A Study on Speaker Diarization based on End-to-end Optimization

エンドツーエンド最適化に基づく話者ダイアライゼーションに関する研究

February, 2024

Yusuke FUJITA
藤田　雄介

A Study on Speaker Diarization based on End-to-end Optimization

エンドツーエンド最適化に基づく話者ダイアライゼーションに関する研究

February, 2024

Waseda University Graduate School of Fundamental Science and Engineering

Department of Computer Science and Communications Engineering, Research on Perceptual Computing

Yusuke FUJITA
藤田　雄介

A Study on Speaker Diarization based on End-to-end Optimization

by

Yusuke Fujita

Submitted to the Department of Computer Science and Communications Engineering
in February 2024, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

This dissertation addresses speaker diarization, the task of partitioning a multi-talker audio recording into speaker-wise segments. Speaker diarization has long been studied as a family of speaker identification problems, where multiple speakers in input audio should be separately identified without the speaker enrollment process. Therefore, most systems have used a speaker identification model as a central module in their pipeline structure. Traditional speaker diarization systems, as represented by x-vector clustering, divide the problem into at least three subproblems: speech activity detection, speaker embedding extraction from the speaker identification model, and clustering. These three subproblems are independently solved using different objectives: speech/non-speech classification accuracy, speaker identification accuracy, and cluster purity, respectively. These objectives are not directly connected to the minimization of diarization errors. In addition, the traditional systems need additional components to deal with overlapping speech. Recent studies have explored objectives that directly minimize diarization errors with "fully-supervised" models. The fully-supervised models use speaker labels in multi-talker audio as training data, minimizing diarization errors. However, such recent models still use a pipeline structure of multiple independently optimized modules. In Chapter 1, we review the history of speaker diarization research and introduce typical solutions and their limitations. The limitations suggest our main research objective: "end-to-end optimization".

Chapter 2 proposes a new formulation of the speaker diarization problem that no longer divides the problem into subproblems. In the new formulation, speaker diarization is a "multi-sequence" estimation composed of multi-speaker speech activity detectors, whereas the traditional systems

define the problem as a “single-sequence” speaker index estimation. The proposed formulation enables us to build an end-to-end optimization model that generates full speaker labels that include speech activity detection, speaker identification, and overlapping speech detection simultaneously. The proposed formulation with the end-to-end optimization is referred to as EEND: end-to-end neural diarization. To optimize the EEND model with the multi-sequence estimation target, we propose a permutation-free objective. We also propose a mixture simulation algorithm to produce sufficient training data to optimize the EEND model. We utilize bidirectional long short-term memory (BLSTM) for the neural network architecture to transform the input audio sequence into the estimation target. Our experiments with simulated data show that the proposed permutation-free objective is an essential component to realize the EEND model training. Moreover, instead of BLSTM, we employ the self-attention mechanism. The self-attentive EEND (SA-EEND) model demonstrates significant performance improvement over the BLSTM-based model and traditional x-vector clustering on the experiments with the CALLHOME two-speaker dataset. We visualize the behavior of the self-attention mechanism in SA-EEND. The visualization indicates that the multi-head self-attention mechanism captures global speaker characteristics distributed to the whole sequence, and the captured speaker information is encoded into frame-level embeddings according to the speaker’s presence per frame.

Though the EEND models have shown superior performance compared to the traditional systems, they still have some limitations. The models generally require large-scale training data that fits the target domain. The EEND model cannot deal with additional prior knowledge other than training data to help diarization performance. The root cause of the limitations is the conditional independence between speaker labels, which blocks the utilization of given speaker labels as context information. Chapters 3 and 4 provide our conditioning approaches to inducing speaker labels as context information for the EEND models.

Chapter 3 introduces a new conditioning method that utilize partial dependency in the speaker labels. The proposed method, dubbed “speaker-wise chain rule”, decomposes the target speaker label into speaker-wise sequences and forms a chain of speaker-wise estimators. By generating a speaker label sequence iteratively with conditioning input of previously estimated sequences, the proposed model can utilize the existence of other speakers as prior knowledge for improved diarization performance. Besides the performance improvement, the speaker-wise chain network can handle a variable number of speakers, by learning when to stop the chain. Experimental re-

sults on CALLHOME with two speakers showed that the speaker-wise chain rule outperformed the baseline EEND model. Furthermore, the experiments with the variable number of speakers demonstrated better speaker counting accuracy than the x-vector clustering system. This chapter also experimented with another conditioning method based on SAD and overlapping speech detection subtasks. We extended the speaker-wise chain rule to accept pre-conditioning input from the different tasks. The experiments showed that the subtask-first model improves the performance of the speaker-wise chain rule.

Chapter 4 presents another conditioning scheme that utilizes "self-conditioning via intermediate predictions." In the proposed self-conditioning model, the speaker labels produced in the middle of the neural network are fed back to the higher-layer network. The proposed method achieves iterative refinement of speaker labels through multiple intermediate predictions layer by layer. Experiments show that EEND models are improved with the proposed methods while keeping the amount of training data the same. To compare with the state-of-the-art method, we implemented self-conditioning on the encoder-decoder-based attractor (EDA) model. However, we found the bottleneck of EDA when used with self-conditioning. Therefore, we proposed the non-autoregressive attractor as a variant of EDA, which replaces the autoregressive computation part in EDA with the non-autoregressive attention-based module. Experiments showed that the proposed method improves both performance and training efficiency compared with the original EDA. The obtained DER is comparable with existing state-of-the-art WavLM models, which use self-supervised pretraining with large-scale training data and far more parameters.

We summarize the contributions of the dissertation in Chapter 5. The introduction of EEND has revolutionized speaker diarization research, marking a significant shift in research focus. Many papers are now engaged in extending the EEND models. We explore the future direction by presenting recent studies built on the EEND concept.

Thesis Committee

Tetsunori Kobayashi	Professor, Faculty of Science and Engineering, Waseda University
Tetsuji Ogawa	Professor, Faculty of Science and Engineering, Waseda University
Daisuke Kawahara	Professor, Faculty of Science and Engineering, Waseda University
Shinji Watanabe	Associate Professor, Carnegie Melon University

Acknowledgments

I would like to express my gratitude to all those who supported me during my research work.

First, I am extremely grateful to my advisor, Prof. Tetsunori Kobayashi, at Waseda University. He always guided me on how to pursue a Ph.D. while working in industry. In collaborative research with LINE Corporation, he and I discussed automatic speech recognition a lot. From the discussion, I learned his immense experience in speech technology, which was valuable in forming the main story of the dissertation.

I am also grateful to my co-advisor, Prof. Tetsuji Ogawa, at Waseda University. He gave me a lot of technical advice in writing conference and journal papers. He also gave me the opportunity to discuss various research topics with his students. Without his support, I could not have built my comfortable research environment at Waseda.

I would like to express my gratitude to Prof. Shinji Watanabe at Carnegie Mellon University. He encouraged me to pursue the Ph.D. and introduced me to Prof. Kobayashi. In our joint research while at Johns Hopkins University and Hitachi, he and I developed EEND, which is obviously the core of the dissertation. During my visit to Hopkins, I learned quite a lot about academic research and how to collaborate with other researchers.

I would like to thank my colleagues for their help while I was at Hitachi, Ltd. I learned fundamental skills to be a professional industry researcher from Prof. Yoshinori Kitahara (currently Emeritus Professor at Tokyo University of Agriculture and Technology). Discussions with Dr. Naoyuki Kanda (currently at Microsoft) were always fun and fruitful. His professionalism always guided our work in the right direction. Dr. Shota Horiguchi supported me in enhancing our EEND work. The dissertation is based on his continual contributions.

I would like to thank my colleagues for their support while I was at LINE Corporation (currently LY Corporation). Dr. Masahito Togami (currently at Amazon Web Services) and Mr. Yusuke Kida (currently at LINE WORKS) kindly accepted and supported my journey to the Ph.D. course while working at LINE. Mr. Tatsuya Komatsu introduced his idea of self-conditioning for automatic speech recognition, resulting in our latest work.

I would like to express my gratitude to the last Prof. Naohisa Komatsu at Waseda University. He invited me to the research field on speech communication. During my undergraduate and master’s courses at Waseda, I enjoyed exploring biometrics, including speaker recognition, with his advice. The fun experience in Komatsu Laboratory brought me on the journey to becoming a speech researcher.

Finally, I would like to thank my wife, Nozomi, and my daughters, Miho and Mayu, for accompanying me everywhere.

Contents

List of Figures

List of Tables

1
Introduction

1.1 Background of speaker diarization

Speech processing plays a crucial role in modern information systems that support our daily lives. It is essential for natural interaction between humans and computer systems. Automatic speech recognition (ASR) enables voice-enabled systems to understand the user's voice content. Speaker identification offers security measures to verify the user who attempts to control the systems. Speech synthesis enables more intuitive interfaces than text-based responses.

Besides human-to-computer interaction, speech processing offers essential tools for understanding human-to-human spoken communication. ASR offers automated subtitle generation to broadcast news and movies containing spoken communication. Call center systems analyze customer-operator interactions by transcribing calls and classifying sentiments and emotions from the voice using various speech processing technologies to continually improve operational efficiency and track customer satisfaction or frustration. Speech translation supports effective communication between humans who speak different languages, lowering language barriers. In general, speech processing of human-to-human communication is relatively hard due to the spontaneous spoken style and multi-talker situation compared to human-to-computer interaction.

Speaker diarization emerges as a particularly important technology for understanding human-to-human communication. Speaker diarization is defined as the task of partitioning a multi-talker audio recording into homogeneous speech segments according to speaker identity; in other words, speaker diarization answers the question of "who spoke when." Here is a short list of the applications that use speaker diarization:

1. **Meeting note**: Speaker diarization helps in accurately transcribing the spoken content by identifying who said it. The speaker's information attributed to a transcribed sentence is vital for future reference of the meeting note.

2. **Legal enforcement**: Some legal statements require correct attribution to the right speaker. Court recordings should be analyzed with speaker diarization to ensure their correctness.

3. **Healthcare insights and productivity improvement**: Doctor-patient interactions require speaker diarization, providing insights into communication patterns. Not only healthcare applications but analyzing communication patterns of business meetings is also needed to improve productivity.

4. **Customer service**: In call centers, distinguishing the customer and the operator is essential for analyzing interaction. The analyzed calls are used for operator training purposes and quantifying customer satisfaction.

In summary, speaker diarization provides speaker labels in the audio recording, which helps understand human-to-human communication by showing the rich transcription along with speaker attribution or analyzing the speaker transition patterns to obtain insights.

Speaker diarization is also used as a preprocessing step for other speech processing tasks. In many applications, speaker diarization is combined with ASR. Since the ASR system is typically tuned with single-speaker audio segments, it fails when just feeding multi-speaker audio into the system. Therefore, speaker diarization is used as a preprocessor for ASR, providing single-speaker speech segments. Numerous domains require speaker diarization as a preprocessor, for example, telephone conversations, broadcast news, interviews, meeting conversations, and web videos.

1.2 Evolution of speaker diarization research

In this section, we provide a brief overview of the evolution of speaker diarization research to explain the background leading to our proposed method: end-to-end neural diarization (EEND).

Speaker diarization can be considered as speaker verification between different speech segments in an audio stream. The history of speaker diarization research has primarily been aligned with the one of speaker verification. In the 2000s, feature representations for speaker verification were based on the Gaussian mixture model and universal background model (GMM/UBM; Reynolds et al. (2000)). In the 2010s, feature transformation methods, like i-vector (Dehak et al. (2011)) and probabilistic linear discriminant analysis (PLDA; Garcia-Romero and Espy-Wilson (2011)), successfully suppressed intra-speaker variability to attain speaker verification accuracy under various environments. Then, deep neural networks have been exploited to replace traditional feature representations with discriminatively optimized speaker embeddings, such as d-vector (Variani et al. (2014)) and x-vector (Snyder et al. (2018)). The improvement in accuracy through the x-vector had a significant impact on the research community, and since then, the enhancement in speaker diarization accuracy has been equated with that in the speaker embedding models.

Clustering has been a key technique to transform the speaker verification task into the speaker diarization task. Similar feature representations in different speech segments are marked as the same speaker using clustering. Agglomerative hierarchical clustering (Chen et al. (1998)) and Spectral clustering (Ning et al. (2006)) have long been used for speaker diarization. In contrast to speaker embeddings, clustering did not undergo significant evolution. Since clustering performs unsupervised optimization, the traditional systems based on clustering cannot be optimized with ground-truth speaker diarization information.

EEND's innovation

Our work (EEND; Fujita et al. (2019a)) arrived at the optimization of diarization itself without following the historical path of speaker verification. We connected the problems of source separation and speaker diarization. We found that mask estimation in source separation (Yu et al. (2017); Hershey et al. (2016)) is equivalent to the speaker diarization task, except for considering the frequency axis. Consequently, speaker diarization transcended the conventional "speaker verification per segment" framework and was redefined as a "multi-speaker speech activity labeling".

Furthermore, following our work, the evaluation metrics of speaker diarization have shifted from the speaker assignment error rates "only in non-overlapping speech segments" to the error rates across all segments, including "non-speech and overlapping speech". This shift has significantly changed the research focus from unrealistic non-overlapping scenarios to realistic overlapping scenarios.

1.3 Review of speaker diarization methods

Comprehensive reviews of speaker diarization methods are presented in Tranter and Reynolds (2006), Anguera et al. (2012), and Park et al. (2022). They all mentioned that, in the 2000s, the Rich Transcription evaluations (NIST RT; National Institute of Standard and Technology (NIST) (2009)) fostered diarization researchers to use standard evaluation protocols and databases to compare different approaches meaningfully. In this section, we introduce the evaluation protocols and databases that our experiments follow. Then, we introduce the traditional and recent systems to provide the prerequisites for the proposed methods in this dissertation.

1.3.1 Evaluation protocols

Speaker diarization systems are required to output a hypothesis of speaker activity. The speaker activity is represented as a sequence of speech segments, including start and end times with speaker labels. The speaker labels only distinguish different speakers in an audio recording, so they do not need to identify the real names. The hypothesis is compared with a ground-truth reference to obtain a diarization error rate (DER). Diarization errors are categorized into miss, false alarm, and

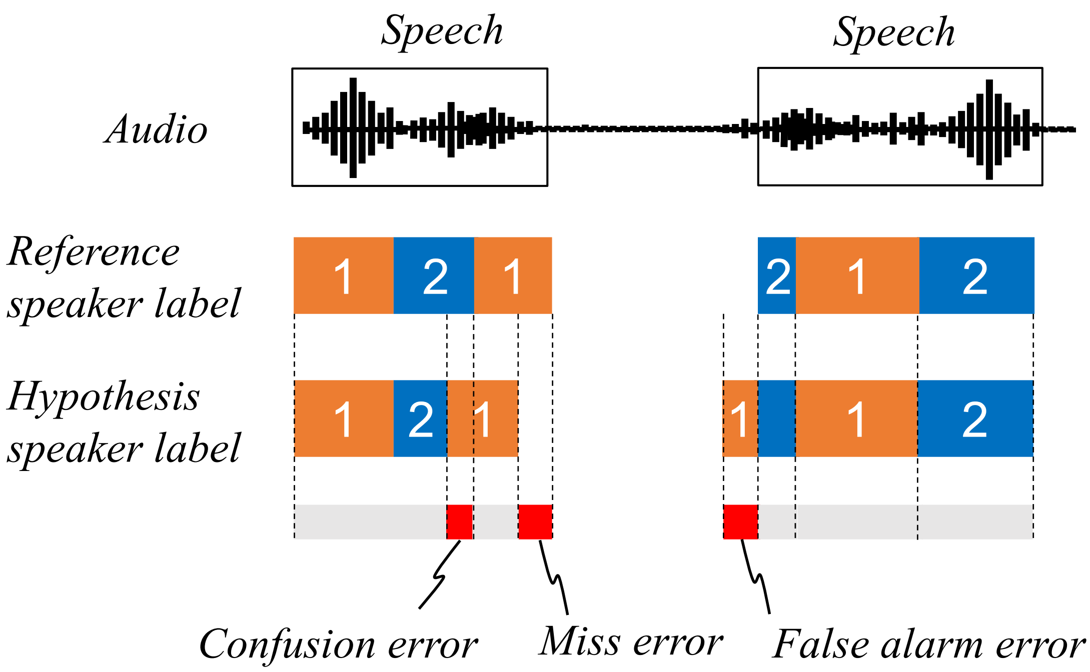


Figure 1-1: Diarization errors.

confusion errors, as shown in Fig. 1-1. Miss errors are speech time in the reference, not in the hypothesis. False alarm errors are speech time in the hypothesis, not in the reference. Confusion errors are speech time when the speaker labels do not match. In addition, overlap errors are also accumulated where the number of speakers in the hypothesis differs from that in the reference. Miss errors are added if the number of speakers in the reference is larger. False alarm errors are added if the number of speakers in the hypothesis is larger. DER is calculated as a ratio of the sum of the three errors to the total speech time. The collar tolerance, generally 250 msec, is set around the reference target boundary to ignore the errors caused by inconsistency in human annotations. NIST defined the diarization output format named RTTM (Rich Transcription Time Marked) and provided a tool ‘md-eval.pl’ to calculate DERs.

Other metrics have also been used in the literature. Word-level diarization error rate (WDER) is a practical metric when combined with ASR (Park and Georgiou (2018)). Jaccard error rate (JER) is a recently introduced metric to evaluate per-speaker error rates (Ryant et al. (2019)). However, we always use DERs throughout this dissertation to compare our experimental results consistently with other literature.

It is important to note that DERs reported in the earlier studies than our EEND paper may not fully represent the diarization performance. This is because these studies often used oracle speech/non-speech labels, which led to the exclusion of misses or false alarm errors in their evaluations. Overlapping speech segments were typically excluded from their evaluations, further impacting the comprehensiveness of the reported DERs. Contrary to the earlier studies, we evaluate all the errors, including overlapping speech segments, because the EEND includes both speech activity detection and overlapping speech detection functionality.

1.3.2 Databases

Prior to the NIST RT, speaker diarization was evaluated through the NIST Speaker Recognition Evaluation (NIST SRE; Doddington et al. (2000)). The NIST SRE provided the evaluation set of telephone conversations (hereinafter called CALLHOME). Our experiments primarily use this evaluation dataset because we can consistently compare our methods with a bunch of highly influenced papers.

Besides telephone conversations, NIST RT focused on broadcast news and meeting conversations. Meeting datasets such as ICSI (Janin et al. (2003); Çetin and Shriberg (2006)) and AMI (Renals et al. (2008)) have also fostered speaker diarization research. Since the late 2010s, researchers have started to tackle various audio domains, such as web videos, dinner parties, and conversations in restaurants. VoxConverse (Chung et al. (2020)), CHiME-6 (Watanabe et al. (2020)), and DIHARD Challenges (Sell et al. (2018); Ryant et al. (2019, 2021)) are popular challenge tasks targeting such various domains. Recent tasks are considering more challenging and rich inputs, such as far-field audio (M2MeT; Yu et al. (2022)), and audio-visual input (MISP; Wang et al. (2023)).

1.3.3 Traditional systems: Speaker embedding clustering

A traditional system of speaker diarization is a pipeline of multiple modules. Fig. 1-2 depicts the pipeline referred to as speaker embedding clustering. The main modules are speech activity detection, speaker embedding extraction, and clustering.

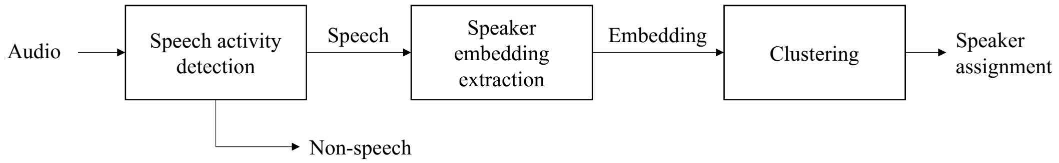


Figure 1-2: Schematic diagram of traditional speaker diarization pipeline.

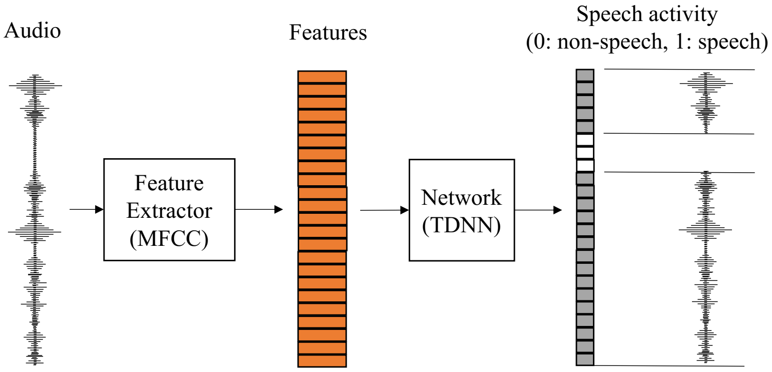
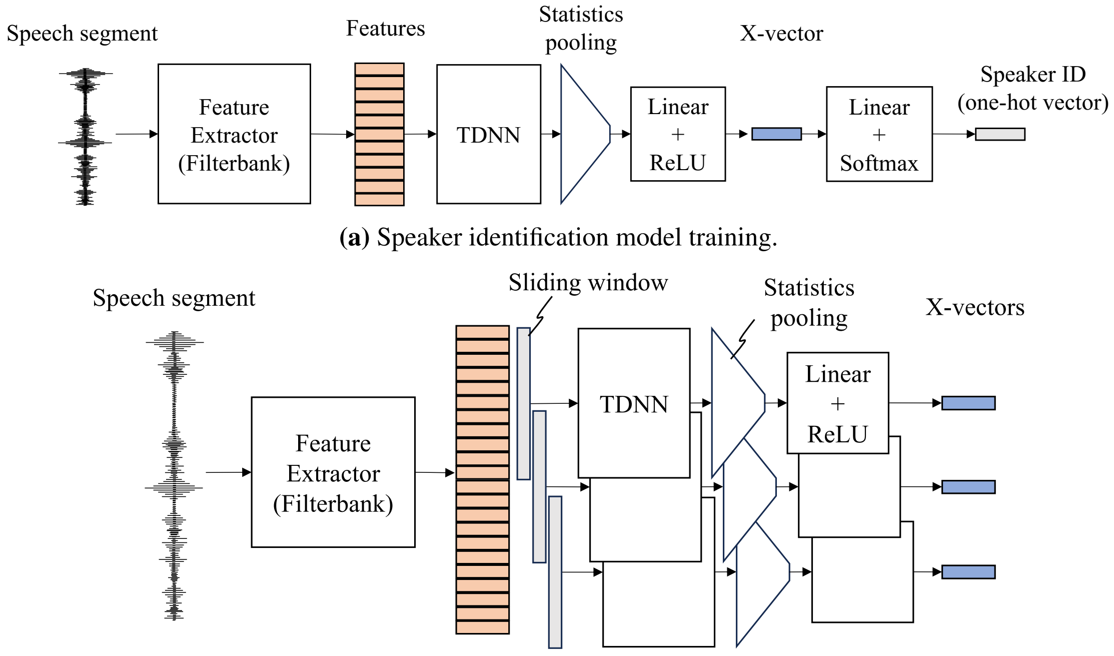


Figure 1-3: Typical DNN-based speech activity detection module.

Speech activity detection

Speech activity detection (SAD) determines speech/non-speech boundaries and selects speech segments for further processing. A simple approach to SAD is to measure signal power and threshold, though it degrades performance in noisy environments. Recent systems typically use a deep neural network (DNN) to determine frame-level speech activity. Fig 1-3 shows a typical SAD module based on a DNN. In the module, audio is transformed into a sequence of features, such as Mel-frequence cepstral coefficients (MFCCs). Then, the time-delay-neural network (TDNN; Peddinti et al. (2015)) receives the sequence of features. For seeing long-term context, the statistics pooling layers (Ghahremani et al. (2016)) are also employed. The network is trained with a cross-entropy objective to estimate a binary label, indicating the frame-level speech activity. The non-speech segments are filtered out of the pipeline, and the speech segments go to the next speaker embedding extraction module.



(a) Speaker identification model training.

(b) Speaker embedding extraction

Figure 1-4: Speaker embedding training and extraction.

Speaker embedding extraction

Speech segments are transformed into speaker embeddings representing speaker identity for each time step. The speaker embedding extractor is fundamentally a speaker identification model. X-vectors (Snyder et al. (2018)) and d-vectors (Wan et al. (2018)) are popular speaker identification models used in speaker diarization. Fig. 1-4a shows a structure of the speaker identification model. A single-speaker speech segment is fed into the feature extractor (filterbank) and TDNN. The statistics pooling layer reduces the time axis, and a linear layer and a rectified linear unit (ReLU) produce an embedding called an x-vector. In the training phase, the x-vector is transformed into speaker ID. This model is trained with a cross-entropy objective using a large collection of single-speaker speech segments with thousands of speakers. In the speaker diarization pipeline, the trained speaker identification model is utilized to extract speaker embeddings with sliding windows, as shown in Fig. 1-4b.

