Turn-to-Diarize: Online Speaker Diarization Constrained by Transformer Transducer Speaker Turn Detection

Authors: *Wei Xia, Han Lu, Quan Wang, Anshuman Tripathi, Yiling Huang, Ignacio Lopez Moreno, Hasim Sak*

Presented by: *Quan Wang*

icassp 2022
Singapore, China, Online

Part 1:

Introduction

Follow our previous work: Watch these videos

- https://www.youtube.com/watch?v=pjxGPZQeeO4
- https://www.youtube.com/watch?v=pGkqwRPzx9U

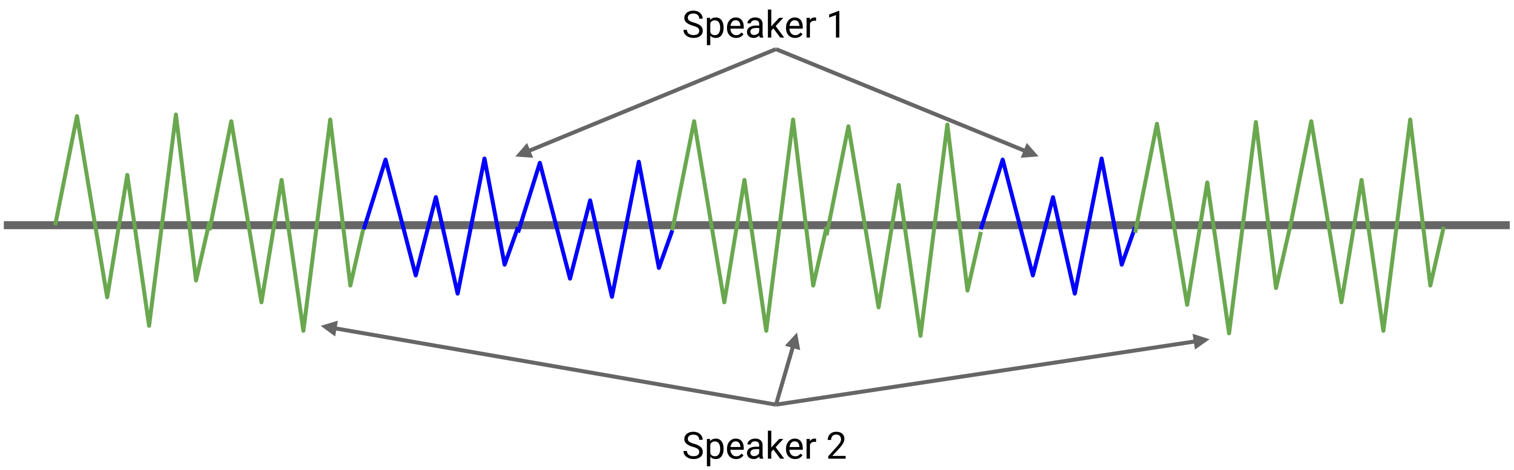
[ICASSP 2018] Google's Diarization System: Speaker Diarization with LSTM

