Increasing speaker invariance in unsupervised speech learning by partitioning probabilistic models using linear siamese networks

Arvid Fahlström Myrman

KTH Royal Institute of Technology
Department of Speech, Music and Hearing

20th June 2017

Why unsupervised speech recognition?

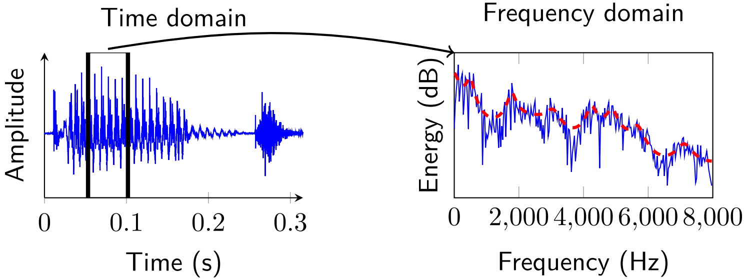
- Speech recognition traditionally supervised
  - Need transcription in addition to audio
  - Very costly to develop data
  - Lack of quality data for most of the world's languages
- Unsupervised recognition: Learn from only audio, without transcription
  - Easier to develop speech systems for low-resource languages
  - Useful for linguistic research
  - Could model language acquisition of infants

Why unsupervised speech recognition?

- Speech recognition traditionally supervised
  - Need transcription in addition to audio
  - Very costly to develop data
  - Lack of quality data for most of the world's languages
- Unsupervised recognition: Learn from only audio, without transcription
  - Easier to develop speech systems for low-resource languages
  - Useful for linguistic research
  - Could model language acquisition of infants

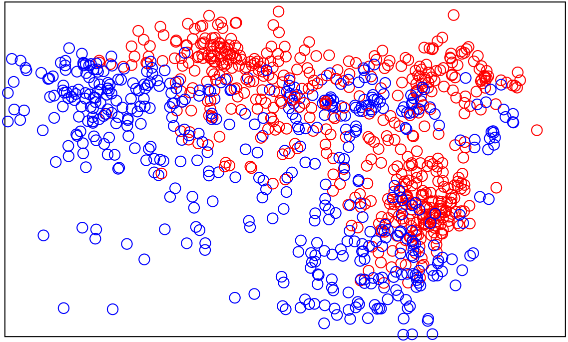
Representation of speech in speech recognition

- Waveform not optimal as a representation of speech
- Instead: Frequency content of short sections of the signal
- Further processing: Filter banks, cosine transform
- Repeat while moving the window to generate frames



Challenges in unsupervised speech recognition

- Speaker variation
- Segmentation
- No knowledge of what sounds exist in the language



Learning speaker-invariant representations

- Standard representations of speech are speaker dependent
- First step: Find speaker-invariant representations
  - Sounds of the same type should be similar
  - Sounds of different types should be dissimilar
  - Not concerned with categorising sounds
- Track 1 of Zero Resource Speech Challenge[1]

[1] Maarten Versteegh et al. (2015). 'The Zero Resource Speech Challenge 2015'. In: *Proc. of Interspeech*.

Previous work

- Deep autoencoder[2]
- Contrastive autoencoder[3]
  - "Auto"encode frame to other frame of same type
- Dirichlet process GMM clustering[4]
  - Surprisingly performant

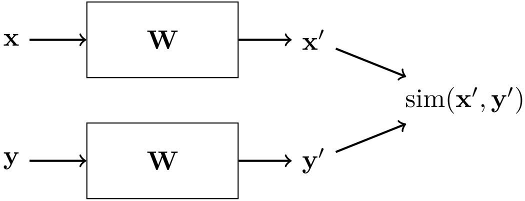
[2] Leonardo Badino et al. (2015). 'Discovering Discrete Subword Units with Binarized Autoencoders and Hidden-Markov-Model Encoders'. In: *Proc. of ISCA*.

[3] Daniel Renshaw et al. (2015). 'A comparison of neural network methods for unsupervised representation learning on the Zero Resource Speech Challenge'. In: *Proc. of Interspeech*.

