JSALT19 Speaker Diarization Tutorial

Paola García, Jesus Villalba, Phani Nidadavolu, Saurabh Kataria

06/20/19

Introduction

- Clone (similar to the done by Jesus):
  - git clone https://github.com/jsalt2019-diadet/jsalt2019-tutorial.git
  - Dependencies:
    - Kaldi: feature and embedding extraction
    - Python3.5
  - Create links to depencies in AWS
    - cd jsalt2019-tutorial
    - make_awk_links.sh
      - It will create links to already compiled anaconda3 and Kaldi in the grid.

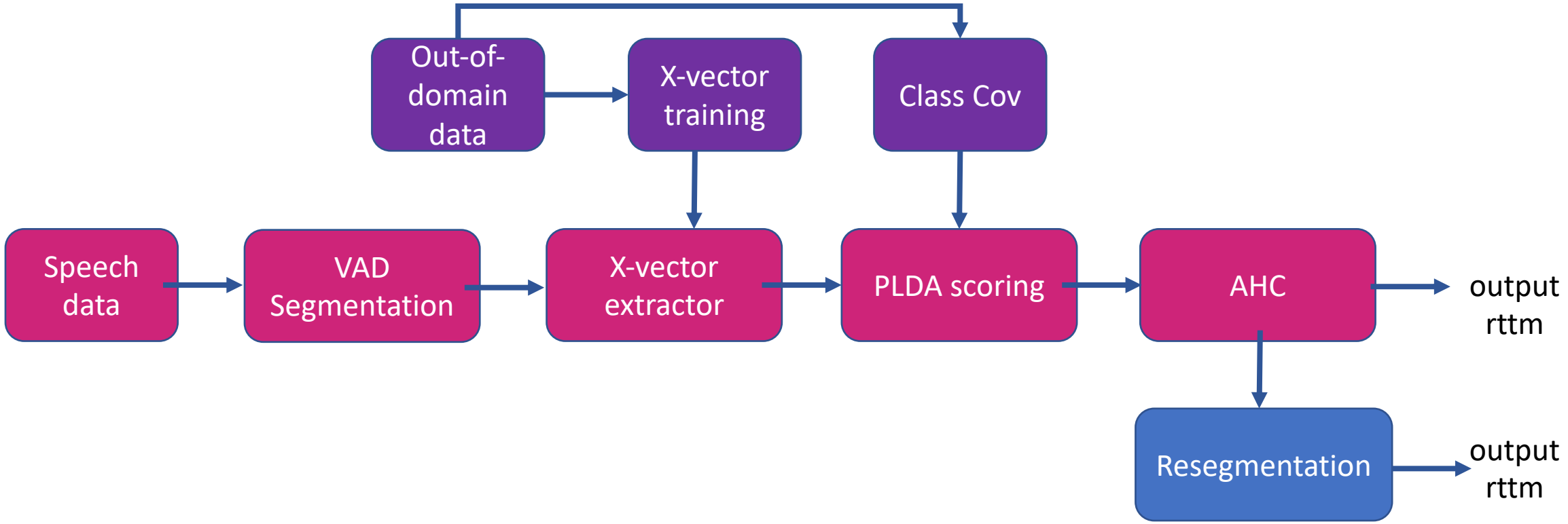
Directory structure

- Egs: recipes
  - sitw_tutorial/v1
    - Speaker verification example.
    - Based on the Speakers in the Wild dataset.
  - callhome_diarization/v1
    - Speaker diarization example.
    - Based on the calhome dataset
- Tools: links to dependencies
  - Hyperion: python tools
    - LDA/PLDA back-end
    - Calibration
  - Kaldi
  - Anaconda Python

```
./jsalt2019-tutorial
./jsalt2019-tutorial/tools
./jsalt2019-tutorial/tools/anaconda
./jsalt2019-tutorial/tools/anaconda/anaconda3.5
./jsalt2019-tutorial/tools/kaldi
./jsalt2019-tutorial/tools/kaldi/kaldi
./jsalt2019-tutorial/tools/hyperion
./jsalt2019-tutorial/tools/hyperion/hyperion
./jsalt2019-tutorial/egs
./jsalt2019-tutorial/egs/sitw_tutorial
./jsalt2019-tutorial/egs/sitw_tutorial/v1
./jsalt2019-tutorial/egs/callhome_diarization
./jsalt2019-tutorial/egs/callhome_diarization/v1
```

Callhome diarization Pipeline

- egs/callhome_diarization/v1



Callhome diarization Pipeline

- VAD (voice activity detection)
  - Energy VAD
  - Using the actual rttm (oracle VAD) can give us a better idea.
- Training:
  - Kaldi based system for i-vector
  - Kaldi based system for x-vector (mainly focused on this one)
- PLDA
  - Out-of-the-box trained using sre + swbd data.
- We don’t have the VB-resegmentation (but we are working on it!)

