Speaker Diarization

A brief catch-up on tools and technology to get started!

By
Simran Sehgal and Huan Yu
June 2020 - December 2020

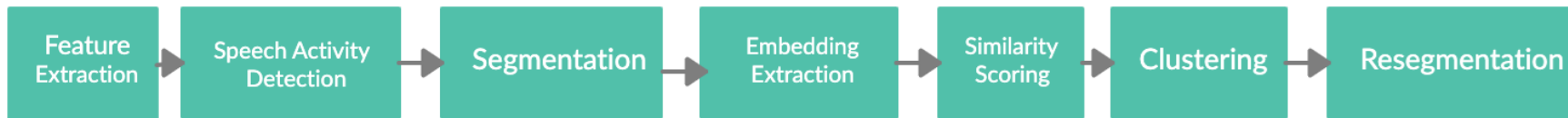*Work done under the supervision of Prof Eng-Siong Chng, Prof Xionghu Zhong and Dr Van Pham Tung*

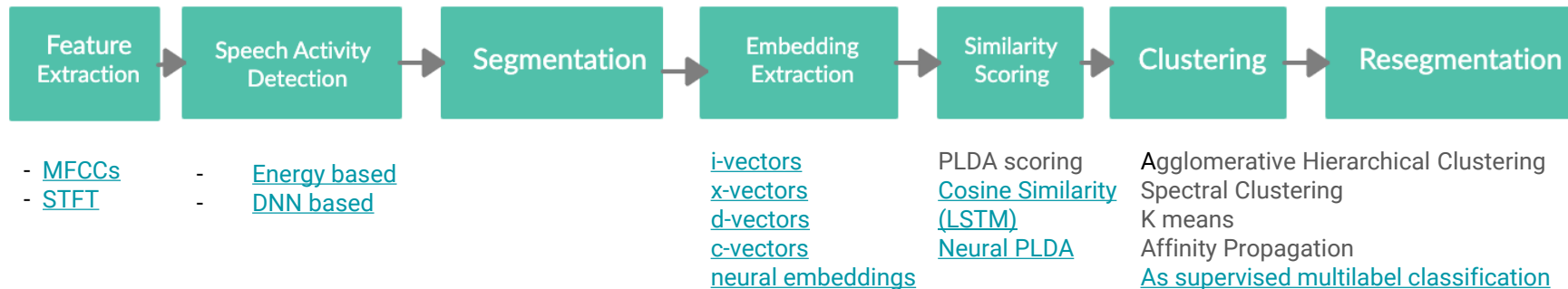Find our work
Find our weekly presentations

Contents

1. Problem Statement
2. A Brief Literature Review
3. Test Dataset
4. Metric
5. Toolkits: LIUM, Kaldi and Pyannote-Audio
6. Recent Research Directions
7. Our work!
8. Future Directions
9. Useful links

Problem Statement

- Determining 'who spoke when' in an audio.
- Identifying and labelling speech frames according to the speaker.
- A typical Speaker Diarization pipeline:

| Feature Extraction | → | Speech Activity Detection | → | Segmentation | → | Embedding Extraction | → | Similarity Scoring | → | Clustering | → | Resegmentation |

A Brief Literature Review

| Feature Extraction | Speech Activity Detection | Segmentation | Embedding Extraction | Similarity Scoring | Clustering | Resegmentation |

- MFCCs
- STFT

- Energy based
- DNN based

i-vectors
x-vectors
d-vectors
c-vectors
neural embeddings

PLDA scoring
Cosine Similarity (LSTM)
Neural PLDA

Agglomerative Hierarchical Clustering
Spectral Clustering
K means
Affinity Propagation
As supervised multilabel classification

Test Dataset: CALLHOME

- Released by Linguistic Data Consortium (LDC) as part of 2000 NIST Speaker Recognition database as Disk 8.
- Oracle labels available at openslr.org, earlier used to be hosted by NIST online
- Used to test or adapt models and pipelines, using k fold cross validation

Features	Callhome
Sampling Frequency	8 KHz
Number of Recordings	500
Number of Hours	17
Language	Multlingual (English, Spanish etc)
Average duration of file	124.5 secs

Metric: Diarization Error Rate

$$DER = \frac{false\ alarm + missed\ detection + confusion}{total}$$

- **Confusion Error:** Reference and Hypothesis speaker labels don't match
- **False Alarm Error (FA):** Hypothesis labels as speech a segment which Reference labels as non-speech
- **Miss Error:** Hypothesis labels as non-speech and does not assign any speaker but Reference labels as speech.

