Encoder-Decoder Based Attractors for End-to-End Neural Diarization

Shota Horiguchi, *Member, IEEE*, Yusuke Fujita, *Member, IEEE*, Shinji Watanabe, *Senior Member, IEEE*, Yawen Xue, and Paola García, *Member, IEEE*

***Abstract*—This paper investigates an end-to-end neural diarization (EEND) method for an unknown number of speakers. In contrast to the conventional cascaded approach to speaker diarization, EEND methods are better in terms of speaker overlap handling. However, EEND still has a disadvantage in that it cannot deal with a flexible number of speakers. To remedy this problem, we introduce encoder-decoder-based attractor calculation module (EDA) to EEND. Once frame-wise embeddings are obtained, EDA sequentially generates speaker-wise attractors on the basis of a sequence-to-sequence method using an LSTM encoder-decoder. The attractor generation continues until a stopping condition is satisfied; thus, the number of attractors can be flexible. Diarization results are then estimated as dot products of the attractors and embeddings. The embeddings from speaker overlaps result in larger dot product values with multiple attractors; thus, this method can deal with speaker overlaps. Because the maximum number of output speakers is still limited by the training set, we also propose an iterative inference method to remove this restriction. Further, we propose a method that aligns the estimated diarization results with the results of an external speech activity detector, which enables fair comparison against cascaded approaches. Extensive evaluations on simulated and real datasets show that EEND-EDA outperforms the conventional cascaded approach.**

***Index Terms*—Speaker diarization, EEND, EDA.**

I. Introduction

SPEAKER diarization is a task of estimating multiple speakers' speech activities from input audio (sometimes referred to as the "who spoke when" problem) [1]. It can be placed as a downstream task of automatic speech recognition (ASR), in which speaker information is tagged to each transcribed utterance [2]–[4]. It can also be used as a prior step to speech separation and the following ASR. For example, in guided source separation [5], speech activities are used as constraints to update time-frequency masks of a complex angular central Gaussian mixture model. The speech-activity-driven speech-extraction neural network [6] takes acoustic features and a target speaker's speech activity to perform fully neural speech separation.

Classical cascaded methods treat speaker diarization as a partition problem. Given a set of time frames, they first detect speaker-active frames and then divide them into clusters by using speaker embeddings extracted with a sliding window. The number of clusters, which represents the number of speakers, is determined in the clustering step during inference. Eigen value analysis on the graph Laplacian of a similarity matrix calculated from frame-wise embeddings is one way to estimate the number of speakers explicitly [7], [8]. If agglomerative hierarchical clustering is employed as a clustering algorithm, a threshold value is usually preset, and the number of clusters, i.e., the number of speakers, is dynamically determined by the threshold value [9]. Either way, the number of clusters can be set flexibly during inference. However, there is one fundamental problem that it basically cannot handle speaker overlaps because each speech frame is usually assigned to one speaker.

Some neural-network-based end-to-end methods, in comparison, naturally handle speaker overlap with a single network. For example, the Recurrent Selective Attention Network (RSAN) [10], [11] decodes speech activity for each speaker one by one until a stopping condition is satisfied. However, it requires clean speech to be trained as a mask-based speech separation model. End-to-end neural diarization (EEND) [12]–[14], which estimates multiple speakers' speech activities at once from input audio, does not require such clean speech for training. The limitation is that the original EEND fixes the output number of speakers; thus, knowing the number of speakers in advance is a requirement.

In our previous study [15], we introduced an encoder-decoder-based attractor calculation module (EDA) as part of the self-attentive EEND model [13] to handle unknown numbers of speakers (EEND-EDA). It calculates attractors from frame-wise embeddings using a sequence-to-sequence method with an LSTM encoder-decoder; thus, the number of attractors can be flexible. In general, sequence-to-sequence methods require a stopping criterion in their decoding process. To decide when to stop the attractor calculation, EDA also estimates whether each calculated attractor really corresponds to a speaker. The diarization results are calculated as dot products between the attractors and frame-wise embeddings. Despite being designed

Manuscript received June 20, 2021; revised January 27, 2022; accepted March 8, 2022. Date of publication March 24, 2022; date of current version April 29, 2022. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Alberto Abad. *(Corresponding author: Shota Horiguchi.)*

Shota Horiguchi and Yawen Xue are with Hitachi, Ltd., Kokubunji-shi, Tokyo 185-8601, Japan (e-mail: shota.horiguchi.wk@hitachi.com; yawen.xue.wn@hitachi.com).

