INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Speaker Diarization for ASR Output with T-vectors: A Sequence Classification Approach

Midia Yousefi, Naoyuki Kanda, Dongmei Wang, Zhuo Chen, Xiaofei Wang, Takuya Yoshioka

Microsoft, One Microsoft Way, Redmond, WA, USA

{midiayousefi, nakanda, dowan, zhuc, xiaofewa, tayoshio}@microsoft.com

Abstract

This paper considers applying speaker diarization (SD) to the output tokens of automatic speech recognition (ASR). We formulate the task to be solved as a sequence classification problem, where we estimate the correct speaker label for each ASR output token based on a sequence of token-level speaker embeddings and candidate speaker profiles. To leverage the information from the ASR model, we utilize a recently proposed t-vector for the speaker embedding estimation. Whereas previous studies performed t-vector classification using cosine similarities with ad hoc post-processing, we propose to use a sequence classification model to leverage the sequential nature of the task more effectively. To handle a variable number of speakers, we use a classification model inspired by a target speaker voice activity detection based on transformers. We conduct experiments using the AMI meeting corpus in both speaker identification and diarization settings and show the effectiveness of our approach.
Index Terms: speaker diarization, multi-talker speech recognition, serialized output training, speaker embedding

1. Introduction

The increasing popularity of podcasts and online videos as well as the widespread use of meeting recordings resulted in a surge of the demand for automatic speech recognition (ASR) of conversation recordings [1, 2, 3]. Speaker diarization (SD) [4], a technology to label audio recordings with classes that correspond to speaker identity, is one of the indispensable components for such conversation transcription systems. SD provides greater readability and also helps subsequent natural language processing. The speaker identity can be represented by either a real name or an anonymous ID.

In an exemplary architecture of a conversation transcription system, SD is first applied, and ASR is then performed for each speech segment detected by SD [5, 6]. With the advancement of the SD technology [7, 8], this approach demonstrated a great capability to deal with challenging conversations containing many overlapped speech segments [2]. However, previous studies [9, 10, 11, 12] showed that the SD accuracy could be greatly improved by incorporating information from ASR, such as word time boundaries and linguistic features.

With the ASR technology improvement in handling long-term contexts and overlapped speech, it has become increasingly common to apply ASR directly to multi-talker audio [13, 14, 15, 16]. When ASR is performed independent of SD, each ASR output token must be labeled by speaker identity afterwards. This could be accomplished by aligning the ASR and SD outputs based on estimated word and speaker-segment time stamps. However, aligning the ASR and SD results can be challenging, as they handle overlaps of multiple talkers' utterances differently.

One solution to this challenge is to estimate a speaker embedding vector for each ASR output token, or a set of consecutive ASR output tokens, and classify them by speakers[1] [17, 18, 19]. A naïve way of estimating speaker embedding vectors based on word time stamps could be insufficient for dealing with overlapped speech. Recently, Kanda *et al.* [19] proposed a speaker embedding vector, called a t-vector, utilizing transformer transducer (TT) ASR models [20]. The t-vector is estimated for each ASR token by leveraging the intermediate representations obtained from the ASR encoder and the recognized tokens. The t-vector model can also be applied to multi-talker ASR models obtained with token-level serialized output training (t-SOT). In [19], the t-vectors were classified into speakers by measuring them against a set of speaker d-vectors (speaker profiles) by cosine similarity. However, some ad hoc post-processing was necessary to smooth out discontinuities in the speaker label assignments.

In this paper, we investigate the use of a sequence classification model that can take an entire t-vector sequence as input and eliminate the need for a separate post-processing step. Since the number of speakers can vary for each audio sequence, a fixed number of output dimensions cannot be assumed. Therefore, we utilize a model inspired by the recently proposed transformer model [21] for target speaker voice activity detection (TS-VAD) [8]. In contrast to the previous t-vector work, which performed evaluation only on simulated data sets [19], we conduct experiments using actual meeting recordings from the AMI corpus [22] and show the effectiveness of our proposed sequence classification model. It is worth noting that extending the t-vector approach to real meeting data poses a challenge due to the lack of sufficient transcribed training data with speaker annotations. In our experiments, we alleviated this by adopting a pseudo-labeling technique to pre-train the t-vector model with data containing only speaker labels. The pre-trained model was then fine-tuned using a limited amount of real transcribed data.