Callhome: general view

- We have all steps in Kaldi
  - Data_prep (40% of your success!)
  - VAD - energy based
  - Training stage (you can do it offline)
    - Augmentation using RIR and MUSAN
    - Training xvectors with sre datasets
  - X-vector extractor
  - Train PLDA with callhome1 and callhome2
  - Score PLDA
  - Clustering

run_training.sh

```
if [ $stage -le 0 ]; then
  local/make_dataPrep.sh
…
if [ $stage -le 1 ]; then
 sid/compute_vad_decision.sh
…
if [ $stage -le 2 ]; then
steps/data/augment_data_dir.py
…
if [ $stage -le 4 ]; then
local/nnet3/xvector/run_xvector.sh
….
if [ $stage -le 7 ]; then
diarization/nnet3/xvector/extract_xvectors.sh
….
if [ $stage -le 8 ]; then
ivector-compute-plda
….
if [ $stage -le 9 ]; then
diarization/nnet3/xvector/score_plda
….
if [ $stage -le 10 ]; then
diarization/cluster.sh
….
```

Callhome: data preparation

- Stage 0
  - Prepare collection of data for train and test
  - SRE + swbd + callhome1 and callhome2:
    - Training data for x-vector and PLDA back-end
    - Wav.scp, utt2spk, spk2utt
      - Sorted by speaker and utterance id

```
spk2utt
100304-f 100304-f-sre2006-kacg-A 100304-f-
sre2006-kbjd-B 100304-f-sre2006-kcbg-B
100304-f-sre2006-kcbz-B 100304-f-sre2006-
kcqt-B 100304-f-sre2006-kcro-B 100304-f-
sre2006-kdfy-A 100304-f-sre2006-kdla-A
100304-f-sre2006-keal-A 100304-f-sre2006-
kemd-A 100304-f-sre2006-kepr-A 100304-f-
sre2006-obtz-B 100304-f-sre2006-oczr-B
100304-f-sre2006-odvl-A 100304-f-sre2006-
oear-B 100304-f-sre2006-oedw-B 100304-f-
sre2006-oesx-A 100304-f-sre2006-ofdw-B
100304-f-sre2006-oilp-B 100304-f-sre2008-
tjsjr-A
```

```
utt2spk
100304-f-sre2006-kacg-A 100304-f
100304-f-sre2006-kbjd-B 100304-f
100304-f-sre2006-kcbg-B 100304-f
```

```
Wav.scp:
100304-f-sre2006-kacg-A sph2pipe -f wav -p -c 1 /export/corpora5/LDC/LDC2011S09/data/train/data/kacg.sph |
100304-f-sre2006-kbjd-B sph2pipe -f wav -p -c 2 /export/corpora5/LDC/LDC2011S09/data/train/data/kbjd.sph |
100304-f-sre2006-kcbg-B sph2pipe -f wav -p -c 2 /export/corpora5/LDC/LDC2011S09/data/train/data/kcbg.sph |
```

Callhome diarization: data preparation

```
Wav.scp:   file_identifier instruction_to_convert_file_with_path  |
100304-f-sre2006-kacg-A sph2pipe -f wav -p -c 1 /export/corpora5/LDC/LDC2011S09/data/train/data/kacg.sph |
100304-f-sre2006-kbjd-B sph2pipe -f wav -p -c 2 /export/corpora5/LDC/LDC2011S09/data/train/data/kbjd.sph |
100304-f-sre2006-kcbg-B sph2pipe -f wav -p -c 2 /export/corpora5/LDC/LDC2011S09/data/train/data/kcbg.sph |

Utt2spk: file_identifier model
100304-f-sre2006-kacg-A 100304-f
100304-f-sre2006-kbjd-B 100304-f
100304-f-sre2006-kcbg-B 100304-f

Spk2utt: model file_identifier
100304-f 100304-f-sre2006-kacg-A 100304-f-sre2006-kbjd-B 100304-f-sre2006-kcbg-B 100304-f-sre2006-kcbz-B
100304-f-sre2006-kcqt-B 100304-f-sre2006-kcro-B 100304-f-sre2006-kdfy-A 100304-f-sre2006-kdla-A 100304-f-
sre2006-keal-A 100304-f-sre2006-kemd-A 100304-f-sre2006-kepr-A 100304-f-sre2006-obtz-B 100304-f-sre2006-
oczr-B 100304-f-sre2006-odvl-A 100304-f-sre2006-oear-B 100304-f-sre2006-oedw-B 100304-f-sre2006-oesx-A
100304-f-sre2006-ofdw-B 100304-f-sre2006-oilp-B 100304-f-sre2008-tjsjr-A
```