Yusuke Fujita was with Hitachi, Ltd., Tokyo 185-8601, Japan. He is now with LINE Corporation, Shinjuku-ku, Tokyo 160-0004, Japan (e-mail: yusuke.fujita@ieee.org).

Shinji Watanabe is with Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: shinjiw@ieee.org).

Paola García is with Johns Hopkins University, Baltimore, MD 21218 USA (e-mail: lgarci27@jhu.edu).

Digital Object Identifier 10.1109/TASLP.2022.3162080

for the diarization of flexible numbers of speakers, it also has performed better than the original EEND under fixed-number-of-speakers conditions. Compared with other EEND extensions for unknown numbers of speakers [16], [17], it performed the best on various datasets including the CALLHOME and DIHARD III datasets [18]. Several studies have also proposed extensions to EEND-EDA to allow online processing [19], [20].

In this paper, we revisit EEND-EDA with more comprehensive discussions and formulations and propose several extensions from the original EEND-EDA presented in [15]. The modifications from the original EEND-EDA study are summarized as follows:

- We discuss the relationship between the original EEND and EEND-EDA, which explains EEND-EDA's better performance in a fixed-number-of-speakers evaluation.
- We also propose refining the training strategy of EEND-EDA, which resulted in a 2.41 % DER improvement on the CALLHOME dataset from the original paper [15].
- In the history of diarization studies, it has been difficult to compare the results of cascaded approaches and EEND-based approaches because the former ones are often evaluated with an oracle speech activity detection (SAD), while EENDs operate SAD and diarization simultaneously. To conduct fair comparisons between cascaded and EEND-based approaches, this paper introduces SAD post-processing to align diarization results from EEND-EDA with external SAD results.
- We also propose an iterative inference for handling the problem of the number of outputs of EEND-EDA being empirically limited by its training dataset.
- We conduct thorough evaluations and analyses on simulated and real datasets including CALLHOME, CSJ, AMI, DIHARD II, and DIHARD III.

II. Related Work

A. Speaker Diarization

Conventional diarization methods are typically a cascade of four modules: 1) speech activity detection (SAD), 2) speaker embedding extraction, 3) embedding clustering, and 4) overlap handling as an optional process. Some methods also include an ASR module [21], [22]. Most studies mainly focus on 2) speech embedding extraction and 3) embedding clustering. For speaker embeddings, i-vectors [23], [24], x-vectors [25]–[27], and d-vectors [7], [28] have been explored. For embedding clustering, earlier works used traditional clustering algorithms, e.g., K-means clustering [29], [30], agglomerative hierarchical clustering (AHC) [9], [31], [32], mean-shift clustering [23], and spectral clustering [7], [33]. Recently, better clustering methods have been proposed, such as variational Bayes hidden Markov model clustering (VBx) [34], [35], auto-tuning spectral clustering [8], or fully supervised clustering [28], [36]. They are usually used for hard clustering, so most cascaded methods (with some exceptions [37]) cannot deal with speaker overlap. To make them able to treat speaker overlap, 4) overlap handling should be considered; however, it has sometimes been excluded from methods and evaluations even in very recent studies [7], [8], [24], [28], [36]. Moreover, 1) speech activity detection has often been ignored in evaluations of cascaded approaches that use oracle speech activities [7], [8], [24], [28], [36].