Clustering with similarity score

The clustering module makes clusters of speaker embeddings according to their similarity. Even though cosine similarity can be used as the simplest similarity metric, traditional speaker diarization systems often use a similarity metric in a dedicated subspace. Gaussian probabilistic linear discriminant analysis (G-PLDA; Prince and Elder (2007)) is a popular tool to score the similarity of two speaker embeddings in the subspace where the embeddings from the same speaker give a positive score and the ones from the different speakers give a negative score. With a modification by Garcia-Romero and Espy-Wilson (2011), it assumes a generative model of speaker embedding as:

$$\boldsymbol{e} = \boldsymbol{\mu} + \boldsymbol{F}\boldsymbol{h} + \boldsymbol{n} \in \mathbb{R}^D, \tag{1.1}$$

where $\boldsymbol{e}$ is a D-dimensional speaker embedding, $\boldsymbol{\mu}$ is a global mean of speaker embeddings, $\boldsymbol{F} \in \mathbb{R}^{D\times d}$ is a matrix composed of bases of the d-dimensional subspace, $\boldsymbol{h}$ is a latent vector in the subspace sampled from the standard normal distribution, and $\boldsymbol{n}$ is a Gaussian noise with zero mean and diagonal covariance $\boldsymbol{\Sigma}$. Maximum-likelihood estimates of the parameter set $\{\boldsymbol{\mu}, \boldsymbol{F}, \boldsymbol{\Sigma}\}$ are obtained from a large-scale dataset using the expectation-maximization algorithm. Given two speaker embeddings, $\boldsymbol{e}_1, \boldsymbol{e}_2$, the similarity score (hereinafter called PLDA score) is calculated as the log-likelihood ratio of the same speaker hypothesis $\mathcal{H}_{\mathsf{s}}$ to the different speaker hypothesis $\mathcal{H}_{\mathsf{d}}$:

$$\mathsf{score}(\boldsymbol{e}_1, \boldsymbol{e}_2) = \log \frac{p(\boldsymbol{e}_1, \boldsymbol{e}_2 \mid \mathcal{H}_{\mathsf{s}})}{p(\boldsymbol{e}_1, \boldsymbol{e}_2 \mid \mathcal{H}_{\mathsf{d}})}. \tag{1.2}$$

Thanks to the properties of Gaussian distributions, the PLDA score is rewritten as:

$$\begin{aligned} \mathsf{score}(\boldsymbol{e}_1, \boldsymbol{e}_2) &= \log \mathcal{N}\left(\begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{bmatrix}; \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu} \end{bmatrix}, \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^\top + \boldsymbol{\Sigma} & \boldsymbol{F}\boldsymbol{F}^\top \\ \boldsymbol{F}\boldsymbol{F}^\top & \boldsymbol{F}\boldsymbol{F}^\top \end{bmatrix} \right) && (1.3) \\ &\quad - \log \mathcal{N}\left(\begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{bmatrix}; \begin{bmatrix} \boldsymbol{\mu} \\ \boldsymbol{\mu} \end{bmatrix}, \begin{bmatrix} \boldsymbol{F}\boldsymbol{F}^\top + \boldsymbol{\Sigma} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{F}\boldsymbol{F}^\top \end{bmatrix} \right). && (1.4) \end{aligned}$$

A popular clustering algorithm used in speaker diarization is agglomerative hierarchical clustering (AHC) (Meignier (2010); Sell and Garcia-Romero (2014); Garcia-Romero et al. (2017);

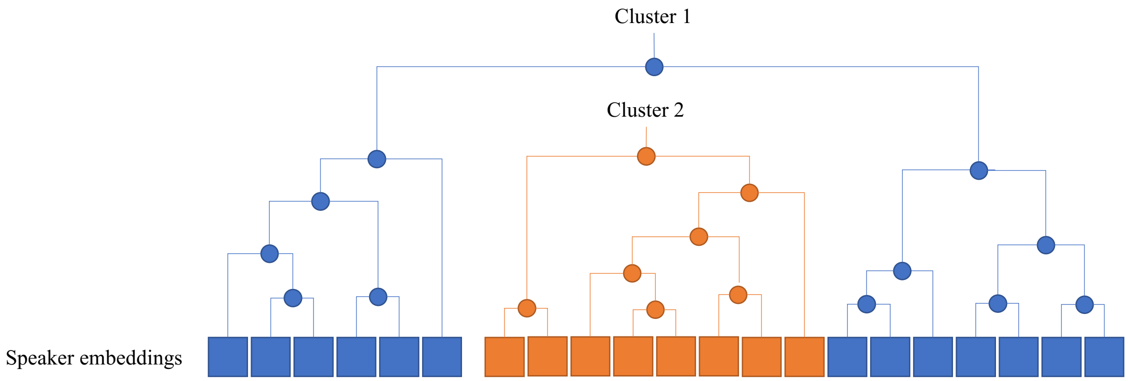


Figure 1-5: Agglomerative hierarchical clustering.

Maciejewski et al. (2018)). AHC initializes clusters with one speaker embedding for each. The AHC process iteratively merges a pair of clusters with the highest PLDA score until the PLDA score meets a stopping threshold. The PLDA score of a pair of *clusters* is an average of the PLDA score between elements of each cluster. Resulting clusters form a dendrogram, as shown in Fig. 1-5. The stopping threshold controls the number of speakers (clusters). According to Eq. 1.2, the threshold should be 0. However, it is determined by sweeping the candidate thresholds using the development data in practice. If the number of speakers is given in advance, AHC stops when the number of clusters meets the condition.

There are many alternative clustering methods to AHC, such as Gaussian mixture models (Meignier (2010); Shum et al. (2013)), mean shift clustering (Senoussaoui et al. (2014)), k-means clustering (Dimitriadis and Fousek (2017); Wang et al. (2018)), Links (Mansfield et al. (2018); Wang et al. (2018)), and spectral clustering (Wang et al. (2018)).

Resegmentation

The speaker assignment obtained from clustering generally ignores temporal characteristics, i.e., a speaker transition model. Landini et al. (2022b) proposed a postprocessing method to refine the clustering-based speaker assignment. Their Bayesian hidden Markov model (VB-HMM) generates a sequence of speaker embeddings considering constraints on speaker transition patterns. The VB-HMM is a factorized generative model of T-length D-dimensional speaker embeddings $\boldsymbol{E} \in \mathbb{R}^{D \times T}$ and T-length speaker index sequence $\boldsymbol{y} \in \mathbb{Z}_{>0}^{T}$. They introduce speaker-specific latent variable $\boldsymbol{H} \in \mathbb{R}^{d \times S}$, which is compatible with PLDA described in the previous subsection.

$$p(\boldsymbol{E}, \boldsymbol{y}, \boldsymbol{H}) = \prod_{t=1}^{T} \underbrace{p(\boldsymbol{E}_{:,t}|\boldsymbol{y}_t, \boldsymbol{H})}_{\text{embedding emission}} \underbrace{p(\boldsymbol{y}_t|\boldsymbol{y}_{t-1})}_{\text{transition}} \prod_{s=1}^{S} \underbrace{p(\boldsymbol{H}_{:,s})}_{\text{speaker-wise latent}} . \tag{1.5}$$

Here, they use an ergodic HMM with a one-to-one correspondence between the HMM states and the speakers to model transition probability $p(\boldsymbol{y}_t|\boldsymbol{y}_{t-1})$ and embedding emission probability $p(\boldsymbol{E}_{:,t}|\boldsymbol{y}_t, \boldsymbol{H})$. The emission probability follows a Gaussian distribution $\mathcal{N}(\boldsymbol{F}\boldsymbol{H}_{:,s}, \boldsymbol{I})$, where $\boldsymbol{F} \in \mathbb{R}^{D \times d}$ is pre-trained with PLDA. $\boldsymbol{H}_{:,s}$ follows a standard normal distribution. Variational Bayes (VB) inference gives the maximum posterior $p(\boldsymbol{y}|\boldsymbol{E})$ by iteratively updating the speaker label $\boldsymbol{y}$ and the speaker-specific latent $\boldsymbol{H}$, initialized with the speaker label from the clustering result.

Difficulties in developing the pipeline

The speaker embedding clustering pipeline has shown effectiveness on various datasets (e.g., Sell et al. (2018); Diez et al. (2018); Sun et al. (2018)). However, the traditional method faced troubles in the development phase.

Firstly, we should train three independent models to be optimized with different criteria: speech v.s. non-speech classification accuracy for the SAD model, speaker ID accuracy for the speaker embedding extractor, and same/different speaker classification accuracy for the PLDA scorer. None of these criteria directly minimizes diarization errors. In general, the speaker embedding extractor critically affects performance. However, Sell and Garcia-Romero (2014) has shown the importance of domain adaptation through calibration of the PLDA scorer. Moreover, the SAD model becomes critical when environmental noise is challenging. There are many best practices to develop the three modules.

Secondly, they have trouble handling overlapping speech. The clustering and resegmentation modules perform the hard assignment of a frame to only one speaker. Overlapping speech can be detected using a speech separation model and can heuristically assign the second speaker based on closeness in time (Landini et al. (2021)). However, the pipeline becomes more complicated.

1.3.4 Recent systems: Fully-supervised models

Since 2019, “fully-supervised” models have been investigated, alleviating the difficulties in optimizing traditional pipeline systems. The key functionality is to use “multi-talker labels in conversations” as training data, whereas the traditional models are generally trained using a collection of “single-speaker labels in segments”. Removing the clustering module, which is an unsupervised process that prohibits direct optimization with speaker diarization labels, is a core idea of the fully-supervised models.

UIS-RNN (Zhang et al. (2019))

The unbounded interleaved-state recurrent neural network (UIS-RNN; Zhang et al. (2019)) is the first fully-supervised model for speaker diarization. An RNN generates a speaker embedding sequence for each speaker. The authors formulated a generative model of speaker embeddings and speaker indices. T-length D-dimensional speaker embeddings $\boldsymbol{E} \in \mathbb{R}^{T\times D}$ and T-length speaker indices $\boldsymbol{y} \in \mathbb{Z}^{T}_{>0}$ are generated in an online manner:

$$p(\boldsymbol{E}, \boldsymbol{y}) = p(\boldsymbol{E}_{:,1}, \boldsymbol{y}_1) \prod_{t=2}^{T} \underbrace{p(\boldsymbol{E}_{:,t}|\boldsymbol{E}_{:,1:t-1}, \boldsymbol{y}_{:,:t})}_{\text{speaker embedding}} \underbrace{p(\boldsymbol{y}_t|\delta_t, \boldsymbol{y}_{1:t-1})}_{\text{speaker assignment}} \underbrace{p(\delta_t|\delta_{1:t-1})}_{\text{speaker change}}, \tag{1.6}$$

where $\delta_t = \mathbb{1}(\boldsymbol{y}_t \neq \boldsymbol{y}_{t-1}) \in \{0, 1\}$ is a speaker change indicator at time t. The first term $p(\boldsymbol{E}_{:,t}|\boldsymbol{E}_{:,1:t-1}, \boldsymbol{y}_{:,:t})$ is the autoregressive speaker embedding generation model, following Gaussian distribution: $\mathcal{N}(\boldsymbol{\mu}_t, \sigma^2\boldsymbol{I})$, where $\boldsymbol{\mu}_t$ is a running mean of the RNN’s output for the speaker index $\boldsymbol{y}_t$ until time t. The second term $p(\boldsymbol{y}_t|\delta_t, \boldsymbol{y}_{1:t-1})$ is the speaker assignment model, following the Chinese restaurant process, a Bayesian nonparametric model that assigns the probability of $\boldsymbol{y}_t$ proportional to the number of speaker turns. The third term is the speaker change prior set to be constant. This joint model is trained using speaker embedding sequences and reference speaker indices in multi-talker audio.

SSGD (von Neumann et al. (2019), Fang et al. (2021))

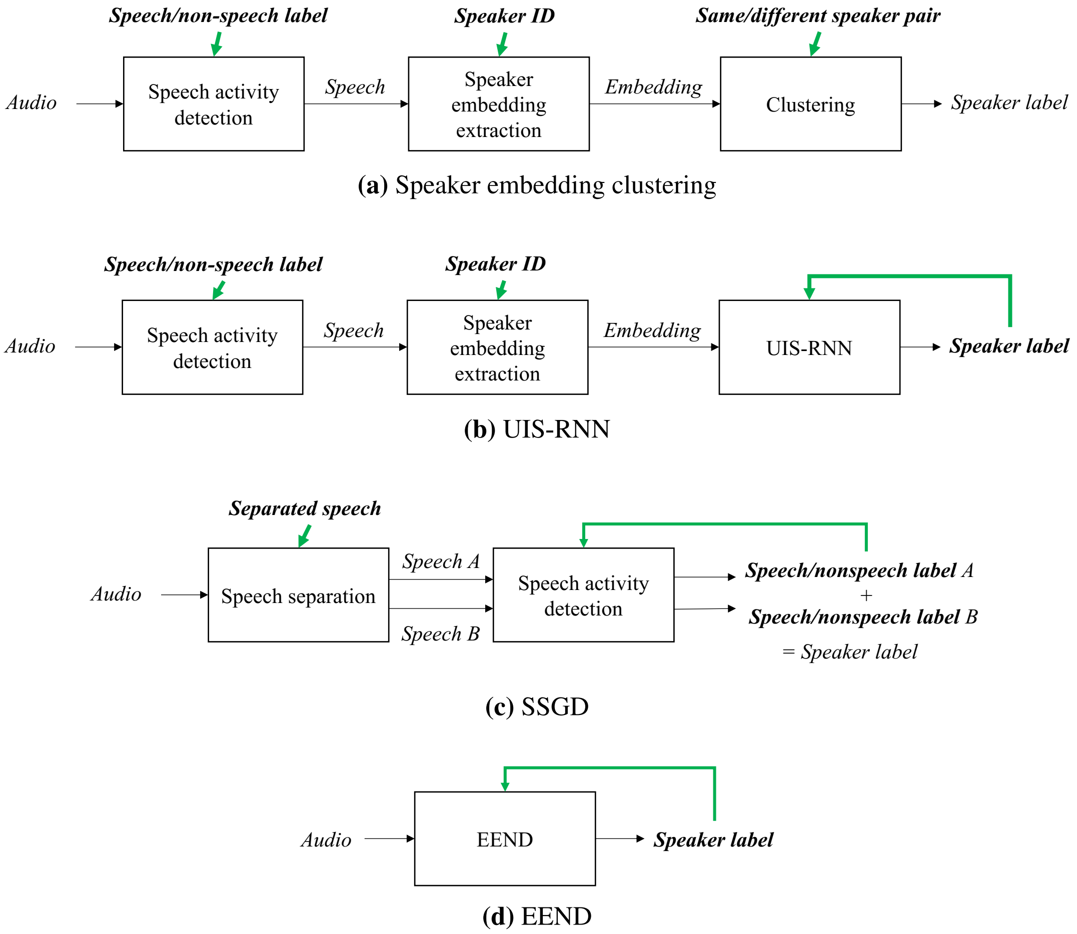
The use of speech separation to solve the diarization problem was investigated by von Neumann et al. (2019). They use a neural speech separation model that continually tracks the speakers in a block-online manner. Speech separation models have usually been trained with fully overlapped speech. Instead, the authors use simulated conversations containing silence, single-speaker, and overlapped segments in an audio stream. To track the separated speakers between blocks, the authors use a guidance vector extracted from the previous block to condition the separation network in the next block.

Fang et al. (2021) studied the complementary nature of speech separation and speaker diarization. They proposed a method called speech separation guided diarization (SSGD). SSGD prepares the traditional diarization pipeline and a conventional speech separation model trained with fully overlapped speech. Then, SSGD uses the traditional diarization pipeline to generate adaptation data for the speech separation model. The speech separation model is fine-tuned using the adaptation data, improving the separation performance on conversational data. Finally, separated speech is fed into the conventional SAD to obtain the speech activity of each speaker.

EEND (Our work; Fujita et al. (2019a,b))

Whereas UIS-RNN and SSGD still use module pipelines, end-to-end neural diarization (EEND; Fujita et al. (2019a)) uses a single module. A neural network receives audio features and outputs multi-speaker speech activity directly. The proposed method solves the speaker diarization problem with a joint multi-sequence classification model, while the earlier systems were composed of single-sequence generation models.

Fig 1-6 compares differences in supervision signals among speaker embedding clustering, UIS-RNN, SSGD, and EEND models. Speaker embedding clustering uses three independent supervision signals, and speaker labels are not used. UIS-RNN uses speaker labels as supervision signals, while it requires independent optimization of three modules. SSGD is a pipeline of two independent modules, and separated (clean) speech is required to train the speech separation model. EEND is solely optimized with speaker labels.



(a) Speaker embedding clustering

(b) UIS-RNN

(c) SSGD

(d) EEND

Figure 1-6: Differences in supervision signals among clustering, UIS-RNN, SSGD, and EEND. Green arrows indicate the supervision signals when used in the training phase.

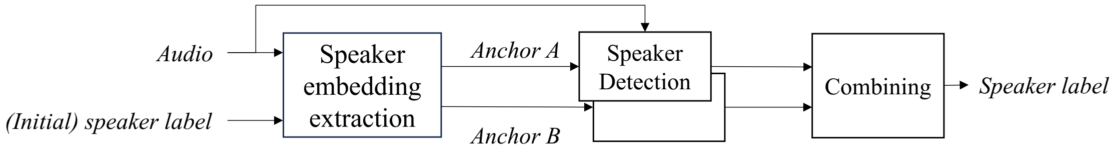


Figure 1-7: Overview of TS-VAD model architecture.

TS-VAD (Medennikov et al. (2020a,b))

Target-speaker voice activity detection (TS-VAD) integrates EEND and the "target-speaker" concepts. Target-speaker ASR (Žmolíková et al. (2017); Kanda et al. (2019)) and Speaker Beam (Delcroix et al. (2018)) introduce "anchor" speaker embeddings that help improve ASR and speech separation, respectively. TS-VAD prepares the anchor speaker embeddings using a traditional pipeline-based diarization system. The anchor speaker embedding is an averaged i-vector (Dehak et al. (2011)) per speaker calculated according to the initial diarization results. The anchor speaker embedding is fed together with audio features into the neural network, and the network outputs multi-speaker speech activity directly, similar to EEND. The neural network comprises a speaker detection network, which accepts anchor speaker embedding with audio features, and a combining network, which uses multiple inputs from the speaker detection network to produce the final diarization output. The TS-VAD architecture is depicted in Fig. 1-7. TS-VAD proved effective in various challenging tasks including CHiME-6 (Watanabe et al. (2020)) and M2MeT (Yu et al. (2022))).

EEND-VC (Kinoshita et al. (2021a,b))

EEND-vector clustering (EEND-VC) is a hybrid method of EEND and speaker embedding clustering. Fig. 1-8 shows the EEND-VC architecture. EEND-VC alleviates a main limitation of the EEND model: it's hard to accept audio containing many speakers. Assuming a short chunk has a limited number of speakers, EEND-VC solves the speaker assignment among chunks by using clustering. The EEND-VC's network generates both frame-wise speaker labels and speaker embeddings for an audio chunk. Speaker embeddings for each speaker are averaged in a chunk according to the estimated speaker labels. Then, the speaker embeddings in multiple chunks are clustered using a constrained clustering algorithm. EEND-VC is particularly effective when there

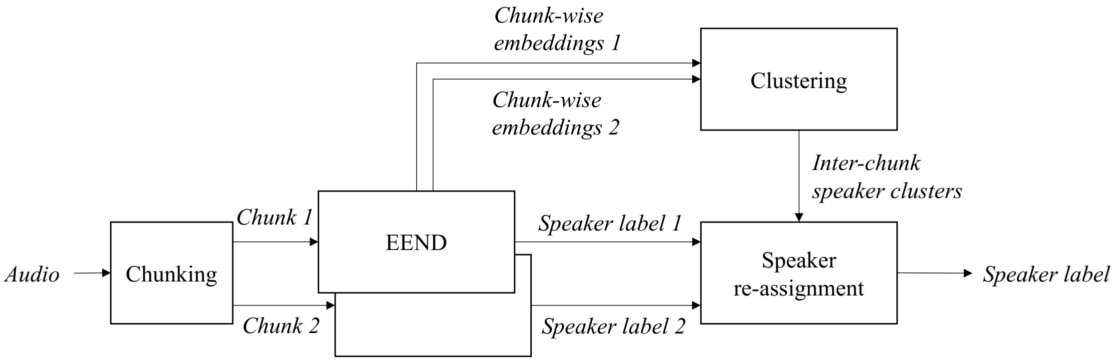


Figure 1-8: Overview of EEND-VC model architecture.

are more than three speakers.

1.3.5 Comparison on optimization targets of existing diarization systems

Each of the aforementioned speaker diarization systems optimizes for different targets. This section uses probabilistic model formulations to elucidate the optimization targets and compare their characteristics.

Speaker embedding clustering Given an input audio sequence $\boldsymbol{X} \in \mathbb{R}^{T'}$, the traditional system optimizes for the speaker index sequence $\boldsymbol{y} \in \mathbb{Z}_{\geq 0}^{T}$ using three independent models:

$$\hat{\boldsymbol{y}} = \arg\max_{y} p(\boldsymbol{y}|\boldsymbol{X}) \approx \arg\max_{y} p(\hat{\boldsymbol{E}}|\boldsymbol{y}, \hat{\boldsymbol{s}}) p(\boldsymbol{y}|\hat{\boldsymbol{s}}), \tag{1.7}$$

$$\hat{\boldsymbol{E}} = \arg\max_{\boldsymbol{E}} p(\boldsymbol{E}|\boldsymbol{X}, \hat{\boldsymbol{s}}), \tag{1.8}$$

$$\hat{\boldsymbol{s}} = \arg\max_{\boldsymbol{s}} p(\boldsymbol{s}|\boldsymbol{X}). \tag{1.9}$$

A SAD model $p(\boldsymbol{s}|\boldsymbol{X})$ determines speech activity $\hat{\boldsymbol{s}} \in \{0,1\}^{T'}$. $p(\boldsymbol{E}|\boldsymbol{X}, \hat{\boldsymbol{s}})$ is a speaker embedding model for speech segments specified by the SAD output $\hat{\boldsymbol{s}}$. Clustering of the speaker embeddings $\hat{\boldsymbol{E}}$ determines $\boldsymbol{y}$ that maximizes $p(\hat{\boldsymbol{E}}|\boldsymbol{y}, \hat{\boldsymbol{s}})$. The optimization target $\boldsymbol{y}$ is the speaker index sequence, so it does not consider overlapping speech.

VB-HMM and UIS-RNN These methods introduce an explicit temporal dependency to the sequence generation model $p(\hat{\boldsymbol{E}}|\boldsymbol{y}, \hat{\boldsymbol{s}})$ (Eq. 1.7):

$$p(\hat{\boldsymbol{E}}|\boldsymbol{y}, \hat{\boldsymbol{s}}) \approx \prod_{t=1}^{T} p(\hat{\boldsymbol{E}}_t|\hat{\boldsymbol{E}}_{1:t-1}, \boldsymbol{y}_{1:t})p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}). \tag{1.10}$$

As with speaker embedding clustering, we need three independently optimized models for UIS-RNN. The sequence generation model is initialized with the clustering output for VB-HMM, requiring an additional optimization target. Although more accurate than clustering, these methods still do not consider overlapping speech.

SSGD This method uses a source separation model before determining diarization result $\boldsymbol{Y} \in \{0, 1\}^{S\times T}$, which is a joint speech activity of S speakers:

$$\arg\max_{\boldsymbol{Y}} p(\boldsymbol{Y}|\boldsymbol{X}) \approx \arg\max_{\boldsymbol{Y}} \prod_{s=1}^{S} p(\boldsymbol{Y}_{s,:}|\boldsymbol{M}'_{s,:}) \tag{1.11}$$

$$\boldsymbol{M}' = \arg\max_{\boldsymbol{M}} p(\boldsymbol{M}|\boldsymbol{X}). \tag{1.12}$$

The source separation model $p(\boldsymbol{M}|\boldsymbol{X})$ produces separated audio $\boldsymbol{M}' \in \mathbb{R}^{S\times T'}$ for S speakers. $p(\boldsymbol{Y}_{s,:}|\boldsymbol{M}'_{s,:})$ is the SAD model (Eq. 1.9) and is applied independently for each speaker's audio stream in $\boldsymbol{M}'$. Two independent models should be optimized. Thanks to the source separation model, it can handle overlapping speech. However, estimating the separated speech $\boldsymbol{M}'$ is generally more challenging than estimating the speaker labels $\boldsymbol{Y}$.