2. Speaker diarization for ASR output

We start by framing the problem we wish to solve using a general term. We consider an ASR system that generates a sequence of tokens $(r_1, \cdots, r_U)$ from an audio input, where the tokens can be subwords or words. Each token r_u has an associated F-dimensional embedding, $\mathbf{e}_u \in \mathbb{R}^F$, that contains information for identifying the speaker of that token. Additionally, we assume that we have a set of speaker profiles, $\mathbf{P} = \{\mathbf{p}_s\}_{s\in\{1,\cdots,S\}}$, where S is the number of potential speakers, and $\mathbf{p}_s \in \mathbb{R}^P$ is a P-dimensional embedding representing

[1]Note that this approach simplifies the SD task from frame-based multi-label classification to a token-based single-label problem.

10.21437/Interspeech.2023-494

the sth potential speaker. The speaker profiles can be obtained from users (speaker identification scenario) or estimated using an independent frame-based diarization method (SD scenario). Our task is to classify each token-level embedding $\mathbf{e}_u$, associated with the ASR output token r_u, into their respective speaker index s, given the speaker profile set $\mathbf{P}$.

In this paper, we use multi-talker TT ASR models based on t-SOT [16] with token-level speaker embeddings called t-vectors [19]. While the proposed framework is applicable to other models,[2] we chose these methods as they produced state-of-the-art benchmarking results for the LibriSpeechMix [15] and LibriCSS [25] corpora. These models are briefly reviewed below.

2.1. T-SOT: multi-talker TT ASR

T-SOT attempts to handle overlapping talkers with TT ASR models [16]. Assume we have an audio recording with multiple active speakers. A t-SOT ASR model generates tokens of transcriptions for all speakers as a single sequence, where all the tokens are sorted chronologically using their vocalized times. To deal with overlapping speech, t-SOT introduces an idea of virtual output channels: each virtual output channel must contain non-overlapping tokens while the tokens in different channels may overlap. The t-SOT ASR model inserts a special token between two adjacent output tokens spoken by different speakers to indicate a change of the virtual output channels. If we assume that a maximum of two talkers can speak at the same time, we need only two virtual output channels and hence one such special symbol, which we denote as $\langle cc \rangle$ following [16].

During training, for each training sample, we align the audio with the reference transcription, sort the tokens of all speakers chronologically, and insert $\langle cc \rangle$ between the adjacent tokens spoken by different speakers. At inference time, the TT ASR model generates a sequence of tokens including $\langle cc \rangle$ and thus can handle overlapping utterances. We refer the reader to [16] for more details.

2.2. T-vector: speaker embedding for TT ASR

T-vector [19] is a speaker embedding designed for use with a TT ASR model. It involves incorporating an additional neural network, which we refer to as a t-vector model, into a pre-trained ASR model. By utilizing the connections from the ASR model layers, the t-vector model can leverage both acoustic and lexical information to improve speaker classification.

The t-vector model comprises a speaker encoder and a speaker decoder. The speaker encoder processes audio input in tandem with the ASR encoder on a frame-by-frame basis, and consists of a Res2Net module [26, 27] followed by a stack of multi-head attention (MHA) layers. The Res2Net module has the same architecture with that of the d-vector model used for speaker profile extraction. The number of MHA layers in the speaker encoder is the same as that of the ASR encoder. Each MHA layer of the speaker encoder uses the key and query inputs from the corresponding MHA layer of the ASR encoder. The speaker decoder consists of two bidirectional long short-term memory (BLSTM) layers, and generates a t-vector for each token based on the speaker encoder output and ASR output tokens.