Neural-network-based methods that directly produce diarization results from audio are emerging [10], [11]. One strength of such methods is that they require no extra modules for SAD or overlap handling. For some methods, models have been trained for speech separation, and diarization results have been obtained as byproducts [10], [11]. Such models have been trained on the basis of clean speech (or time-frequency masks calculated from clean speech); thus, they cannot be trained on real mixtures like DIHARD datasets [38], [39]. However, EEND-based models are trained to output multiple speakers' speech activities; they do not require clean speech for training and real mixtures can be used. The original EEND [12]–[14] can output diarization results for a fixed number of speakers. To extend the EEND for an unknown number of speakers, two approaches have been investigated. One is an attractor-based approach [15], [19], and the other is a speaker-wise conditional EEND (SC-EEND) [16], [17]. In this paper, we investigate the attractor-based EEND because it showed better performance compared to SC-EEND.

B. Speech Processing Based on Neural Networks for Unknown Numbers of Speakers

While some methods have achieved promising results with a fixed number of output speakers in diarization [12], [13], [40] and speech separation [41]–[44] contexts, it is challenging to make them able to deal with unknown numbers of speakers. The difficulty of neural-network-based speech processing for unknown numbers of speakers is that we cannot fix the output dimension.

One possible approach is to determine the maximum number of speakers to decode. In this case, the number of outputs is set to a sufficiently large value. Some methods treat a flexible number of speakers by outputting null speech activities if the number of outputs is smaller than the network capacity [45]. However, this approach did not work well with EEND (see [16]). In other methods, the number-of-speaker-wise output branches are trained independently, and the most probable is used during inference [46]. In this case, we have to know the maximum number of speakers. One of the strengths of EEND is that it can be finetuned using a target domain dataset from a pretrained model, but we usually cannot access the maximum number of speakers of the target domain beforehand. Therefore, a method that does not require that the maximum number of speakers be defined would be preferable.

Another approach is to decode speakers one by one until a stopping condition is satisfied, like SC-EEND [16]. For speech separation, RSAN [10], [11] and one-and-rest permutation invariant training (OR-PIT) [47] can be used. The key difference between speech separation and diarization is whether or not the residual output can be defined. RSAN uses a mask-based approach, in which each time-frequency bin is softly assigned to each speaker so that the process finishes when all the elements of the residual mask become zero. OR-PIT is time-domain speech separation by which residual output is determined as a mixture

that contains other speakers rather than the target speaker. Both require clean recordings to determine oracle masks or signals. However, they are not always accessible in the diarization context, in which only multi-talker recordings and speech segments are provided.

In this paper, we adopted an attractor-based approach like deep attractor networks (DANet) [45], [48]. While the number of speakers [48] or maximum number of speakers [45] is fixed for the original DANet, in this paper, we calculated a flexible number of attractors without defining them.

C. Neural-Network-Based Representative Vector Calculation

There have been several efforts to calculate representative vectors from a sequence of embeddings in an end-to-end trainable fashion. For example, Set Transformer [49] enables set-to-set transformation, which can be used to calculate cluster centroids from a set of embeddings. However, the number of outputs has to be known in advance, so it cannot be used for our purpose. Meier *et al.* proposed an end-to-end clustering framework [50], in which clustering for all possible number of clusters $K \in \{1, \dots, K_{\max}\}$ is performed and the result of the most probable number of clusters is used. The framework performs the clustering of a flexible number of clusters in an end-to-end manner, but the maximum number of clusters is limited by $K_{\max}$. EDA in this paper, in comparison, determines a flexible number of attractors from an input embedding without prior knowledge of the number of speakers. Thus, we can use datasets of the different maximum number of speakers during pretraining and finetuning.

III. Method

In this section, we first introduce the conventional EEND in Section III-A followed by an explanation of a natural extension of the method called attractor-based EEND in Section III-B. We also provide novel inference techniques in Section III-C.