EEND Our proposed method optimizes directly for $p(\boldsymbol{Y}|\boldsymbol{X})$. This is a single and holistic model optimization using speaker labels. Like SSGD, due to the output label $\boldsymbol{Y}$, EEND can handle overlapping speech. We further elaborate on our formulation of EEND in detail and superiority over other existing methods with experimental results in Chapter 2.

TS-VAD This method estimates EEND's speaker labels $\boldsymbol{Y}$ from the output of speaker embedding clustering:

$$\arg\max_{\boldsymbol{Y}} p(\boldsymbol{Y}|\boldsymbol{X}) \approx \arg\max_{\boldsymbol{Y}} p(\boldsymbol{Y}|\hat{\boldsymbol{E}}, \hat{\boldsymbol{y}}), \tag{1.13}$$

where $\hat{\boldsymbol{y}}$ maximizes $p(\hat{\boldsymbol{E}}|\boldsymbol{y}, \hat{\boldsymbol{s}})p(\boldsymbol{y}|\hat{\boldsymbol{s}})$ (Eq. 1.7). Four models (three models for speaker embedding clustering and another model for speaker label estimation) should be optimized. TS-VAD can consider overlapping speech like EEND, and the pretrained speaker embedding model contributes to improved accuracy.

EEND-VC This method utilizes a joint model that produces provisional speaker labels $\hat{\boldsymbol{Y}}^{(\mathsf{local})}$ and speaker embeddings $\hat{\boldsymbol{E}}^{(\mathsf{local})}$ with block-wise processing of local audio segments. Then, clustering of $\hat{\boldsymbol{E}}^{(\mathsf{local})}$ with constraint on $\hat{\boldsymbol{Y}}^{(\mathsf{local})}$ determines global speaker labels $\boldsymbol{Y}$.

$$\arg\max_{\boldsymbol{Y}} p(\boldsymbol{Y}|\boldsymbol{X}) \approx \arg\max_{\boldsymbol{Y}} p(\hat{\boldsymbol{E}}^{(\mathsf{local})}|\boldsymbol{Y}, \hat{\boldsymbol{Y}}^{(\mathsf{local})}) \tag{1.14}$$

$$\hat{\boldsymbol{Y}}^{(\mathsf{local})}, \hat{\boldsymbol{E}}^{(\mathsf{local})} = \arg\max_{\boldsymbol{Y},\boldsymbol{E}} \prod_{b=1}^{B} p(\boldsymbol{Y}_{:,\mathcal{T}(b)}, \boldsymbol{E}_{:,\mathcal{S}(b)}|\boldsymbol{X}_{\mathcal{T}'(b)}), \tag{1.15}$$

where B is the number of blocked segments. For the b-th block, $\mathcal{T}(b)$ and $\mathcal{T}'(b)$ denote the corresponding frame indices and audio time indices, respectively. $\mathcal{S}(b)$ denotes the cumulative speaker indices for the b-th block, which distinguishes speaker indices in different blocks. This method employs a single model that simultaneously optimizes speaker diarization and speaker identification. In local temporal segments, it is equivalent to EEND, which handles overlapping speech. At the same time, its global characteristic is similar to speaker embedding clustering, contributing to improved accuracy in long audio with many speakers.

In summary, most existing methods adopt modular architecture utilizing multiple optimization targets, as listed in Table 1.1. In contrast, EEND is based on a single module with overall optimization of the speaker labels.

1.4 Research objectives

The main research objective of this dissertation is to formulate the speaker diarization problem as a simple end-to-end optimization problem. We tackle the issue of independent optimization of traditional speaker diarization pipeline systems. Our proposed method, EEND, simultaneously solves all the speaker diarization problems: speech activity detection, speaker label assignment, and over-

Table 1.1: Optimization targets and overlap handling capability of existing systems.

Method	Optimization targets	Overlap
Speaker embedding clustering	3 (SAD, spk. emb., clustering)	no
VB-HMM	4 (SAD, spk. emb., clustering, seq. generation)	no
UIS-RNN	3 (SAD, spk. emb., seq. generation)	no
SSGD	2 (speech separation, SAD)	yes
EEND	1 (speaker label)	yes
TS-VAD	4 (SAD, spk. emb., clustering, speaker label)	yes
EEND-VC	2 (speaker label, clustering)	yes

lapping speech detection. The simplicity of EEND fosters general machine learning researchers and engineers to tackle the complicated speaker diarization problem more easily. Moreover, the model with a single-neural network makes it easier to transfer knowledge from other research outcomes. We demonstrate that 1) permutation-free training (Hershey et al. (2016); Du et al. (2016)), developed for speech separation, is an essential component to solve the speaker diarization problem, and 2) the self-attention mechanism (Lin et al. (2017); Vaswani et al. (2017)), developed for natural language processing, is beneficial to aggregate global speaker characteristics contributing to speaker diarization.

The second objective is to build a better conditioning strategy specific to the speaker diarization models on top of the EEND framework. We propose a conditioning scheme based on latent variables in the EEND model, enabling us to import several findings from traditional speaker diarization systems. For instance, speaker-wise latent variables in the resegmentation module are imported to EEND, resulting in performance improvement and enabling speaker counting. It also chains a traditional pipeline of SAD and clustering. We refer to the strategy as “speaker-wise chain rule”. Furthermore, we import the idea of an iterable diarization approach (Shum et al. (2013)) into the EEND model. Intermediate predictions are the latent variables that iteratively refine the speaker label. The proposed idea is closely related to self-conditioning (Nozaki and Komatsu (2021)), originally developed for ASR. Through the introduction of self-conditioning into the EEND models, we demonstrate the superior performance of a fully non-autoregressive model that incorporates iterative refinement.

This dissertation uses a simple yet practical experimental setup: single-channel, language-agnostic, and offline processing. Whereas multi-channel inputs with a microphone array hold spatial information that helps diarization, our evaluation does not assume the availability of microphone arrays. Linguistic cues are not explicitly used since our main test dataset contains multilingual audio that requires a language-agnostic system. We leave online processing as future work, although it is preferred in real-time applications. Though speaker diarization can be jointly modeled and evaluated with ASR or speech source separation, as shown in several studies, we focus on general speaker diarization problems and do not focus on such joint optimization problems.

1.5 Dissertation organization

This dissertation comprises five chapters. Chapter 1 (this chapter) reviewed existing diarization systems and highlighted our research objective: a simple end-to-end optimization.

Chapter 2 proposes EEND, a new formulation of speaker diarization that enables end-to-end optimization. The new formulation and experiments highlight differences between the traditional and proposed EEND systems. The experimental results of EEND are strong compared to traditional systems. However, it has two main problems: 1) the number of speakers should be fixed, and 2) the conditional independence assumption in the EEND architecture causes performance degradation.

Chapter 3 addresses the problem of the fixed number of speakers. We propose a new conditioning scheme “speaker-wise chain rule”, which performs speaker-wise iterative estimation. The experiments show that the proposed method can accurately detect the number of speakers compared to the clustering-based method. The experiments also show that the iterative estimation conditioned on partially estimated speaker labels performs better than the original EEND model, indicating the importance of relaxing the conditional independence assumption.

Chapter 4 further investigates how to mitigate the performance degradation due to the conditional independence assumption in EEND models. We propose “self-conditioning”, where intermediate speaker label predictions are utilized to refine the output speaker label layer by layer. The proposed method performs conditional inference on intermediate speaker labels, which relaxes the conditional independence assumption. The experimental results show that self-conditioning boosts the performance of EEND models. We also explore efficient architectures for EEND with self-conditioning and propose a non-autoregressive attractor model. The proposed model not only achieved better performance but also requires fewer parameters compared to existing models.

Chapter 5 summarizes the dissertation with the future directions by showing some recent work utilizing the concept of EEND.

2

Formulation of speaker diarization with end-to-end optimization

2.1 Introduction

This chapter introduces a new formulation of the speaker diarization problem that no longer divides the problem into subproblems. Firstly, we develop the formulation of the traditional system. Then, we propose the formulation of a novel end-to-end optimal system. The discrepancy between the two formulations reveals the superiority of the proposed end-to-end optimal system. Finally, experiments demonstrate the effectiveness of the proposed method.

2.2 Traditional system: Single-sequence embedding generation model

Traditional systems assume that speaker diarization is done by partitioning audio segments so that each time frame belongs to only one speaker or non-speech. Therefore, the estimation target is a *single* sequence of speaker indices, written in a vector form as $\boldsymbol{y} \in \mathbb{Z}_{\geq 0}^{T}$, where T is the number of output frames and $\mathbb{Z}_{\geq 0}$ is a set of non-negative integers. $\boldsymbol{y}_t = c$ means that time t is assigned to c-th speaker. $\boldsymbol{y}_t = 0$ means non-speech at time t. An input audio stream is assumed to be a sequence of feature vectors $\boldsymbol{X} \in \mathbb{R}^{D' \times T'}$, where D' is a feature vector (e.g., filterbank) dimension, and T' is the number of input frames. Using probabilistic modeling, we write the objective of speaker diarization as follows:

$$\hat{\boldsymbol{y}} = \arg\max_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{X}). \tag{2.1}$$

Here, we write $p(\boldsymbol{y}|\boldsymbol{X})$ to describe a probability mass function of a multivariate random variable $\boldsymbol{y}$ given $\boldsymbol{X}$.

The model comprises three independent models: speech activity detection, speaker embedding extraction, and clustering (speaker assignment). To do that, we first introduce a variable $\boldsymbol{s} = \min(\boldsymbol{y}, \mathbf{1}) \in \{0,1\}^{T'}$ corresponding to speech activity detection, where $\min$ is element-wise minimum, and $\mathbf{1}$ is the all-one vector. Then, the marginal probability over $\boldsymbol{s}$ is approximated with the most probable value of $\boldsymbol{s}$:

$$\hat{\boldsymbol{y}} = \arg\max_{\boldsymbol{y}} \sum_{\boldsymbol{s}} p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{s}) p(\boldsymbol{s}|\boldsymbol{X}) \tag{2.2}$$

$$\approx \arg\max_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{X}, \hat{\boldsymbol{s}}) p(\hat{\boldsymbol{s}}|\boldsymbol{X}), \tag{2.3}$$

$$\approx \arg\max_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{X}, \hat{\boldsymbol{s}}), \tag{2.4}$$

where $\hat{\boldsymbol{s}} = \arg\max_{\boldsymbol{s}} p(\boldsymbol{s}|\boldsymbol{X})$ is the output of speech activity detector. $p(\boldsymbol{y}|\boldsymbol{X}, \hat{\boldsymbol{s}})$ means speaker embedding extraction and clustering considering given speech activity $\hat{\boldsymbol{s}}$. In a similar way, we introduce another variable $\boldsymbol{E} \in \mathbb{R}^{D \times T}$ corresponding to a sequence of T-length D-dimensional speaker embeddings, and marginalizing it out:

$$\arg\max_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{X}, \hat{\boldsymbol{s}}) \approx \arg\max_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{X}, \hat{\boldsymbol{E}}, \hat{\boldsymbol{s}}), \tag{2.5}$$

where $\hat{\boldsymbol{E}} = \arg\max_{\boldsymbol{E}} p(\boldsymbol{E}|\boldsymbol{X}, \hat{\boldsymbol{s}})$ is the speaker embedding extracted from $\boldsymbol{X}$ with the guide of speech activity label $\hat{\boldsymbol{s}}$. Then we rewrite the discriminative model of $\boldsymbol{y}$ as a generative model of speaker embeddings $\hat{\boldsymbol{E}}$ given speaker indices $\boldsymbol{y}$, using the Bayes rule:

$$\arg\max_{\boldsymbol{y}} p(\boldsymbol{y}|\boldsymbol{X}, \hat{\boldsymbol{E}}, \hat{\boldsymbol{s}}) = \arg\max_{\boldsymbol{y}} p(\hat{\boldsymbol{E}}|\boldsymbol{y}, \hat{\boldsymbol{s}}) p(\boldsymbol{y}|\hat{\boldsymbol{s}}). \tag{2.6}$$

Here, the traditional systems omit $\boldsymbol{X}$ because clustering is performed solely on speaker embeddings $\hat{\boldsymbol{E}}$. This approximation heavily relies on the quality of frame-level speaker embeddings, i.e., the speaker identification model. $p(\boldsymbol{y}|\hat{\boldsymbol{s}})$ means that we deterministically assign “non-speech” labels according to the speech activity label. $p(\hat{\boldsymbol{E}}|\boldsymbol{y}, \hat{\boldsymbol{s}})$ corresponds to the clustering process. Given two speaker embeddings, a scorer, such as a PLDA scorer, can estimate a log-likelihood ratio of the same speaker hypothesis over the different speaker hypothesis. $p(\hat{\boldsymbol{E}}|\boldsymbol{y}, \hat{\boldsymbol{s}})$ is maximized by finding $\boldsymbol{y}$ so that the total log-likelihood ratio over all pairs of speaker embeddings is maximized.

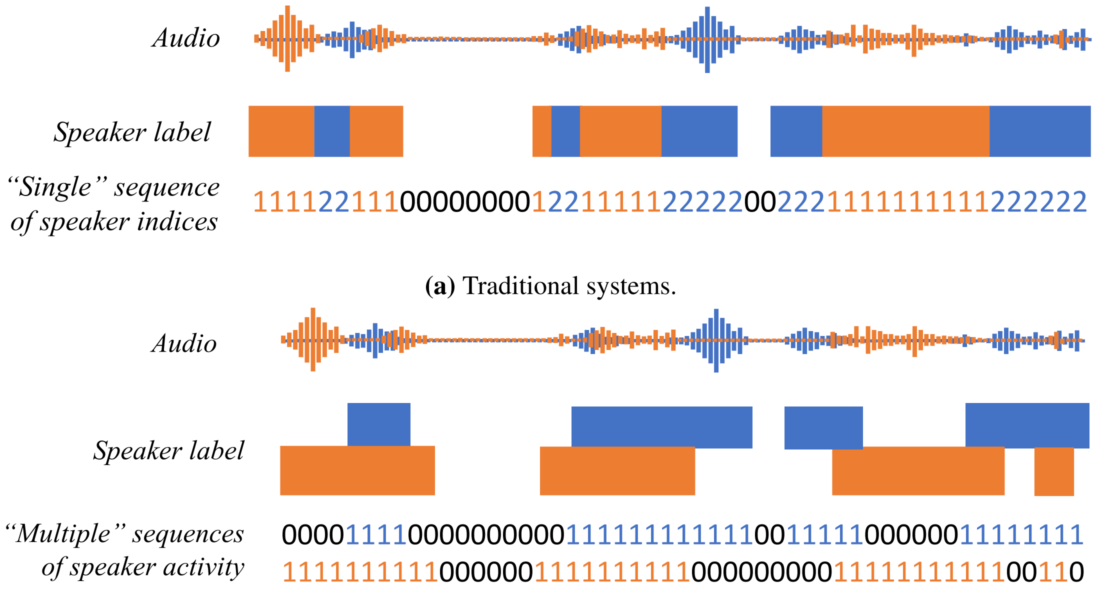
Finally, we obtain three independent models: (1) $p(\hat{\boldsymbol{s}}|\boldsymbol{X})$ as a speech activity detector, (2) $p(\hat{\boldsymbol{E}}|\boldsymbol{X}, \hat{\boldsymbol{s}})$ as a speaker embedding extractor, and (3) $p(\hat{\boldsymbol{E}}|\boldsymbol{y}, \hat{\boldsymbol{s}})$ as a scorer for clustering.

From the formulation, we can see many drawbacks in the traditional system, as follows.

1. These approximations cause performance degradation because we cannot recover the SAD error in the later pipeline. $\hat{\boldsymbol{s}}$ in the later pipeline is only used to ignore the non-speech frames; the decision is deterministic.

2. Speaker embedding and clustering, i.e., the PLDA scorer, are independently optimized. Note that Garcia-Romero et al. (2017) is an exception, which proposed a joint model of the speaker embedding and PLDA scorer. However, most traditional systems utilize a speaker embedding model trained with speaker identification criteria and then train a scoring model using fixed speaker embeddings.

3. It cannot be optimized using the target $\boldsymbol{y}$, because clustering is an unsupervised process.



(a) Traditional systems.

(b) EEND.

Figure 2-1: Difference in speaker label target between traditional and EEND systems.

Though the PLDA scorer can be optimized to guide the clustering process, it cannot use sequential (local) input or output characteristics because the scorer only considers a pair of speaker embeddings without time information. Note that UIS-RNN (Zhang et al. (2019)) replaces the clustering part with a supervised embedding generation model trained with pairs of $\boldsymbol{E}$ and $\boldsymbol{y}$. However, their RNN model should be optimized independently from the speaker embedding extractor.

4. With the definition of target $\boldsymbol{y}$ as a single sequence of speaker indices, we cannot handle overlapping speech without considering another model. In Landini et al. (2021), overlapping speech is detected using another speech separation model and heuristically assigns the *second* speaker based on closeness in time. However, the pipeline becomes more complicated to deal with overlapping speech.

2.3 EEND: Multi-sequence end-to-end model

This section provides a novel formulation of speaker diarization that no longer divides the problem into subproblems. First, we introduce a new speaker label target $\boldsymbol{Y} \in \{0,1\}^{S \times T}$ instead of $\boldsymbol{y} \in \mathbb{Z}^{T}_{\geq 0}$. S is the number of speakers in audio, and the row s of $\boldsymbol{Y}$ corresponds to the speech activity sequence of s-th speaker. The new target comprises *multiple* (S) sequences. This target can describe the overlapping speech by putting multiple ones in the same column, e.g., $\boldsymbol{Y}_{1,t} = \boldsymbol{Y}_{2,t} = 1$. Fig 2-1 depicts the difference in speaker label target between the traditional system $\boldsymbol{y}$ and the EEND system $\boldsymbol{Y}$. We write the objective of speaker diarization with the new target $\boldsymbol{Y}$:

$$\hat{\boldsymbol{Y}} = \arg\max_{\boldsymbol{Y}} p(\boldsymbol{Y}|\boldsymbol{X}). \tag{2.7}$$

End-to-end optimization uses a neural network to learn function f that maps input audio $\boldsymbol{X}$ to the speaker label posterior, $f : \boldsymbol{X} \mapsto p(\boldsymbol{Y}|\boldsymbol{X})$. However, computing the joint posteriors of all frames and speakers is difficult. Therefore, we instead estimate the frame-wise and speaker-wise posterior $p(\boldsymbol{Y}_{s,t}|\boldsymbol{X})$, which is conditioned on all input frames but independent of outputs from other frames and speakers, yielding parallel computation of all the posteriors. This can be interpreted as an approximation of $p(\boldsymbol{Y}|\boldsymbol{X})$ in the objective Eq. 2.7:

$$\begin{aligned} p(\boldsymbol{Y}|\boldsymbol{X}) &= \prod_{t=1}^{T}\prod_{s=1}^{S} p(\boldsymbol{Y}_{s,t}|\boldsymbol{X}, \boldsymbol{Y}_{<s,:}, \boldsymbol{Y}_{s,<t}) &(2.8)\\ &\approx \prod_{t=1}^{T}\prod_{s=1}^{S} p(\boldsymbol{Y}_{s,t}|\boldsymbol{X}), &(2.9) \end{aligned}$$

where $\boldsymbol{Y}_{<s,:}$ is a submatrix of $\boldsymbol{Y}$ containing rows $[0, s-1]$, $\boldsymbol{Y}_{s,<t}$ is a subvector of row s of $\boldsymbol{Y}$ containing columns $[0, t-1]$, and these conditions are omitted. Consequently, we train a neural network function f to estimate the speaker label posterior $\hat{\boldsymbol{Z}} \in [0,1]^{S \times T}$ as:

$$\hat{\boldsymbol{Z}} = f(\boldsymbol{X}), \tag{2.10}$$

where the element $\hat{\boldsymbol{Z}}_{s,t}$ is a posterior of s-th speaker activity at time t: $p(\boldsymbol{Y}_{s,t}|\boldsymbol{X})$ [1]. With the

[1] The actual output is $p(\boldsymbol{Y}_{s,t} = 1|\boldsymbol{X}) \in [0,1]$ as the target is a binary variable

network, the inference is achieved by $\hat{\boldsymbol{Y}}_{s,t} = \mathbb{1}[\hat{\boldsymbol{Z}}_{s,t} > 0.5]$, where $\mathbb{1}$ is the indicator function that returns 1 if the condition in the argument is true and returns 0 otherwise.

Another difficulty of the optimization is that the model must consider speaker label permutations: changing the order of speaker indices within a correct speaker label $\boldsymbol{Y}$ is also regarded as correct. Formally, with any permutation matrix $\boldsymbol{P} \in \{0,1\}^{S \times S}$, $\boldsymbol{PY}$ is equivalent to $\boldsymbol{Y}$. The label permutations obstruct the training of the neural network when we use a standard binary cross-entropy loss function.

To solve the label permutation problem, we employ a permutation-free training scheme that considers all the permutations of the reference speaker label. The permutation-free training scheme has been used in research on source separation (Hershey et al. (2016); Yu et al. (2017); Kolbæk et al. (2017)). Here, we apply a permutation-free loss function to the speaker label.

$$\mathcal{L}_{\mathsf{PF}}(\boldsymbol{Y}, \hat{\boldsymbol{Z}}) = \min_{\boldsymbol{P} \in \mathcal{P}(S)} \mathsf{BCE}(\boldsymbol{PY}, \hat{\boldsymbol{Z}}), \tag{2.11}$$

$$\mathsf{BCE}(\boldsymbol{\Psi}, \boldsymbol{\Omega}) = \frac{1}{ST} \sum_{s=1}^{S} \sum_{t=1}^{T} -\boldsymbol{\Psi}_{s,t} \log \boldsymbol{\Omega}_{s,t} - (1 - \boldsymbol{\Psi}_{s,t}) \log(1 - \boldsymbol{\Omega}_{s,t}), \tag{2.12}$$

where BCE computes element-wise binary cross-entropy between target label elements and estimated posteriors, and $\mathcal{P}(S)$ is the set of $S \times S$ permutation matrices. Fig. 2-2 depicts the training of EEND with permutation-free loss in a two-speaker case. As shown in the figure, the training process evaluates both permutations and selects the one that minimizes binary cross-entropy. The network is learned to select the order of speakers in a self-organizing manner, which is theoretically better than any rule-based order, such as "first-observed speaker as the first," or "most-speaking speaker as the first."

2.4 Neural network architecture for EEND

We investigate two different architectures for the EEND model. Firstly, the BLSTM model is used to demonstrate the effectiveness of the permutation-free loss on the simulated dataset. Secondly, the self-attention-based model is used to demonstrate the importance of the "speaker aggregation layer" and superior performance compared to the BLSTM model and traditional speaker embed-

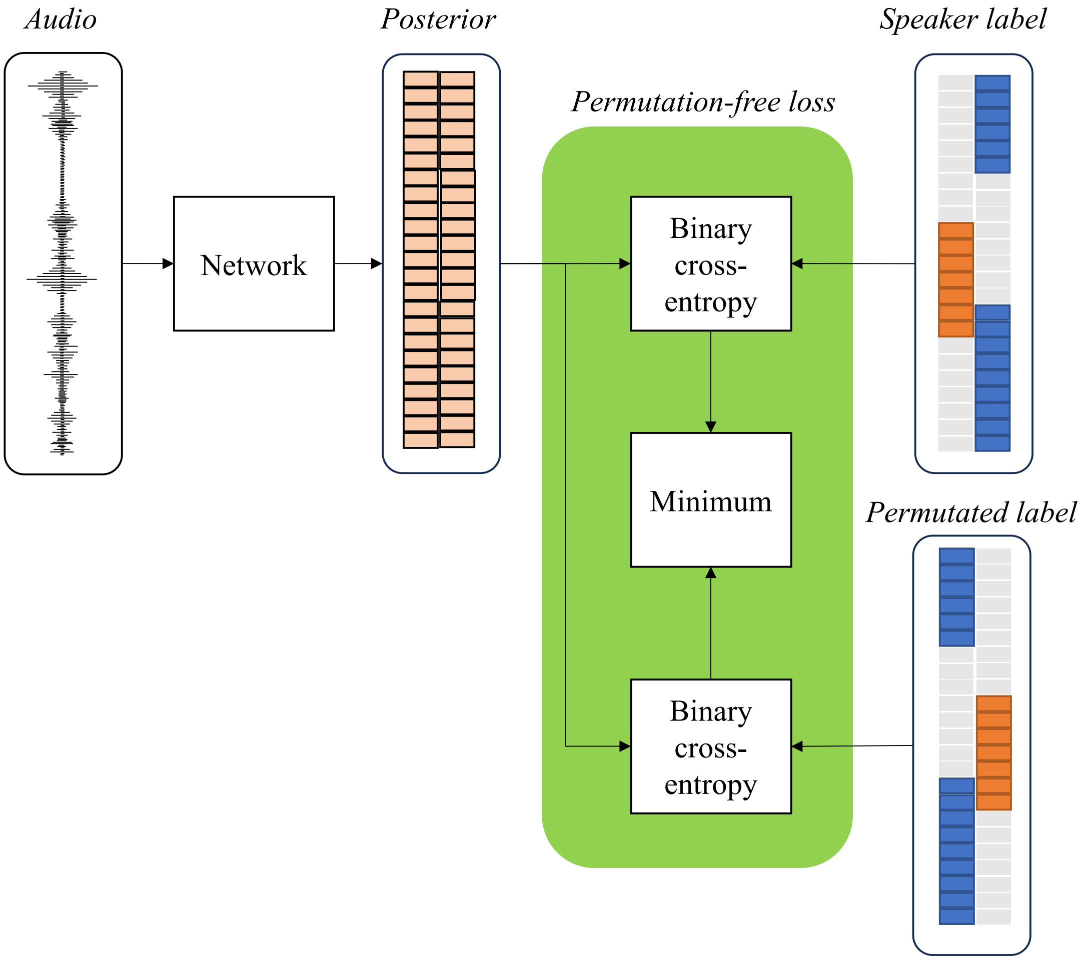


Figure 2-2: EEND model trained with permutation-free loss.

ding clustering models.