[2]For instance, a potentially less powerful, yet simpler, approach would be to employ a standard ASR module that generates a time-stamped word sequence. Meanwhile, we execute a speaker embedding extraction model on the input audio to acquire frame-level speaker embedding vectors, such as d-vectors or x-vectors [23, 24]. Next, for every word produced by the ASR model, we would compute the average speaker embedding for the duration of time that the word occupies.

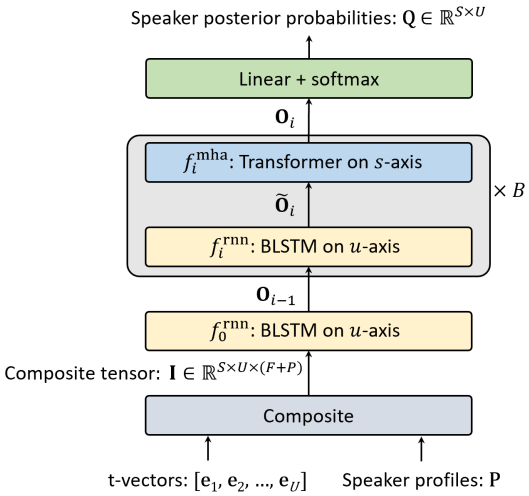

Figure 1: *Sequence classification model for t-vectors.*

The t-vector model is trained using a speaker identification cross entropy loss. During training, the cosine similarities between the t-vector and candidate speaker profiles are calculated and used to compute the following loss function:

$$L = \sum_{u|_{r_u \neq \langle cc \rangle}} -\log \frac{e^{\cos(\mathbf{e}_u, \mathbf{p}_{s_u})}}{e^{\cos(\mathbf{e}_u, \mathbf{p}_{s_u})} + \sum_{s \neq s_u} e^{\cos(\mathbf{e}_u, \mathbf{p}_s)}}. \quad (1)$$

Here, $\mathbf{e}_u$ is the t-vector for the u-th token, s_u is the index of the correct speaker for token r_u. For each training sample, the speaker profile set $\mathbf{P}$ is obtained by randomly picking S enrollment audio files from the entire training set and extracting d-vectors by using a pre-trained Res2Net-based model [26, 27].

2.3. Speaker diarization with t-vector classification

During inference, once we obtain ASR output tokens (r_u) and associated t-vector sequence $(\mathbf{e}_u)$, we classify each t-vector, $\mathbf{e}_u$, to the correct speaker of the corresponding ASR output token r_u. The speaker label is defined as an index of the speaker profile set $\mathbf{P}$. In [19], speaker classification was performed for each embedding independently by calculating the cosine similarity between $\mathbf{e}_u$ and each speaker profile $\mathbf{p}_s$ and choosing the speaker with the maximum cosine similarity.

However, the previously adopted token-by-token classification approach may not be optimal for taking into account the statistical dependency that exists between neighboring t-vectors. Such statistical dependency results from the fact that consecutive ASR tokens are likely to be spoken by the same person. Therefore, it could be beneficial to adopt a sequence classification approach to leverage the inter-t-vector dependency. Additionally, such an approach would pave the way for modeling the turn-taking dynamics between multiple talkers, although we leave the analysis of this aspect for future work.

3. Sequence classification of t-vectors

Our goal is to develop a sequence classification model that can classify a sequence of t-vectors based on a set of candidate speaker profiles. Since the candidate speaker profiles are different for each test audio (i.e., each meeting), we cannot use a sequence classification model with a fixed number of output

Table 1: *Comparison of token-by-token t-vector classification using cosine similarity and proposed sequence classification in terms of speaker classification error rate (SCErr) and cpWER on LibriSpeech-based simulated data.*

Model	SCErr (%)	cpWER (%)
Token-by-token classification	1.5	11.9
Sequence classification	0.9	11.2

dimensions. To address this, we employ a classification model that is inspired by a transformer-based TS-VAD model [21], as described below. The proposed model is also depicted in Fig. 1.