A. Conventional End-to-End Neural Diarization

End-to-end neural diarization (EEND) [12], [13] is a method for estimating multiple speakers' speech activities simultaneously from an input recording. Given frame-wise F-dimensional acoustic features $(\boldsymbol{x}_t)_{t=1}^T$, where $t \in \{1, \dots, T\}$ is a frame index, EEND estimates speech activities $(\boldsymbol{y}_t)_{t=1}^T$. Here, $\boldsymbol{y}_t := [y_{1,t}, \dots, y_{s,t}, \dots, y_{S,t}]^\mathsf{T}$ denotes speech activities of S speakers at t defined as

$$y_{s,t} = \begin{cases} 0 & (\text{Speaker } s \text{ is inactive at } t) \\ 1 & (\text{Speaker } s \text{ is active at } t) \end{cases}. \tag{1}$$

EEND assumes that $y_{s,t}$ is conditionally independent given the acoustic features, namely,

$$P\left(\boldsymbol{y}_1, \dots, \boldsymbol{y}_T \mid \boldsymbol{x}_1, \dots, \boldsymbol{x}_T\right) = \prod_{t=1}^{T} \prod_{s=1}^{S} P\left(y_{s,t} \mid \boldsymbol{x}_1, \dots, \boldsymbol{x}_T\right). \tag{2}$$

With this assumption, speaker diarization can be regarded as a multi-label classification problem and can thus be easily modeled using a neural network f_{EEND} as

$$(\boldsymbol{p}_1, \dots, \boldsymbol{p}_T) = f_{\mathsf{EEND}}\left(\boldsymbol{x}_1, \dots, \boldsymbol{x}_T\right), \tag{3}$$

where $\boldsymbol{p}_t := [p_{1,t}, \dots, p_{S,t}]^\mathsf{T} \in (0,1)^S$ is the posterior probabilities of S speakers' speech activities at frame index t. The estimation of speech activities $(\hat{\boldsymbol{y}}_t)_{t=1}^T$ is

$$\hat{\boldsymbol{y}}_1, \dots, \hat{\boldsymbol{y}}_T = \underset{\boldsymbol{y}_1, \dots, \boldsymbol{y}_T}{\arg\max} P\left(\boldsymbol{y}_1, \dots, \boldsymbol{y}_T \mid x_1, \dots, x_T\right), \tag{4}$$

$$= \left(\mathbb{1}\left(p_{s,t} > 0.5\right)\right)_{\substack{1 \le s \le S \\ 1 \le t \le T}}, \tag{5}$$

where $\mathbb{1}(\text{cond})$ is an indicator function that returns 1 if cond is satisfied and 0 otherwise. Note that the threshold value in (5) is always set to 0.5 in this paper for simplicity.

The conventional EEND is implemented as a composition of an embedding part $g : \mathbb{R}^{F \times T} \to \mathbb{R}^{D \times T}$ and a classification part $h : \mathbb{R}^{D \times T} \to (0,1)^{S \times T}$, i.e.,

$$f_{\mathsf{EEND}} = h \circ g. \tag{6}$$

The first embedding part g converts input acoustic features into D-dimensional frame-wise embeddings. It is implemented with N-stacked encoders, each of which converts a flexible length of embedding sequence $(\boldsymbol{e}_t^{(n-1)})_{t=1}^T$ into the same length of embedding sequence $(\boldsymbol{e}_t^{(n)})_{t=1}^T$ as

$$\boldsymbol{e}_1^{(n)}, \dots, \boldsymbol{e}_T^{(n)} = g^{(n)}\left(\boldsymbol{e}_1^{(n-1)}, \dots, \boldsymbol{e}_T^{(n-1)}\right), \tag{7}$$

$$\boldsymbol{e}_t^{(0)} = \boldsymbol{x}_t \quad (1 \le t \le T), \tag{8}$$

where $g^{(n)}$ is the n-th encoder layer. As examples of encoders, bi-directional long short-term memories (BLSTM) [12] and Transformers [13] are exploited in the conventional studies. In this paper, we used Transformer encoders but without positional encodings to prevent the outputs from being affected by the absolute position of the frames. Hereafter, for simplicity, we use $\boldsymbol{e}_t$ to denote the embeddings from the last encoder, i.e., $\boldsymbol{e}_t := \boldsymbol{e}_t^{(N)}$ for $t \in \{1, \dots, T\}$.