2.4.1 BLSTM-based neural network with Deep Clustering loss

Bidirectional long short-term memory (BLSTM; Graves and Schmidhuber (2005)) is a popular network architecture to process a sequence of features. The input audio $\boldsymbol{X}$ is transformed as follows:

$$\boldsymbol{X}^{(0)} = \mathsf{SubSample}(\boldsymbol{X}) \in \mathbb{R}^{D' \times T} \tag{2.13}$$

$$\boldsymbol{H}^{(l)} = \mathsf{BLSTM}^{(l)}(\boldsymbol{H}^{(l-1)}) \in \mathbb{R}^{2D \times T} \quad (1 \leq l \leq L) \tag{2.14}$$

$$\hat{\boldsymbol{Z}} = \sigma \circ \mathsf{Linear}^{(\mathsf{o})}(\boldsymbol{H}^{(L)}). \tag{2.15}$$

Here, $\mathsf{SubSample}$ reduces the number of frames from T' to T because a frameshift of audio features is generally smaller than that of the output speaker label. $\mathsf{BLSTM}^{(1)}$ is the first BLSTM layer which accepts a sequence of D'-dimensional vectors and produces a sequence of concatenated vector of D-dimenstional forward and backward LSTM outputs. For $l > 1$, $\mathsf{BLSTM}^{(l)}$ accepts $(2D \times T)$ matrix as the previous layer output. $\mathsf{Linear}^{(o)}$ is a linear layer to project $2D$-dimensional vectors into S-dimensional vectors. σ is the element-wise sigmoid function. L is the number of BLSTM layers.

Besides the main branch to produce $\hat{\boldsymbol{Z}}$, we add another branch to generate frame-wise speaker embeddings in the lower layers, which works as a regularizer. The M-th BLSTM layer output $\boldsymbol{H}^{(M)}$ obtained from Eq. 2.14 is transformed into normalized V-dimensional embedding:

$$\boldsymbol{V} = \mathrm{Normalize} \circ \mathrm{Tanh} \circ \mathsf{Linear}^{(\mathsf{v})}(\boldsymbol{H}^{(M)}) \in \mathbb{R}^{V \times T}, \tag{2.16}$$

where $\mathsf{Linear}^{(\mathsf{v})}$ is a linear layer to convert dimension from $2D$ to V, Tanh is the element-wise hyperbolic tangent function, and $\mathrm{Normalize}$ is the L2 normalization function. The Deep Clustering (DC) loss function (Hershey et al. (2016)) is applied to $\boldsymbol{V}$ so that the embedding vectors are partitioned into speaker-dependent clusters and overlapping and non-speech clusters. For example, four clusters (non-speech, speaker 1, speaker 2, and overlapping) are involved in a two-speaker audio. The DC loss function is expressed as follows:

$$\mathcal{L}_{\mathsf{DC}} = \|\boldsymbol{V}^\top \boldsymbol{V} - \boldsymbol{Y'}^\top \boldsymbol{Y'}\|_F^2, \tag{2.17}$$

where $\boldsymbol{Y'} \in \mathbb{R}^{2^S \times T}$ is a matrix in which each column represents a one-hot vector converted from $\boldsymbol{Y}_{:,t}$ to represent the cluster index in the power set of speakers. $\|\cdot\|_F$ is the Frobenius norm. The loss function encourages the two embeddings at different time indices to be close together if they are in the same cluster and encourages them to be far apart otherwise. We mix the two objectives with a mixing parameter α:

$$\mathcal{L}_{\mathsf{PF+DC}} = (1-\alpha)\mathcal{L}_{\mathsf{PF}} + \alpha\mathcal{L}_{\mathsf{DC}}. \tag{2.18}$$

2.4.2 Self-attention-based neural network

Using BLSTM, each output frame is conditioned solely on its previous hidden state, subsequent hidden state, and current input frame. In contrast, by utilizing a self-attention mechanism (Lin et al. (2017)), each frame-level output is conditioned on all the input frames by computing the pairwise similarity between all pairs of input frames. Self-attention can aggregate global information from all frames based on similarity, which fits the speaker diarization task since it requires global speaker characteristics distributed to the whole audio input and requires the frame-level assignment based on the characteristics.

Here, we use a self-attention-based neural network using Transformer encoders (Vaswani et al. (2017)) instead of BLSTM. The input features are transformed as follows:

$$\boldsymbol{E}^{(0)} = \mathsf{Linear}^{(\mathsf{i})} \circ \mathsf{Subsample}(\boldsymbol{X}) \in \mathbb{R}^{D \times T} \tag{2.19}$$

$$\boldsymbol{E}^{(l)} = \mathsf{Enc}^{(l)}(\boldsymbol{E}^{(l-1)}) \in \mathbb{R}^{D \times T} \quad (1 \le p \le L), \tag{2.20}$$

$$\hat{\boldsymbol{Z}} = \sigma \circ \mathsf{Linear}^{(\mathsf{o})} \circ \mathsf{LayerNorm}(\boldsymbol{E}^{(L)}) \tag{2.21}$$

Here, $\mathsf{Linear}^{(\mathsf{i})}$ projects input vectors into D-dimensional vectors. $\mathrm{Enc}^{(l)}$ is the l-th Transformer encoder block. We use L encoder blocks followed by layer normalization (Lei Ba et al. (2016)), a linear layer, and sigmoid activations to obtain the posteriors. Note that the configuration of the

Table 2.1: Statistics of EEND training and test sets.

		Num. of mixtures	Avg. duration (s)	Overlap ratio (%)
Training				
SimBeta2	Simulated ($\beta = 2$)	100,000	87.6	34.4
Real	SWBD+SRE	26,172	304.7	3.7
SimLarge	Simu. ($\beta = 2, 3, 5, 7$)	400,000	126.4	23.4
Comb	Real+SimLarge	426,172	137.3	20.5
Test				
1	Simulated ($\beta = 2$)	500	87.3	34.4
2	Simulated ($\beta = 3$)	500	103.8	27.2
3	Simulated ($\beta = 5$)	500	137.1	19.5
4	CALLHOME	148	72.1	13.0
5	CSJ	54	766.3	20.1

encoder block is almost the same as the one in the Speech-Transformer introduced in Dong et al. (2018), but without positional encoding.

2.5 Experimental setup

2.5.1 Data

We verified the effectiveness of EEND for various overlap situations. We included four sets of training data and five sets of test data. These sets are categorized into simulated and real datasets. The statistics of the training and test sets are listed in Table 2.1. The overlap ratio is the ratio of the audio time of overlapping segments over the total speech segments.

The training data for speaker embedding clustering differs from the EEND training data. The clustering-based methods use single-speaker segments for training the speaker embedding models. Instead, EEND uses mixed audio from multiple speakers as training data.

Simulated datasets

We developed a mixture simulation method. Algorithm 1 describes the simulation method. Unlike the well-known mixture simulation algorithm for speech separation study (Hershey et al. (2016)), we prepared *conversation-style* mixtures: each mixture has multiple utterances per speaker with

Algorithm 1: Conversation-style mixture simulation algorithm.

```
Input: $\mathcal{S}, \mathcal{N}, \mathcal{I}, \mathcal{R}$          // Speakers, noises, RIRs and SNRs
       $\mathcal{U} = \{U_s\}_{s \in \mathcal{S}}$                // Utterances of speaker s
       $N_{\text{spk}}$                            // #speakers per mixture
       $N_{\text{umax}}, N_{\text{umin}}$      // Max.  and min.  #utterances per speaker
       $\beta$                                // Average silence interval
Output: $\mathbf{y}$                                             // Mixture
Sample a set of $N_{\text{spk}}$ speakers $\mathcal{S}'$ from $\mathcal{S}$
$\mathcal{X} \leftarrow \emptyset$ forall $s \in \mathcal{S}'$ do
    $\mathbf{x}_s \leftarrow \emptyset$ Sample $\mathbf{i}$ from $\mathcal{I}$ Sample $N_u$ from $\{N_{\text{umin}}, \ldots, N_{\text{umax}}\}$
    for $u = 1$ to $N_u$ do
        Sample $\delta \sim \frac{1}{\beta} \exp\left(-\frac{\delta}{\beta}\right)$                // Silence interval
        $\mathbf{x}_s \leftarrow \mathbf{x}_s \oplus \mathbf{0}^{(\delta)} \oplus U_s[u] * \mathbf{i}$      // Append silence and utterance
    $\mathcal{X}$.add$(\mathbf{x}_s)$
$L_{\max} = \max_{\mathbf{x} \in \mathcal{X}} |\mathbf{x}|$
$\mathbf{y} \leftarrow \sum_{\mathbf{x} \in \mathcal{X}} \left(\mathbf{x} \oplus \mathbf{0}^{(L_{\max} - |\mathbf{x}|)}\right)$                          // Mix-down
Sample $\mathbf{n}$ from $\mathcal{N}$ Sample $r$ from $\mathcal{R}$ Determine a mixing scale $p$ from $r, \mathbf{y}$, and $\mathbf{n}$
$\mathbf{n}' \leftarrow$ repeat $\mathbf{n}$ until the length of $\mathbf{y}$ is reached
$\mathbf{y} \leftarrow \mathbf{y} + p \cdot \mathbf{n}'$
```

randomly sampled silence between the utterances. A hyperparameter β controls the average silence interval. Large β generates large silence intervals, resulting in less overlap.

The audio sources were telephone speech, comprised of the Switchboard-2 (Phase I, II, III), Switchboard Cellular (Part 1, Part2), and NIST Speaker Recognition Evaluation datasets (2004, 2005, 2006, 2008). All audio sources sampled at 8 kHz. We split the dataset, which had 6,381 speakers, into 5,743 training speakers and 638 test speakers. The split is identical to that of the Kaldi recipe: CALLHOME diarization v2 (Povey et al. (2011)), enabling a fair comparison with the clustering-based methods in the recipe. Since there are no speech activity annotations in the datasets, we used a speech activity detector based on TDNN and statistics pooling[2]. A set of 37 background noises was from the MUSAN corpus (Snyder et al. (2015)). The set of 10,000 room impulse responses (RIRs) was from the Simulated Room Impulse Response Database used in Ko et al. (2017). The candidate SNR values were 10, 15, and 20 dB. These noises and RIRs were also used for training the x-vector and SAD models in the x-vector clustering-based method.

[2] The SAD model: `http://kaldi-asr.org/models/m4`

We generated two-speaker mixtures for each speaker with 10-20 utterances ($N_{\text{spk}} = 2, N_{\text{umin}} = 10, N_{\text{umax}} = 20$). For the simulated training set, we generated 100,000 mixtures with $\beta = 2$ (SimBeta2). We also prepared four sets of 100,000 mixtures with different values of β (2, 3, 5, and 7), and combined them to form 400,000 mixtures (SimLarge). For the simulated test set, we generated 500 mixtures with $\beta = 2$, 3, and 5. The resulting overlap ratios of the simulated mixtures were from 19.5 to 34.4%.

Real datasets

We prepared telephone speech recordings as the real training set (Real). The real training set comprised 26,172 two-speaker recordings, which were extracted from Switchboard-2 (Phase I, II, III), Switchboard Cellular (Part 1, Part 2), and NIST Speaker Recognition Evaluation datasets. The overlap ratio of the training data was 3.7%, which is significantly less than that of the simulated mixtures.

We evaluated the proposed method on real telephone conversations in the CALLHOME dataset (NIST (2000)). We split the two-speaker audio from the CALLHOME dataset into 155 adaptation data and 158 test data. The overlap ratio of the real test set was 13.0%.

In addition, we prepared another test set from the dialogue part of the Corpus of Spontaneous Japanese (CSJ; Maekawa (2003)). The original corpus contains 58 two-speaker dialogues, recorded using headset microphones in soundproof rooms. To make the test test, we excluded four dialogues that contain speakers in the official ASR evaluation sets. The overlap ratio of the CSJ test set was 20.1%, which is larger than the CALLHOME test set.

Combined datasets

To generalize a model to various environments, we conducted experiments using both the simulated training set (SimLarge) and the real training set (Real). We refer to the dataset as the combined training set (Comb).

2.5.2 Model configuration

Clustering-based systems

The proposed method was compared against two traditional clustering-based methods: the i-vector and x-vector systems, which were developed using the Kaldi CALLHOME diarization v1 and v2 recipes (Sell et al. (2018)). These recipes used AHC and PLDA. We set the fixed number of speakers to two. Unlike the original recipes that utilize oracle speech/non-speech labels, we employed the SAD model, configured as detailed in Sec. 2.5.1.

BLSTM-based EEND system

The BLSTM-EEND system, as detailed in Sec. 2.4.1, was set up with specific configurations. We used 23-dimensional log-Mel-filterbanks as input features, with a frame length of 25 ms and a frame shift of 10 ms. Features from the previous and subsequent seven frames were concatenated with each feature. To manage long audio sequences in our neural networks, we subsampled these concatenated features by a factor of ten, resulting in a (23×15)-dimensional input feature being fed into the network every 100 ms.

We used a five-layer BLSTM, each layer having 256 hidden units. The output from the second layer of the BLSTM was transformed into a 256-dimensional embedding. This embedding was used to compute the Deep Clustering loss. The mixing parameter α was set at 0.5. For optimization, we utilized the Adam (Kingma and Ba (2015)) optimizer with a learning rate of 10^{-3} and a batch size of 10. The training iterated over 20 epochs.

The neural network output is the probability of speech activity per speaker. To make a decision on speech activity for each frame, we set a threshold at 0.5. Additionally, to avoid creating excessively short segments, we applied 11-frame median filtering.

For domain adaptation, we retrained the neural network using the CALLHOME adaptation set, employing the Adam optimizer with a learning rate of 10^{-6} for five epochs. In postprocessing, we modified the threshold to 0.6, optimizing the Diarization Error Rate (DER) for the adaptation set.

Self-attention-based EEND system

The self-attention-based EEND model (SA-EEND) utilized the same input features as the BLSTM-EEND system. However, due to the higher memory consumption of the SA-EEND system, the sequence length during training was limited to 500, equivalent to 50 seconds of audio time. Consequently, input audio recordings were divided into non-overlapping 50-second segments. In the inference stage, we processed the full sequence for each recording.

The model was composed of two encoder blocks, each with 256 attention units and four heads. The position-wise feed-forward layer within these blocks contained 1024 units. For optimization, the Adam optimizer was employed along with a learning rate scheduler as described in Vaswani et al. (2017). The learning rate scheduler included 25,000 warm-up steps, and the batch size was set to 64. The training was conducted over 100 epochs.

After 100 epochs, an averaged model was created by averaging the parameters of the models from the last ten epochs. As with the BLSTM-EEND system, 11-frame median filtering was applied.

For domain adaptation, this averaged model was further trained using the CALLHOME adaptation set. The training used the Adam optimizer with a learning rate of 10^{-5} for an additional 100 epochs. Then, an averaged model was again obtained by averaging the model parameters from the final ten epochs.

2.5.3 Performance metric

We evaluated the systems with DER (NIST (2009)). Note that DERs reported in numerous previous studies may not fully represent the performance of diarization systems. This is because these studies often used oracle speech/non-speech labels, which led to the exclusion of misses or false alarm errors in their evaluations. Moreover, overlapping speech segments were typically not considered in their evaluations, further impacting the comprehensiveness of the reported DERs.

Instead, we evaluated all the errors, including overlapping speech segments, because the proposed method includes both speech activity detection and overlapping speech detection functionality. As is typically done, we used a collar tolerance of 250 ms at the start and end of each segment.

Table 2.2: Effect of loss functions evaluated on simulated speech generated with $\beta = 2$. We trained BLSTM-based models using 10,000 mixtures generated with $\beta = 2$.

Permutation-free loss	DC loss	DER (%)
-	-	41.74
✓	-	25.14
✓	✓	23.79

Table 2.3: Effect of training data size evaluated on simulated speech generated with $\beta = 2$. We trained BLSTM-based models using simulated mixtures with $\beta = 2$.

Number of training mixtures	DER(%)
10,000	23.79
20,000	14.66
100,000	12.28

2.6 Results

2.6.1 Effect of loss functions and training data size

We first evaluated the effect of the proposed loss functions. Without the permutation-free loss, we used a standard binary cross-entropy loss with the fixed permutation by sorting the speaker names in a lexical order. With permutation-free and DC losses, we set the mixing parameter $\alpha = 0.5$. Table 2.2 shows the results. The results demonstrate that the permutation-free loss is essential for training the EEND network, and DC loss helps improve performance.

The comparison with different training data sizes is shown in Table 2.3. Clearly, performance was improved with increasing training data size.

2.6.2 Evaluation on simulated mixtures

DERs on various test sets are shown in Table 2.4. The performances of clustering-based systems were weak in scenarios with high overlap in simulated mixtures. The results were anticipated as these systems do not account for speaker overlaps, leading to increased misses in high-overlap scenarios.

In contrast, the BLSTM-EEND system, trained on the SimBeta2 dataset, demonstrated a sig-

Table 2.4: DERs (%) on various test sets. For EEND systems, the CALLHOME (CH) results were obtained with domain adaptation.

	Simulated			Real	
	$\beta = 2$	$\beta = 3$	$\beta = 5$	CH	CSJ
Clustering-based					
i-vector	33.74	30.93	25.96	12.10	27.99
x-vector	28.77	24.46	19.78	11.53	22.96
BLSTM-EEND					
trained with SimBeta2	12.28	14.36	19.69	26.03	39.33
trained with Real	36.23	37.78	40.34	23.07	25.37
SA-EEND					
trained with SimBeta2	7.91	8.51	9.51	13.66	22.31
trained with Real	32.72	33.84	36.78	**10.76**	**20.50**
trained with SimLarge	**6.81**	6.60	6.40	14.03	21.84
trained with Comb	6.92	**6.54**	**6.38**	11.99	22.26

nificant DER reduction on simulated mixtures compared to the clustering-based systems. The system was particularly effective in conditions with the highest overlap ($\beta = 2$), reflecting the system's ability to handle overlapping speech similar to those in the training data.

The SA-EEND system, also trained on the simulated dataset, outperformed the BLSTM-EEND system across all test sets, achieving significantly lower DERs. Like the BLSTM-EEND system, it showed optimal performance in the highest overlap condition ($\beta = 2$). Notably, the SA-EEND system exhibited less performance degradation in low overlap conditions compared to the BLSTM-EEND system. The results suggest that the inclusion of self-attention blocks enhanced its robustness to varying degrees of overlap.

Further, training the SA-EEND model with various overlap conditions (SimLarge) resulted in improvements across all test sets over training with a single overlap condition (SimBeta2). The results indicate that training with diverse overlap scenarios can help reduce the risk of overfitting to a specific overlap ratio.

2.6.3 Evaluation on real test sets

Despite its strong performance on simulated mixtures, the BLSTM-EEND system showed less effective results on real test sets when compared to the clustering-based systems. Even after switch-

Table 2.5: DERs (%) on the CALLHOME with and without domain adaptation.

	w/o adaptation	with adaptatation
x-vector clustering	11.53	N/A
BLSTM-EEND		
trained with SimBeta2	43.84	26.03
trained with Real	31.01	23.07
SA-EEND		
trained with SimBeta2	17.42	13.66
trained with SimLarge	16.31	14.03
trained with Real	12.66	**10.76**
trained with Comb	14.50	11.99

ing its training data from simulated to real, the DERs of the BLSTM-EEND system remained higher than those of the clustering-based systems.

Conversely, the SA-EEND system trained with the SimBeta2 dataset exhibited notable improvements on real test sets of CALLHOME and CSJ. These improvements highlight the generalization capability of the self-attention blocks in the system. For the CSJ test set, the SA-EEND system outperformed the x-vector clustering-based method even without domain adaptation. Training the SA-EEND model with a variety of overlap ratio conditions (SimLarge) further enhanced its generalization to real test sets.

The SA-EEND system trained with real data (Real) demonstrated superior performance on real test sets compared to the SimLarge model. However, its performance on simulated test sets was not as strong, due to the limited diversity and lower overlap ratios in the real training set. In contrast, the SA-EEND system trained with a combined dataset (Comb), incorporating various overlap ratios, demonstrated excellent generalization abilities. The results suggest that exposing the model to a wide range of overlap conditions during training can significantly enhance its adaptability to different test scenarios.

2.6.4 Effect of domain adaptation

The EEND models trained with simulated datasets displayed overfitting to the specific overlap ratio presented in the training set. Domain adaptation was expected to mitigate such overfitting.

Table 2.6: Detailed DERs (%) evaluated on the CALLHOME. DER is composed of Miss (MI), False alarm (FA), and Confusion (CF) errors. The SAD errors are composed of Miss (MI) and False alarm (FA) errors.

		DER breakdown			SAD errors	
Method	DER	MI	FA	CF	MI	FA
Clustering						
i-vector	12.10	7.74	0.54	3.82	1.4	0.5
x-vector	11.53	7.74	0.54	3.25	1.4	0.5
SA-EEND						
no-adapt	12.66	7.42	3.93	1.31	3.3	0.6
adapted	**10.76**	6.68	2.40	1.68	2.3	0.5

Indeed, as shown in Table 2.5, domain adaptation significantly decreased the DERs on the CALLHOME dataset, leading to even better results than those achieved by the x-vector-based system.

A more detailed analysis of DERs on the CALLHOME test set is presented in Table 2.6. The clustering-based systems exhibited fewer SAD errors, benefiting from a robust SAD model trained on diverse, noise-augmented data. However, these systems faced challenges with misses and confusion errors, primarily due to their inability to handle speaker overlaps.

In contrast, the proposed EEND models resulted in significantly fewer confusion and miss errors than the clustering-based systems. Furthermore, the application of domain adaptation led to a reduction in all types of errors except for confusion errors. These results indicate that while domain adaptation improves overall diarization performance, there may still be room for enhancing its effectiveness in reducing confusion errors.

2.6.5 Visualization of self-attention

The analysis of the self-attention mechanism in the EEND model provides insights into how it processes audio data. Fig. 2-3 shows the attention weight matrix at the second encoder block.

Heads 1 and 2 display vertical lines at different positions within the matrix. These vertical lines correlate with the activity of each speaker, indicating that these heads are transforming the input features into a weighted mean of frames corresponding to the same speaker. This suggests that heads 1 and 2 are capturing global speaker characteristics, essential features for speaker di-

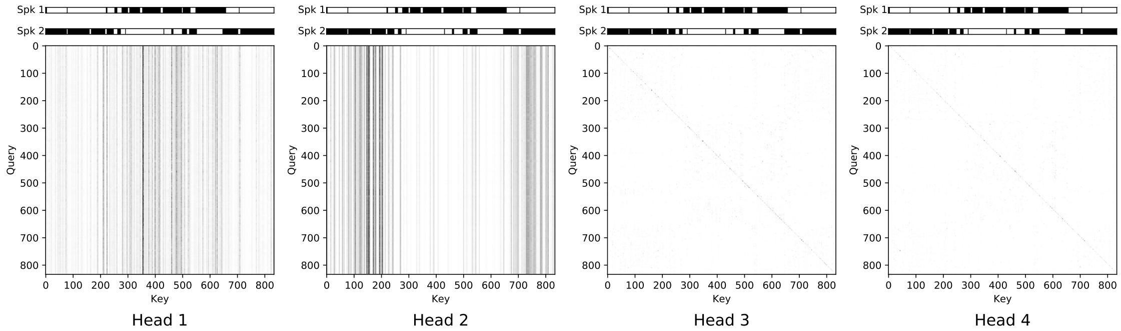


Figure 2-3: Attention weight matrices at the second encoder block. The input was the CALLHOME test set (recording id: iagk). The model was trained with the real training set, followed by domain adaptation. The top two rows show the reference speech activity of two speakers.

