GHENT
UNIVERSITY

FACULTY OF ENGINEERING
AND ARCHITECTURE

Sound separation

using ad-hoc distributed microphone arrays

Martijn Meeldijk — martijn.meeldijk@ugent.be

GHENT
UNIVERSITY

Overview

Introduction: Ad-hoc microphone arrays

Ad-hoc microphone arrays

Also known as acoustic sensor networks (ASN):

- Several microphones
- Unknown locations in a room

Increasing interest due to:

- Declining cost of receivers
- Edge devices such as laptops, phones, tablets

Challenges

Often different nodes are only weakly synchronized

- Algorithm should be able to deal with this

Limited bandwidth

- Try to exchange little information
- Limit amount of computations

Problems

Clustering

- Divide receivers into clusters dominated by a source
- Optional extra cluster for background noise or interference

Classification

- E.g. speech, music, noise, ...

Separation

- Enhancing the signal for a given source

Clustering source dominated microphones

GHENT
UNIVERSITY

Fuzzy Clustering

Instead of hard clustering

- Assign fuzzy membership values (FMV)
- Indicates how much a microphone belongs to a cluster

Fuzzy Clustering
Steps

Extract Mod-MFCC features

- Feature vector based on modulations om Mel-Frequency cepstral coeficcients
  - Processed in blocks of 4s
- Averaged over time, so behaves more stationary compared to short-time audio features.

Fuzzy Clustering

Steps

Assign fuzzy-membership values using the Fuzzy C-means Algorithm by evaluating least-squared error functional

$$J_m = \sum_{d=1}^{D} \sum_{n=1}^{N} (\mu_{nd})^{\alpha} ||\mathbf{v}_d - \mathbf{u}_n||_{\beta}^2$$

- N clusters
  - N cluster centers $\mathbf{u}_n = (\mathbf{u}_{n,1}, \mathbf{u}_{n,2}, ..., \mathbf{u}_{n,A})^T$
- D observations $\{\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_D\}$
- Weighting exponent α and distance norm $||||_{\beta}$ with weighting matrix β

Detecting the background cluster

- An extra cluster is added for microphones where background noise is dominant
- Microphones assigned to a source cluster should have the background cluster as second highest FMV

Fuzzy Clustering

Background cluster FMV

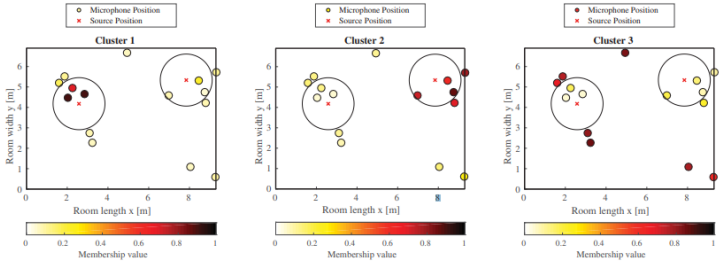
Mic. d	Cluster n		
	1	2	3
1	0.1	*0.3*	**0.6**
2	**0.5**	*0.3*	0.2
3	**0.7**	*0.2*	0.1
4	*0.25*	0.2	**0.55**
5	*0.25*	**0.6**	0.15
6	0.15	**0.65**	*0.2*
7	0.05	*0.15*	**0.8**

Figure: FMV's for each microphone

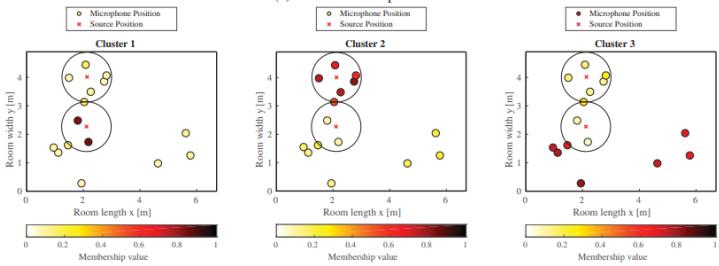
	Cluster n		
	1	2	3
Mics. in Cluster 1	**0.6**	*0.25*	0.15
Mics. in Cluster 2	*0.2*	**0.63**	0.17
Mics. in Cluster 3	0.13	*0.22*	**0.65**

Figure: Average FMV's per cluster

PERFORMANCE



(a) Distant source positions.