For more nuance, see how pyannote-metric calculates it, and NIST defines it.

Toolkits

- LIUM SpkDiarization
- Kaldi
- Pyannote

Some frameworks/languages to learn:

- Pytorch/Tensorflow
- Python, Basics of Shell Script.

LIUM SpkDiarization

About the Toolkit

- Written in Java, based on i-vector, optimised for radio and TV shows.
- Hard to operate due to less documentation
- Was state-of-the-art once, but no further developments after i-vector

  DER on Callhome: 51.28 %

Some links:
- https://projets-lium.univ-lemans.fr/spkdiarization/
- https://github.com/StevenLOL/LIUM

Kaldi

- Open source state-of-the-art toolkit for Speech Recognition and other applications written in C++, Python and Bash Scripts.
- Contains a lot of folders and recipes, familiarize yourself with 'egs/callhome_diarization' first.

- Some python wrappers to use Kaldi's functionalities outside of command line environment:

  Bob: Only for SAD

  Kaldi-io: To convert from kaldi data (x-vectors, mfcc in .ark) to python numpy matrices

  Pykaldi: Python wrappers for the C++, low level function code in kaldi

Installation

- Clone kaldi from git and follow Install instructions present in repository (http://jrmeyer.github.io/asr/2016/01/26/Installing-Kaldi.html)

- If you are having problems with compatibility of some tools (like Intel MKL) on OS, consider installing Kaldi in a docker (https://maelfabien.github.io/signal/kaldi/#)

Running the Callhome Recipe

- Learn about data files that Kaldi needs from [here](#) and [here](#): wav.scp, segments, spk2utt, utt2spk.
- Go to egs/callhome_diarization/v2 and find an x-vector recipe there (run.sh).
- Get the callhome dataset from Dr Tung.
- Try to run it step by step from terminal and observe the pipeline, the input and output of each module
- Exclude the training parts of the recipe.
- Practice: Make a run.sh script for diarization of a single file, say iaaa.sph from Callhome

Refer to these if stuck:

- https://towardsdatascience.com/speaker-diarization-with-kaldi-e30301b05cc8
- https://github.com/kaldi-asr/kaldi/issues/2523#issuecomment-408935477

Verify Results

Once you have successfully run the run.sh i.e 2 fold callhome experiment, verify the results for 0.25 seconds collar with known number of speakers.

Model	DER for Overlap Included	DER for Overlap Excluded	Inference Time
Kaldi: X-vectors +PLDA scoring+ AHC clustering	20.14%	9.46%	5%
Kaldi: X-vectors +PLDA scoring+ AHC clustering + VB resegmentation	18.61%	7.33%	19%

This would be an important baseline.

Pyannote-Audio
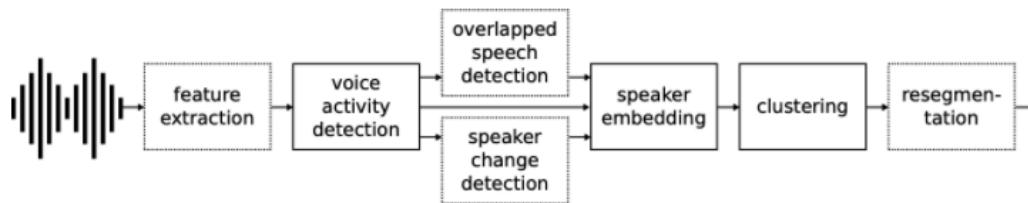
- Open source toolkit for speaker diarization based on Pytorch framework
- Provides trainable neural network blocks for Speech Activity Detection (SAD), Speaker Change Detection (SCD) and Embedding extraction.
- Models make up Speaker Diarization Pipeline
- All NN models are LSTM-based

Available Pipelines:

	Pipeline	Models used internally	Development set
✅	dia or dia_dihard	{ sad_dihard , scd_dihard , emb_voxceleb }	DIHARD.custom.dev
✅	dia_ami	{ sad_ami , scd_ami , emb_ami }	AMI.dev
❌	dia_etape	{ sad_etape , scd_etape , emb_etape }	Etape.dev

Pipelines marked with ❌ are not available yet but will be released at some point.

Getting Started

Pyannote-audio's [repository](#) has good tutorials.

A good colab introduction for pyannote-audio's APIs [here](#).