Table 2.7: DERs (%) with different number of heads. The models are trained with SimBeta2.

	Simulated			Real	
Num. heads	$\beta = 2$	$\beta = 3$	$\beta = 5$	CH	CSJ
2	12.60	13.42	16.12	16.49	26.05
4	7.91	8.51	9.51	13.66	**22.31**
8	**6.84**	**7.06**	**7.85**	13.44	23.58
16	7.19	7.52	7.88	**13.28**	24.35

arization, by calculating similarities between distant frames. Conversely, heads 3 and 4 present diagonal matrices, indicating their operation as local linear transforms. These heads likely perform speech/non-speech detectors, focusing on more immediate frame-to-frame changes rather than global patterns. The coexistence of these different types of attention heads within the EEND system enhances its overall effectiveness.

2.6.6 Effect of varying number of heads in self-attention blocks

The investigation presented in Sec. 2.6.5 revealed that different heads in the self-attention mechanism were representing different speakers. To further explore the significance of having multiple heads in the model, we conducted experiments with models having varying numbers of heads. The results are displayed in Table 2.7. It showed performance improvement with an increase in the number of heads. This trend indicates that the SA-EEND models were effectively trained to distinguish between speakers by leveraging the global speaker characteristics identified by the dif-

Table 2.8: DERs (%) for different numbers of encoder blocks and warm-up steps with/without residual connections. The models were trained with SimBeta2

Enc.	Warm.	Res.	Simulated			Real	
blocks	steps	con.	$\beta = 2$	$\beta = 3$	$\beta = 5$	CH	CSJ
2	25k	N	7.91	8.51	9.51	13.66	22.31
2	25k	Y	7.36	7.59	7.78	12.50	23.38
4	25k	Y	5.66	5.39	5.01	10.16	**20.39**
4	50k	Y	5.01	4.64	4.10	10.25	21.50
4	100k	Y	**4.56**	**4.50**	**3.85**	**9.54**	20.48
x-vector clustering			28.77	24.46	19.78	11.53	22.96

ferent heads. It implies that the minimum required number of heads in the model should be at least equal to the number of speakers present in the audio. Furthermore, the results suggest that having additional heads beyond this minimum threshold can further enhance the model's performance.

2.6.7 Effect of varying number of encoder blocks

In this subsection, the focus was on exploring additional encoder blocks with residual connections into the EEND system. The impact of varying the number of encoder blocks on DER is detailed in Table 2.8. The results reveal that as the number of encoder blocks increased, there was a significant improvement in the performance.

Specifically, the enhanced EEND system achieved a DER of 9.54% on the CALLHOME dataset, outperforming the x-vector clustering-based system, which recorded a DER of 11.53%. On the CSJ dataset, the EEND system's performance was also superior, a DER of 20.39%, compared to the 22.96% DER of the x-vector clustering-based system.

Moreover, the EEND system demonstrated superior performance on the simulated test set, with DERs ranging from 4.56% to 3.85%. In contrast, the x-vector clustering-based system showed significantly higher DERs, between 19.78% and 28.77%.

These results highlight the effectiveness of deeper model configurations in the EEND system, particularly when employing more encoder blocks with residual connections. The improvement in DERs across various datasets, especially in comparison to the x-vector clustering-based system, demonstrates the potential of EEND models in enhancing accuracy.

2.7 Conclusion

This chapter introduced a novel approach to speaker diarization known as end-to-end neural diarization (EEND). The key innovation of EEND lies in its end-to-end modeling, which allows the neural network to directly produce speaker label probabilities for multi-speaker audio inputs. A notable aspect of this method is the use of a permutation-free objective function, specifically designed to minimize diarization errors.

The EEND models were tested on both simulated speech mixtures and real conversational datasets. The results showed that EEND consistently outperformed the state-of-the-art x-vector clustering-based methods, and showed its capability to handle overlapping speech.

We explored the neural network architecture optimal for EEND. The results demonstrate the pivotal role of self-attention-based neural networks in achieving high performance. This architecture was found to be particularly effective due to its ability to capture both global speaker characteristics and local speech activity dynamics. This dual capability is a crucial factor in addressing the speaker diarization problem.

Further experiments with the neural network architecture revealed a correlation between the number of encoder blocks and the performance of the EEND model. It was observed that increasing the number of encoder blocks in the model led to better performance. This suggests that more complex network structures can more effectively enhance the overall performance in real test data.

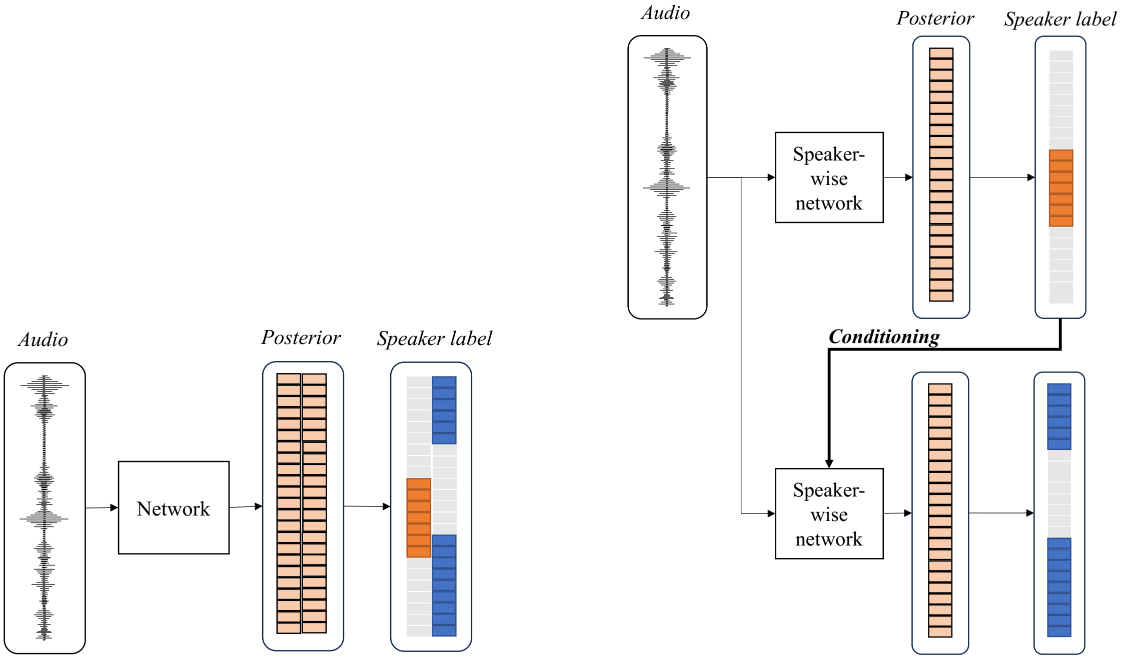
We found that the proposed method has two limitations. First, the number of speakers S should be fixed in advance since the neural network output is $S \times T$ matrix. The next chapter addresses the problem of the fixed number of speakers. Second, the conditional independence assumption introduced in Eq. 2.9 may cause a performance bottleneck. Chapter 4 proposes a method to relax the conditional independence assumption in EEND models.

3

Speaker-wise conditioning for end-to-end speaker diarization

3.1 Introduction

In this chapter, we introduce a conditioning method called “speaker-wise chain rule”. By generating a speaker label sequence iteratively with conditioning input of previously estimated sequences, the proposed model can utilize the existence of other speakers as prior knowledge for improved diarization performance. The proposed speaker-wise chain rule handles the variable number of speakers because it can iteratively produce a new speaker label sequence.



(a) Conventional EEND method

(b) Proposed SW-EEND method

Figure 3-1: System diagrams of the conventional EEND method and the proposed SW-EEND method.

3.2 Speaker-wise sequence decomposition

In the EEND model, the neural network output comprises multiple sequences of the fixed number of speakers, as shown in Fig. 3-1a. Instead, the proposed neural network produces a single sequence of one speaker, as shown in Fig. 3-1b. The network is used iteratively to produce a different sequence of the new speaker, considering the condition of previously estimated speakers. According to given multi-speaker audio, the model can handle a variable number of speakers by stopping the iteration when no speech activity is found from the output sequence.

This new model architecture is formulated as a speaker-wise sequence decomposition of the conventional EEND model. Recall Eq. 2.8 and we do not omit the condition $\boldsymbol{Y}_{<s,:}$ to approximate

the objective:

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{t=1}^{T} \prod_{s=1}^{S} p(\boldsymbol{Y}_{s,t}|\boldsymbol{X}, \boldsymbol{Y}_{<s,:}, \boldsymbol{Y}_{s,<t}) \tag{3.1}$$

$$\approx \prod_{s=1}^{S} p(\boldsymbol{Y}_{s,:}|\boldsymbol{X}, \boldsymbol{Y}_{<s,:}). \tag{3.2}$$

The proposed speaker-wise neural network f_{sw} estimates the posterior $p(\boldsymbol{Y}_{s,t}|\boldsymbol{X}, \boldsymbol{Y}_{<s,:})$:

$$\hat{\boldsymbol{Z}}_{s,:} = f_{\mathsf{sw}}(\boldsymbol{X}, \hat{\boldsymbol{Y}}_{<s,:}) \in [0,1]^T \tag{3.3}$$

With this model, each speaker's speech activity is sequentially decoded using previously estimated speech activities as conditions. This model is similar to the chain-rule-based autoregressive models such as sequence-to-sequence ASR models. Whereas the autoregressive models consider the conditions on the time axis, our "speaker-wise chain rule" considers the conditions on the speaker axis. We still use the independence assumption on the time axis of the current speaker, i.e., $\boldsymbol{Y}_{s,<t}$, the proposed model can see the whole sequence of previous speakers, which relaxes the independence on the time axis to some extent.

To generate a variable number of speakers, Eq. 3.3 is iteratively applied to the next speaker until *no speech activity* is found, i.e., $\hat{\boldsymbol{Y}}_{s,:}$ equals to the all-zero vector.

3.3 Network architecture for speaker-wise neural network

Since the proposed speaker-wise neural network generates the output for a variable number of times, the encoder-decoder type of the neural network is a suitable choice. For the encoder part, we use the same transformer encoders as Eq.2.20 in the conventional EEND model to obtain $\boldsymbol{E}^{(L)}$ after L encoder blocks. For the decoder part, the neural network output $\hat{\boldsymbol{Z}}_{s,:}$ for s-th iteration is computed as follows:

$$\boldsymbol{G}^{(s)} = \begin{bmatrix} \boldsymbol{E}^{(L)} \\ \mathsf{Linear}^{(\mathsf{sw1})}(\hat{\boldsymbol{Y}}_{s-1,:}) \end{bmatrix} \in \mathbb{R}^{2D \times T} \tag{3.4}$$

$$\boldsymbol{R}^{(s)}, \boldsymbol{K}^{(s)} = \mathsf{LSTM}(\boldsymbol{G}^{(s)}, \boldsymbol{R}^{(s-1)}, \boldsymbol{K}^{(s-1)}) \in \mathbb{R}^{D \times T}, \mathbb{R}^{D \times T} \tag{3.5}$$

$$\hat{\boldsymbol{Z}}_{s,:} = \sigma(\mathsf{Linear}^{(\mathsf{sw2})}(\boldsymbol{R}^{(s)}) \in [0,1]^T, \tag{3.6}$$

where LSTM is a uni-directional LSTM that maps $2D$-dimensional input vector to D-dimensional vector while keeping D-dimensional hidden state and memory cell for each time index. Note that the LSTM performs its recurrent behavior solely on the speaker axis and does not consider a sequence in the time axis. Finally, a linear projection with a sigmoid activation σ produces a T-dimensional vector as a neural network output.

The neural network accepts $\hat{\boldsymbol{Y}}_{s-1,:}$, a speech activity vector of the previous speaker index estimated at the previous decoder iteration (for the first iteration, we use the zero vector). However, the estimation error at the previous iteration hurts the performance at the next iteration. To reduce the error, we use the teacher-forcing (Williams and Zipser (1989)) technique, which boosts the performance by exploiting ground-truth labels. During training, Eq. 3.3 is replaced with as follows:

$$\hat{\boldsymbol{Z}}^{(\mathrm{TF})}_{s,:} = f^{(\mathsf{sw})}(\boldsymbol{X}, \boldsymbol{Y}_{<s,:}), \tag{3.7}$$

Here, $\boldsymbol{Y}_{<s,:}$ is a set of ground-truth speech activity of speaker index from 1 to $s-1$. However, a problem arises with training loss computation in Eq. 2.7. As described in Sec. 2.3, the order of speakers is determined during training. One cannot determine a speaker index $s-1$ before computing the permutation-free loss, which requires estimates of all speakers. To alleviate this problem, we examine two kinds of loss computation strategies, as follows.

Speaker-wise greedy loss

In each decoding iteration, the system selects the most suitable speaker index by aiming to reduce the binary cross-entropy loss among all speaker indices. Following this selection, the speech

Algorithm 2: Two-stage permutation-free loss

```
Input: $\boldsymbol{X}, \boldsymbol{Y}$
Output: $L_{\mathsf{PF2}}$
// First stage
$\hat{\boldsymbol{Z}}_{1,:} = f^{(\mathsf{sw})}(\boldsymbol{X}, \mathbf{0})$,                          // Eq. 3.3
$\hat{\boldsymbol{Y}}_{1,:} = [\mathbb{1}(\hat{\boldsymbol{Z}}_{1,t} > 0.5) \mid t = 1, \ldots, T]$      // Threshold
for $s = 2$ to $S$ do
    $\hat{\boldsymbol{Z}}_{s,:} = f^{(\mathsf{sw})}(\boldsymbol{X}, \hat{\boldsymbol{Y}}_{<s,:})$,                // Eq. 3.3
    $\hat{\boldsymbol{Y}}_{s,:} = [\mathbb{1}(\hat{\boldsymbol{Z}}_{s,t} > 0.5) \mid t = 1, \ldots, T]$  // Threshold
$\boldsymbol{P}^{*} = \arg\min_{\boldsymbol{P} \in \mathcal{P}(S)} \mathsf{BCE}(\boldsymbol{P}\boldsymbol{Y}, \hat{\boldsymbol{Z}})$   // Optimal order (Eq. 2.11)
$\hat{\boldsymbol{Z}}^{(\mathsf{TF})}_{1,:} = \hat{\boldsymbol{Z}}_{1,:}$
// Second stage
for $s = 2$ to $S$ do
    $\hat{\boldsymbol{Z}}^{(\mathsf{TF})}_{s,:} = f^{(\mathsf{sw})}(\boldsymbol{X}, [\boldsymbol{P}^{*}\boldsymbol{Y}]_{<s,:})$    // Eq. 3.7
// Loss with the optimal order
$L_{\mathsf{PF2}} = \mathsf{BCE}(\hat{\boldsymbol{Z}}^{(\mathsf{TF})}, \boldsymbol{P}^{*}\boldsymbol{Y})$
// Last output with no speech activity
$\hat{\boldsymbol{Z}}^{(\mathsf{last})} = f^{(\mathsf{sw})}(\boldsymbol{X}, [\boldsymbol{P}^{*}\boldsymbol{Y}; \mathbf{0}])$
$L_{\mathsf{PF2}} \mathrel{+}= \mathsf{BCE}(\hat{\boldsymbol{Z}}^{(\mathsf{last})}, \mathbf{0})$
```

activity of the selected speaker is used as input for the next decoding iteration.

Two-stage permutation-free loss

The computation of the two-stage permutation-free loss follows the steps outlined in Algorithm 2. In the first stage, the outputs of the neural network are calculated without using teacher-forcing, as defined in Eq. 3.3. Following this, the optimal speaker order is identified based on Eq. 2.11. The second stage then involves recalculating the neural network outputs, this time implementing teacher-forcing and utilizing the optimally determined speaker order. The final loss is derived by comparing these second-stage outputs with the ordered labels that were determined in the first stage. Note that this two-stage process is time-efficient, because backward computation is only necessary during the second stage.

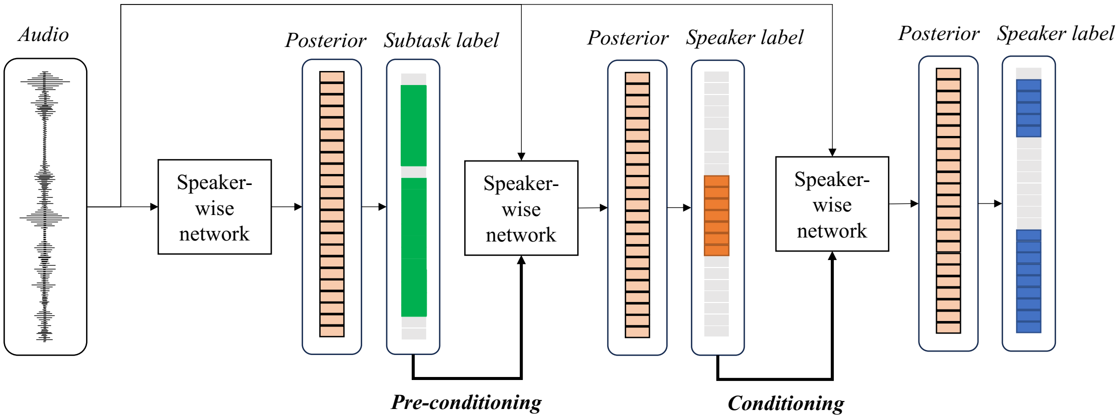


Figure 3-2: Overview of the subtask-first speaker-wise chain rule.

3.4 Preconditioning with subtask predictions

The speaker-wise chain rule can infuse any type of context information through the LSTM. In addition to previous speakers, we propose to precondition the network using speech activity detection and overlap detection subtasks. Fig. 3-2 depicts the overview of the proposed subtask-first system with the speaker-wise chain rule.

We introduce speech activity detection target $\boldsymbol{u} \in [0,1]^T$:

$$\boldsymbol{u}_t = \max(\boldsymbol{Y}_{:,t}) \quad (1 \le t \le T). \tag{3.8}$$

Similarly, we introduce overlap speech detection target $\boldsymbol{v} \in [0,1]^T$:

$$\boldsymbol{v}_t = \mathbb{1}(\sum_{s=1}^{S} \boldsymbol{Y}_{s,t} > 1) \quad (1 \le t \le T). \tag{3.9}$$

Then, we augment these latent variables and approximate the objective function with the most probable values:

$$\begin{aligned}\arg\max_{\boldsymbol{Y}} p(\boldsymbol{Y}|\boldsymbol{X}) &= \sum_{u}\sum_{v} p(\boldsymbol{Y}|\boldsymbol{X},\boldsymbol{v},\boldsymbol{u})p(\boldsymbol{v}|\boldsymbol{X},\boldsymbol{u})p(\boldsymbol{u}|\boldsymbol{X}) && (3.10)\\ &\approx \arg\max_{\boldsymbol{Y}} p(\boldsymbol{Y}|\boldsymbol{X},\hat{\boldsymbol{v}},\hat{\boldsymbol{u}}) && (3.11)\\ &\approx \arg\max_{\boldsymbol{Y}} \prod_{s=1}^{S} p(\boldsymbol{Y}_{s,:}|\boldsymbol{X},\boldsymbol{Y}_{<s,:},\hat{\boldsymbol{v}},\hat{\boldsymbol{u}}), && (3.12)\end{aligned}$$

where $\hat{\boldsymbol{u}} = \arg\max_{\boldsymbol{u}} p(\boldsymbol{u}|\boldsymbol{X})$, and $\hat{\boldsymbol{v}} = \arg\max_{\boldsymbol{v}} p(\boldsymbol{v}|\boldsymbol{X},\hat{\boldsymbol{u}})$. To compute $p(\boldsymbol{u}|\boldsymbol{X})$ and $p(\boldsymbol{v}|\boldsymbol{X},\hat{\boldsymbol{u}})$, we reuse the same function with Eqs. 3.4-3.6. $p(\boldsymbol{u}|\boldsymbol{X})$ is computed by replacing the conditional input $\hat{\boldsymbol{Y}}_{s-1,:}$ with the zero vector, and $p(\boldsymbol{v}|\boldsymbol{X},\hat{\boldsymbol{u}})$ is computed using the conditional input $\hat{\boldsymbol{u}}$. At the first speaker iteration, $p(\boldsymbol{Y}_{s,:}|\boldsymbol{X},\boldsymbol{Y}_{<s,:},\hat{\boldsymbol{v}},\hat{\boldsymbol{u}})$ is computed using the conditional input $\hat{\boldsymbol{v}}$.

3.5 Experimental setup

3.5.1 Data

We prepared simulated training and test sets for both two-speaker and variable-speaker audio mixtures. We also prepared real adaptation/test sets from CALLHOME (NIST (2000)). The statistics of the datasets are listed in Table 3.1. For the simulated dataset with a variable number of speakers (Simulated-vspk), the overlap ratio is adjusted to be similar among the different numbers of speakers. The simulation method was the same as in Chapter 2. For the CALLHOME-2spk, we use the same test set as Chapter 2. For the CALLHOME-vspk sets, we used the same test set of the Kaldi CALLHOME diarization v2 recipe (Povey et al. (2011)).

3.5.2 Model configuration

x-vector clustering-based (x-vector+AHC) model

We used the same clustering-based system as Chapter 2. The system uses AHC with the probabilistic linear discriminant analysis (PLDA) scoring scheme. The number of clusters was fixed to be two for the two-speaker experiments, while it was estimated using a PLDA score for the variable-speaker experiments.

Table 3.1: Statistics of training/adaptation/test sets.

	# speaker	# mixture	Avg. duration	Overlap ratio
Training sets				
Simulated-2spk	2	100,000	87.6	34.4
Simulated-vspk	1-4	100,000	128.1	30.0
Adaptation sets				
CALLHOME-2spk	2	155	74.0	14.0
CALLHOME-vspk	2-7	249	125.8	17.0
Test sets				
Simulated-vspk	1-4	2,500	128.1	30.0
CALLHOME-2spk	2	148	72.1	13.0
CALLHOME-vspk	2-6	250	123.2	16.7

Self-attention-based EEND (SA-EEND) and the proposed speaker-wise chain rule (SW-EEND)

We built self-attention-based EEND (SA-EEND) models and the proposed speaker-wise chain rule (SW-EEND) models, mostly based on the configuration described in Chapter 2. The configurations have small differences between the two-speaker and variable-speaker experiments, as follows.

For the two-speaker experiments, we used four encoder blocks with 256 attention units containing four heads. For the variable-speaker experiments, we used four encoder blocks with 384 attention units containing six heads. We used a subsampling ratio of 20 for variable-speaker experiments, which is twice larger than that of two-speaker experiments (10). Note that conventional EEND does not handle a variable number of speakers. We trained a fixed four-speaker model with zero-padded labels for three or fewer speakers in the training data.

In preparing the SA-EEND models and the proposed SW-EEND models, we adhered to the configuration in Chapter 2, with minor adjustments to suit the specific requirements of two-speaker and variable-speaker experiments.

For experiments involving two speakers, we used four encoder blocks, each equipped with 256 attention units and containing four heads. In contrast, for the variable-speaker experiments, we scaled up the configuration to have four encoder blocks with 384 attention units and six heads. Additionally, the subsampling ratio for the variable-speaker experiments was set to 20, which is

Table 3.2: DERs on two-speaker CALLHOME.

Model	Training	DER
x-vector+AHC	-	11.53
SA-EEND	PF	9.70
Proposed SW-EEND	PF	9.95
Proposed SW-EEND	Greedy+TF	9.01
Proposed SW-EEND	PF2+TF	**8.86**

double the ratio used in the two-speaker experiments (10).

Note that the conventional EEND approach does not accommodate a variable number of speakers. To address this limitation, we trained the EEND model for four speakers. In cases where the training data included three or fewer speakers, we implemented zero-padding for the labels to adapt to this fixed four-speaker model configuration.

3.6 Results

3.6.1 Experiments on fixed two-speaker models

Table 3.2 shows the DERs on the two-speaker CALLHOME. The proposed SW-EEND without teacher-forcing (TF) was slightly worse than conventional SA-EEND. With teacher-forcing, DER was significantly reduced and outperformed the conventional SA-EEND. For the loss computation strategy, two-stage permutation-free loss (PF2+TF) was slightly better than speaker-wise greedy loss (Greedy+TF). These results indicate that the conditional inference on partially estimated speaker labels helps improve the diarization performance.