(b) Close source positions.

Fuzzy Clustering

Source separation

1. Estimate relative time-differences-of-arrival (TDOA)
2. Combine with FMV in beamforming stage
3. Apply cluster-related spectral masks to the output of the beamformers

Estimate relative time-differences-of-arrival (TDOA)

- Select reference microphone (highest FMV)
- Initial source signal estimation
  - Calculate spectral mask for reference microphone
  - $$\mathscr{M}_n(k,b) = \begin{cases} 1 & |X_{R_n}(k-b)| > \frac{1}{B}\sum |X_{R_j}(k,b)|, \\ & \quad j = 1,\ldots,N \text{ and } j \neq n \\ 0 & \text{otherwise} \end{cases} \quad (1)$$
  - Apply to all microphones in cluster
- Correlation analysis with respect to the reference microphone to compute TDOAs

CLUSTERING-STEERED BEAMFORMING

$$\hat{s}_{n,\,\text{W-DSB}}(l) = \sum_{i_n} w_{n,i_n} x_{i_n}(l + D_{i_n})$$

- l: discrete time index
- D_{i_n}: relative TDOA's
- w_{n,i_n}: the weights allocated to each microphone i_n of cluster n
- These are proportional to the FMV

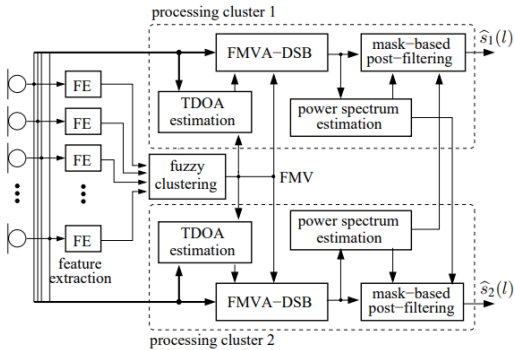
By using a weighted combination of the microphone signals, we can make a better DSB with a better output SNR than uniform weighting.

APPLY CLUSTER-RELATED SPECTRAL MASKS

TO THE OUTPUT OF THE BEAMFORMERS

- Compute a post-filtering mask, similar to what we did before
- $$\mathscr{M}_n(k,b) = \begin{cases} 1 & |\hat{S}_{n,\text{FMVA-DSB}}(k,b)| > \frac{1}{B}\sum |\hat{S}_{j,\text{FMVA-DSB}}(k,b)|, \\ & \quad j = 1,\dots,N \text{ and } j \neq n \\ 0 & \text{otherwise} \end{cases} \tag{2}$$
- Yields final enhanced estimate of the source signal

OVERVIEW
(FOR TWO SOURCES)



This approach yields a slightly better performance than simple DSB.

Federated Learning

GHENT
UNIVERSITY

Federated Learning

Is a machine learning technique that that trains a model distributed over multiple decentralized devices. Data is not shared between nodes. Instead, a three step process is followed:

1. The clients synchronize with the server by downloading the latest model parameters
   - This is a column vector θ^{τ}
2. Each client independently improves its own parameters θ_i^{τ} with stochastic gradient descent on their own data
3. Each client uploads its model parameter updates $\Delta\theta_i^{\tau}$ to the server, where they are aggregated.

$$\boldsymbol{\theta}^{\tau+1} = \sum_{i=1}^{M} \frac{|\boldsymbol{D}_i|}{|\boldsymbol{D}|} \Delta\boldsymbol{\theta}_i^{\tau} \tag{3}$$

- M: number of clients
- $|\boldsymbol{D}_i|$ the cardinality of the dataset of the i-th client
- $|\boldsymbol{D}|$: the cardinality of the total dataset

CLUSTERED FEDERATED LEARNING

Problem

- Doesn't work if the different clients' data comes from different (incongruent) distributions.
- There's no single set of parameter updates that can optimally minimize the loss of all clients at the same time.