Suppose we have the t-vector sequence $(\mathbf{e}_1 \cdots \mathbf{e}_U)$ and the candidate speaker profile set $\mathbf{P} = \{\mathbf{p}_s\}_{s\in\{1,\cdots,S\}}$ as defined in Section 2. We formulate our goal to estimate 2D tensor $\mathbf{Q} \in \mathbb{R}^{S\times U}$ whose value at position (s, u) provides the posterior probability of the uth token, r_u, being spoken by the sth speaker. This is achieved by using the model described below.

First, we transform the t-vector sequence and the candidate speaker profile set into a 3D tensor $\mathbf{I} \in \mathbb{R}^{S\times U\times(F+P)}$, which we call a composite input tensor. The 1D slice of $\mathbf{I}$ at position (s, u) of the first two dimensions, i.e., $\mathbf{I}[s, u, :]$ with a slice notation, is the concatenation of the uth t-vector and the sth profile, namely $\mathbf{e}_u$ and $\mathbf{p}_s$, respectively. We preprocess the composite input tensor $\mathbf{I}$ by applying a BLSTM to the t-vector sequence dimension, i.e., the second dimension indexed by u, to obtain another 3D tensor $\mathbf{O}_0$, which forms an input to the subsequent dual-path blocks. That is, we have

$$\mathbf{O}_0 = [f_0^{\text{rnn}}(\mathbf{I}[s, :, :])]_{s=1,\cdots,S}, \tag{2}$$

where f_0^{rnn} denotes the sequence mapping defined by the BLSTM. The ith dual-path block transforms $\boldsymbol{O}_{i-1}$ to another 3D tensor $\boldsymbol{O}_i$ by first applying a BLSTM to the t-vector sequence dimension and then a transformer encoder to the speaker dimension as follows:

$$\tilde{\mathbf{O}}_i = [f_i^{\text{rnn}}(\mathbf{O}_{i-1}[s, :, :])]_{s=1,\cdots,S} \tag{3}$$

$$\mathbf{O}_i = [f_i^{\text{mha}}(\tilde{\mathbf{O}}_i[:, u, :])]_{u=1,\cdots,U}, \tag{4}$$

where f_i^{rnn} and f_i^{mha} denotes the sequence mappings defined by the BLSTM and the transformer encoder, respectively. Note that the BLSTM is applied to the second dimension indexed by u in Eq. (3), whereas the transformer encoder is applied to the first dimension indexed by s in Eq. (4). After applying B dual-path blocks, we linearly project each vector in the last dimension to a scalar to generate logits, i.e.,

$$\tilde{\mathbf{Q}} = [f^{\text{lin}}(\mathbf{O}_B[s, u, :])]_{(s,u)=(1,1),\cdots,(S,U)}. \tag{5}$$

Finally, we obtain the desired posterior probability tensor by applying softmax function to the speaker dimension. That is,

$$\mathbf{Q} = f^{\text{softmax}}(\tilde{\mathbf{Q}}). \tag{6}$$

The parameters involved in this model are optimized with the same cross entropy loss as Eq. (1). More specifically, $\cos(\mathbf{e}_u, \mathbf{p}_s)$ in Eq. (1) is replaced by the (s, u) element of $\tilde{\mathbf{Q}}$.

4. Experiments

Section 4.1 reports a proof-of-concept experimental result obtained using simulated data. Section 4.2 describes our results on real meeting recordings. For both experiments, we used the same d-vector model based on Res2Net [26] for speaker profile generation. We employed the VoxCeleb corpora [28, 29] for the d-vector model training, following the recipe of [27].