[ICASSP 2019] Fully Supervised Speaker Diarization: Say Goodbye to clustering

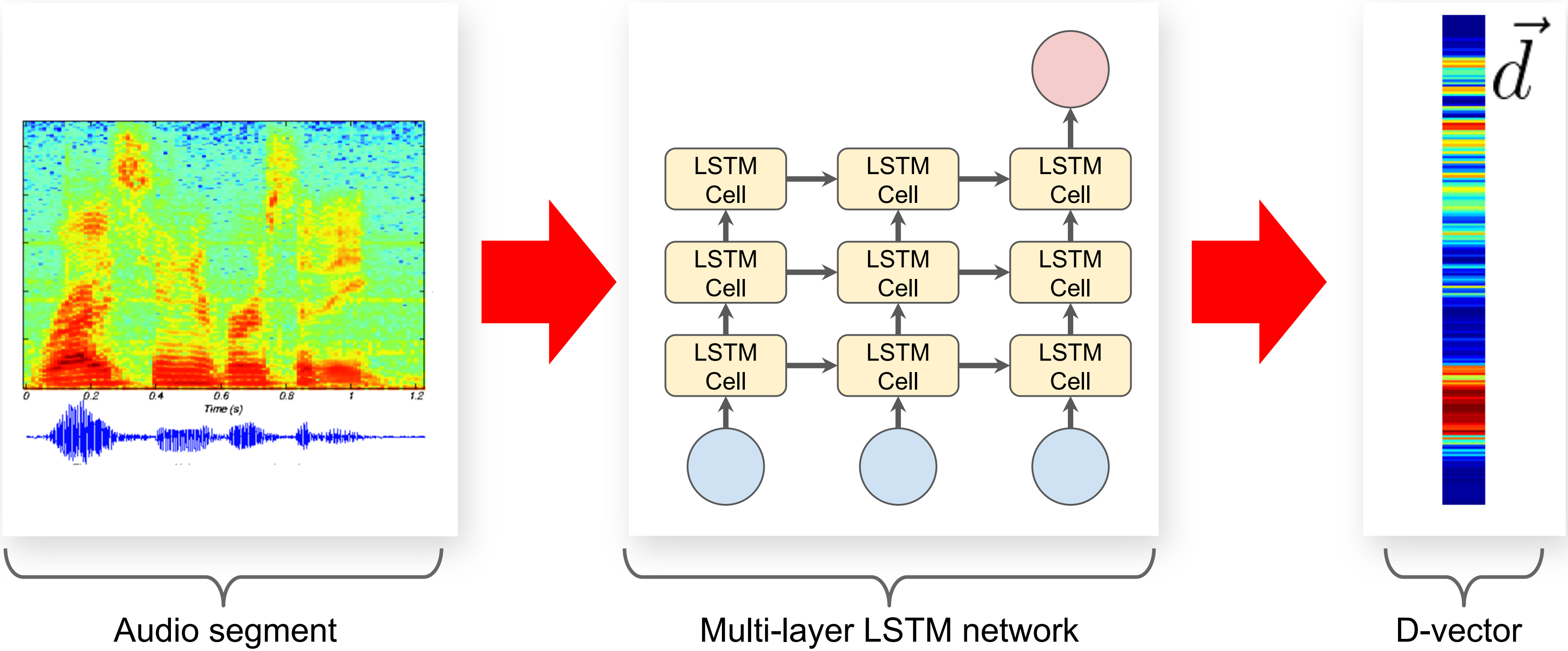
Recap: What is speaker diarization?

- The process of partitioning an input audio stream into homogeneous segments according to the speaker identity