3.6.2 Experiments on a variable number of speakers

Table 3.3 shows the DERs on the variable-speaker simulated test set. For SW-EEND without teacher-forcing, we observed no significant improvement from the conventional SA-EEND. With teacher-forcing, again, significant improvement was observed, particularly on a large number of speakers. The proposed two-stage permutation-free loss was significantly better than the speaker-wise greedy loss. The results indicate that the permutation-free loss is particularly important when

Table 3.3: DERs on variable-speaker simulated test set.

		Num. of speakers			
Model	Training	1	2	3	4
SA-EEND	PF	1.16	6.40	11.59	21.75
Proposed SW-EEND	PF	0.96	6.32	11.75	22.52
Proposed SW-EEND	Greedy+TF	0.85	5.25	10.56	18.28
Proposed SW-EEND	PF2+TF	**0.76**	**4.31**	**8.31**	**12.50**

Table 3.4: DERs on variable-speaker CALLHOME. Note that Greedy+TF adaptation model† was evaluated at 20^{th} epoch, because the adaptation was not stable after the epoch.

Model	Training	DER
x-vector+AHC	-	19.01
SA-EEND	PF	20.47
Proposed SW-EEND	PF	17.42
Proposed SW-EEND	Greedy+TF	18.07†
Proposed SW-EEND	PF2+TF	**15.75**

the number of speakers is large.

DERs on the variable-speaker CALLHOME are shown in Table 3.4. Even without teacher-forcing, the SW-EEND outperformed the conventional SA-EEND and x-vector+AHC methods. SW-EEND with teacher-forcing with the two-stage permutation-free loss significantly boosted performance.

3.6.3 Analysis on speaker counting

In the variable-speaker CALLHOME experiments, our analysis focused on the accuracy of speaker counting. The results of this analysis are presented in Table 3.5. It was observed that the proposed method demonstrated superior accuracy in counting speakers compared to the x-vector+AHC method. This improvement indicates the effectiveness of the proposed approach in identifying the number of speakers in a conversation. However, while the proposed method showed better performance in speaker counting, it still faced challenges in scenarios involving more than four speakers.

Table 3.5: Speaker counting results on variable-speaker CALLHOME. SW-EEND models were trained with PF2+TF.

(a) x-vector+AHC (Acc: 54.6%)

		Estimated				
		2	3	4	5	6
Reference	2	**84**	62	2	0	0
	3	18	**51**	5	0	0
	4	2	12	**6**	0	0
	5	0	4	1	**0**	0
	6	0	1	2	0	**0**

(b) SW-EEND (Acc: 74.8%)

	Estimated				
	2	3	4	5	6
2	**130**	17	1	0	0
3	17	**54**	3	0	0
4	4	13	**3**	0	0
5	0	3	2	**0**	0
6	0	2	1	0	**0**

3.6.4 Experiments with preconditioning for fixed two-speaker models

We verified the effectiveness of the preconditioning with SAD and overlapping speech detection (OD) on two-speaker mixtures as shown in Table 3.6.

First, we observed the performance of the models using SAD as the subtask. SAD-first SW-EEND achieved a 6.2% relative improvement over SW-EEND. The proposed SAD-first model achieved comparable SAD-level performance with clustering-based methods that have the SAD module trained separately. The results indicate that the subtask-first conditioning leverages the subsequent diarization task. Next, we discuss the performance of models using OD as the subtask. The OD-first SW-EEND outperformed SW-EEND, and showed slightly worse performance than SAD-first SW-EEND. FA errors in DER breakdown were significantly reduced compared with the SAD-first model. The results suggest that the overlap information helps prevent the over-generation of overlapping segments in the diarization results.

Furthermore, when both SAD and OD subtasks are used for preconditioning, we observed the best performance among the evaluated methods. SAD-OD-first SW-EEND showed 3.18% and 6.16% relative DER improvements over SAD-first and OD-first approaches, respectively. Similar to SAD-first SW-EEND, we observed a significant reduction in SAD errors. FA errors in DER breakdown were significantly reduced compared with SAD-first SW-EEND owing to the conditioning on OD. The results demonstrate that conditioning on the subtasks contributes to significant performance improvement for EEND models.

Table 3.6: Detailed DERs (%) evaluated on CALLHOME-2spk. DER is composed of Misses (MI), False alarms (FA), and Confusion errors (CF). The SD errors are composed of Misses (MI) and False alarms (FA) errors.

		DER breakdown			SAD	
Method	DER	MI	FA	CF	MI	FA
Clustering-based						
i-vector	12.10	7.74	0.54	3.82	1.4	0.5
x-vector	11.53	7.74	0.54	3.25	1.4	0.5
EEND-based						
SA-EEND	10.32	5.66	3.25	1.40	3.0	0.5
SW-EEND	9.39	4.96	2.73	1.70	2.2	0.4
Subtask-first SW-EEND						
SAD-first	8.81	4.11	2.96	1.74	1.4	0.8
OD-first	9.09	5.25	1.86	1.98	2.3	0.4
SAD-OD-first	**8.53**	4.22	2.33	1.98	1.6	0.7

Table 3.7: DERs (%) on CALLHOME-vspk.

Method	DER
Clustering-based	
x-vector	19.01
EEND-based	
SW-EEND	15.57
SAD-first SW-EEND	15.36
OD-first SW-EEND	16.37
SAD-OD-first SW-EEND	**15.32**

3.6.5 Experiments with preconditioning for variable number of speakers

We also experimented with the variable number of speakers for CALLHOME test set. For this particular experiment, we randomly disabled the SAD subtask losses at the frame level with a ratio of 0.7 and multiplied the losses by 0.1, because our preliminary experiments showed overfitting to the SAD subtask. Furthermore, we used the outputs of the SAD subtask network to determine non-speech frames regardless of the diarization outputs.

Table 3.7 shows the DERs. The SAD-first SW-EEND achieved 15.36%, which corresponds to 19.2% and 1.35% relative DER improvements over the conventional x-vector clustering method

Table 3.8: Detailed DERs (%) associated with each number of speaker on CALLHOME-vspk.

	Num. of speakers				
Model	2	3	4	5	6
SW-EEND	9.0	14.4	**19.1**	34.6	39.5
SAD-OD-first SW-EEND	**8.0**	**13.5**	23.1	**30.0**	**35.2**

and SW-EEND without preconditioning. SAD-OD-first SW-EEND reached 15.32% DER, which is the best performance among the evaluated methods. The results indicate that the proposed SAD-first approach is also effective in a variable-speaker setting. However, OD-first models did not outperform the conventional SW-EEND, although they outperformed the conventional x-vector clustering method. The results suggest that we need a careful training strategy, such as scheduled learning, since OD is more difficult than SAD.

Table 3.8 shows the detailed DER breakdown of the SAD-OD-first SW-EEND and the conventional SW-EEND without preconditioning in CALLHOME-vspk (Table 3.7) for each number of speakers. SAD-OD-first SW-EEND is better than the SW-EEND without preconditioning in most cases except for the four-speaker case. The results indicate that the preconditioning is robust to the large number of speakers.

Finally, we compared the proposed method with other systems as shown in Table 3.9. In this comparison, we only evaluated single-speaker regions, i.e., ignored the errors in overlapped and non-speech segments, as with the traditional evaluation protocol. For this comparison, we used oracle SAD and OD information as preconditions, and filtered out non-speech frames of the estimated diarization result using the oracle SAD information. Although our proposed method could not achieve state-of-the-art performance, it outperformed the system of Zhang et al. (2019). The results suggest that the proposed preconditioned model can use external subtask information.

3.7 Conclusion

In this chapter, we proposed the speaker-wise chain rule, an end-to-end speaker diarization conditioned on previous speaker labels and speech activity detection subtasks. The experiments demonstrated that the proposed speaker-wise chain rule outperforms the SA-EEND thanks to the label

Table 3.9: DERs (%) evaluated on CALLHOME-vspk with oracle SAD information. Overlapping segments were omitted from the DER computation. The evaluation set for the proposed method was different from that for other systems. We used a random subset of CALLHOME, whereas other systems used the whole CALLHOME evaluation set.

Method	DER
McCree et al. (2019)	7.1
SAD-OD-first SW-EEND	7.4
Zhang et al. (2019)	7.6

dependency. We also demonstrated the SAD and OD subtasks further improve performance. In particular, the subtask-first model exhibits robustness in the large number of speakers.

We found that conditioning on previous speaker labels improved performance, indicating that the estimated speaker labels can infuse label dependency that relaxes the conditional independence assumption in the original EEND model. The next chapter focuses on effectively utilizing estimated speaker labels to mitigate the performance bottleneck caused by the conditional independence assumption in EEND models.

4

Self-conditioning via intermediate predictions

4.1 Introduction

As described in Chapter 2, EEND assumes conditional independence between speaker labels. The assumption blocks the utilization of given speaker labels as context information. In this chapter, we focus on relaxing the conditional independence assumption.

We propose a new conditioning scheme that utilizes “intermediate predictions”. The speaker labels produced in the middle of the neural network are fed back to the higher-layer network. The proposed method, called “self-conditioning”, achieves iterative refinement of speaker labels through multiple intermediate predictions.

To achieve state-of-the-art performance, we investigate the use of encoder-decoder-based attractor (EDA; Horiguchi et al. (2020, 2022)) and find the performance bottleneck of EDA in the autoregressive calculation module when used with the proposed self-conditioning. Therefore, we propose a different method to calculate attractors called "non-autoregressive attractors", which produces the attractors simultaneously in a non-autoregressive manner.

4.2 Self-conditioning via intermediate predictions

The proposed self-conditioning mechanism is a latent variable model that introduces label dependency into the EEND model. The primary objective, as outlined in Eq. 2.7, is to identify the most probable speaker labels $\boldsymbol{Y}$ given audio features $\boldsymbol{X}$:

$$\hat{\boldsymbol{Y}} = \arg\max_{\boldsymbol{Y} \in \mathcal{Y}} p(\boldsymbol{Y}|\boldsymbol{X}), \tag{4.1}$$

where $\mathcal{Y}$ is a set of all possible speaker labels. The proposed method introduces a latent variable $\boldsymbol{Y}^{(L-1)}$ for the intermediate speaker label prediction at the $(L-1)$-th layer, where L is the number of encoder layers. The introduced variable is the marginalized out as:

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \sum_{\boldsymbol{Y}^{(L-1)} \in \mathcal{Y}} p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{Y}^{(L-1)}) p(\boldsymbol{Y}^{(L-1)}|\boldsymbol{X}). \tag{4.2}$$

Instead of the full marginalization, we approximate it by using the most probable value of the intermediate prediction $\hat{\boldsymbol{Y}}^{(L-1)}$:

$$p(\boldsymbol{Y}|\boldsymbol{X}) \approx p(\boldsymbol{Y}|\boldsymbol{X}, \hat{\boldsymbol{Y}}^{(L-1)}) p(\hat{\boldsymbol{Y}}^{(L-1)}|\boldsymbol{X}) \tag{4.3}$$

This process is extended to introduce another latent variable $\boldsymbol{Y}^{(l-1)}$ for each layer l from $L-1$ to 2, leading to a decomposition as follows:

$$p(\boldsymbol{Y}|\boldsymbol{X}) \approx p(\boldsymbol{Y}|\boldsymbol{X}, \hat{\boldsymbol{Y}}^{(L-1)}) \left(\prod_{l=2}^{L-1} p(\hat{\boldsymbol{Y}}^{(l)}|\boldsymbol{X}, \hat{\boldsymbol{Y}}^{(l-1)}) \right) p(\hat{\boldsymbol{Y}}^{(1)}|\boldsymbol{X}). \tag{4.4}$$

The decomposed posteriors, i.e., $p(\boldsymbol{Y}|\boldsymbol{X}, \hat{\boldsymbol{Y}}^{(L-1)})$, $p(\hat{\boldsymbol{Y}}^{(l)}|\boldsymbol{X}, \hat{\boldsymbol{Y}}^{(l-1)})$, and $p(\hat{\boldsymbol{Y}}^{(1)}|\boldsymbol{X})$, are estimated using the intermediate prediction and self-conditioning functions in the proposed neural network. The neural network architecture, implemented in a four-layer self-attention-based EEND

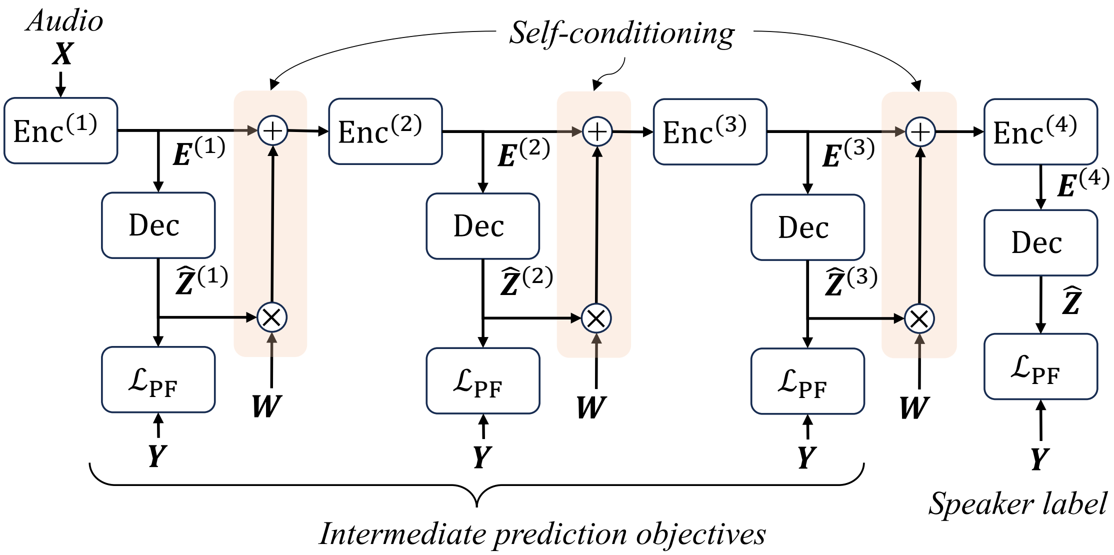


Figure 4-1: Overview of proposed method with four-layer self-attention-based EEND model. Shared decoder Dec accepts outputs from each layer $\mathsf{Enc}^{(l)}(l = 1, 2, 3)$. The intermediate prediction $\hat{\boldsymbol{Z}}^{(l)}$ is optimized with the same permutation-free binary cross entropy objective $\mathcal{L}_{\mathsf{PF}}$. The intermediate prediction is fed back to the subsequent encoder layer through a shared linear projection matrix $\boldsymbol{W}$.

model, is illustrated in Fig. 4-1. In the context of the proposed model, we refer to the generalized function in Eq.2.21 as a "decoder" denoted by Dec. The function maps the encoder layer output $\boldsymbol{E}^{(L)}$ to the speaker label posterior $\hat{\boldsymbol{Z}}$:

$$\hat{\boldsymbol{Z}} = \mathsf{Dec}(\boldsymbol{E}^{(L)}). \tag{4.5}$$

4.2.1 Intermediate prediction

The intermediate prediction $p(\hat{\mathbf{Y}}^{(l)}|\boldsymbol{X})$ is estimated by feeding the intermediate encoder layer output to the decoder:

$$\hat{\boldsymbol{Z}}^{(l)} = \mathsf{Dec}(\boldsymbol{E}^{(l)}) \qquad (1 \leq l \leq L-1). \tag{4.6}$$

Note that the decoder parameter is shared among all the layers. For optimizing the intermediate predictions, the same training objective as Eq. 2.11 is applied to the intermediates:

$$\mathcal{L}_{\mathsf{inter}} = \mathcal{L}_{\mathsf{PF}}(\boldsymbol{Y}, \hat{\boldsymbol{Z}}) + \frac{1}{L-1}\sum_{l=1}^{L-1} \mathcal{L}_{\mathsf{PF}}(\boldsymbol{Y}, \hat{\boldsymbol{Z}}^{(l)}). \tag{4.7}$$

Here, we mix the main and auxiliary losses without tuning the weight in this work. The higher encoders are indirectly conditioned on the intermediate speaker labels by optimizing the intermediate speaker labels.

4.2.2 Self-conditioning

To estimate $p(\hat{\boldsymbol{Y}}^{(l)}|\boldsymbol{X}, \hat{\boldsymbol{Y}}^{(l-1)})$, we need to augment the conditioning input $\hat{\boldsymbol{Y}}^{(l-1)}$ to the intermediate prediction $p(\hat{\boldsymbol{Y}}^{(l)}|\boldsymbol{X})$ described in the previous section. To this end, we use $\hat{\boldsymbol{Z}}^{(l-1)}$ as the conditioning input [1]:

$$\boldsymbol{E}^{(l)} = \mathsf{Enc}^{(l)}(\mathsf{Condition}(\boldsymbol{E}^{(l-1)})), \tag{4.8}$$

$$\mathsf{Condition}(\boldsymbol{E}^{(l-1)}) = \boldsymbol{E}^{(l-1)} + \boldsymbol{W}\hat{\boldsymbol{Z}}^{(l-1)}, \tag{4.9}$$

where $\boldsymbol{W} \in \mathbb{R}^{D \times C}$ is a linear layer that projects the intermediate predictions back to the encoder's dimension. In this way, the intermediate speaker label information is encoded in the frame-level embedding space, and the higher layer encoder can utilize the whole sequence of encoded speaker label information thanks to the self-attention mechanism in the Transformer encoder. Note that $\boldsymbol{W}$ is shared among all the intermediate layers.

[1] We could use $\hat{\boldsymbol{Y}}^{(l-1)}$. However, our preliminary experiment did not show a significant difference. Therefore we use $\hat{\boldsymbol{Z}}^{(l-1)}$ to reduce computation.

4.3 Non-autoregressive Attractor

Instead of using a linear layer in Eq. 2.21, EDA (Horiguchi et al. (2020)) generates speaker-wise attractors $\boldsymbol{A} = [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_C] \in \mathbb{R}^{D \times C}$:

$$\boldsymbol{A} = \mathsf{EDA}(\boldsymbol{E}^{(L)}). \tag{4.10}$$

The function EDA is composed of two unidirectional LSTM layers:

$$(\boldsymbol{h}_t, \boldsymbol{c}_t) = \mathsf{LSTM}_1(\boldsymbol{h}_{t-1}, \boldsymbol{c}_{t-1}, \boldsymbol{E}^{(L)}_{:,t}) \quad (1 \leq t \leq T), \tag{4.11}$$

$$(\boldsymbol{a}_c, \boldsymbol{d}_c) = \mathsf{LSTM}_2(\boldsymbol{a}_{c-1}, \boldsymbol{d}_{c-1}, \boldsymbol{0}) \quad (1 \leq c \leq C), \tag{4.12}$$

where $\boldsymbol{h}_t \in \mathbb{R}^D$ is a hidden state, $\boldsymbol{c}_t \in \mathbb{R}^D$ is a cell state of LSTM, $\boldsymbol{E}^{(L)}_{:,t}$ is a column t of the matrix $\boldsymbol{E}^{(L)}$. Here, $\mathsf{LSTM}_1()$ consumes $\boldsymbol{E}^{(L)}$ through T consecutive steps in an autoregressive manner. Then, $\mathsf{LSTM}_2()$ produces C attractors sequentially. Then, the decoder estimates the speaker label by comparing the embedding sequence $\boldsymbol{E}^{(L)}$ with the speaker-wise attractors $\boldsymbol{A}$:

$$\hat{\boldsymbol{Z}} = \sigma(\boldsymbol{A}^\top \boldsymbol{E}^{(L)}). \tag{4.13}$$

Though EDA contains an autoregressive submodule (Eq. 4.11), the posteriors $\hat{\boldsymbol{Z}}$ are generated in parallel with the given attractors and the encoder outputs.

When applying the self-conditioning to the EDA model, training throughput is down due to the autoregressive submodule in Eq. 4.11. To mitigate the issue, we propose a fully non-autoregressive architecture for speaker-wise attractor extraction. A system diagram of the proposed non-autoregressive attractor is shown in Fig. 4-2. Instead of using LSTMs, we employ a multi-head cross-attention module to extract attractors:

$$\boldsymbol{A} = \mathsf{MHA}(\boldsymbol{Q}, \boldsymbol{E}^{(L)}, \boldsymbol{E}^{(L)}), \tag{4.14}$$

where $\mathsf{MHA}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V})$ is a multi-head attention function used in Transformer decoders Vaswani et al. (2017). Here, query vectors $\boldsymbol{Q} \in \mathbb{R}^{C \times D}$ are prepared as learnable parameters, and keys/val-

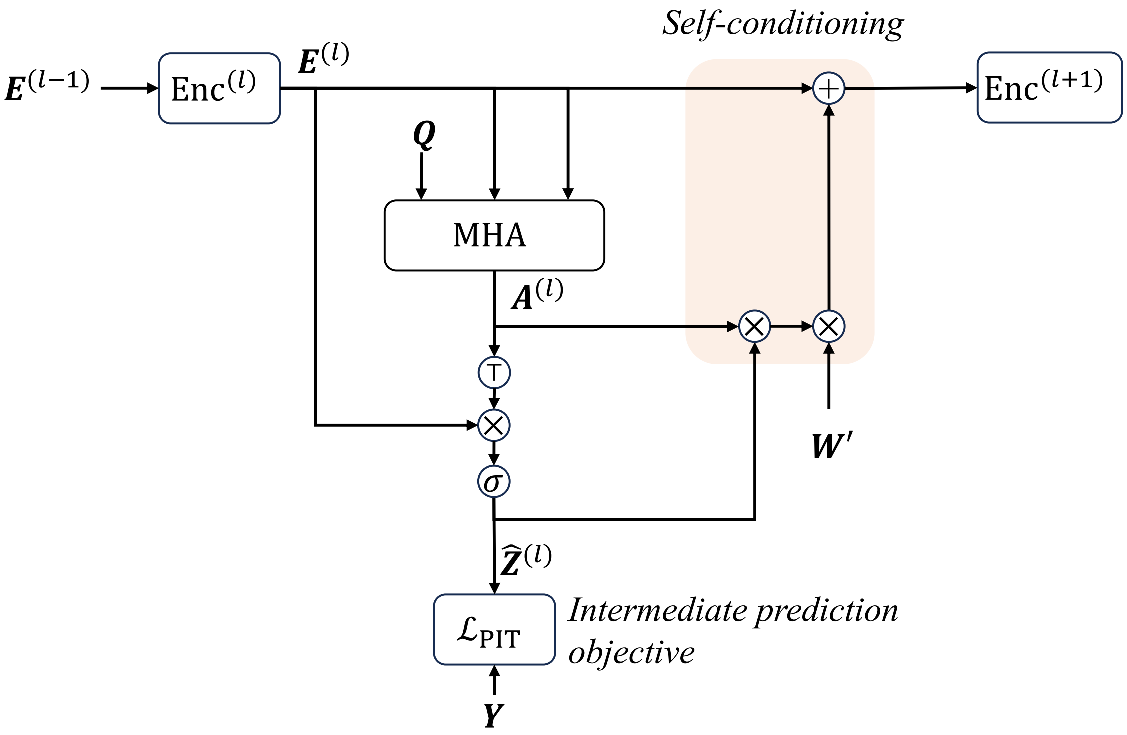


Figure 4-2: Schematic diagram of non-autoregressive attractor with self-conditioning. Multi-head self-attention MHA processes encoder output $\boldsymbol{E}^{(l)}$ and trainable query matrix $\boldsymbol{Q}$ to produce intermediate attractor $\boldsymbol{A}^{(l)}$. Intermediate prediction $\hat{\boldsymbol{Z}}^{(l)}$ is calculated using transpose ($\top$), matrix multiplication ($\times$) and sigmoid (σ) operations. Self-conditioning processes the intermediate attractor and the intermediate prediction with a shared linear matrix $\boldsymbol{W}'$ to condition the subsequent encoder layer.

ues are the encoder output $\boldsymbol{E}^{(L)}$.

To generate intermediate labels with the non-autoregressive attractor, we first extract intermediate attractors:

$$\boldsymbol{A}^{(l)} = \mathsf{MHA}(\boldsymbol{Q}, \boldsymbol{E}^{(l)}, \boldsymbol{E}^{(l)}) \quad (1 \leq l \leq L-1). \tag{4.15}$$

Then, we get the intermediate labels using the same function as Eq. 4.13:

$$\hat{\boldsymbol{Z}}^{(l)} = \sigma(\boldsymbol{A}^{(l)\top}\boldsymbol{E}^{(l)}) \quad (1 \leq l \leq L-1). \tag{4.16}$$

Table 4.1: Training, adaptation, and test data statistics.

