Learning blind one-microphone speech separation

Francis Bach  
Ecole des Mines de Paris  
francis.bach@mines.org

Michael Jordan  
UC Berkeley  
jordan@cs.berkeley.edu
Blind one-microphone speech separation

- Two or more speakers \( s_1, \ldots, s_m \) - one microphone \( x \)
- Ideal acoustics: \( x = s_1 + s_2 + \cdots + s_m \)
- **Goal**: recover \( s_1, \ldots, s_m \) from \( x \)
- **Blind**: without knowing the speakers in advance
Approaches to one-microphone speech separation

- Mixing model: \( x = s_1 + s_2 + \cdots + s_m \)
- Two types of approaches:
  1. Generative
     - Learn source model \( p(s) \), then "simply" an inference problem
     - Model too simple: does not separate
     - Model too complex: inference potentially intractable
     - Works for non blind situations (Roweis, 2001, Lee et al., 2002)
Approaches to one-microphone speech separation

- Mixing model: \( x = s_1 + s_2 + \cdots + s_m \)
- Two types of approaches:
  1. Generative
     - Learn source model \( p(s) \), then "simply" an inference problem
     - Model too simple: does not separate
     - Model too complex: inference potentially intractable
     - Works for non blind situations (Roweis, 2001, Lee et al., 2002)
  2. Discriminative: model of separation task, not of speakers
Spectrogram
Sparsity and superposition

\[ s_1 + s_2 = x \]
Reformulation as segmentation

- *Empirical property*: there exists a segmentation that leads to audibly acceptable signals, e.g., take \( \text{arg max}(\|S_1\|, \|S_2\|) \)

Spectrogram of the mix

“Optimal” segmentation

cf. *time frequency masking*
Reformulation as segmentation

- *Empirical property:* there exists a segmentation that leads to audibly acceptable signals, e.g., take $\arg \max(|S_1|, |S_2|)$

  cf. *time frequency masking*

- Requires new way of segmenting images
Segmenting images for speech separation

- “Speech segments” are very different from “vision segments”
Segmenting images for speech separation

- “Speech segments” are very different from “vision segments”
- Designing segmenter by hand is cumbersome
- Why not learn it directly from data? Requires:
  1. labelled examples
  2. machine learning algorithm
Learning problem

• Data:
  – Artificially generated spectrograms
  – Corresponding segmentations
• Goal: learn how to segment new spectrograms
Learning problem

• Data:
  – Artificially generated spectrograms
  – Corresponding segmentations

• Goal: learn how to segment new spectrograms

• We propose a two stage approach:
  1. Build features adapted to speech segments
  2. Learn how to segment from those features (clustering)
Features for speech separation

• Usual grouping cues from speech psycho-physics and computational auditory scene analysis (CASA)
Features for speech separation

- Usual grouping cues from speech psycho-physics and computational auditory scene analysis (CASA)
- Non harmonic cues (same as in vision)
  - Continuity
  - Common fate
    - Common offsets/onsets
    - Frequency co-modulation (frequencies move in sync)
Features for speech separation

- Usual grouping cues from speech psycho-physics and computational auditory scene analysis (CASA)
- Non harmonic cues (same as in vision)
  - Continuity
  - Common fate
    - Common offsets/onsets
    - Frequency co-modulation (frequencies move in sync)
- Harmonic cues
  - Pitch
  - Timbre
Building features

- For each cues, build a “feature map”
Building features

• For each cues, build a “feature map”
• Feature I: continuity
  – Time/frequency are usual features for continuity
Features II: common fate cues

- Oriented edge filters used in vision
  - Vertical: common offsets and onsets
  - Other angles: frequency co-modulation
Features III: harmonic cues

- Estimation of pitch for multiple speakers:
  - Simple estimation based on independent frames and spline smoothing
Features for speech separation

- Characteristics of features …
  - Numerous
  - Noisy or very noisy
- … impose constraints on clustering algorithm
  - Robust to noise
  - Flexible enough to account for various cluster shapes
- Spectral clustering
Spectral clustering

- Consider N data points (e.g., pixels) as weighted graph
  - N vertices: one vertex per data point
  - Weight: $W_{ij} \geq 0, \ i, j \in \{1, \ldots, N\}$
  - $W_{ij}$ large if points i and j likely to be in the same cluster
- $W \in \mathbb{R}^{N \times N}$ = similarity matrix
- Goal: find clusters with high intra-similarity and low inter-similarity
Spectral clustering

- $W \in \mathbb{R}^{N \times N}$ = similarity matrix
- Goal: find clusters with high intra-similarity and low inter-similarity
- Criterion: normalized cut = \[
\frac{\text{Sum of inter-cluster weights}}{\text{Sum of intra-cluster weights}}
\]
- Goal: find partition that minimizes the normalized cut
Spectral clustering

• $W \in \mathbb{R}^{N \times N}$ = similarity matrix

• Goal: find clusters with high intra-similarity and low inter-similarity

• Criterion: normalized cut = $\frac{\text{Sum of inter-cluster weights}}{\text{Sum of intra-cluster weights}}$