Then, the classification part h in (6) converts the embeddings $(\boldsymbol{e}_t)_{t=1}^T$ to posteriors of speech activities $(\boldsymbol{p}_t)_{t=1}^T$ in (3). It is implemented by using a fully connected layer and an element-wise sigmoid function $\sigma(\cdot)$ that takes a tensor as an argument:

$$[\boldsymbol{p}_1, \dots, \boldsymbol{p}_T] = h(\boldsymbol{e}_1, \dots, \boldsymbol{e}_T; W_{\text{cls}}, \boldsymbol{b}_{\text{cls}}) \tag{9}$$

$$= \sigma\left(W_{\text{cls}}^\mathsf{T}[\boldsymbol{e}_1, \dots, \boldsymbol{e}_T] + \boldsymbol{b}_{\text{cls}} \mathbf{1}_D^\mathsf{T}\right) \in (0,1)^{S \times T}, \tag{10}$$

where $(\cdot)^\mathsf{T}$ denotes the matrix transpose, $\mathbf{1}_D$ is D-dimensional all-one vector, and $W_{\text{cls}} \in \mathbb{R}^{D \times S}$ and $\boldsymbol{b}_{\text{cls}} \in \mathbb{R}^S$ are the weight and bias of the fully connected layer, respectively.

EEND outputs posteriors of multiple speakers simultaneously but without any conditions to decide the order of the speakers. Such a network is optimized by using a permutation-free objective [41], [51], which was originally proposed for multi-talker speech separation. It computes the loss for all possible speaker assignments between predictions $(\boldsymbol{p}_t)_{t=1}^T$, as introduced in (3),

and groundtruth labels $(\boldsymbol{y}_t)_{t=1}^T$, and it picks the minimum one for backpropagation as follows.

$$\mathcal{L}_{\text{diar}} = \frac{1}{TS} \min_{\phi \in \Phi(S)} \sum_{t=1}^{T} H\left(\boldsymbol{y}_t^{\phi}, \boldsymbol{p}_t\right), \tag{11}$$

where $\Phi(S)$ is a set of all possible permutations of the sequence $(1, \ldots, S)$, $\boldsymbol{\phi} := (\phi_1, \ldots, \phi_S)$ is the permuted sequence, $\boldsymbol{y}_t^{\phi} := [y_{\phi_1,t}, \ldots, y_{\phi_S,t}]^{\mathsf{T}} \in \{0,1\}^S$ is the permuted groundtruth labels using $\boldsymbol{\phi}$, and $H(\cdot,\cdot)$ is the binary cross entropy defined as

$$H\left(\boldsymbol{y}_t, \boldsymbol{p}_t\right) := \sum_{s=1}^{S} \left\{-y_{s,t} \log p_{s,t} - (1 - y_{s,t}) \log\left(1 - p_{s,t}\right)\right\}. \tag{12}$$

Compared with cascaded approaches, EEND has two significant strengths. One is that the cascaded approaches conduct diarization by dividing frame-wise speaker embeddings, so they require SAD as pre-processing and overlap detection and assignment as post-processing. In contrast, EEND estimates each speaker's speech activities independently, so no extra modules for speech activity detection and overlap detection are needed. The other strength is that the EEND model can be adapted to the desired domain's dataset, while cascaded approaches typically tune only probabilistic linear discriminant analysis (PLDA) parameters to optimize intra- and inter-speaker similarity between speaker embeddings [9], [18], [52].

B. Attractor-Based End-to-End Neural Diarization

The limitation of the conventional EEND is in the classification part h in (6); the number of output speakers S is fixed by the fully connected layer as in (10). One possible way to treat a flexible number of speakers with this fixed-output architecture is to set the number of outputs to be large enough. However, as discussed in Section II-B, it requires knowing the maximum number of speakers in advance, and it has been already verified that such a strategy results in poor performance (see [16]). It is also a problem that the calculation cost of the permutation-free loss increases if we set a large number of speakers to be output. Therefore, a significant research question is how to output diarization results for a flexible number of speakers.