Table 2: *cpWER (%) results for ASR + SID evaluation setting. Speaker-Agnostic WERs were 16.3%, 18.9%, 21.6%, and 25.3% for IHM-mix-dev, IHM-mix-eval, SDM-dev, and SDM-eval, respectively. Utterance-group segmentations were used. The number of parameters of t-SOT TT ASR model was 82M.*

t-vector model	Classification model	IHM-mix dev	IHM-mix eval	SDM dev	SDM eval
28M	token-by-token	27.6	26.5	32.2	32.3
32M	token-by-token	27.4	26.5	33.1	32.5
28M	seq-cls (4M)	**25.1**	**24.9**	**30.3**	**31.0**

4.1. Experiment with simulated data

Data: We used the LibriSpeech [30] corpus to simulate conversations. The training data were simulated by randomly selecting at most five utterances from the train-960 subset and mixing them with random delays without amplitude rescaling. Randomly generated room impulse responses were applied to the individual utterances. Also, a randomly chosen noise sample was added to the mixed signal at a random signal-to-noise ratio. The d-vector of each speaker in the mixture was created by averaging d-vectors extracted from five randomly picked utterances of the same speaker. The evaluation data were generated in the same way based on the dev-clean subset.

Model configuration and training: We trained all models, namely the ASR model, the t-vector extraction model, and the t-vector sequence classification model, on the LibriSpeech mixture data described above. Our ASR model was based on a t-SOT TT with an 18-layer transformer encoder, and built by using the configuration and recipe described in [16]. We employed the same configuration as [19] for the t-vector model except that we used a non-causal Res2Net in the speaker encoder and a 2-layer BLSTM with 512 cells in each direction for the speaker decoder. Finally, our t-vector sequence classification model comprised 2 dual-path blocks using the following configurations: f_0^{rnn} with a 2-layer BLSTM with 128 cells in each direction, f_i^{rnn} with a single layer BLSTM including 160 cells in each direction, and f_i^{mha} with a transformer encoder layer with 320-dim MHA with 4 attention heads followed by a 320-dim position-wise feed-forward layer, for $i > 0$. The t-SOT TT ASR model and the t-vector extraction model were first trained by following the training configuration in [19]. The t-vector sequence classification model was then trained by freezing the parameters of t-SOT TT ASR and t-vector models. We employed an AdamW optimizer with a linear decay learning rate schedule with a peak learning rate of 1.5e-4 after 10k warm-up iterations. The training was performed for 60k iterations with 16 GPUs, each of which consumed a mini-batchd of 12k frames.

Results: Table 1 shows the speaker classification error rate (SCErr) and concatenated minimum-permutation word error rate (cpWER) [2] results. The SCErr was obtained by using ground-truth ASR tokens and was defined as the percentage of the incorrectly speaker-identified tokens. This allows us to prevent both t-vector estimation and performance measurement from being affected by ASR errors. Our proposed model improved the SCErr by 40% relative, which demonstrates its effectiveness under the oracle ASR condition. When using estimated ASR tokens, the cpWER was improved by 5.9% relative, from 11.9% to 11.2%.

4.2. Experiment with real meeting recordings

We further conducted experiments using the AMI corpus [22] to evaluate the proposed model in more realistic settings. We

Table 3: *cpWER (%) results for ASR + SD evaluation setting. 28M-parameter model was used for t-vector computation.*

Classification model	Segmentation	IHM-mix		SDM	
		dev	eval	dev	eval
token-by-token	utt-grp	26.9	28.1	31.3	33.5
	vad	26.1	27.3	31.0	33.8
seq-cls model	utt-grp	24.8	26.0	29.9	32.0
	vad	**23.9**	**24.9**	**29.5**	**31.9**

considered two recording conditions: one using single distant microphone recordings (SDM), and one using mixtures of independent headset microphone recordings (IHM-mix).

4.2.1. Two-stage training

Since the quantity of the AMI training set is only about 80 audio hours, we adopted a two-stage training approach. For ASR, we pre-trained a t-SOT TT model on a 75k-hours of in-house speech dataset comprising 64 million anonymized, transcribed English utterances collected from various domains such as voice search and dictation [31]. When creating a mini-batch, we used a mixture of randomly picked two utterances instead of an original utterance at a 0.5 probability. The pre-trained model was then fine-tuned on the AMI training set. We adopted the model configuration and training recipe of [16] and thus used an 18-layer TT, in which each transformer layer consisted of a 512-dim MHA with 8 heads and a 2048-dim point-wise feed-forward layer. The ASR decoder was a prediction network with 2 layers of 1024-dim LSTM. The input to the ASR model was 80-dim log mel-filterbank (LMFB) extracted every 10 msec.