- ***"Who Spoke When?"***

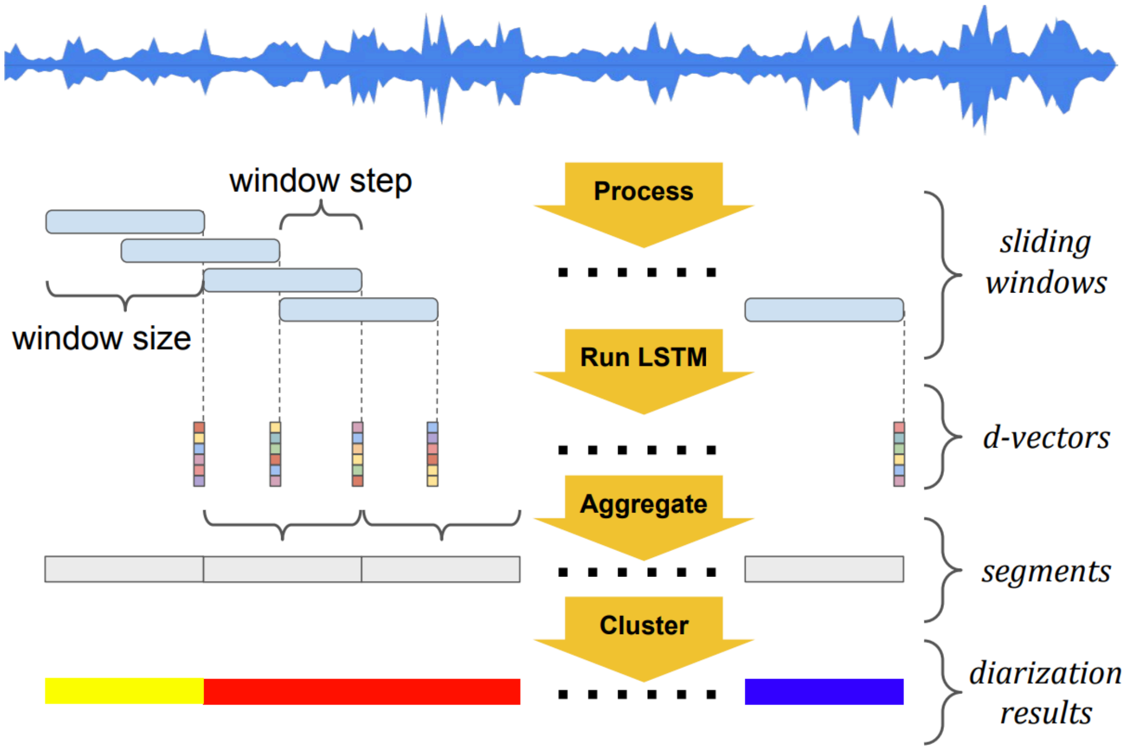


Recap: Speaker recognition with d-vector



Wan, Li, Quan Wang, Alan Papir, and Ignacio Lopez Moreno. "Generalized end-to-end loss for speaker verification." In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4879-4883. IEEE, 2018.

Recap: Baseline "dense d-vector" diarization system



Wang, Quan, Carlton Downey, Li Wan, Philip Andrew Mansfield, and Ignacio Lopz Moreno. "Speaker diarization with LSTM." In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5239-5243. IEEE, 2018.

The rise of supervised diarization!

- Fully supervised approaches for diarization became popular!
- They replace the unsupervised clustering algorithms
- Representative works:
  - UIS-RNN, [ICASSP 2019]
  - Discriminative Neural Clustering, [SLT 2021]
  - Permutation invariant training (E2E diarization)
    - [Google Patent 2018]
    - [Interspeech 2019]

Annotating the training data

- To train a supervised diarization model, what kind of data do we need?
  - Conversational audio
  - **Time-annotated** speaker labels, e.g.:
    - `start: 0.0, end: 1.2, speaker: A`
    - `start: 1.3, end: 4.4, speaker: B`
    - `start: 6.7, end: 9.4, speaker: A`
- This kind of annotation is extremely **expensive** and **error-prone**!
  - Annotator needs to frequently **go back** to compare with previous speakers
  - Single pass annotation of **10 min** of audio takes about **2 hours**!

Part 2:

Speaker turn detection

How to make annotations easier?

- From “who spoke **when**” to “who spoke **what**”
  - In most applications of diarization, we care about “what” more than “when”
  - “what” means the text transcript from ASR

```
start: 0.0, end: 1.2, speaker: A
start: 1.3, end: 4.4, speaker: B
start: 6.7, end: 9.4, speaker: A
```

Who spoke when

```
spkA: good morning
spkB: morning how are you
spkA: good what about you
```

Who spoke what

How to make annotations easier?

- From “speaker **labels**” to “speaker **turns**”
  - Annotating speaker labels requires frequently checking previous speakers from long term history
  - Annotating speaker turns:
    - Only focuses on short term context
    - Minimal incremental efforts on top of regular ASR annotation

```
spkA: good morning
spkB: morning how are you
spkA: good what about you
```

Speaker labels

```
good morning <st>
morning how are you <st>
good what about you <st>
```

Speaker turns

Motivation

- **Speaker turn annotations** are much easier to obtain at large scale, compared with time-annotated speaker labels
- How can we make use of them for speaker diarization?
  - Train an ASR-alike speaker turn detection model
  - Compute turn-wise speaker embeddings
  - Use speaker turns to constrain the unsupervised clustering algorithm

Speaker turn detection

- We treat speaker turn as a new special token <st>
- It is jointly trained with the ASR model

```
good morning morning how are you
good what about you
```

Transcript to train ASR

```
good morning <st> morning how are
you <st> good what about you <st>
```