In this paper, we extend the conventional EEND to handle a flexible number of speakers. We assume that the embedding part g in (6) is implemented in the same manner as the conventional EEND described in Section III-A. Given frame-wise D-dimensional embeddings $\{\boldsymbol{e}_t\}_{t=1}^T$, our goal is to produce posteriors for a flexible number of speakers in the classification part h. To achieve this goal, we propose a method to calculate a flexible number of speaker-wise attractors from embeddings and then calculate diarization results on the basis of attractors and embeddings. The proposed method is depicted in Fig. 1.

1) EDA: Encoder-Decoder-Based Attractor Calculation: EDA converts frame-wise embeddings into speaker-wise attractors using a sequence-to-sequence method with an LSTM encoder-decoder. The LSTM encoder h^{enc} takes the frame-wise embeddings as input and updates its hidden state $\boldsymbol{h}_t^{\mathsf{enc}}$ and cell state $\boldsymbol{c}_t^{\mathsf{enc}}$ as

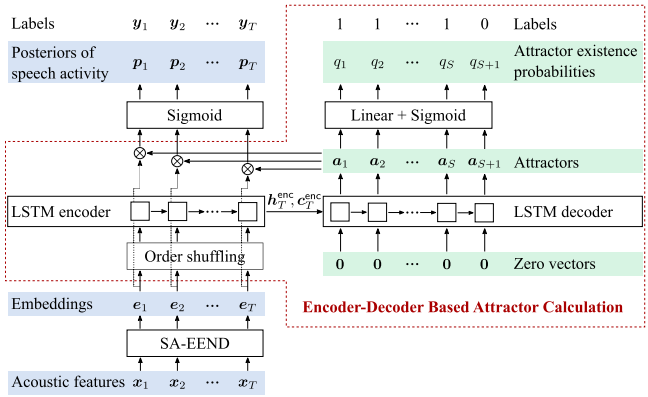


Fig. 1. EEND with encoder-decoder-based attractor calculation (EEND-EDA).

$$\boldsymbol{h}_t^{\mathsf{enc}}, \boldsymbol{c}_t^{\mathsf{enc}} = h^{\mathsf{enc}}\left(\boldsymbol{e}_t, \boldsymbol{h}_{t-1}^{\mathsf{enc}}, \boldsymbol{c}_{t-1}^{\mathsf{enc}}\right) \qquad (t = 1, \ldots, T). \tag{13}$$

The hidden and cell states of the encoder are initialized with zero vectors, i.e., $\boldsymbol{h}_0^{\mathsf{enc}} = \boldsymbol{c}_0^{\mathsf{enc}} = \boldsymbol{0}$. The LSTM decoder h^{dec} estimates speaker-wise attractors as

$$\boldsymbol{h}_s^{\mathsf{dec}}, \boldsymbol{c}_s^{\mathsf{dec}} = h^{\mathsf{dec}}\left(\boldsymbol{0}, \boldsymbol{h}_{s-1}^{\mathsf{dec}}, \boldsymbol{c}_{s-1}^{\mathsf{dec}}\right) \qquad (s = 1, 2, \ldots). \tag{14}$$

We treat the hidden state at each step $\boldsymbol{h}_s^{\mathsf{dec}} =: \boldsymbol{a}_s \in (-1, 1)^D$ as speaker s's attractor, whose dimensionality D is the same as that of the frame-wise embeddings $\boldsymbol{e}_t$. The hidden and cell states of the decoder are initialized by the final hidden and cell states of the encoder as