Callhome: compute MFCC features and VAD

- Stage 1
  - Compute MFCC features

```
File_id mfcc_path
100304-f-sre2006-kacg-A /export/c03/leibny/NTU/kaldi-ntu/egs/callhome_diarization/v2/mfcc/raw_mfcc_train.1.ark:24
100304-f-sre2006-kbjd-B /export/c03/leibny/NTU/kaldi-ntu/egs/callhome_diarization/v2/mfcc/raw_mfcc_train.1.ark:690253
100304-f-sre2006-kcbg-B /export/c03/leibny/NTU/kaldi-ntu/egs/callhome_diarization/v2/mfcc/raw_mfcc_train.1.ark:1380482
100304-f-sre2006-kcbz-B /export/c03/leibny/NTU/kaldi-ntu/egs/callhome_diarization/v2/mfcc/raw_mfcc_train.1.ark:2070711
```

  - Energy VAD : removes the silences

```
mfccs/
vad_callhome1.10.ark
vad_callhome1.10.scp
vad_callhome1.11.ark
vad_callhome1.11.scp
vad_callhome1.12.ark
```

Callhome: prepare features for x-vector NNet

- prepare_feats.sh
  - Normalize training features with short-time CMN
  - It creates a new data directory in data/callhome1_cmn and data/callhome2_cmn

```
local/nnet3/xvector/prepare_feats.sh
```

Callhome: augmentation

- Stage 2: augmentation
  - We augment the training data with reverberation, noise, music, and babble, and combined it with the clean data.
  - The combined list will be used to train the x-vector DNN.
  - The SRE subset will be used to train the PLDA model
  - 5 copies of the data will be available

`steps/data/reverberate_data_dir.py`

`steps/data/augment_data_dir.py`

`utils/combine_data.sh`

`utils/subset_data_dir.sh`

Callhome diarization: prepare features

- Stage 3 (you already have the models trained)
  - Prepare the features to generate examples for x-vector training.
  - This script applies CMN and removes nonspeech frames.
  - Removes features that are too short after removing silence (500 frames) per utterance.
  - several utterances per speaker. We throw out speakers with fewer than 8 utterances.

```
if [ $stage -le 3 ]; then
local/nnet3/xvector/prepare_feats_for_egs.sh
```

```
utils/filter_scp.pl
```

```
utils/filter_scp.pl
```

Callhome: training the x-vector DNN

- Quantum leap
  - Stages 4 to 6
    - Stage 4: get the neural network training examples

```
sid/nnet3/xvector/get_egs.sh
```

    - Stage 5: create the neural network config according to what you want

```
steps/nnet3/xconfig_to_configs.py \
      --xconfig-file $nnet_dir/configs/network.xconfig \
      --config-dir $nnet_dir/configs/
```

    - Stage 6: train the actual DNN

```
local/nnet3/xvector/tuning/run_xvector_1a.sh --stage $stage --train-stage -1 \
  --data data/train_combined_cmn_no_sil --nnet-dir $nnet_dir \
  --egs-dir $nnet_dir/egs
```

Callhome: extract x-vector

- Stage 7: extract x-vectors
  - Extract x-vectors for the two subsets: callhome1 and callhome2

```
diarization/nnet3/xvector/extract_xvectors.sh
```

  - Extract the x-vector for the PLDA (same instruction)
    - PLDA model is also given!

```
diarization/nnet3/xvector/extract_xvectors.sh
```

Callhome: train PLDA

- Stage 8: train PLDA (models are given)
  - Train using SRE and callhome1 to whiten.
    - later use this to score x-vectors in callhome2
  - Train using SRE and callhome2 to whiten.
    - later use this to score x-vectors in callhome1

```
ivector-compute-plda ark:$nnet_dir/xvectors_sre_segmented_128k/spk2utt \
  "ark:ivector-subtract-global-mean \
  scp:$nnet_dir/xvectors_sre_segmented_128k/xvector.scp ark:- \
  | transform-vec $nnet_dir/xvectors_callhome1/transform.mat ark:- ark:- \
  | ivector-normalize-length ark:- ark:- |" \
$nnet_dir/xvectors_callhome1/plda
```