We used the VoxCeleb corpora [28, 29] to train a t-vector extraction model. Because the VoxCeleb corpora provides only speaker labels, we used our in-house ASR model to generate pseudo transcriptions for the training utterances. Based on the VoxCeleb utterances, we generated multi-talker training samples with additional random noise and reverberation in the same way as Section 4.1. With the ground-truth speaker labels and pseudo transcriptions, we pre-trained the t-vector model for 60k iterations on 16 GPUs using 12k-frame mini-batches. The pre-trained t-vector model was fine-tuned on the AMI SDM and IHM-mix training set for another 500 iterations. The t-vector model consisted of a speaker encoder and decoder. The first block in the speaker encoder was a Res2Net model followed by 18 layers of 128-dim, 8-head MHA to extract raw speaker representations. The speaker decoder was a 2-layer BLSTM network with 512 cells in each direction. The input to the t-vcetor model was 80-dim LMFB. Note that the ASR model parameters were frozen during the training so that the use of the pseudo transcriptions would have no negative impact on the ASR accuracy.

Finally, the t-vector sequence classification model was also pre-trained on VoxCeleb-based simulated data for 60k iterations on 16 GPUs using 12k-frame mini-batches. The pre-trained model was then fine-tuned on the SDM and IHM-mix audio data of the AMI training set for another 500 updates. We used the same model configuration as the one used in Section 4.1. When we trained the t-vector sequence classification model, both the ASR and t-vector extraction model paramters were fixed.

4.2.2. Results for ASR + SID setting

First, we performed evaluation by using an SID setting. In this experiment, for each speaker of each test meeting, we obtained isolated audio signals of the speaker from the corresponding IHM audio channel of the other meetings involving the speaker. This ensured the number of profiles to be the same as the number of attending speakers while it created an acoustic condition mismatch between the speaker profiles and the SDM test set. We used the utterance-group segmentations as described in [32] and measured the performance with cpWER.

Table 2 compares the cpWERs of the proposed sequence classification (seq-cls) model and the cosine-similarity-based (token-by-token) method. For the latter, we also experimented with a slightly larger t-vector model for a fair comparison with our sequence classification model in terms of the total model size. This t-vector model had a speaker decoder consisting of a 2-layer BLSTM with 615 cells in each direction while the original t-vector model had 512 cells in each direction. We can see that performing sequence classification on the t-vectors reduced the cpWER by 6.0 % and 4.0% relative for the IHM-mix-eval and SDM-eval sets, respectively. It is also noteworthy that further increasing the t-vector model capacity produced no cpWER gains with the cosine-similarity-based classification method. This indicates that the performance improvement provided by the proposed model can be attributed to the sequence modeling architecture rather than the increased model size.

4.2.3. Results for ASR + SD setting

We also evaluated the proposed model in a SD setting by estimating the speaker profiles directly from each test meeting. This was achieved by using a clustering-based SD method and calculating an average d-vector for each detected speaker from the meeting audio. Specifically, we divided each session into short segments based on the WebRTC Voice Activity Detection (VAD) system [33]. Then, we extracted a 128-dim d-vector for every 0.75 second with a window of 1.5 seconds. Based on the obtained d-vectors, we estimated the number of speakers in the meeting audio with the normalized maximum eigengap method [34]. With the estimated number of speakers, we applied spectral clustering for SD.

In addition to the utterance-group (utt-grp) segmentation used in the previous experiment in Sec. 4.2.2, we used automatically generated segments. For this, we applied WebRTC VAD to generate segments of duration 20–40 seconds.