CLUSTERED FEDERATED LEARNING

Problem

- Doesn't work if the different clients' data comes from different (incongruent) distributions.
- There's no single set of parameter updates that can optimally minimize the loss of all clients at the same time.

Solution

Cluster clients with a similar distribution and train a seperate server model for each cluster.

CLUSTERED FEDERATED LEARNING

- Calculate cosine similarity between update vectors of clients i and j, represented in matrix A, where:

$$a_{i,j} = \frac{\langle \Delta\boldsymbol{\theta}_i, \Delta\boldsymbol{\theta}_j \rangle}{||\Delta\boldsymbol{\theta}_i, \Delta\boldsymbol{\theta}_j||} \tag{4}$$

  - $\langle \cdot \rangle$ denotes the inner product
  - $||\cdot||$ denotes the L_2 norm
- Cluster hierachically using bi-partitioning
  - Resulting two clusters c_1 and c_2 derived such that

$$\max_{\forall i \in c_1, k \in c_2} (a_{i,k}) < \min(\min_{\forall i,j \in c_1} (a_{i,j}), \min_{\forall k,l \in c_2} (a_{i,j})) \tag{5}$$

  - maximum cross-cluster cosine similarity is always smaller than the minimum of either intra-cluster cosine similarities

CLUSTERED FEDERATED LEARNING

- Verification using the mean and maximum Euclidian norms of weight update vectors

$$\Delta\bar{\theta}_c = \left\| \frac{1}{|c|} \sum_{i \in c} \Delta\boldsymbol{\theta}_i \right\| \text{ and } \Delta\hat{\theta}_c = \max_{i \in c}(\|\Delta\boldsymbol{\theta}_i\|) \tag{6}$$

  - If server reaches stationary solution but clients are still converging
  - $\Delta\bar{\theta}_c$ will be low and $\Delta\hat{\theta}_c$ high, this prompts bi-partitioning

MEMBERSHIP VALUES

To assess contribution of each node to its respective cluster. For two clusters:

- Calculate average intra-cluster similarities for each client i , arranged in vector $\boldsymbol{q}$

$$q_i = \frac{1}{|c_x| - 1} \sum_{j \in c_x / i} a_{i,j} \tag{7}$$

- Calculate average cross-cluster similarities for each client i, arranged in vector $\boldsymbol{r}$

$$r_i = \frac{1}{|c_y|} \sum_{k \in c_y} a_{i,k} \tag{8}$$

- $\forall i \in c_x$ and $(c_x, c_y) \in \{(c_1, c_2), (c_1, c_1)\}$, where $|\cdot|$ denotes the cardinality

MEMBERSHIP VALUES

- Now apply min-max normalization to obtain vector $\boldsymbol{p}$ with:

$$p_i = \lambda \frac{q_i - \min(\boldsymbol{q})}{\max(\boldsymbol{q}) - \min(\boldsymbol{q})} + (1 - \lambda) \frac{r_i - \min(\boldsymbol{r})}{\max(\boldsymbol{r}) - \min(\boldsymbol{r})} \tag{9}$$

- Select the node with the smallest p_i as reference node
- The membership value of a node is the cosine similarity between this node and the reference node

$$\mu_i = a_{i,\arg\min(p_j)}, \forall i, j \text{ and } c_x \in \{c_1, c_2\} \tag{10}$$