Callhome: PLDA scoring

- Scoring on all pairs of segments for each recordings
  - We have to do this for callhome1 and callhome 2

```
diarization/nnet3/xvector/score_plda.sh --cmd "$train_cmd --mem 4G" \
  --nj 20 $nnet_dir/xvectors_callhome2 $nnet_dir/xvectors_callhome1 \
  $nnet_dir/xvectors_callhome1/plda_scores
```

Callhome: Clustering threshold

- Threshold that minimizes the DER on each partition of callhome
  - This threshold is in terms of the log likelihood ratio provided by the PLDA scores.
  - For a perfectly calibrated system the threshold is 0
  - We evaluate the clustering on a heldout dataset using reasonable thresholds.
    - callhome1 is heldout for callhome2 and vice-versa)

```
for threshold in -0.3 -0.2 -0.1 -0.05 0 0.05 0.1 0.2 0.3; do
     diarization/cluster.sh --cmd "$train_cmd --mem 4G" --nj 20 \
       --threshold $threshold $nnet_dir/xvectors_$dataset/plda_scores \
       $nnet_dir/xvectors_$dataset/plda_scores_t$threshold
```

Callhome: Clustering oracle # of speakers

- Clustering if the number of speakers per recording is known in advance (meaning the number of clusters).

```
  diarization/cluster.sh --cmd "$train_cmd --mem 4G" \
    --reco2num-spk data/callhome1/reco2num_spk \
    $nnet_dir/xvectors_callhome1/plda_scores
$nnet_dir/xvectors_callhome1/plda_scores_num_spk
```

- An important output is in:

```
$nnet_dir/xvectors_callhome1/plda_scores
```

- If you are curious and want to take a look at the scores:

```
copy-matrix scp: $nnet_dir/xvectors_callhome1/plda_scores scores.scp ark,t:- |head
```

Callhome: evaluation DER

- Evaluation is performed using md-eval

```
md-eval.pl -1 -c 0.25 -r data/$dataset/ref.rttm \
       -s $nnet_dir/xvectors_$dataset/plda_scores_t$threshold/rttm
```

- Diarization Error Rate (DER)
  - Miss (speaker in reference but not in hypothesis)
  - False Alarm (speaker in hypothesis but not in reference)
  - Speaker Confusion (confusing one speaker's speech as from another)

$$DER = \frac{false\_alarm + miss\_detection + speaker\_confusion}{total}$$

Callhome: results and RTTM

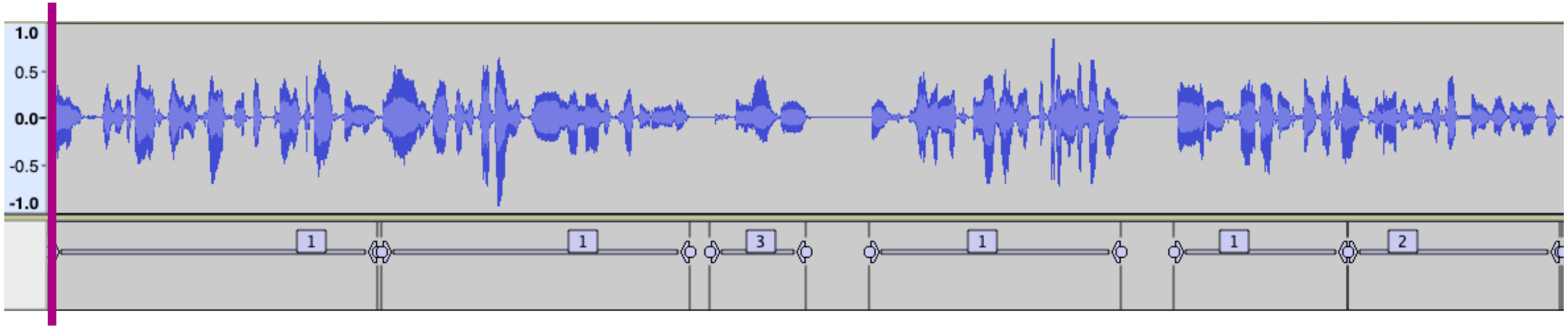
- Results:

```
$nnet_dir/DER_num_spk.txt  $nnet_dir/ DER_threshold.txt
```

- Important columns 2,4,5 and 8.