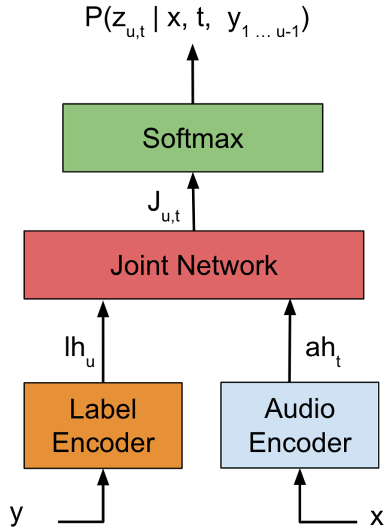
Transcript to train ASR + speaker turn

Transformer transducer model

- We use the transformer transducer architecture
  - Zhang, Qian, et al. "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss." *ICASSP*. IEEE, 2020.
  - 75 possible graphemes in the output

Table 1. Hyper-parameters of a Transformer block.

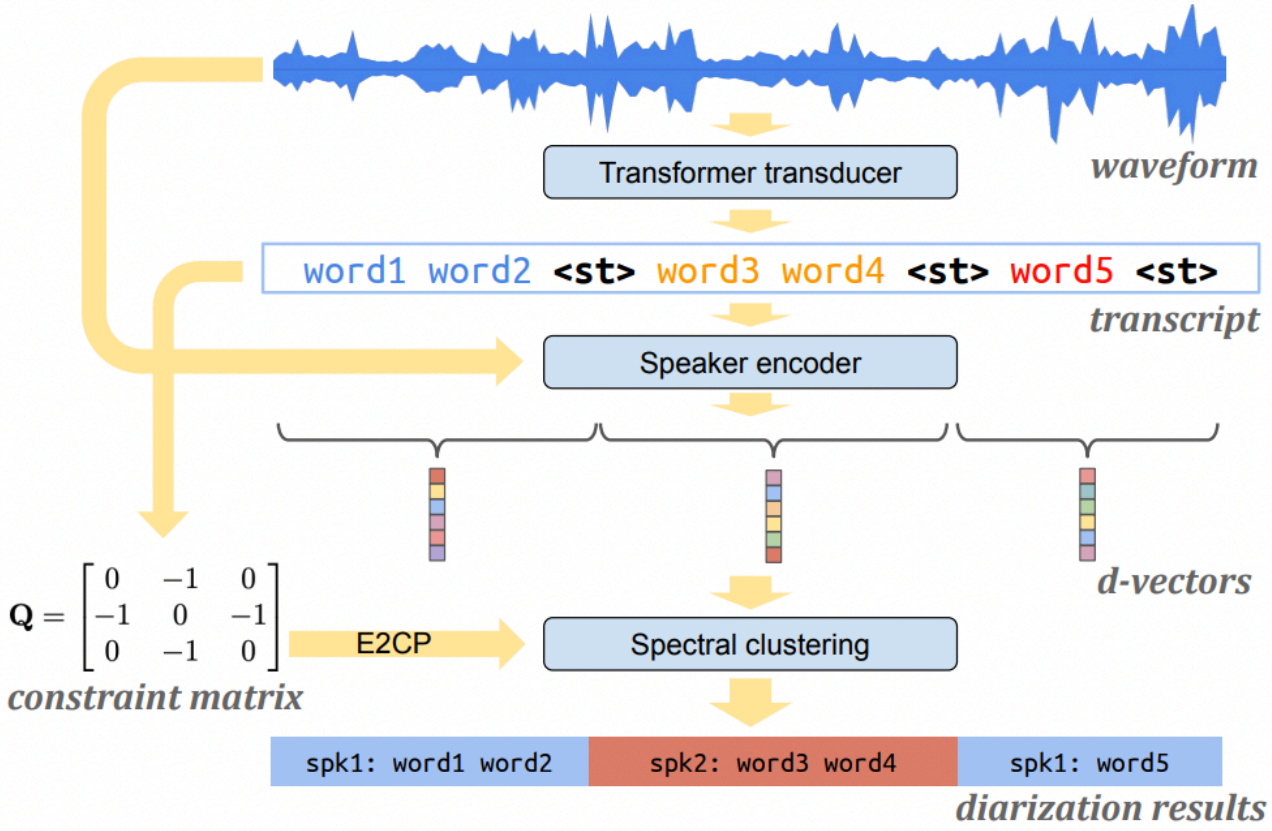
Input feature projection	256
Dense layer 1	1024
Dense layer 2	256
Number attention heads	8
Head dimension	64
Dropout ratio	0.1