Table 3 shows the experimental results. As with the SID evaluation setting, our proposed t-vector sequence classification model outperformed the cosine-similarity-based classification method for both segmentation schemes. With the VAD-based segmentations, the proposed model achieved relative cpWER improvements of 8.8 % and 5.6 % for IHM-mix-eval and SDM-eval, respectively. It is worth noting that using the VAD-based segmentations consistently resulted in lower cpWERs when the proposed model was used for the t-vector classification. Considering that VAD tended to produce longer segments than the utterance-group segments which had an average duration of 4 seconds, this may be an indication that the proposed model leveraged the longer contexts more efficiently.

5. Conclusion

We addressed the problem of speaker diarization for ASR output tokens. By using a t-vector, which is a token-level speaker representation, we described a sequence classification model that determines speaker labels for the t-vectors utilizing the entire t-vector sequence and a set of speaker profiles as input. Experimental results using the AMI meeting corpus showed cpWER improvements in both the SID and SD settings, compared with token-by-token classification using cosine similarities .

6. References

[1] T. Yoshioka, I. Abramovski, C. Aksoylar *et al.*, "Advances in online audio-visual meeting transcription," *Proc. ASRU*, pp. 276–283, 2019.

[2] S. Watanabe, M. Mandel, J. Barker *et al.*, "CHiME-6 challenge: Tackling multispeaker speech recognition for unsegmented recordings," *CHiME 2020-6th International Workshop on Speech Processing in Everyday Environments*, 2020.

[3] F. Yu, S. Zhang, Y. Fu *et al.*, "M2met: The icassp 2022 multi-channel multi-party meeting transcription challenge," *Proc. ICASSP*, pp. 6167–6171, 2022.

[4] T. J. Park, N. Kanda, D. Dimitriadis *et al.*, "A review of speaker diarization: Recent advances with deep learning," *Computer Speech & Language*, vol. 72, p. 101317, 2022.

[5] I. Medennikov, M. Korenevsky *et al.*, "The STC system for the CHiME-6 challenge," *CHiME 2020 Workshop on Speech Processing in Everyday Environments*, 2020.

[6] D. Raj, P. Denisov, Z. Chen *et al.*, "Integration of speech separation, diarization, and recognition for multi-speaker meetings: System description, comparison, and analysis," *Proc. SLT*, pp. 897–904, 2021.

[7] Y. Fujita, N. Kanda, S. Horiguchi *et al.*, "End-to-end neural speaker diarization with permutation-free objectives," *Proc. Interspeech*, pp. 4300–4304, 2019.

[8] I. Medennikov, M. Korenevsky, T. Prisyach *et al.*, "Target-speaker voice activity detection: A novel approach for multi-speaker diarization in a dinner party scenario," *Proc. Interspeech*, pp. 274–278, 2020.

[9] J. Huang, E. Marcheret, K. Visweswariah, and G. Potamianos, "The IBM RT07 evaluation systems for speaker diarization on lecture meetings," *Multimodal Technologies for Perception of Humans*, pp. 497–508, 2007.

[10] A. Khare, E. Han, Y. Yang *et al.*, "ASR-aware end-to-end neural diarization," *Proc. ICASSP*, pp. 8092–8096, 2022.

[11] N. Kanda, X. Xiao, Y. Gaur *et al.*, "Transcribe-to-diarize: Neural speaker diarization for unlimited number of speakers using end-to-end speaker-attributed asr," *Proc. ICASSP*, pp. 8082–8086, 2022.

[12] W. Xia, H. Lu, Q. Wang *et al.*, "Turn-to-diarize: Online speaker diarization constrained by transformer transducer speaker turn detection," *Proc. ICASSP*, pp. 8077–8081, 2022.

[13] X. Chang, W. Zhang, Y. Qian *et al.*, "MIMO-SPEECH: End-to-end multi-channel multi-speaker speech recognition," *ASRU*, pp. 237–244, 2019.

[14] A. Tripathi, H. Lu, and H. Sak, "End-to-end multi-talker overlapping speech recognition," *Proc. ICASSP*, pp. 6129–6133, 2020.