[4] Hongjie Chen et al. (2015). 'Parallel Inference of Dirichlet Process Gaussian Mixture Models for Unsupervised Acoustic Modeling: A Feasibility Study'. In: *Proc. of Interspeech*.

Siamese networks for representation learning[5]

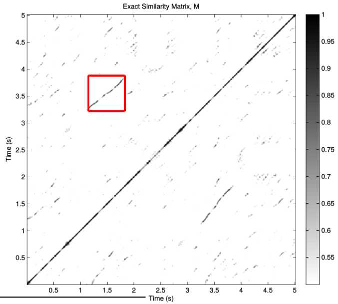
- Input: Pairs of same-class and different-class frames
- Adjust weights to make same-class frames more similar



[5] Gabriel Synnaeve et al. (2014). 'Phonetics embedding learning with side information'. In: *Proc. of IEEE SLT*. IEEE, pp. 106–111.

Finding same-class frames[6]

- Wish to find same-class frame pairs without supervision
- Simple clustering yields speaker-dependent units
- Idea: Patterns are easier to find at larger time scales



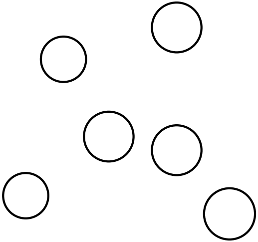
[6] Figure taken from Jansen and Van Durme (2011)

Proposed method – motivation

- Term discovery only covers a fraction of the data
  - Would like to make more efficient use of all data
- Large neural networks are prone to overfitting and difficult to interpret
- Idea: Use the whole data to find representations that are speaker dependent, but that can be used with simpler models
- Use a term discovery data to make these representations more speaker invariant

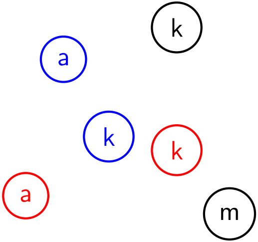
Proposed method

- Infer a probabilistic model from the whole data
- The model will find speaker-dependent phonetic classes
- Use same-class and different-class frame pairs to merge the phonetic classes
- The classes are described using probabilities → merging is simple addition



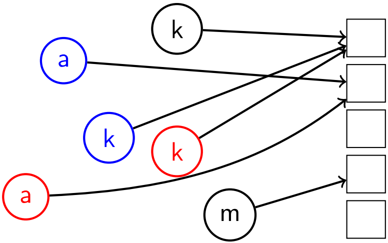
Proposed method

- Infer a probabilistic model from the whole data
- The model will find speaker-dependent phonetic classes
- Use same-class and different-class frame pairs to merge the phonetic classes
- The classes are described using probabilities $\rightarrow$ merging is simple addition



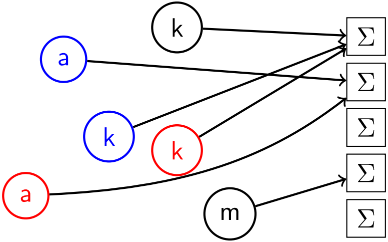
Proposed method

- Infer a probabilistic model from the whole data
- The model will find speaker-dependent phonetic classes
- Use same-class and different-class frame pairs to merge the phonetic classes
- The classes are described using probabilities → merging is simple addition



Proposed method

- Infer a probabilistic model from the whole data
- The model will find speaker-dependent phonetic classes
- Use same-class and different-class frame pairs to merge the phonetic classes
- The classes are described using probabilities → merging is simple addition



Proposed method (cont.)

- Represent input as a probability vector $\mathbf{x}$
  - Each element is the probability of a latent class
- Merging the classes is done using a linear transform: $\mathbf{xW}$
- $\mathbf{W}$ describes a proper partitioning of the input iff:
  1. Each element in $\mathbf{W}$ is either 0 or 1
  2. Each row in $\mathbf{W}$ contains exactly one 1
- We instead constrain $\mathbf{W}$ as follows:
  1. Each element is positive
  2. Each row sums to 1
- Output of the model is a lower-dimensional probability vector

Proposed method (cont.)