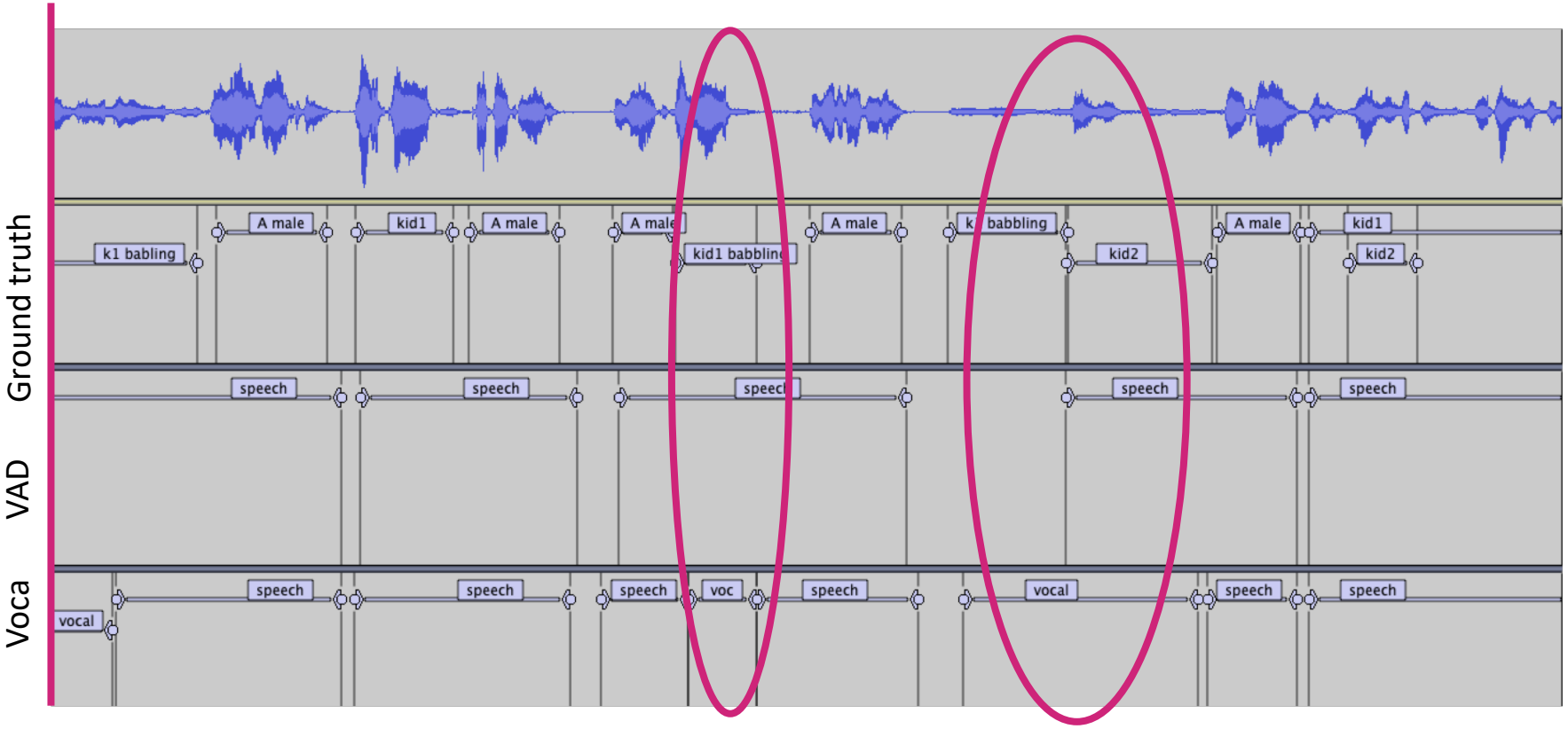
```
SPEAKER iaaa 0   0.000   3.270 <NA> <NA> 2 <NA> <NA>
SPEAKER iaaa 0   3.390   2.060 <NA> <NA> 3 <NA> <NA>
SPEAKER iaaa 0   5.510   0.440 <NA> <NA> 3 <NA> <NA>
SPEAKER iaaa 0   6.140   2.990 <NA> <NA> 3 <NA> <NA>
SPEAKER iaaa 0   9.240   2.625 <NA> <NA> 3 <NA> <NA>
SPEAKER iaaa 0  11.865   3.355 <NA> <NA> 1 <NA> <NA>
SPEAKER iaaa 0  15.640   2.130 <NA> <NA> 1 <NA> <NA>
SPEAKER iaaa 0  17.990   1.810 <NA> <NA> 1 <NA> <NA>
SPEAKER iaaa 0  20.160   1.330 <NA> <NA> 1 <NA> <NA>
SPEAKER iaaa 0  22.070   5.625 <NA> <NA> 3 <NA> <NA>
SPEAKER iaaa 0  27.695   1.155 <NA> <NA> 2 <NA> <NA>
```

RTTM in audacity



Diarization: true example

- While speaking



CENTER FOR LANGUAGE AND SPEECH PROCESSING

Thank you!