[15] N. Kanda, Y. Gaur, X. Wang *et al.*, "Serialized output training for end-to-end overlapped speech recognition," *Proc. Interspeech*, pp. 2797–2801, 2020.

[16] N. Kanda, J. Wu, Y. Wu *et al.*, "Streaming multi-talker ASR with token-level serialized output training," *Proc. Interspeech*, pp. 3774–3778, 2022.

[17] D. Dimitriadis and P. Fousek, "Developing on-line speaker diarization system," *Proc. Interspeech*, pp. 2739–2743, 2017.

[18] N. Kanda, Y. Gaur, X. Wang *et al.*, "Joint speaker counting, speech recognition, and speaker identification for overlapped speech of any number of speakers," *Proc. Interspeech*, pp. 36–40, 2020.

[19] N. Kanda, J. Wu, Y. Wu *et al.*, "Streaming speaker-attributed ASR with token-level speaker embeddings," *Proc. Interspeech*, pp. 521–525, 2022.

[20] Q. Zhang, H. Lu, H. Sak *et al.*, "Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss," *Proc. ICASSP*, pp. 7829–7833, 2020.

[21] D. Wang, X. Xiao, N. Kanda *et al.*, "Target speaker voice activity detection with transformers and its integration with end-to-end neural diarization," *arXiv preprint arXiv:2208.13085*, 2022.

[22] J. Carletta, S. Ashby, S. Bourban *et al.*, "The AMI meeting corpus: A pre-announcement," *Machine Learning for Multimodal Interaction: Second International Workshop, MLMI 2005, Edinburgh, UK, July 11-13, 2005, Revised Selected Papers 2*, pp. 28–39, 2006.

[23] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," *Proc. ICASSP*, pp. 4052–4056, 2014.

[24] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," *Proc. ICASSP*, pp. 5329–5333, 2018.

[25] Z. Chen, T. Yoshioka, L. Lu *et al.*, "Continuous speech separation: Dataset and analysis," *Proc. ICASSP*, pp. 7284–7288, 2020.

[26] S.-H. Gao, M.-M. Cheng, K. Zhao *et al.*, "Res2Net: A new multi-scale backbone architecture," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 2, pp. 652–662, 2019.

[27] X. Xiao, N. Kanda, Z. Chen *et al.*, "Microsoft speaker diarization system for the voxceleb speaker recognition challenge 2020," *Proc. ICASSP*, pp. 5824–5828, 2021.

[28] A. Nagrani, J. S. Chung, and A. Zisserman, "VoxCeleb: a large-scale speaker identification dataset," *Telephony*, vol. 3, pp. 33–039, 2017.

[29] J. Chung, A. Nagrani, and A. Zisserman, "VoxCeleb2: Deep speaker recognition," *Proc. Interspeech*, 2018.

[30] V. Panayotov, G. Chen, D. Povey *et al.*, "LibriSpeech: an asr corpus based on public domain audio books," *Proc. ICASSP*, pp. 5206–5210, 2015.

[31] N. Kanda, G. Ye, Y. Wu, Y. Gaur, X. Wang, Z. Meng, Z. Chen, and T. Yoshioka, "Large-Scale Pre-Training of End-to-End Multi-Talker ASR for Meeting Transcription with Single Distant Microphone," in *Proc. Interspeech 2021*, 2021, pp. 3430–3434.

[32] N. Kanda, G. Ye, Y. Wu *et al.*, "Large-scale pre-training of end-to-end multi-talker ASR for meeting transcription with single distant microphone," *Proc. Interspeech*, pp. 3430–3434, 2021.

[33] "py-webrtcvad," https://github.com/wiseman/py-webrtcvad, accessed: 03-05-2023.

[34] T. J. Park, K. J. Han, M. Kumar *et al.*, "Auto-tuning spectral clustering for speaker diarization using normalized maximum eigengap," *IEEE Signal Processing Letters*, vol. 27, pp. 381–385, 2019.