- Represent input as a probability vector $\mathbf{x}$
  - Each element is the probability of a latent class
- Merging the classes is done using a linear transform: $\mathbf{xW}$
- $\mathbf{W}$ describes a proper partitioning of the input iff:
  1. Each element in $\mathbf{W}$ is either 0 or 1
  2. Each row in $\mathbf{W}$ contains exactly one 1
- We instead constrain $\mathbf{W}$ as follows:
  1. Each element is positive
  2. Each row sums to 1
- Output of the model is a lower-dimensional probability vector

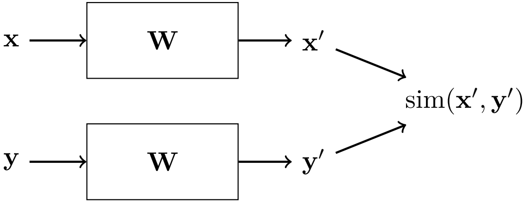
Proposed method (cont.)

- Represent input as a probability vector $\mathbf{x}$
  - Each element is the probability of a latent class
- Merging the classes is done using a linear transform: $\mathbf{xW}$
- $\mathbf{W}$ describes a proper partitioning of the input iff:
  1. Each element in $\mathbf{W}$ is either 0 or 1
  2. Each row in $\mathbf{W}$ contains exactly one 1
- We instead constrain $\mathbf{W}$ as follows:
  1. Each element is positive
  2. Each row sums to 1
- Output of the model is a lower-dimensional probability vector

Proposed method (cont.)

- Represent input as a probability vector $\mathbf{x}$
  - Each element is the probability of a latent class
- Merging the classes is done using a linear transform: $\mathbf{xW}$
- $\mathbf{W}$ describes a proper partitioning of the input iff:
  1. Each element in $\mathbf{W}$ is either 0 or 1
  2. Each row in $\mathbf{W}$ contains exactly one 1
- We instead constrain $\mathbf{W}$ as follows:
  1. Each element is positive
  2. Each row sums to 1
- Output of the model is a lower-dimensional probability vector

Proposed method (cont.)



Loss function

- The output is a probability vector
- We can measure similarity using a statistical divergence measure
- Here: The root Jensen-Shannon divergence

$$L(\mathbf{W}; \mathbf{x}, \mathbf{y}) = \sqrt{JS(\mathbf{xW} || \mathbf{yW})}$$

- Minimize if $\mathbf{x}$ and $\mathbf{y}$ are the same speech sound; maximize otherwise
- Need to balance same-class and different-class losses over a minibatch

Loss function

- The output is a probability vector
- We can measure similarity using a statistical divergence measure
- Here: The root Jensen-Shannon divergence

$$L(\mathbf{W}; \mathbf{x}, \mathbf{y}) = \sqrt{JS(\mathbf{xW} || \mathbf{yW})}$$

- Minimize if $\mathbf{x}$ and $\mathbf{y}$ are the same speech sound; maximize otherwise
- Need to balance same-class and different-class losses over a minibatch

Loss function

- The output is a probability vector
- We can measure similarity using a statistical divergence measure
- Here: The root Jensen-Shannon divergence

$$L(\mathbf{W}; \mathbf{x}, \mathbf{y}) = \sqrt{JS(\mathbf{x}\mathbf{W} || \mathbf{y}\mathbf{W})}$$

- Minimize if $\mathbf{x}$ and $\mathbf{y}$ are the same speech sound; maximize otherwise
- Need to balance same-class and different-class losses over a minibatch

Entropy penalty for encouraging sparsity

- Merging corresponds to partitioning the speaker-dependent classes
- However, $\mathbf{W}$ as defined is not a proper partitioning
- Using an entropy penalty on the model output we can encourage $\mathbf{W}$ to be an approximate partitioning

$$L_H(\mathbf{W}; \mathbf{x}, \mathbf{y}) = H(\mathbf{x}\mathbf{W}) + H(\mathbf{y}\mathbf{W})$$

- Prevents the probability mass from being spread out over multiple outputs
- Implicitly makes the model sparse

Entropy penalty for encouraging sparsity

- Merging corresponds to partitioning the speaker-dependent classes
- However, $\mathbf{W}$ as defined is not a proper partitioning
- Using an entropy penalty on the model output we can encourage $\mathbf{W}$ to be an approximate partitioning