Part 3:

Turn-to-Diarize

System architecture



Core components

- Speaker turn detection
  - Use the transformer transducer jointly trained with ASR
  - If a turn is longer than 6 seconds, insert a "**fake turn**" (compensate for false rejects)
- Speaker encoder
  - The d-vector model
  - Reset states at speaker turn boundaries
  - One embedding for **each turn** (instead of fixed-length segment)
- Spectral clustering
  - Cluster the turn-wise d-vectors – makes clustering **drastically cheaper**!
  - Constrained by speaker turns

Spectral clustering (unconstrained)

- Basic steps:
    - Construct an affinity matrix based on cosine similarities
    - Refine the affinity matrix:
        - **No Gaussian blur** (only useful for dense d-vectors)
        - Row-wise soft-thresholding with **auto-tune**
        - Symmetrization
    - Eigen-decomposition of **Laplacian matrix**
    - Estimate number of speakers via eigen-gap
    - K-means on re-normalized spectral embeddings

Speaker turn priors

- We want to integrate prior information from speaker turns into the clustering process
- For the transformer transducer model:
  - Some `<st>` tokens are **fake** tokens to avoid very long turns
  - Each **true** `<st>` token has a confidence: $c(\texttt{<st>})$
- Neighboring segments:
  - "Cannot-link" (CL): If they are segmented by a true `<st>` token
  - "Must-link" (ML): If they are segmented by a fake `<st>` token

Constraint matrix

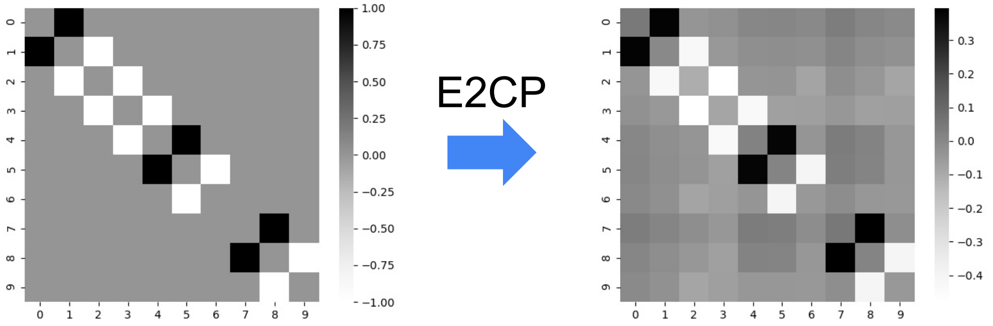
- The constraint matrix $\mathbf{Q}$ has the same shape as the affinity matrix $\boldsymbol{A}$
- Heuristics:
  - Segments around **highly confident** `<st>` must be from **different** speakers
  - Segments around **fake** `<st>` must be from the **same** speaker

$$\mathbf{Q}_{ij} = \begin{cases} -1, & \text{If } (i,j) \in \text{CL and } c(\texttt{<st>}) > \sigma; \\ +1, & \text{If } (i,j) \in \text{ML}; \\ 0, & \text{Otherwise.} \end{cases}$$

Propagating the constraints

- We want to propagate constraints to **non-neighboring** segments
  - Example: $A = B, A \neq C \Rightarrow B \neq C$
- We use Exhaustive and Efficient Constraint Propagation (E2CP)

$$\mathbf{Q}^* = (1-\alpha)^2 (I - \alpha\bar{\mathbf{A}})^{-1}\mathbf{Z}(I - \alpha\bar{\mathbf{A}})^{-1}.$$