	# mixtures	Average duration (s)	Overlap ratio (%)
Training	24,179	368.8	8.1
Adaptation	155	74.0	14.0
Test	148	72.1	13.0

Self-conditioning with the non-autoregressive attractor is a bit customized compared with Eq. 4.9. We utilize the intermediate attractors themselves by computing the weighted sum of them according to the intermediate label posteriors:

$$\mathsf{Condition}(\boldsymbol{E}^{(l)}) = \boldsymbol{E}_l + \boldsymbol{W}'\boldsymbol{A}^{(l)}\hat{\boldsymbol{Z}}^{(l)}, \tag{4.17}$$

where $\boldsymbol{W}' \in \mathbb{R}^{D \times D}$ is learnable to control the weights of intermediate predictions. The learnable parameters for $\boldsymbol{W}'$ are shared among $L - 1$ layers.

4.4 Experimental setup

4.4.1 Test Data

We conducted diarization experiments on the CALLHOME two-speaker dataset (NIST (2000)) as in Chapter 2.

4.4.2 Training Data

In this chapter, we exploit a similar but different mixture simulation method from the one used in previous chapters. This method, outlined in Landini et al. (2022a), creates conversation-style simulated mixtures that reflect the statistical properties of the adaptation data. Additionally, the algorithm mixes randomly selected noise from the MUSAN corpus (Snyder et al. (2015)).

The statistics for the training, adaptation, and test datasets are shown in Table 4.1. The source audio samples used for this simulation process remain consistent with those described in Chapter 2. These sources include the Switchboard-2 dataset (Phases I, II, III), Switchboard Cellular (Parts 1 and 2), and the NIST Speaker Recognition Evaluations from 2004, 2005, 2006, and 2008.

4.4.3 Model hyperparameters

To prepare the SA-EEND and EDA models as our baselines, the configuration in the EDA paper (Horiguchi et al. (2020)) was closely followed.

The audio features used were 23-dimensional log-Mel-filterbanks. These were extracted using a window size of 25 msec and a hop size of 10 msec. The final audio features were obtained by concatenating 15 consecutive frames, resulting in 345-dimensional features. These features were then subsampled at intervals of 100 msec.

During the training process, the length of the audio input was limited to 50 seconds. The Transformer encoders were configured with 256 attention units and four attention heads. The models had either four or eight encoder blocks.

Each model was trained for 100 epochs, with a batch size set to 32. The Adam optimizer was used alongside the Noam learning rate scheduler. Gradient clipping was also employed with a norm threshold of 5.0. The number of warmup steps was set to 200,000.

In the adaptation stage, the learning rate was fixed at 10^{-5}, and the models were run for an additional 100 epochs. After both the training and adaptation stages, the model checkpoints from the last ten epochs were averaged to create the final model.

4.4.4 Metrics

For evaluating the performance of the diarization models, DERs were calculated with a tolerance collar of 250 msec. Errors were counted not only in speech segments but also in non-speech and overlapped segments. Additionally, as part of the performance evaluation, the training throughput was recorded. This measurement was expressed in terms of the number of batches processed per second. The hardware used for this experiment was a Tesla V100 32G GPU,

4.5 Results

4.5.1 Performance improvement with intermediate prediction and self-conditioning

Table 4.2 presents the DERs for the four-layer models on the CALLHOME two-speaker test set. The results clearly show that the implementation of intermediate predictions led to a reduction in

Table 4.2: Diarization error rates (%) on CALLHOME two-speaker test for the four-layer models, showing effect of intermediate prediction, self-conditioning, and non-autoregressive attractor.

Method	w/o adaptation	w/ adaptation
SA-EEND	9.38	8.69
+ Intermediate pred.	9.31	8.34
+ Self-conditioning	8.64	7.80
Non-autoregressive Attractor	9.16	8.96
+ Intermediate pred.	8.37	8.37
+ Self-conditioning	**8.28**	**7.05**

DERs across all conditions. This outcome indicates that the technique of indirect conditioning, achieved by incorporating intermediate predictions at lower layers, significantly enhances the final speaker diarization performance.

Furthermore, the introduction of self-conditioning into the models resulted in an even greater reduction in DERs. This improvement was consistent in scenarios both with and without adaptation, suggesting the robustness of self-conditioning in various contexts. Notably, self-conditioning achieved the best performance in both cases. The results demonstrate that explicitly conditioning the higher layers of the model with intermediate predictions is more effective than merely adding intermediate predictions. The performance improvements were particularly evident during the adaptation stage. Self-conditioning seems to have facilitated faster convergence with smaller datasets, a feature highly desirable in domain adaptation scenarios. This capability suggests that self-conditioning not only improves the model's accuracy but also enhances its efficiency and adaptability, making it well-suited for applications where data availability is limited or where rapid model adaptation is required.

4.5.2 Training efficiency improvement with non-autoregressive attractor

Table 4.3 provides a comparison of attractor-based models in terms of training throughput and the number of parameters. The proposed non-autoregressive attractor model demonstrated higher training throughput and required fewer parameters compared to the EDA model. This indicates a more efficient use of computational resources and a potentially more streamlined model architecture.

Table 4.3: Training throughput (#batches/sec) and the number of parameters of attractor-based models. All models contain four-layer Transformer encoders. DERs were obtained with adaptation.

Method	Throughput	# params	DER
EDA	3.30	6.4M	7.74
+ Intermediate pred.	1.20	6.4M	8.11
+ Self-conditioning	1.03	6.5M	9.13
Non-autoregressive Attractor	4.15	5.6M	8.96
+ Intermediate pred.	3.88	5.6M	8.37
+ Self-conditioning	3.79	5.7M	**7.05**

Integrating the proposed intermediate prediction and self-conditioning into the conventional EDA model was unsuccessful. One of the primary reasons for this was identified as the LSTM encoder used in EDA, which significantly reduced the training throughput to about one-third.

In contrast to the EDA model, the proposed non-autoregressive attractor exhibited better training efficiency. This suggests that the modifications in the model design, specifically the move away from autoregressive components, contributed to its training efficiency. Even with the addition of self-conditioning, the proposed method maintained a faster training speed compared to the EDA model. Moreover, it achieved lower DERs than the numbers with EDA. This aspect highlights the effectiveness of the self-conditioning in enhancing both accuracy and efficiency.

4.5.3 Effect of Layer-normalization with non-autoregressive attractor

The experiment comparing different Transformer configurations, specifically pre-Layer Normalization (pre-LN) versus post-Layer Normalization (post-LN), provided insightful results, as detailed in Table 4.4. This comparison is grounded in the broader context of understanding Transformer architectures, as discussed in Liu et al. (2020).

In Fujita et al. (2023), it was observed that the performance of the non-autoregressive attractor in four-layer models was inferior to that of the EDA model. This performance gap was primarily attributed to the use of the pre-LN architecture in conjunction with the non-autoregressive attractor, whereas the conventional SA-EEND and EDA models employed a post-LN architecture.

The results in Table 4.4 clearly indicate that, within the experimental setup used, the post-LN

Table 4.4: Diarization error rates (%) on CALLHOME two-speaker test for layer-normalization effect on four-layer non-autoregressive attractor models.

Method	w/o adaptation	w/ adaptation
Pre-LN	11.15	11.34
+ Intermediate pred.	9.33	8.23
+ Self-conditioning	8.81	7.77
Post-LN	9.16	8.96
+ Intermediate pred.	8.37	8.37
+ Self-conditioning	**8.28**	**7.05**
EDA (Post-LN)	8.66	7.74

configuration outperformed the pre-LN setup. The proposed non-autoregressive attractors demonstrated a better fit with the post-LN Transformer configuration. Our hypothesis from the observations is that insufficient normalization of encoder outputs in the pre-LN setup adversely affects the performance of non-autoregressive attractor computation. This implies that the quality and effectiveness of layer normalization are critical for the optimal functioning of non-autoregressive attractors in Transformer-based models.

4.5.4 Layer-by-layer progressive refinement

In Fig. 4-3, the DERs are displayed layer-by-layer, showing the impact of intermediate predictions within the model. The figure clearly illustrates a progressive reduction in diarization errors as the input passes through each successive layer of the network. This trend demonstrates the effectiveness of self-conditioning in optimizing speaker label prediction at lower layers, which subsequently leads to improved overall performance.

An interesting observation from the results is that the performance at layer 7 was marginally better than at the final layer (layer 8). This phenomenon can be attributed to what is known as "overthinking" in deep networks. In some cases, as explained in Kaya et al. (2019) and Berrebbi et al. (2023), deep networks may reach the correct prediction at an intermediate layer but then diverge slightly from this optimal point in subsequent layers. A common approach to mitigating the overthinking effect is to implement a confidence-based early exit. However, in this particular experiment, a simpler strategy of early exit was employed, where the process was terminated at

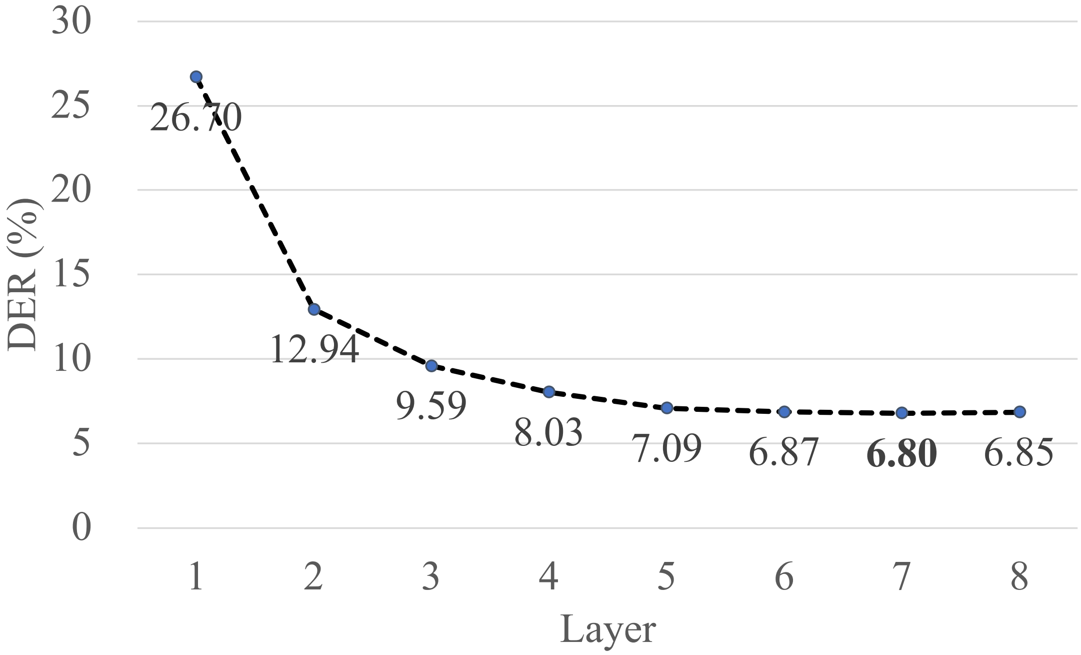


Figure 4-3: Diarization error rates with intermediate predictions layer-by-layer. Results were obtained using non-autoregressive attractor-based model with eight-layer Post-LN Transformer encoders.

layer 7, taking advantage of the optimal predictions observed at this layer.

4.5.5 Comparison with other existing models

Table 4.5 presents a performance comparison between the proposed method and existing state-of-the-art models in speaker diarization. The WavLM paper (Chen et al. (2022)) reports the best results on the CALLHOME diarization task. WavLM Base+ and WavLM Large are models that benefit from self-supervised learning (SSL) and have been trained on a large dataset comprising 94K hours. These models demonstrate that performance can be significantly enhanced with extensive training data, a strategy not employed in our models. Despite having far fewer parameters, the proposed method achieves performance levels comparable to those of the SSL models. This highlights the efficiency and effectiveness of the proposed method's architecture.

EEND-VC (Kinoshita et al. (2021b)), EDA & clustering (Horiguchi et al. (2021)), and WavLM & EEND-VC (Chen et al. (2022)), incorporate a clustering module, which requires independent

Table 4.5: Diarization error rates (%) on CALLHOME two-speaker test with adaptation for comparison with existing state-of-the-art models. Proposed models were trained with non-autoregressive attractors and self-conditioning.

Method	# params	DER
EEND-only		
SA-EEND ♮	5.4M	8.69
EDA ♮	6.4M	7.74
Proposed 4-layer	5.7M	7.05
Proposed 8-layer	10.9M	**6.80**
EEND & clustering		
EEND-VC (Kinoshita et al. (2021b)) †	8.1M	7.96
EDA & clustering (Horiguchi et al. (2021)) †	6.4M	7.11
WavLM Base+ & EEND-VC †	94.7M + 8.1M	6.99
WavLM Large & EEND-VC †	316.6M + 8.1M	**6.46**

♮ indicates that the numbers are our reproduced results.
† indicates that the numbers are from Chen et al. (2022).

fine-tuning of hyperparameters, to achieve performance improvement. In contrast, the proposed method, which does not rely on a clustering module, outperforms most of these methods. Note that the DERs reported in the referenced studies (Kinoshita et al. (2021b); Horiguchi et al. (2021); Chen et al. (2022)) are based on datasets with a variable number of speakers, ranging from two to six. Their training and adaptation data also include recordings with three or more speakers, whereas our models were trained with two-speaker data only. These studies have shown better results in their two-speaker subsets compared to models trained solely on two-speaker data. The training data preparation methods and the use of SSL models in these referenced studies suggest that incorporating similar strategies could further enhance the performance of the proposed method.

4.6 Conclusion

In this chapter, we proposed an end-to-end neural speaker diarization model that incorporates self-conditioning through intermediate predictions. This approach effectively integrates speaker label dependency into existing non-autoregressive EEND models by using intermediate speaker label predictions. The effectiveness of the proposed method was validated through experiments conducted using the CALLHOME two-speaker dataset. These experiments demonstrated that the self-conditioning significantly enhances diarization performance. In exploring efficient architectures for EEND with self-conditioning, it was found that the proposed non-autoregressive attractor model not only achieved better performance but also required fewer parameters compared to existing EEND models.

5

Conclusions

This chapter summarizes the contributions of the dissertation. Then, we explore the future directions by introducing recent studies based on EEND.

5.1 Contributions

This dissertation addressed speaker diarization, which is essential in processing multi-talker audio to understand human-to-human communication. We first reviewed traditional and recent speaker diarization systems and revealed that the traditional system is not optimal in two aspects: 1) a complex pipeline of independently optimized modules and 2) limited capability of handling overlapping speech. The review also introduced some recent systems that employ fully-supervised methods, which utilize speaker labels in conversations. However, most systems, except for EEND, still have complex pipelines and are not end-to-end optimal.

We proposed EEND, the first end-to-end optimal system with a single neural network that deals with full diarization problems, including overlapping speech detection. In Chapter 2, we developed the new formulations of both traditional and EEND systems based on probabilistic modeling. The formulations revealed that EEND optimizes to generate a "multi-sequence" target, whereas traditional systems generate a "single-sequence" target. To optimize the multi-sequence target, we found that the permutation-free loss function is essential. Through the experiments of the CALLHOME two-speaker dataset, we demonstrated that EEND significantly outperforms the traditional x-vector clustering system. To achieve sufficient accuracy on the real dataset, self-attention architecture in SA-EEND plays a key role. Experiments with attention visualization show the clear advantage of the SA-EEND.

Chapter 3 addressed the two limitations of EEND: 1) the conditional independence assumption between speaker labels and 2) the fixed number of speakers. We proposed the "speaker-wise chain rule," enabling conditional inference on previous speaker labels. The formulation of the speaker-wise chain rule is based on the decomposition of the multi-sequence target to speaker-wise sequences. The speaker-wise chain network produces speaker labels one by one; it can iteratively generate a variable number of speakers. Experimental results on CALLHOME with two speakers showed that the speaker-wise chain rule outperformed SA-EEND. Furthermore, the experiments with the variable number of speakers demonstrated better speaker counting accuracy than the x-vector clustering system. This chapter also experimented with another conditioning method based on SAD and overlapping speech detection subtasks. We extended the speaker-wise chain rule to accept pre-conditioning input from the different tasks. The experiments showed that the subtask-first model improves the performance of the speaker-wise chain rule.

In Chapter 4, our proposed "self-conditioning" has shown significant performance improvement on the CALLHOME two-speaker dataset. The proposed method utilizes intermediate predictions at the middle layers of SA-EEND and EDA. Then, it performs conditional inference on intermediate speaker labels. To compare with the state-of-the-art method, i.e., EDA, we implemented self-conditioning on top of the EDA model. However, we found the bottleneck of EDA when used with self-conditioning. Therefore, we proposed the non-autoregressive attractor as a variant of EDA, which replaces the autoregressive computation part in EDA with the non-

Table 5.1: Comparison of speaker diarization methods with output target and functions to be optimized. Traditional clustering outputs a single sequence $\boldsymbol{y}$ by optimizing three functions. EEND outputs a two-dimensional matrix $\boldsymbol{Y}$ of multiple sequences by optimizing a single function. SW-EEND introduces the previous speaker labels $\boldsymbol{Y}_{<s,:}$ as conditions. Self-conditioning introduces the intermediate speaker label $\boldsymbol{Y}^{(M)}$ at the M-th intermediate layer.

Method	Output target	Functions to be optimized
Clustering	$\boldsymbol{y} \in \mathbb{Z}_{\geq 0}^{T}$	$\underbrace{p(\boldsymbol{E}\|\boldsymbol{y},\boldsymbol{s})}_{\text{clustering}}, \underbrace{p(\boldsymbol{E}\|\boldsymbol{X},\boldsymbol{s})}_{\text{speaker embedding}}, \underbrace{p(\boldsymbol{s}\|\boldsymbol{X})}_{\text{SAD}}$
EEND	$\boldsymbol{Y} \in \{0,1\}^{S \times T}$	$p(\boldsymbol{Y}_{s,t}\|\boldsymbol{X})$
SW-EEND	$\boldsymbol{Y} \in \{0,1\}^{S \times T}$	$p(\boldsymbol{Y}_{s,t}\|\boldsymbol{X}, \boldsymbol{Y}_{<s,:})$
Self-conditioning	$\boldsymbol{Y} \in \{0,1\}^{S \times T}$	$p(\boldsymbol{Y}_{s,t}\|\boldsymbol{X}, \boldsymbol{Y}^{(M)})$

autoregressive attention-based module. Experiments showed that the proposed method improves both performance and training efficiency. The obtained DER is comparable with existing state-of-the-art WavLM models, which use self-supervised pretraining with large-scale training data and far more parameters.

Table 5.1 summarizes our formulations developed in this dissertation, which compares the traditional clustering and the proposed EEND methods in terms of the output target and functions to be optimized. Traditional clustering outputs a single sequence $\boldsymbol{y}$ by optimizing three functions. EEND outputs a two-dimensional matrix $\boldsymbol{Y}$ of multiple sequences by optimizing a single function. SW-EEND introduces the previous speaker labels $\boldsymbol{Y}_{<s,:}$ as conditions. Self-conditioning introduces the intermediate speaker label $\boldsymbol{Y}^{(M)}$ at the M-th intermediate layer.

5.2 Future directions

The introduction of EEND has revolutionized speaker diarization research, marking a significant shift in research focus. Many papers are now engaged in extending the EEND models. We explore the future directions by presenting recent studies built on the EEND concept.

5.2.1 Speaker aggregation module

The proposed EEND with self-attention (SA-EEND) has shown the capability of aggregating global speaker information distributed to the whole sequence into each frame if the same speaker is present. This behavior suggests that the EEND learns a scoring metric similar to the traditional clustering module, such as the PLDA scorer. The speaker aggregation module based on SA-EEND is an active research area.

Based on SA-EEND, EDA (Horiguchi et al. (2020, 2022)) is proposed as an explicit speaker aggregation module inside the network. Besides EDA's ability to handle the variable number of speakers, it even outperformed SA-EEND for the two-speaker-only case. The results suggest that the explicit speaker aggregation module is better than the implicit one with SA-EEND. Broughton and Samarakoon (2023) proposed to use a learned summary vector to produce speaker-wise attractors, which proved effective when there were many speakers. Improving the attractor-based model architecture is an important research direction.

To assist in aggregating speaker information, Kinoshita et al. (2022) proposed to use auxiliary speaker identification loss to the EEND model. The encoder output is trained with both speaker diarization and speaker identification objectives through multi-task learning. Jeoung et al. (2023) proposed an auxiliary loss to attention heads in SA-EEND. The auxiliary loss encourages the attention weight matrix of each head to be close to the affinity matrix of each speaker. The auxiliary loss enforces the correspondence between the attention head and a speaker, which is based on our observation from the attention matrix visualization in Chapter 2. TS-VAD (Medennikov et al. (2020a,b)) is considered as EEND with an explicit speaker aggregation module as an auxiliary input. TS-VAD first extracts speaker embedding of each speaker using the traditional clustering-based pipeline. Then, the EEND network is conditioned on these embeddings.

The aforementioned studies suggest that utilizing some prior speaker-level information helps improve EEND. However, it generally makes the training and inference processes complicated. We believe there is room for improvement with a simple yet effective model.

5.2.2 Local temporal dynamics and linguistic clues

Though speaker aggregation is a primary concern in speaker diarization, local temporal features are also important clues for the task. For example, a sudden change in audio volume or in the harmonic structure implies a speaker change. Some word sequences may also imply the end or start of the utterance. Our EEND models have not considered such clues.

Maiti et al. (2021) proposed to use a time-dilated convolutional neural network and showed the effectiveness in capturing local features before aggregating them via self-attention. Liu et al. (2021) proposed a Conformer-based architecture and demonstrated the importance of local features by utilizing the convolution module inside the Transformer network.

Khare et al. (2022) proposed to use the time-aligned phones, position-in-word information, and word boundaries from the ASR model as additional features for diarization. They extended the EEND model to estimate the additional ASR-based features and jointly trained using multi-task learning. Kanda et al. (2022) proposed to use end-to-end multi-speaker ASR for diarization. The end-to-end ASR model transcribes each speaker conditioned on the speaker embedding like TS-VAD. The proposed method utilizes linguistic clues since the ASR model can consider language models through training.

Since our EEND model was trained on the multilingual dataset to obtain language-agnostic performance, it was not easy to integrate with mono-lingual ASR. Methods for adapting to specified language will be a demanded research direction.

5.2.3 Consistent diarization for long-form recording

In EEND training, the audio length is limited to 50 sec. On the other hand, in the inference phase, EEND accepts a whole sequence of audio, which requires a lot of memory. The original EEND did not consider block processing of long-form audio to reduce memory or perform online processing.

Xue et al. (2021) extends the SA-EEND for the block online processing. The proposed method utilizes a speaker tracing buffer, which summarizes the features and the estimated labels from the previous blocks, to obtain consistent labels by permutation alignment between the current block and the tracing buffer. Kinoshita et al. (2021b) integrated EEND and clustering, achieving consistent diarization with block processing, as the inter-block permutation alignment is solved using clustering. Online processing is one of the important practical issues to be continually tackled.

5.2.4 Integration of speaker diarization, separation, and ASR

Speaker diarization is used as a preprocessing step for other speech processing, as written in Chapter 1. Joint optimization of speaker diarization and the following speech processing task is also an active research area. EEND fosters researchers to investigate such joint optimal systems because the EEND's single neural network can be easily integrated with other neural network-based processes.

Maiti et al. (2022) proposed EEND-SS, a speech separation model jointly trained with EEND. The results show that the joint model outperforms the independent speech separation model in terms of speech separation metrics and also outperforms the independent EEND model in terms of DER. Already shown in Sec. 5.2.2, Khare et al. (2022) and Kanda et al. (2022) investigated the integration of EEND and ASR.

The integrated models generally require a large amount of training data and model parameters. Efficient training and inference strategy for integrated models is a promising research direction. The use of self-supervised pretraining models, such as WavLM (Chen et al. (2022)), will encourage this line of research.

Finally, the use of multimodal pretraining models for audio and images, such as AV-HuBERT (Shi et al. (2022)), and large language models like GPT-3 (Brown et al. (2020)), which have seen significant advancements in recent years, are anticipated as further extensions of EEND. Audio-visual diarization is believed to be extremely effective for the diarization of video content. Moreover, if confident linguistic context can be obtained using large language models, it is not only expected to improve accuracy but also to apply to real-time diarization. If turn-taking can be detected in real-time, new applications, such as dialogue robots facilitating human-to-human communication, could be enabled. For such new applications as well, the extension of EEND is demanded.