Try diarizing Callhome/ Run the pyannote tutorial for AMI dataset here and verify results for 0.25 seconds collar. The results on Callhome aren't very good because the pipeline is trained on 16 Khz data.
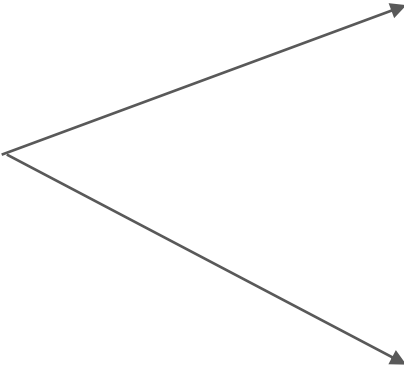
Verify Results

Dataset	DER with overlap	DER without overlap	Pyannote pipeline
AMI meeting set	17.57%	14.20%	dia_ami
Callhome	54.62%		dia_ami
Callhome	49.30%		dia_dihard

Note: We tried to finetune the dia_ami pipeline on the AMI eval set using this tutorial so that ultimately we could finetune it on Callhome using 5 fold evaluation and improve results but the code always broke down and we could not configure an environment for it to run perfectly. This might be because pyannote is still in development.

Recent Research Directions and Leading Works

Approach

Improve component of pipeline (SAD/ embedding/ clustering/ resegmentation)

Reduce Pipeline, often by combining 2 or more modules in a neural network and make system end-to-end

	Paper	Year	Authors	Methodology	Available implementation	Baseline
1.	Speaker Diarization with Region Proposal Network	2020	*Zili Huang, Shinji Watanabe, et al*	Borrow a CV technique- Region Proposal Network (RPN) and reduce typical pipeline to 2 steps: SAD+Embedding/ Clustering.	https://github.com/HuangZili Andy/RPNSD Trained model available and can be adapted for CALLHOME!	Kaldi's CALLHOME diarization V2 recipe (X-vectors) with TDNN SAD model + VB resegmentation.
2.	End-to-End Neural Speaker Diarization with Permutation-Free Objectives	2019	*Yusuke Fujita, Naoyuki Kanda,et al.*	Instead of separate clustering and embedding modules, one pipeline- a BLSTM neural network	https://github.com/hitachi-speech/EEND Trained model not available	Kaldi's CALLHOME diarization v1 (i-vectors) and v2 (x-vectors) recipe
3.	Fully supervised speaker diarization	2019	*Aonan Zhang, Quan Wang,et al.*	Supervised system. Built on 2018 work using d-vectors. Replaces clustering model by a parameter sharing RNN which models each speaker and generates instances of RNN according to a probablistic model	https://github.com/google/ui s-rnn	(Paper 4)
4.	Speaker Diarization with LSTM	2018	*Carlton Downey, Li Wan, et al.*	Use LSTM-based d vectors for speaker embeddings+ non-parametric clustering	(only code for spectral clustering algorithm available)	Train their own i-vector network on anonymized voice searches

Our Work

From June to November, 3 primary experiments were performed and the models, code and results were established:

1. Kaldi's diarization recipe
2. Kaldi's x-vector with LSTM similarity scoring
3. RPNSD

Find them [here](#).

Secondary Tasks

- Transformer for multi-label classification. Achieved a DER of ~30% on Callhome.
- SOMs for clustering RPNSD embeddings- worked well for 2 speaker conversations but did not do well with more number of speakers
- Joint optimisation of Kaldi's x-vector with the LSTM model for generating similarity matrix. Training data was too less to do this adequately. Contact Huan Yu for any questions.

Future Directions

- Replace LSTM similarity matrix with LSTM+attention similarity matrix.

- Generate new embeddings from existing embeddings optimised for cluster assignment and feature representation [1]

- Generate new embeddings using triplet loss.

- Concatenate i-vectors and x-vectors using a neural network to generate new embeddings (similar to c-vectors)

- Refine overlapped frame labelling script to reduce speaker confusion error.

[1] Unsupervised Deep Embedding for Clustering Analysis Junyuan Xie, University of Washington Ross Girshick, Facebook AI Research (FAIR) Ali Farhadi, University of Washington

Useful Links

⭐ https://github.com/YoavRamon/awesome-kaldi

⭐ https://wq2012.github.io/awesome-diarization/

⭐ https://paperswithcode.com/task/speaker-diarization

https://pytorch.org/tutorials/index.html (Sequence to Sequence model tutorials are typically available for text, but nowadays they are being adapted for speech tasks)

https://colab.research.google.com/github/facundodeza/transfomer-audio-classification/blob/master/audio_classification_transformer.ipynb#scrollTo=_PqhqY0iMgjo (A rare instance of transformer code with audio)

https://www.danielpovey.com/kaldi-lectures.html

https://ai.googleblog.com/2018/11/accurate-online-speaker-diarization.html