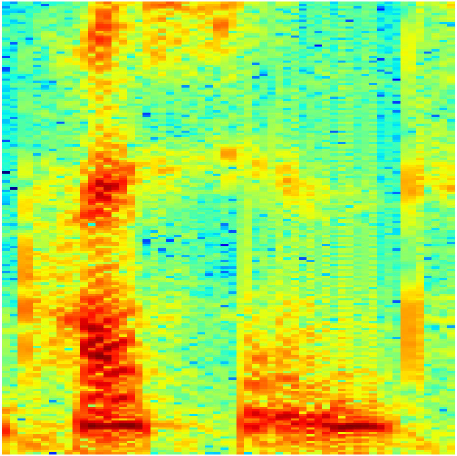
$$L_H(\mathbf{W}; \mathbf{x}, \mathbf{y}) = H(\mathbf{x}\mathbf{W}) + H(\mathbf{y}\mathbf{W})$$

- Prevents the probability mass from being spread out over multiple outputs
- Implicitly makes the model sparse

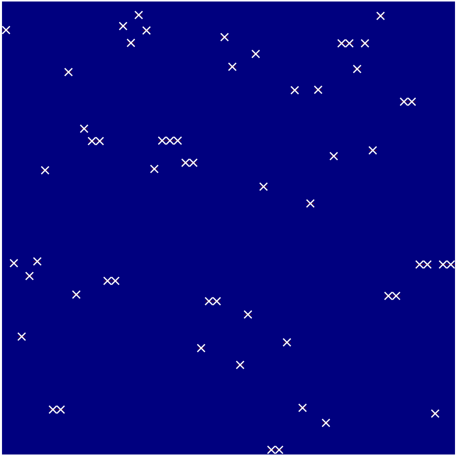
Discretisation

- The trained model can be discretised
- Set largest element on each row to 1
- Set all other elements to 0
- Yields a proper partitioning

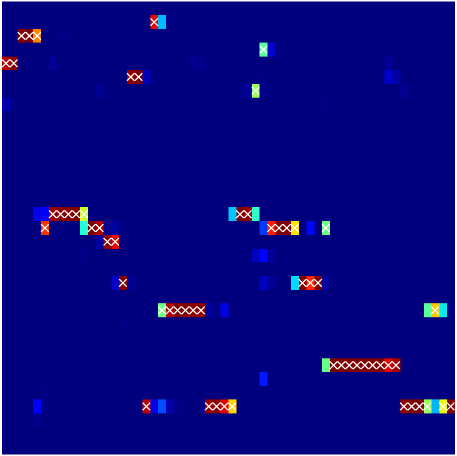
Spectrogram features



GMM features (input)



Merged GMM features (output)



Data

- Two corpora: One of casual English, and one of read Xitsonga
- The data is clustured using 1024-component GMMs
- For each frame we extract posterior probabilities from the GMM
- The proposed model is trained with 64 outputs

Minimal-pair ABX

- The evaluation is done using the minimal-pair ABX task
- Three utterances: A, B and X
- Either A or B is the same category as X
- Representing frames using the output of the model, the utterance most similar to X is chosen

Results

	English		Xitsonga	
Model	Within	Across	Within	Across
GMM posteriors	12.3	23.8	11.4	23.2
Proposed model	12.8	19.8	14.0	23.2
Binary $\mathbf{W}$	12.0	19.3	12.7	21.9
ABnet[7]	12.0	17.9	11.7	16.6
DPGMM + LDA[8]	10.6	16.0	8.0	12.6

[7] Roland Thiolliere et al. (2015). 'A hybrid dynamic time warping-deep neural network architecture for unsupervised acoustic modeling'. In: *Proc. of Interspeech*.

[8] Michael Heck et al. (2016). 'Unsupervised Linear Discriminant Analysis for Supporting DPGMM Clustering in the Zero Resource Scenario'. In: *Procedia*

Discussion

- The model significantly decreases the dimensionality of the input
- At the same time, it empirically improves the speaker invariance of the representation
- Worse performance for Xitsonga – sensitive to dimensionality?
- The model has few parameters, making it fast to train and robust against overfitting
- Can use probability vectors from any model as input

Future work

- Other models for generating the probability vectors
- Alternative loss functions

Thank you for listening!