Constrained spectral clustering

- Once we have the propagated constraint matrix $\mathbf{Q}^*$, we adjust the affinity matrix:

$$\hat{\mathbf{A}}_{ij} = \begin{cases} 1 - \left(1 - \mathbf{Q}^*_{ij}\right)\left(1 - \mathbf{A}_{ij}\right), & \text{If } \mathbf{Q}^*_{ij} \geq 0\,; \\ \left(1 + \mathbf{Q}^*_{ij}\right)\mathbf{A}_{ij}, & \text{If } \mathbf{Q}^*_{ij} < 0\,. \end{cases}$$

- This happens before the refinement operations
- Workflow: affinity → constraint → refinement → Laplacian

Part 4:

Experiments

Datasets

- Speaker turn detection training sets:
  - Fisher, Callhome American English (training subset), ~7500 hours of YouTube videos (internal)
- Speaker encoder training sets:
  - Vendor collected speech queries (37 locales), LibriVox, CN-Celeb, TIMIT, VCTK
- Diarization eval sets:
  - “Outbound” telephone speech: 450 conversations, each with 2 speakers
  - “Inbound” telephone speech : 250 conversations, each with 2~10 speakers
  - Callhome American English (eval subset)

Experiment protocol

- We compare two systems:
  - “Dense d-vector”: Cluster d-vectors from fixed-length segments (400ms)
  - “Turn-to-diarize”: Cluster d-vectors from speaker turns (max 6s, average ~4s)
- We report:
  - Confusion and Diarization Error Rate (DER)
  - Floating point operations to process one second of audio (FLOP/s) after 10min and 1h
    - Assume clustering runs every 4s
    - Assume auto-tune searches 10 steps

Results: computational cost

Table 2. Confusion (%), total DER (%) and GFLOP/s on three datasets for different embeddings and methods.

System	Method	Inbound		Outbound		Callhome Eval		GFLOP/s at 10min	GFLOP/s at 1h
		Conf.	DER	Conf.	DER	Conf.	DER		
Dense d-vector	Dense	17.98	22.13	10.66	15.97	5.39	7.76	0.85	36.54
	Dense + Auto-tune	14.09	18.24	9.56	14.88	5.42	7.79	4.76	361.37
Turn-to-diarize	Turn	17.87	19.43	8.41	10.34	8.23	10.08	1.00	1.18
	Turn + E2CP	17.21	18.77	7.94	9.86	3.56	5.41	1.00	1.18
	Turn + Auto-tune	13.83	15.39	7.01	8.93	5.11	6.95	1.02	2.81
	Turn + E2CP + Auto-tune	**13.66**	**15.22**	**6.86**	**8.78**	**3.49**	**5.33**	1.02	2.81

- Clustering a long sequence of dense d-vectors is **extremely expensive**
  - Mostly due to **eigen-decomposition**
  - Especially after running for a **long time**
  - Even more expensive if we want to **auto-tune** the refinement threshold

Results: computational cost

Table 2. Confusion (%), total DER (%) and GFLOP/s on three datasets for different embeddings and methods.

System	Method	Inbound		Outbound		Callhome Eval		GFLOP/s at 10min	GFLOP/s at 1h
		Conf.	DER	Conf.	DER	Conf.	DER		
Dense d-vector	Dense	17.98	22.13	10.66	15.97	5.39	7.76	0.85	36.54
	Dense + Auto-tune	14.09	18.24	9.56	14.88	5.42	7.79	4.76	361.37
Turn-to-diarize	Turn	17.87	19.43	8.41	10.34	8.23	10.08	1.00	1.18
	Turn + E2CP	17.21	18.77	7.94	9.86	3.56	5.41	1.00	1.18
	Turn + Auto-tune	13.83	15.39	7.01	8.93	5.11	6.95	1.02	2.81
	Turn + E2CP + Auto-tune	**13.66**	**15.22**	**6.86**	**8.78**	**3.49**	**5.33**	1.02	2.81

- Turn-to-diarize made the sequence much shorter!
  - Clustering becomes very cheap
  - Even with auto-tune
  - Even after processing 1h of audio

Results: quality

Table 2. Confusion (%), total DER (%) and GFLOP/s on three datasets for different embeddings and methods.

