effects we want to reverse

• How do we fill the spectral holes?
  – Lots of statistical approaches
Problem definition

• Given an input spectrogram $S_t(f)$ with missing values how can we obtain the most likely to be correct reconstruction?
  – We will only look into the magnitude part of the spectrogram (phase is easy to recover)

$$S_t(f) = \begin{cases} 
S_t^o(f), \text{the observed values} \\
S_t^m(f), \text{the missing values}
\end{cases}$$

• We will gather available statistics from the input $S_t^o(f)$ in order to guess $S_t^m(f)$
  – Can also measure other representative examples

• This is a typical missing data imputation setup in the statistics literature
Traditional imputation approaches

• K-NN missing data imputation
  – 1. Ignoring missing values, find nearest neighbor to input
  – 2. Substitute missing values with neighbor’s values
    • Optionally average from multiple neighbors

• SVD-based missing data imputation
  – 1. Fill in random values for missing data
  – 2. Perform SVD
  – 3. Replace missing data with model approximation
  – 4. Go to 2 and repeat process until convergence

• Shortcomings
  – K-NN approach cannot model sound mixtures
    • Each datum can only take values from a known sample
  – SVD approach will produce negative values
    • This results into audible noise in reconstruction
Fully observable spectra

- Learning from intact examples
  - All frequencies exist in training data
  - Straightforward EM estimation

- Estimation equations:

  \[
P_t(z | f) = \frac{P_t(z)P(f | z)}{\sum_{z'} P_t(z')P(f | z')}, \quad P_t(z) = \sum_f P_t(z | f)S_t(f) \quad \frac{\sum_i P_t(z | f)S_t(f)}{\sum_f R \sum_i P_t(z' | f)S_t(f)}\]

- Reconstruction equations:

  \[
  \hat{S}_t^m(f) = \alpha_t \sum_z P_t(z)P(f | z), \forall f \in F^m, \quad \alpha_t = \sum_{\forall f \in F} \frac{S_t^o(f)}{\sum_z P_t(z)P(f | z)}
  \]

Partially observable spectra

- Previous formulation doesn’t work columns with missing data
- Estimation equations from input with missing data:

\[
P_t(z \mid f) = \frac{P_t(z)P(f \mid z)}{\sum_{z'} P_t(z')P(f \mid z')}
\]

\[
\hat{S}_t(f) = \begin{cases} 
    \alpha_i \sum_z P_t(z)P(f \mid z), & \forall f \in \mathcal{F}^m \\
    S_t(f), & \forall f \in \mathcal{F}^o
\end{cases}, \quad \alpha_i = \sum_{\forall f \in \mathcal{F}^o} \sum_z P_t(z)P(f \mid z)
\]

- Similar to previous case, this time we iterate over an interim estimate
- We now simultaneously learn and fill-in the data
The simple version

- Fill holes with random values

- Repeat until convergence:
  - Learn a PLCA model using the filled hole data
  - Approximate the holes with PLCA model
Large-scale missing data example

- Filling in a large time/frequency gap
  - 4.3 sec × 3 kHz at its widest
- Model was trained on various instances of piano music in addition to the input itself
Random mask example

- 60% of input was missing through a random mask
- Training tool place on the input only

Upsampling example

- Filling in 75% of the upper frequency register
- Training was done on other rock music examples
Note that mixtures are resolved
Real-world usage – Sound removal

• Removing a sound from a recording
  – User deletes offending area in spectrogram
  – Model is learned from the recording and is used to reconstruct any of the missing data

Spectrogram of output
Real-world usage – Sound removal

- Removing a sound from a recording
  - User deletes offending area in spectrogram
  - Model is learned from the recording and is used to reconstruct any of the missing data

*Spectrogram of mixture*

Real-world usage – Compression recovery

- Extreme compression routines create spectral gaps
  - Same effect with aggressive background subtraction
- These gaps result in “musical noise”
- Using the proposed method we can fill the gaps and resolve the noise issues

[Images of missing data input, restored output, and ground truth]
Temporal Separation

- Temporally-aware Separation
- Dereverberation
- Markov Model-based Separation
PLCA vs Shift Invariant PLCA on audio

- Finding *repeating chunks* instead of *slices*
- Extracts short-time temporal structure
  - More informative description
  - Also less expressive
- Applicable to all previous examples
  - Separation, recognition, denoising, etc.
- Also probabilistic and easy to plug into meta-learning algorithms
Using Shift-Invariance to Remove Echoes

- Formulate reverberation as a time smearing:
  - Same as one component shift-invariant model
    \[ P(f, t) = P_k(f, t) * P_I(t) \]
    \( P \) **Observation**  \( P_k \) **Repeating Kernel**  \( P_I \) **Reverberation (original source)**  \( t \) **Impulse**
  - Very efficient for large problems