• Goal: find partition that minimizes the normalized cut

• NP hard – but can be relaxed in an eigenvalue problem
Overview of spectral clustering algorithm: clustering into R clusters

- Given similarity matrix $W \in \mathbb{R}^{N \times N}$
  1. Find first R eigenvectors $U = (u_1, \ldots, u_R) \in \mathbb{R}^{N \times R}$
  2. Cluster U (considered as N points in R dimensions) using K-means → output partition E(W)
Overview of spectral clustering algorithm: clustering into R clusters

- Given similarity matrix $W \in \mathbb{R}^{N \times N}$
  1. Find first R eigenvectors $U = (u_1, \ldots, u_R) \in \mathbb{R}^{N \times R}$
  2. Cluster U (considered as N points in R dimensions) using K-means → output partition E(W)

- Properties:
  - Flexible clusters
  - State-of-the-art in vision (Malik et al.)
  - Naïve running time complexity $O(N^3)$
Overview of spectral clustering algorithm: clustering into R clusters

- Given similarity matrix $W \in \mathbb{R}^{N \times N}$
  1. Find first $R$ eigenvectors $U = (u_1, \ldots, u_R) \in \mathbb{R}^{N \times R}$
  2. Cluster $U$ (considered as $N$ points in $R$ dimensions) using K-means → output partition $E(W)$

- Properties:
  - Flexible clusters
  - State-of-the-art in vision (Malik et al.)
  - Naïve running time complexity $O(N^3)$

- Two challenges:
  - (1) learning from examples
  - (2) complexity
Learning spectral clustering

• Spectral clustering: Given similarity matrix \( W \in \mathbb{R}^{N \times N} \)
  1. Find first \( R \) eigenvectors \( U = (u_1, \ldots, u_R) \in \mathbb{R}^{N \times R} \)
  2. Cluster \( U \) (considered as \( N \) points in \( R \) dimensions) using \( K \)-means \( \rightarrow \) output partition \( E(W) \)

• Learning spectral clustering:
  – Given \( E \), find \( W \) such that \( E \) and \( E(W) \) are close
  – Solution proposed in earlier work
    (Bach & Jordan, NIPS 2004)
    • Designing appropriate differentiable cost function
Linear time complexity

- Naïve approaches using full matrices: $O(N^3)$
- Linear complexity: $O(N)$
  - Sparse matrices (short range interactions)
  - Low-rank approximations (long range interactions)
  - Band diagonal matrices (medium range interactions)
- Ranges of interactions in speech
Spectral clustering for speech separation

- **TEST**: Given spectrogram with N pixels to segment:
  - build features: \( x \in \mathbb{R}^{D \times N} \)
  - build (parameterized) similarity matrix
    \[
    W_{ij} = e^{-\sum_k \alpha_k (x_{ki} - x_{kj})^2}
    \]
  - Cluster using spectral clustering
  - Obtain speech signal by spectrogram inversion
Spectral clustering for speech separation

• **TEST**: Given spectrogram with N pixels to segment:
  – build features: $x \in \mathbb{R}^{D \times N}$
  – build (parameterized) similarity matrix
    \[ W_{ij} = e^{-\sum_k \alpha_k (x_{ki} - x_{kj})^2} \]
  – Cluster using spectral clustering
  – Obtain speech signal by spectrogram inversion

• **TRAIN**: Given spectrograms and segmentations, learn parameters $\alpha \in \mathbb{R}^{D}$
  – Feature weighting and feature selection
Experiments

- Two datasets of speakers (one for train, one for test)
  - "optimal" segmentation
  - Segmentation result

- Testing time (Matlab/C) : $T = \text{duration of signal (in sec)}$
  - Building features: $\approx 4 \times T$
  - Segmentation: $\approx 30 \times T$
## Sound demos

<table>
<thead>
<tr>
<th>Input (mixed signal)</th>
<th>Outputs (separated signals)</th>
</tr>
</thead>
<tbody>
<tr>
<td>English 1</td>
<td></td>
</tr>
<tr>
<td>English 2</td>
<td></td>
</tr>
<tr>
<td>French 1</td>
<td></td>
</tr>
<tr>
<td>French 2</td>
<td></td>
</tr>
</tbody>
</table>
### Sound demos

<table>
<thead>
<tr>
<th>Input (mixed signal)</th>
<th>Outputs (separated signals)</th>
</tr>
</thead>
<tbody>
<tr>
<td>English 1</td>
<td></td>
</tr>
<tr>
<td>English 2</td>
<td></td>
</tr>
<tr>
<td>French 1</td>
<td></td>
</tr>
<tr>
<td>French 2</td>
<td></td>
</tr>
</tbody>
</table>

- **Issues**
  - Male vs. female
  - French is easier than English

- **Usual problems**
  - Full overlap of some harmonics
  - Switching between speakers (requires oversegmentation)
Conclusion and future work

• Discriminative approach to speech separation
• Learning how to segment spectrograms from examples
  – Clustering of large set of “physical” features
• Current/future work:
  – Benchmarks and separability measure
  – Mixing conditions: allow some form of echo or delay
  – Speaker vs. speaker → speaker vs. non stationary noise
  – Better post processing of spectrogram segmentation?
  – Iterate feature extraction and separation