Bibliography

X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals. Speaker diarization: A review of recent research. *IEEE Trans. on ASLP*, 20(2):356–370, 2012. ISSN 1558-7916. doi: 10.1109/TASL.2011.2125954.

Dan Berrebbi, Brian Yan, and Shinji Watanabe. Avoid overthinking in self-supervised models for speech recognition. In *Proc. ICASSP*, 2023. doi: 10.1109/ICASSP49357.2023.10095335.

Samuel J. Broughton and Lahiru Samarakoon. Improving end-to-end neural diarization using conversational summary representations. In *Proc. INTERSPEECH*, 2023. doi: 10.21437/Interspeech.2023-2401.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901, 2020.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, Xiangzhan Yu, and Furu Wei. Wavlm: Large-scale self-supervised pre-training for full stack speech processing. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1505–1518, 2022. doi: 10.1109/JSTSP.2022.3188113.

Scotte Chen, Ponani S. Gopalakrishnan, and Ibm Thomas J. Watson. Speaker, environment and channel change detection and clustering via the bayesian information criterion. In *DARPA Speech Recognition Workshop*, 1998.

Joon Son Chung, Jaesung Huh, Arsha Nagrani, Triantafyllos Afouras, and Andrew Zisserman. Spot the Conversation: Speaker Diarisation in the Wild. In *Proc. Interspeech*, pages 299–303, 2020. doi: 10.21437/Interspeech.2020-2337.

N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet. Front-end factor analysis for speaker verification. *IEEE Trans. on ASLP*, 19(4):788–798, 2011. ISSN 1558-7916. doi: 10.1109/TASL.2010.2064307.

M. Delcroix, K. Zmolikova, K. Kinoshita, A. Ogawa, and T. Nakatani. Single channel target speaker extraction and recognition with speaker beam. In *Proc. ICASSP*, pages 5554–5558, 2018. doi: 10.1109/ICASSP.2018.8462661.

Mireia Diez, Federico Landini, Lukáš Burget, Johan Rohdin, Anna Silnova, Kateřina Žmolíková, Ondřej Novotný, Karel Veselý, Ondřej Glembek, Oldřich Plchot, Ladislav Mošner, and Pavel Matějka. BUT system for DIHARD speech diarization challenge 2018. In *Proc. Interspeech*, pages 2798–2802, 2018. doi: 10.21437/Interspeech.2018-1749.

Dimitrios Dimitriadis and Petr Fousek. Developing on-line speaker diarization system. In *Proc. Interspeech*, pages 2739–2743, 2017. doi: 10.21437/Interspeech.2017-166.

G Doddington, Mark Przybocki, Alvin Martin, and D Reynolds. NIST speaker recognition evaluation – overview, methodology, systems, results, perspective. *Speech Communication*, 2000. doi: 10.1016/S0167-6393(99)00080-1.

Linhao Dong, Shuang Xu, and Bo Xu. Speech-Transformer: A No-Recurrence Sequence-to-Sequence Model for Speech Recognition. In *Proc. ICASSP*, pages 5884–5888, 2018.

Jun Du, Yan-Hui Tu, Lei Sun1, Feng Ma, Hai-Kun Wang, Jia Pan, Cong Liu, Jing-Dong Chen, and Chin-Hui Lee. The ustc-iflytek system for chime-4 challenge. In *CHiME-4*, pages 36–38, 2016.

Xin Fang, Zhen-Hua Ling, Lei Sun, Shu-Tong Niu, Jun Du, Cong Liu, and Zhi-Chao Sheng. A deep analysis of speech separation guided diarization under realistic conditions. In *2021 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 667–671, 2021.

Yusuke Fujita, Naoyuki Kanda, Shota Horiguchi, Kenji Nagamatsu, and Shinji Watanabe. End-to-end neural speaker diarization with permutation-free objectives. In *Proc. Interspeech*, 2019a.

Yusuke Fujita, Naoyuki Kanda, Shota Horiguchi, Yawen Xue, Kenji Nagamatsu, and Shinji Watanabe. End-to-end neural speaker diarization with self-attention. In *Proc. ASRU*, 2019b.

Yusuke Fujita, Tatsuya Komatsu, Robin Scheibler, Yusuke Kida, and Tetsuji Ogawa. Neural diarization with non-autoregressive intermediate attractors. In *Proc. ICASSP*, pages 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10094824.

D. Garcia-Romero, D. Snyder, G. Sell, D. Povey, and A. McCree. Speaker diarization using deep neural network embeddings. In *Proc. ICASSP*, pages 4930–4934, 2017. doi: 10.1109/ICASSP.2017.7953094.

Daniel Garcia-Romero and Carol Y. Espy-Wilson. Analysis of i-vector length normalization in speaker recognition systems. In *Proc. Interspeech*, pages 249–252, 2011. doi: 10.21437/Interspeech.2011-53.

Pegah Ghahremani, Vimal Manohar, Daniel Povey, and Sanjeev Khudanpur. Acoustic Modelling from the Signal Domain Using CNNs. In *Proc. Interspeech*, pages 3434–3438, 2016. doi: 10.21437/Interspeech.2016-1495.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 18(5):602 – 610, 2005. ISSN 0893-6080. doi: 10.1016/j.neunet.2005.06.042. IJCNN 2005.

J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe. Deep clustering: Discriminative embeddings for segmentation and separation. In *Proc. ICASSP*, pages 31–35, 2016. doi: 10.1109/ICASSP.2016.7471631.

Shota Horiguchi, Yusuke Fujita, Shinji Watanabe, Yawen Xue, and Kenji Nagamatsu. End-to-End Speaker Diarization for an Unknown Number of Speakers with Encoder-Decoder Based Attractors. In *Proc. Interspeech*, pages 269–273, 2020. doi: 10.21437/Interspeech.2020-1022.

Shota Horiguchi, Shinji Watanabe, Paola García, Yawen Xue, Yuki Takashima, and Yohei Kawaguchi. Towards neural diarization for unlimited numbers of speakers using global and local attractors. In *Proc. ASRU*, pages 98–105, 2021. doi: 10.1109/ASRU51503.2021.9687875.

Shota Horiguchi, Yusuke Fujita, Shinji Watanabe, Yawen Xue, and Paola García. Encoder-decoder based attractors for end-to-end neural diarization. *IEEE/ACM Trans. on ASLP*, 30:1493–1507, 2022. doi: 10.1109/TASLP.2022.3162080.

A. Janin, D. Baron, J. Edwards, D. Ellis, D. Gelbart, N. Morgan, B. Peskin, T. Pfau, E. Shriberg, A. Stolcke, and C. Wooters. The ICSI meeting corpus. In *Proc. ICASSP*, volume I, pages 364–367, 2003. doi: 10.1109/ICASSP.2003.1198793.

Ye-Rin Jeoung, Joon-Young Yang, Jeong-Hwan Choi, and Joon-Hyuk Chang. Improving transformer-based end-to-end speaker diarization by assigning auxiliary losses to attention heads. In *Proc. ICASSP*, 2023.

Naoyuki Kanda, Shota Horiguchi, Ryoichi Takashima, Yusuke Fujita, Kenji Nagamatsu, and Shinji Watanabe. Auxiliary Interference Speaker Loss for Target-Speaker Speech Recognition. In *Proc. Interspeech*, pages 236–240, 2019. doi: 10.21437/Interspeech.2019-1126.

Naoyuki Kanda, Xiong Xiao, Yashesh Gaur, Xiaofei Wang, Zhong Meng, Zhuo Chen, and Takuya Yoshioka. Transcribe-to-diarize: Neural speaker diarization for unlimited number of speakers using end-to-end speaker-attributed asr. In *Proc. ICASSP*, 2022.

Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. Shallow-deep networks: Understanding and mitigating network overthinking. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3301–3310. PMLR, 09–15 Jun 2019.

Aparna Khare, Eunjung Han, Yuguang Yang, and Andreas Stolcke. ASR-Aware End-to-end Neural Diarization. In *Proc. ICASSP*, 2022.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2015.

Keisuke Kinoshita, Marc Delcroix, and Naohiro Tawara. Integrating end-to-end neural and clustering-based diarization: Getting the best of both worlds. In *Proc. ICASSP*, pages 7198–7202, 2021a.

Keisuke Kinoshita, Marc Delcroix, and Naohiro Tawara. Advances in Integration of End-to-End Neural and Clustering-Based Diarization for Real Conversational Speech. In *Proc. Interspeech 2021*, pages 3565–3569, 2021b. doi: 10.21437/Interspeech.2021-1004.

Keisuke Kinoshita, Marc Delcroix, and Tomoharu Iwata. Tight integration of neural- and clustering-based diarization through deep unfolding of infinite gaussian mixture model. In *Proc. ICASSP*, 2022.

T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur. A study on data augmentation of reverberant speech for robust speech recognition. In *Proc. ICASSP*, pages 5220–5224, 2017. doi: 10.1109/ICASSP.2017.7953152.

M. Kolbæk, D. Yu, Z. Tan, and J. Jensen. Multitalker speech separation with utterance-level permutation invariant training of deep recurrent neural networks. *IEEE/ACM Trans. on ASLP*, 25(10):1901–1913, 2017. ISSN 2329-9290. doi: 10.1109/TASLP.2017.2726762.

Federico Landini, Alicia Lozano-Diez, Lukáš Burget, Mireia Diez, Anna Silnova, Kateřina Žmolíková, Ondřej Novotný, Pavel Matějka, Themos Stafylakis, and Niko Brümmer. But system description for the third dihard speech diarization challenge. In *The Third DIHARD Speech Diarization Challenge Workshop*, 2021.

Federico Landini, Alicia Lozano-Diez, Mireia Diez, and Lukáš Burget. From Simulated Mixtures to Simulated Conversations as Training Data for End-to-End Neural Diarization. In *Proc. Interspeech*, pages 5095–5099, 2022a. doi: 10.21437/Interspeech.2022-10451.

Federico Landini, Ján Profant, Mireia Diez, and Lukáš Burget. Bayesian hmm clustering of x-vector sequences (vbx) in speaker diarization: Theory, implementation and analysis on standard tasks. *Computer Speech & Language*, 71:101254, 2022b. ISSN 0885-2308. doi: https://doi.org/10.1016/j.csl.2021.101254.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. A structured self-attentive sentence embedding. In *ICLR*, 2017.

Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. Understanding the difficulty of training transformers. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5747–5763, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.463.

Yi Chieh Liu, Eunjung Han, Chul Lee, and Andreas Stolcke. End-to-End Neural Diarization: From Transformer to Conformer. In *Proc. Interspeech*, pages 3081–3085, 2021. doi: 10.21437/Interspeech.2021-1909.

Matthew Maciejewski, David Snyder, Vimal Manohar, Najim Dehak, and Sanjeev Khudanpur. Characterizing performance of speaker diarization systems on far-field speech using standard methods. In *Proc. ICASSP*, pages 5244–5248, 2018.

Kikuo Maekawa. Corpus of spontaneous japanese: Its design and evaluation. In *ISCA & IEEE Workshop on Spontaneous Speech Processing and Recognition*, 2003.

Soumi Maiti, Hakan Erdogan, Kevin Wilson, Scott Wisdom, Shinji Watanabe, and John R. Hershey. End-to-end diarization for variable number of speakers with local-global networks and discriminative speaker embeddings. In *Proc. ICASSP*, pages 7183–7187, 2021. doi: 10.1109/ICASSP39728.2021.9414841.

Soumi Maiti, Yushi Ueda, Shinji Watanabe, Chunlei Zhang, Meng Yu, Shi-Xiong Zhang, and Yong Xu. Eend-ss: Joint end-to-end neural speaker diarization and speech separation for flexible number of speakers. In *Proc. SLT*, 2022.

Philip Andrew Mansfield, Quan Wang, Carlton Downey, Li Wan, and Ignacio Lopez Moreno. Links: A high-dimensional online clustering method. *arXiv preprint arXiv:1801.10123*, 2018.

Alan McCree, Gregory Sell, and Daniel Garcia-Romero. Speaker diarization using leave-one-out gaussian PLDA clustering of DNN embeddings. In *Proc. Interspeech*, pages 381–385, 2019.

Ivan Medennikov, Maxim Korenevsky, Tatiana Prisyach, Yuri Khokhlov, Mariya Korenevskaya, Ivan Sorokin, Tatiana Timofeeva, Anton Mitrofanov, Andrei Andrusenko, Ivan Podluzhny, Aleksandr Laptev, and Aleksei Romanenko. The STC System for the CHiME-6 Challenge. In *Proc. 6th International Workshop on Speech Processing in Everyday Environments (CHiME 2020)*, pages 36–41, 2020a. doi: 10.21437/CHiME.2020-9.

Ivan Medennikov, Maxim Korenevsky, Tatiana Prisyach, Yuri Khokhlov, Mariya Korenevskaya, Ivan Sorokin, Tatiana Timofeeva, Anton Mitrofanov, Andrei Andrusenko, Ivan Podluzhny, Aleksandr Laptev, and Aleksei Romanenko. Target-Speaker Voice Activity Detection: A Novel Approach for Multi-Speaker Diarization in a Dinner Party Scenario. In *Proc. Interspeech*, pages 274–278, 2020b. doi: 10.21437/Interspeech.2020-1602.

Sylvain Meignier. LIUM_SPKDIARIZATION: An open source toolkit for diarization. In *CMU SPUD Workshop*, 2010.

National Institute of Standard and Technology (NIST). Rich transcription evaluation. https://www.nist.gov/itl/iad/mig/rich-transcription-evaluation, 2009.

Huazhong Ning, Ming Liu, Hao Tang, and Thomas S. Huang. A spectral clustering approach to speaker diarization. In *Proc. INTERSPEECH*, 2006.

NIST. 2000 NIST Speaker Recognition Evaluation. https://catalog.ldc.upenn.edu/LDC2001S97, 2000.

NIST. The 2009 (RT-09) rich transcription meeting recognition evaluation plan. http://www.itl.nist.gov/iad/mig/tests/rt/2009/docs/rt09-meeting-eval-plan-v2.pdf, 2009.

Jumon Nozaki and Tatsuya Komatsu. Relaxing the Conditional Independence Assumption of CTC-Based ASR by Conditioning on Intermediate Predictions. In *Proc. Interspeech*, pages 3735–3739, 2021. doi: 10.21437/Interspeech.2021-911.

Tae Jin Park and Panayiotis Georgiou. Multimodal Speaker Segmentation and Diarization Using Lexical and Acoustic Cues via Sequence to Sequence Neural Networks. In *Proc. Interspeech 2018*, pages 1373–1377, 2018. doi: 10.21437/Interspeech.2018-1364.

Tae Jin Park, Naoyuki Kanda, Dimitrios Dimitriadis, Kyu J. Han, Shinji Watanabe, and Shrikanth Narayanan. A review of speaker diarization: Recent advances with deep learning. *Comput. Speech Lang.*, 72:101317, 2022.

Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur. A time delay neural network architecture for efficient modeling of long temporal contexts. In *Proc. Interspeech*, pages 3214–3218, 2015. doi: 10.21437/Interspeech.2015-647.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The Kaldi speech recognition toolkit. In *Proc. ASRU*, 2011.

Simon J.D. Prince and James H. Elder. Probabilistic linear discriminant analysis for inferences about identity. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, 2007. doi: 10.1109/ICCV.2007.4409052.

S. Renals, T. Hain, and H. Bourlard. Interpretation of multiparty meetings the AMI and Amida projects. In *2008 Hands-Free Speech Communication and Microphone Arrays*, pages 115–118, 2008. doi: 10.1109/HSCMA.2008.4538700.

Douglas A. Reynolds, Thomas F. Quatieri, and Robert B. Dunn. Speaker verification using adapted gaussian mixture models. *Digit. Signal Process.*, 10:19–41, 2000.

Neville Ryant, Kenneth Church, Christopher Cieri, Alejandrina Cristia, Jun Du, Sriram Ganapathy, and Mark Liberman. The Second DIHARD Diarization Challenge: Dataset, Task, and Baselines. In *Proc. Interspeech*, pages 978–982, 2019.

Neville Ryant, Prachi Singh, Venkat Krishnamohan, Rajat Varma, Kenneth Church, Christopher Cieri, Jun Du, Sriram Ganapathy, and Mark Liberman. The Third DIHARD Diarization Challenge. In *Proc. Interspeech*, pages 3570–3574, 2021. doi: 10.21437/Interspeech.2021-1208.

G. Sell and D. Garcia-Romero. Speaker diarization with PLDA i-vector scoring and unsupervised calibration. In *Proc. SLT*, pages 413–417, 2014. doi: 10.1109/SLT.2014.7078610.

Gregory Sell, David Snyder, Alan McCree, Daniel Garcia-Romero, Jesús Villalba, Matthew Maciejewski, Vimal Manohar, Najim Dehak, Daniel Povey, Shinji Watanabe, and Sanjeev Khudanpur. Diarization is hard: Some experiences and lessons learned for the JHU team in the inaugural DIHARD challenge. In *Proc. Interspeech*, pages 2808–2812, 2018. doi: 10.21437/Interspeech.2018-1893.

M. Senoussaoui, P. Kenny, T. Stafylakis, and P. Dumouchel. A study of the cosine distance-based mean shift for telephone speech diarization. *IEEE/ACM Trans. on ASLP*, 22(1):217–227, 2014. ISSN 2329-9290. doi: 10.1109/TASLP.2013.2285474.

Bowen Shi, Wei-Ning Hsu, Kushal Lakhotia, and Abdelrahman Mohamed. Learning audio-visual speech representation by masked multimodal cluster prediction. In *International Conference on Learning Representations*, 2022.

S. H. Shum, N. Dehak, R. Dehak, and J. R. Glass. Unsupervised methods for speaker diarization: An integrated and iterative approach. *IEEE Trans. on ASLP*, 21(10):2015–2028, 2013. ISSN 1558-7916. doi: 10.1109/TASL.2013.2264673.

D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur. X-vectors: Robust DNN embeddings for speaker recognition. In *Proc. ICASSP*, pages 5329–5333, 2018. doi: 10.1109/ICASSP.2018.8461375.

David Snyder, Guoguo Chen, and Daniel Povey. MUSAN: A music, speech, and noise corpus. *arXiv preprints arXiv:1510.08484*, 2015.

Lei Sun, Jun Du, Chao Jiang, Xueyang Zhang, Shan He, Bing Yin, and Chin-Hui Lee. Speaker diarization with enhancing speech for the first DIHARD challenge. In *Proc. Interspeech*, pages 2793–2797, 2018.

S. E. Tranter and D. A. Reynolds. An overview of automatic speaker diarization systems. *IEEE Trans. on ASLP*, 14(5):1557–1565, 2006. ISSN 1558-7916. doi: 10.1109/TASL.2006.878256.

Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez. Deep neural networks for small footprint text-dependent speaker verification. In *Proc. ICASSP*, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017.

Thilo von Neumann, Keisuke Kinoshita, Marc Delcroix, Shoko Araki, Tomohiro Nakatani, and Reinhold Haeb-Umbach. All-neural online source separation, counting, and diarization for meeting analysis. In *Proc. ICASSP*, 2019.

L. Wan, Q. Wang, A. Papir, and I. L. Moreno. Generalized end-to-end loss for speaker verification. In *Proc. ICASSP*, pages 4879–4883, 2018. doi: 10.1109/ICASSP.2018.8462665.

Q. Wang, C. Downey, L. Wan, P. A. Mansfield, and I. L. Moreno. Speaker diarization with LSTM. In *Proc. ICASSP*, pages 5239–5243, 2018. doi: 10.1109/ICASSP.2018.8462628.

Zhe Wang, Shilong Wu, Hang Chen, Mao-Kui He, Jun Du, Chin-Hui Lee, Jingdong Chen, Shinji Watanabe, Sabato Siniscalchi, Odette Scharenborg, Diyuan Liu, Baocai Yin, Jia Pan, Jianqing Gao, and Cong Liu. The multimodal information based speech processing (misp) 2022 challenge: Audio-visual diarization and recognition. In *Proc. ICASSP*, pages 1–5, 2023. doi: 10.1109/ICASSP49357.2023.10094836.

Shinji Watanabe, Michael Mandel, Jon Barker, Emmanuel Vincent, Ashish Arora, Xuankai Chang, Sanjeev Khudanpur, Vimal Manohar, Daniel Povey, Desh Raj, David Snyder, Aswin Shanmugam Subramanian, Jan Trmal, Bar Ben Yair, Christoph Boeddeker, Zhaoheng Ni, Yusuke Fujita, Shota Horiguchi, Naoyuki Kanda, Takuya Yoshioka, and Neville Ryant. CHiME-6 Challenge: Tackling Multispeaker Speech Recognition for Unsegmented Recordings. In *Proc. 6th International Workshop on Speech Processing in Everyday Environments (CHiME 2020)*, pages 1–7, 2020. doi: 10.21437/CHiME.2020-1.

R. J. Williams and D. Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1(2):270–280, 1989.

Yawen Xue, Shota Horiguchi, Yusuke Fujita, Shinji Watanabe, and Kenji Nagamatsu. Online end-to-end neural diarization with speaker-tracing buffer. In *Proc. SLT*, 2021.

D. Yu, M. Kolbæk, Z. Tan, and J. Jensen. Permutation invariant training of deep models for speaker-independent multi-talker speech separation. In *Proc. ICASSP*, pages 241–245, 2017. doi: 10.1109/ICASSP.2017.7952154.

Fan Yu, Shiliang Zhang, Pengcheng Guo, Yihui Fu, Zhihao Du, Siqi Zheng, Weilong Huang, Lei Xie, Zheng-Hua Tan, DeLiang Wang, Yanmin Qian, Kong Aik Lee, Zhijie Yan, Bin Ma, Xin Xu, and Hui Bu. Summary on the icassp 2022 multi-channel multi-party meeting transcription grand challenge. In *Proc. ICASSP*, pages 9156–9160, 2022. doi: 10.1109/ICASSP43922.2022.9746270.

Aonan Zhang, Quan Wang, Zhenyao Zhu, John Paisley, and Chong Wang. Fully supervised speaker diarization. In *Proc. ICASSP*, pages 6301–6305, 2019.

Özgür Çetin and Elizabeth Shriberg. Overlap in meetings: ASR effects and analysis by dialog factors, speakers, and collection site. In *MLMI*, pages 212–224, 2006.

Kateřina Žmolíková, Marc Delcroix, Keisuke Kinoshita, Takuya Higuchi, Atsunori Ogawa, and Tomohiro Nakatani. Speaker-Aware Neural Network Based Beamformer for Speaker Extraction in Speech Mixtures. In *Proc. Interspeech*, pages 2655–2659, 2017. doi: 10.21437/Interspeech.2017-667.

Publications related to this dissertation

JOURNAL PAPER

2023 Yusuke Fujita, Tetsuji Ogawa, Tetsunori Kobayashi, Self-conditioning via Intermediate Predictions for End-to-end Neural Speaker Diarization, IEEE Access, 2023, doi: 10.1109/ACCESS.2023.3340307

INTERNATIONAL CONFERENCE PAPERS

(PEER REVIEWED)

2023 Yusuke Fujita, Tatsuya Komatsu, Robin Scheibler, Yusuke Kida, Tetsuji Ogawa, “Neural Diarization with Non-autoregressive Intermediate Attractors,” IEEE ICASSP, 2023, doi: 10.1109/ICASSP49357.2023.10094824

2021 Yuki Takashima, Yusuke Fujita, Shinji Watanabe, Shota Horiguchi, Paola García, Kenji Nagamatsu, “End-to-end speaker diarization conditioned on speech activity and overlap detection,” IEEE SLT, 2021, doi: 10.1109/SLT48900.2021.9383555

2019 Yusuke Fujita, Naoyuki Kanda, Shota Horiguchi, Yawen Xue, Kenji Nagamatsu, Shinji Watanabe, “End-to-End Neural Speaker Diarization with Self-attention,” IEEE ASRU, 2019, doi: 10.1109/ASRU46091.2019.9003959

2019 Yusuke Fujita, Naoyuki Kanda, Shota Horiguchi, Kenji Nagamatsu, Shinji Watanabe, “End-to-End Neural Speaker Diarization with Permutation-Free Objectives,” Interspeech, 2019, doi: 10.21437/Interspeech.2019-2899

TUTORIAL TALK

2021 Keisuke Kinoshita, Yusuke Fujita, Naoyuki Kanda, Shinji Watanabe, “T-9: Distant conversational speech recognition and analysis,” IEEE ICASSP, 2021

DOMESTIC CONFERENCE PAPER

2023 Yusuke Fujita, Tatsuya Komatsu, Robin Scheibler, Yusuke Kida, Tetsuji Ogawa, Neural Speaker Diarization with Intermediate Predictions, Acoustic Society of Japan (ASJ) 2023 Spring Meeting, March 2023.