- Now, apply min-max normalization again to the vector $\boldsymbol{\mu}$

CLUSTER MEMBERSHIP VALUES

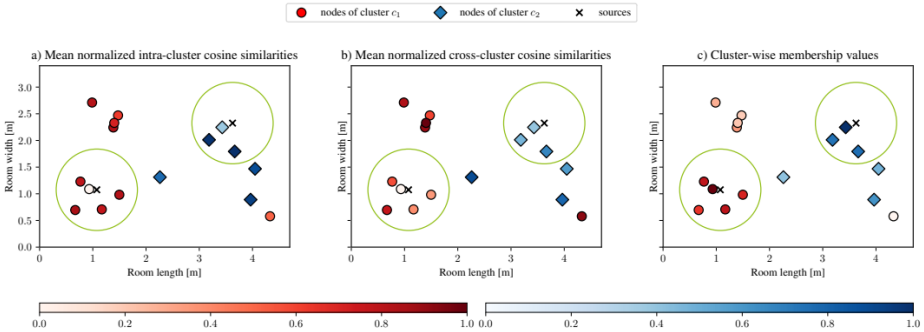


Figure: FMV's for unsupervized CFL

Coherence-based clustering

GHENT
UNIVERSITY

Coherence-based clustering

With non-negative matrix factorization

- A signal consisting of clean speech and interference:

$$x_m(t) = s_m(t) + v_m(t) \tag{11}$$

- We'll consider the signal as a frame of observation samples in vector form

$$\begin{aligned} \boldsymbol{x}_m(t) &= [x_m(t)x_m(t-1)\cdots x_m(t-T+1)]T \\ &= \boldsymbol{s}_m(t) + \boldsymbol{v}_m(t) \end{aligned}$$

- With T denoting frame size

Coherence-based clustering

With non-negative matrix factorization

The proposed clustering algorithm works in two steps:

1. Compute the magnitude squared coherence between the microphone observations to measure their degree of linear dependency
2. Apply NMF to the output of the previous step to find the optimal clustering

MAGNITUDE-SQUARED COHERENCE

- We can measure the coherence between two signals $x(t)$ and $y(t)$ by computing the FFT of the signals.
- Then, we measure the coherence as a function of the center frequency of the filter.

$$\Gamma_{xy}(f) = \frac{|S_{xy}(f)|^2}{S_{xx}(f)S_{yy}(f)} \tag{12}$$

- $S_{xy}(f)$ is the cross-spectral density, which is really a spectral correlation density. This tells how the two signals are correlated in the time domain at a certain frequency and can be computed as:

$$S_{xy}(f) = \sum_{k=1-T}^{T-1} R_{xy}(k)e^{-i2\pi fk} \tag{13}$$

MAGNITUDE-SQUARED COHERENCE

$R_{xy}(k)$ from the previous slide is the cross-correlation function between $x(t)$ and $y(t)$ and can be estimated by:

$$R_{xy}(k) = \begin{cases} \frac{1}{T}\sum_{0}^{T-1-k} x(t)y(t+k) & k = 0, \dots, T-1 \\ R_{xy}(-k) & k = -(T-1), \dots, -1 \end{cases} \tag{14}$$

MAGNITUDE-SQUARED COHERENCE

Now we'll make a matrix consisting of all the correlations between the different microphone signals. We'd like to give the same weight to all frequency bins, regardless of their power. So we'll use the following coherence metric:

$$C_{xy} = \frac{\sum_{f=0}^{F} \Gamma_{xy}(f)}{F} \in [0, 1] \tag{15}$$

- With F denoting the number of frequency bins

Arranging all of the coherence measures between observations into a matrix, we obtain $\boldsymbol{C}$

$$\boldsymbol{C} = \begin{bmatrix} 1 & \cdots & \cdots & C_{1M} \\ C_{12} & 1 & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ C_{1M} & C_{2M} & \cdots & 1 \end{bmatrix} \tag{16}$$

Non-negative matrix factorization

- Every value C_{ij} contains the degree of correlation between the i-th and j-th observation. This means that microphones close to a certain source will be correlated.
- The matrix $\boldsymbol{C}$ can be modelled as:

$$\boldsymbol{C} = \boldsymbol{B}\boldsymbol{B}^T \odot (\mathbf{1} - \boldsymbol{I}) + \boldsymbol{I} \tag{17}$$