System	Method	Inbound		Outbound		Callhome Eval		GFLOP/s at 10min	GFLOP/s at 1h
		Conf.	DER	Conf.	DER	Conf.	DER		
Dense d-vector	Dense	17.98	22.13	10.66	15.97	5.39	7.76	0.85	36.54
	Dense + Auto-tune	14.09	18.24	9.56	14.88	5.42	7.79	4.76	361.37
Turn-to-diarize	Turn	17.87	19.43	8.41	10.34	8.23	10.08	1.00	1.18
	Turn + E2CP	17.21	18.77	7.94	9.86	3.56	5.41	1.00	1.18
	Turn + Auto-tune	13.83	15.39	7.01	8.93	5.11	6.95	1.02	2.81
	Turn + E2CP + Auto-tune	**13.66**	**15.22**	**6.86**	**8.78**	**3.49**	**5.33**	1.02	2.81

- For turn-to-diarize:
  - Auto-tune is critical for the performance (especially for >2 speakers)
  - Speaker turn constraints offer additional improvement

Results: quality

Table 2. Confusion (%), total DER (%) and GFLOP/s on three datasets for different embeddings and methods.

System	Method	Inbound		Outbound		Callhome Eval		GFLOP/s at 10min	GFLOP/s at 1h
		Conf.	DER	Conf.	DER	Conf.	DER		
Dense d-vector	Dense	17.98	22.13	10.66	15.97	5.39	7.76	0.85	36.54
	Dense + Auto-tune	14.09	18.24	9.56	14.88	5.42	7.79	4.76	361.37
Turn-to-diarize	Turn	17.87	19.43	8.41	10.34	8.23	10.08	1.00	1.18
	Turn + E2CP	17.21	18.77	7.94	9.86	3.56	5.41	1.00	1.18
	Turn + Auto-tune	13.83	15.39	7.01	8.93	5.11	6.95	1.02	2.81
	Turn + E2CP + Auto-tune	**13.66**	**15.22**	**6.86**	**8.78**	**3.49**	**5.33**	1.02	2.81

- Best turn-to-diarize significantly outperforms best “dense d-vector”

Other benefits

- One of the biggest challenges in diarization:
    - Knowing "when to diarize"
    - Does this utterance has at least two speakers?
        - Yes - We need diarization
        - No - No need of diarization
- Turn-to-Diarize allows a "smart mode":
    - Start with the transformer transducer only, disable anything else
    - Only enable speaker encoder and clustering after the first `<st>` has been detected

Part 5:

Python library

The spectralcluster package

- We offer a Python re-implementation
  - Only implements constrained spectral clustering, not the full diarization system
  - Available at: https://github.com/wq2012/SpectralCluster
  - Similar APIs with standard scikit-learn clustering algorithms
  - Unit tests with >90% coverage
  - Detailed documentation
  - PyPI packaging:

```
$ pip3 install spectralcluster
```

Features

- Refinement operations on affinity matrix
- Different types of Laplacian matrix
- Customized distance for K-means
- Auto-tune of the threshold
- Constrained spectral clustering with E2CP
- Built-in configurations used by our paper:

```
from spectralcluster import configs

labels = configs.turntodiarize_clusterer.predict(embeddings, constraints)
```

Part 6:

Conclusions and future work

Conclusions

- We proposed Turn-to-Diarize:
  - Easier annotation: No need for time-annotated speaker labels, just need speaker turns in transcripts
  - Transformer transducer model for joint ASR and speaker turn detection
  - One speaker embedding for each turn
  - Constrained spectral clustering via E2CP
- Experimental results:
  - Drastically reduces computational cost of clustering
  - Significant improvement with auto-tune and E2CP
  - Outperforms the best dense d-vector system

Future work

- Multilinguality
  - Our current transformer transducer model is only trained with English
  - We want to train on massively multilingual datasets to generalize to more languages
- Multimodality
  - The constrained spectral clustering idea can generalize to other modalities
  - E.g. visual signals, which offer stronger confidence on user identities

Questions?