- Same idea applies to telephony echo cancellation
  - Robust to non-linear processing
  - No need for doubletalk detection
Markov Models for Separation

• We can also take a statistical look at temporal structure, and ignore rigid dependencies

• A PLCA Markov Model
  – Allows us to separate better when time matters
The Static Model

Dictionary of Source 1

Dictionary of Sound Mixture

Dictionary of Source 2

ICASSP 2011 Tutorial: Applications of Topic
Models for Signal Processing – Smaragdis, Raj
The HMM model

• Impose a Markov chain on dictionary transitions

Each source is modeled by a sequence of transitions across multiple dictionaries
N-FHMM Separation Comparison

- Source 1 - Saxophone playing C-E-G four times
- Source 2 - Saxophone playing G-E-C four times on the same octave
- Factorization completely fails. Both of the “separated” results sound and look like the mixture.
- The N-FHMM model still does a good job of separation.

Pitch Tracking Models

• Using the shift-invariant model for pitch tracking

• Naturally extends to polyphonic pitch tracking
PLCA for pitch tracking

- Constant-Q transform makes transposition a shift in frequency
  - Use shift-invariant analysis on that axis too
- Kernel will be source spectrum
- Impulse will be pitch probability function
- Doesn’t matter if the source is harmonic or not!
Tricky case 1: Double notes

- All common pitch trackers fail at tracking multiple notes
- Additive model/features are working out fine for us though
- Makes pitch tracking in noise easier
Tricky case 2: Inharmonic sounds

- Pitch percept and actual frequency are different things
- This approach relates to how we perceive pitch, not how physics explains it
- Output here is what we hear, not what’s technically true
Tricky case 3: Multiple pitch tracks

- Two-part example
  - Voice + bell mixture
  - “Weights” provide a rough pitch track for each source
  - Inharmonicity of bell is not an issue
  - Both spectral profiles are extracted

- Additivity saves us again here

- Once more this is still a zero assumption model!!
Applications on images

- Images can be interpreted as distributions as well
  - \{x,y,r,g,b\} photon counter
- Shift-invariant PLCA finds repeating patterns
- Extensions with other types of kernel variations (rotation, scale, ...)

Audio-visual example

- Input with correlated images and pictures
  - Results into multimodal features
  - Reveals key/note relationship
Removing audio/visual objects
Object-based editing

- Using modern tools, visual object selection is a trivial process
  - Enabling a new breed of content makers, which was not possible in the past
- This allows us to perform object-based editing

Moving towards the audio world

• Sounds superimpose
  – i.e., are transparent
  – How do you select the layer you want?
  – What’s behind matters now!

• Sounds are inherently 1-D
  – How do you select an object that’s all over the place?
Object-based editing in audio

- In audio things are not that intuitive
  - Popular visualizations do not have clear connections to auditory objects

- What is this input? Where are the objects?
Rethinking the audio interface

- Graphical interfaces for audio don’t provide the user with intuition

- Instead of drawing the target, have the user “mimic” the target
  - Vocalize, whistle, synthesize, ... anything that provides some hints that help segmentation

- This isn’t straightforward though ...
Fitting to the PLCA framework

\[
\left\{ P_u(f \mid z), P_u(t \mid z) \right\}
\]

\[
\begin{bmatrix}
P_t(f \mid z) \\
P_b(f \mid z)
\end{bmatrix}
\begin{bmatrix}
P_t(t \mid z) \\
P_b(t \mid z)
\end{bmatrix}
\]

Provides a dictionary and activations that fit intended source

Decompose mixture such that some of the extracted components use known elements as priors

Update with priors

\[
P_t(f \mid z) \propto \sum_i P(f, t) P(z \mid f, t) + \kappa P_u(f \mid z)
\]

Separate mixture using known approach, components with priors correspond to target

User input

Mixture input

Analysis

Segmentation result

Back to the original example

- User sings the vocal part to select
Back to the original example

- Coughing the snare drum
Inexact guidance

- Whistling to point to a guitar
Another inexact case

- Dense mixture with an approximate user guidance
More examples ...

• Idiot removal

• Singing off-key

• What are the lyrics?
Recap

• Source Separation Models
  – Supervised and semi-supervised
• Recognition in Mixtures
• Missing Data Applications
• Temporal Models
• Pitch Tracking
• Images and Multimodal data
• User Interfaces
Parting Thoughts

• Mixtures are important
  – We can’t rely on the good old $+ n(t)$

• Topic Models gracefully address the issue
  – Can easily substitute the Gaussian in many cases

• There is a wealth of models that can be modified as such and deal properly with mixtures
Bibliography

- Singh, R., B. Raj and P. Smaragdis. 2010. Latent-variable decomposition based dereverberation of monaural and multi-channel signals, in proceedings IEEE International Conference on Audio and Speech Signal Processing, Dallas, TX, USA March 2010. [PDF]