  - $B \in \mathbb{R}^{M \times K}$ is the cluster matrix, where K is the amount of speakers (the amount of clusters)
  - Because $\boldsymbol{C}$ is symmetric, we model it as $\boldsymbol{B}\boldsymbol{B}^T$
  - $\odot$ is the element-wise product
  - $\boldsymbol{I}$ is the identity matrix
  - $\mathbf{1}$ is the all-ones matrix
  - The latter two are introduced because the main diagonal of $\boldsymbol{C}$ does not provide any relevant information in the learning process of $\boldsymbol{B}$

NON-NEGATIVE MATRIX FACTORIZATION

- Based on Euclidian divergence, $\boldsymbol{B}$ is estimated with multiplicative update rules. It is first initialized with random values in the algorithm:

$$\boldsymbol{B} \leftarrow \boldsymbol{B} \odot \frac{(\boldsymbol{C} \odot (\mathbf{1} - \boldsymbol{I}))\boldsymbol{B}}{(\boldsymbol{B}\boldsymbol{B}^{\boldsymbol{T}} \odot (\mathbf{1} - \boldsymbol{I}))\boldsymbol{B}} \tag{18}$$

- Now each column of $\boldsymbol{B}$ contains the contribution of a microphone to each cluster. We can obtain the clustering result with:

$$\gamma_m = \{j \in [1, K] : B_{mj} \geq B_{mk}, \forall k \in [1, K]\} \tag{19}$$

- γ_m denotes the cluster assigned to the m-th microphone. This is simply the largest value of column m.

For those wondering, the variance ratio criterion (VRC) is used to obtain the optimal number of clusters

Non-negative matrix factorization

Results

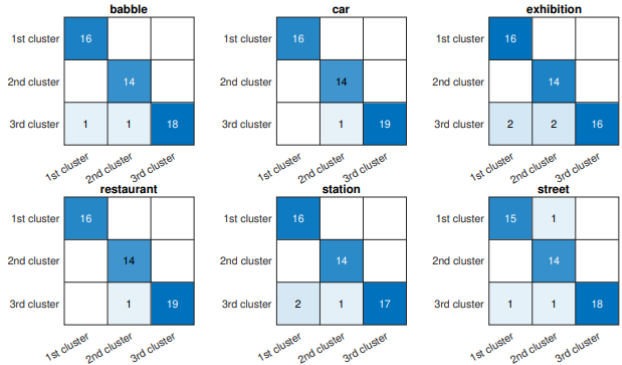


Figure: Confusion matrix for a simulation with 3 clusters and 50 microphones. For reverberation time $T_{60} = 200ms$

Non-negative matrix factorization

Results

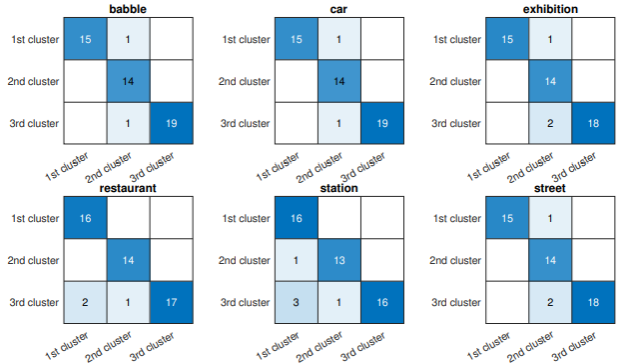


Figure: Confusion matrix for a simulation with 3 clusters and 50 microphones. For reverberation time $T_{60} = 400ms$

- Allows clustering without any prior knowledge of the acoustic scene
- Improved clustering performance in comparison to other methods

Overview

Questions?

GHENT
UNIVERSITY

FACULTY OF ENGINEERING
AND ARCHITECTURE

Sound separation

using ad-hoc distributed microphone arrays

Martijn Meeldijk — martijn.meeldijk@ugent.be

GHENT
UNIVERSITY