Target Speaker - Voice Activity Detection (TS-VAD)

Ivan Medennikov et al., STC & ITMO University, Russia.

Presented by: Desh Raj
What is TS-VAD?

- STC’s new system for speaker diarization (“who spoke when”)
- Used for the first time in the CHiME-6 challenge
- Paper submitted to Interspeech 2020
- Helped them get **DER of 36%** in the challenge (second best was 65%)
The CHiME-6 challenge
Cocktail Party Problem

• 4 speakers having real conversations in their home.
• Recorded on 6 Kinect arrays (each containing 4 microphones)
• Very noisy (utensils, appliances, cooking etc.)
• Up to 50% overlap
• Example: https://chimechallenge.github.io/chime6/overview.html
The CHiME-6 challenge
Cocktail Party Problem

- Only external data allowed -> VoxCeleb2
- 1 million untranscribed utterances from 6k celebrities; used mostly to train neural speaker embedding extractors (x-vectors)
- CHiME-6 training data is only about 50 hours :

**Conclusion**: CHiME-6 is HARD!
A standard diarization system


Cannot detect overlapping speaker segments
How to solve this?

- **EEND [1]**: Outputs frame-level probability of each speaker independently, but requires large amount of training data and short utterances (up to 10 mins)

- Target-speaker speech extraction:
  - Target-speaker ASR [2]
  - SpeakerBeam [3]
  - VoiceFilter [4]

These methods take some speaker features as additional input and estimate time-frequency mask for that speaker

---

How to solve this?

- **Key difficulty:** We do not have clean pre-computed speaker features.

- Need to estimate from the highly overlapping speech.
Suppose we are given i-vectors for each speaker (from oracle segmentation), how can we detect their speech in overlapping conditions?

• Very similar to Personal VAD model [5]

• Takes i-vector and MFCCs of recording

• Outputs frame-level probability of speaker

• DER 66.8% on dev :(  

Single speaker TS-VAD

With oracle i-vectors

- **Trick:** If posterior probability of a speaker is less than the maximum probability on that frame by more than a threshold, then set to 0.

- Example: 0.97, **0.65, 0.58, 0.21**

- Simple single-speaker TS-VAD will output speakers 1,2,3

- But with threshold (say 0.3), only outputs 1.

- DER 46.1%
Multi-speaker TS-VAD

With oracle i-vectors

- Single-speaker TS-VAD predicts output for 1 speaker at a time.
- Can we borrow some ideas from EEND, which performs multi-label classification?
- Important: number of speakers is 4 (fixed)
Multi-speaker TS-VAD

With oracle i-vectors

- Trained on sum of binary cross-entropies
- Forced alignment to obtain training targets
Multi-speaker TS-VAD

With oracle i-vectors

• Data augmentation: on-the-fly random permutation of speakers during training

• “mixup” training [6]
  
  \[ \tilde{x} = \lambda x_i + (1 - \lambda)x_j, \quad \text{where } x_i, x_j \text{ are raw input vectors} \]
  
  \[ \tilde{y} = \lambda y_i + (1 - \lambda)y_j, \quad \text{where } y_i, y_j \text{ are one-hot label encodings} \]

• DER 37.4% (without the “trick” used in single-speaker TS-VAD)

Multi-speaker TS-VAD

With estimated i-vectors

• But all of these are using i-vectors estimated from oracle speaker segments!

• Can it give similar gains with i-vectors estimated from the baseline diarization system?

• Iterative process:
  • Start with i-vectors from baseline CHiME-6 system
  • Next step: i-vectors from TS-VAD iteration 1…. And so on
  • DER improvement ~15% (but requires more iterations)
Multi-speaker TS-VAD

With estimated i-vectors

• Better initialization: 34 layer Wide ResNet x-vector extractor + spectral clustering

• 47% DER (compared to 63% baseline)

• Use this for initializing TS-VAD ivectors:
  • Converges in 2 iterations
  • ~35% DER finally (similar to using oracle i-vectors)
Multi-channel TS-VAD

4-layer CNN -> Speaker Detection 2-layer BLSTMP (shared weights) -> output1, output2, output3, output4

Channel 1: SD1, SD2, SD3, SD4
Channel 2: SD1, SD2, SD3, SD4
... Channel N: SD1, SD2, SD3, SD4

Combining 1-layer BLSTMP
**TS-VAD post-processing**

- Frame-level posteriors are actually used as emission probabilities for an HMM.
- HMM contains 11 states:
  - Silence state
  - 4 states corresponding to single speaker
  - 6 states corresponding to 2 speaker combinations (4 choose 2)
- Transition from silence to 2-speaker state and vice-versa prohibited
- Viterbi decoding to get final output sequence
- 2% DER improvement
Thank You!