A ZOOM FILTER FOR APPLAUSE AND LAUGHTER

Meeting 26.01.22

Where we left off

- The iterations take a long time
  - Only 30 batches in 2 hours
    - Which equals 30*32 = 960 segments
      - 8s audio per minute
- Job is removed after 2 hours

Debugging

- Only ~20% of GPU were utilised when running training

- RTF for loading ICSI:
  - Best case RTF: 1.2
  - worst case RTF: 2.5

- RTF for evaluation on ICSI:
  - Best case: 0.10
  - Worst case: 1.49

RTF= time_to_process/length_of_segment

SINGLE MEETING

- 6 different audio files
  - command: `s_train.audio_path.unique().size`

on Thinkstation CPU - Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz

num_of_batches	total_time [s]	av_time_per_batch [s]	num_of_workers
1	60.73	60.73	8
3	191.08	63.69	8
5	300.04	60.01	8

on AT GPU Machine

num_of_batches	total_time [s]	av_time_per_batch [s]	num_of_workers
5	194.13	38.83	8

on MLP-Cluster GPU Machine -> **loading from DFS**

- allocated memory: 16000MB

num_of_batches	total_time [s]	av_time_per_batch [s]	num_of_workers
1	79.26	79.26	8
5	392.16	78.43	8

on MLP-Cluster GPU Machine -> **loading from scratch disk**

num_of_batches	total_time [s]	av_time_per_batch [s]	num_of_workers
1	80.71	80.71	8
1	75.07	75.07	24 + 32GB mem
5	392.16	78.43	8

Loading from different offsets

- Switchboard average conversation length: 6.5min = 390s

- ICSI average meeting length: 56min = 3360s

Segment length [s]	Offset [s]	Total_time [s]	Average_time [s]	RTF
1	0	1.65	0.17	0.17
1	390	3.70	0.37	0.37
1	3360	18.79	1.88	1.88
20	0	1.65	0.16	0.01
20	390	3.80	0.38	0.02
20	3360	18.96	1.90	0.09

RTF= time_to_process/length_of_segment

Current plan

1. Split create_data_df to match the train/val/test split used by lhotse
2. load those data_dfs in lhotse data_loader script
3. create cuts for each in the data_dfs
4. Compute and store features for all these cuts in the format used by Gillick et al.
5. Create pytorch dataloader for these cuts, that loads the already computed feature from disk