$$\boldsymbol{h}_0^{\mathsf{dec}} = \boldsymbol{h}_T^{\mathsf{enc}}, \tag{15}$$

$$\boldsymbol{c}_0^{\mathsf{dec}} = \boldsymbol{c}_T^{\mathsf{enc}}, \tag{16}$$

which is shown as a right arrow from the LSTM encoder to the LSTM decoder in Fig. 1. In general applications of a sequence-to-sequence method, e.g., speech recognition or machine translation, the output is sentences, i.e., a sequence of words, so the order of output is fixed. However, EDA cannot determine the order of output speakers in advance because this order is determined by minimizing cross entropy as in (11). Even if the order could be predetermined, it would not be possible to determine the optimal attractor outputs. Thus, the well-known strategy of teacher forcing, for which the optimal outputs with their order have to be known in advance, cannot be used. Furthermore, the s-th attractor can correspond to any speaker that is not contained in the first $(s-1)$ attractors. To make this attractor calculation procedure fully order-free, we input a zero vector as input at each step as in (14). Using zero vectors as inputs provides flexibility to change the number of output speakers across pretraining and finetuning rather than using, for example, trainable parameters. This is why we chose an LSTM-based encoder-decoder rather than Transformer encoder-decoder, which requires input queries rather than zero vectors.

Here, the input order to the EDA encoder affects the output attractors because EDA is based on a sequence-to-sequence method. To investigate the effect of the input order, we tried

two types of input orders: chronological and shuffled orders. In the chronological order setting, embeddings are input in the order of frame indexes as in (13). In the shuffled order setting, we use the following instead of (13):

$$\boldsymbol{h}_t^{\mathsf{enc}}, \boldsymbol{c}_t^{\mathsf{enc}} = h^{\mathsf{enc}}\left(\boldsymbol{e}_{\psi_t}, \boldsymbol{h}_{t-1}^{\mathsf{enc}}, \boldsymbol{c}_{t-1}^{\mathsf{enc}}\right) \qquad (t = 1, \ldots, T)\,, \quad (17)$$

where $(\psi_1, \ldots, \psi_T)$ is a randomly chosen permutation of $(1, \ldots, T)$.

The diarization results $\boldsymbol{p}_t$ in (3) are calculated on the basis of the dot product of the frame-wise embeddings and speaker-wise attractors ($\otimes$ in Fig. 1):

$$\boldsymbol{p}_t = \sigma\left(A^{\mathsf{T}} \boldsymbol{e}_t\right) \in (0,1)^S, \quad (18)$$

where $A := [\boldsymbol{a}_1, \ldots, \boldsymbol{a}_S]$ are the speaker-wise attractors. The posteriors are optimized by using (11) in the same manner as the conventional EEND. This posterior calculation no longer depends on the fully connected layer, which determines the output number of speakers as in (10); therefore, EDA-based diarization can vary the output number of speakers.

Comparing (10) and (18), the conventional EEND can also be regarded as using fixed attractors W_{cls} (with bias $\boldsymbol{b}_{\text{cls}}$). In comparison, EDA calculates attractors from an input sequence of embeddings, which makes attractors adaptive to the embeddings. This makes EEND-EDA more accurate even under the fixed-number-of-speakers condition (see Table III).

2) Attractor Existence Probability: As in (14), we can obtain an infinite number of attractors. To decide when to stop the attractor calculation, we calculate the attractor existence probabilities from the calculated attractors by using a fully connected layer followed by sigmoid activation:

$$q_s = \sigma\left(\boldsymbol{w}_{\text{exist}}^{\mathsf{T}} \boldsymbol{a}_s + b_{\text{exist}}\right), \quad (19)$$

where $\boldsymbol{w}_{\text{exist}} \in \mathbb{R}^D$ and $b_{\text{exist}} \in \mathbb{R}$ are trainable weights and bias parameters of the fully connected layer, respectively.

During training, we know the oracle number of speakers S, so the training objective of the attractor existence probabilities is based on the first $(S+1)$-th attractors using the binary cross entropy defined in (12):

$$\mathcal{L}_{\text{exist}} = \frac{1}{S+1} H\left(\boldsymbol{l}, \boldsymbol{q}\right), \quad (20)$$

where

$$\boldsymbol{l} := [\underbrace{1, \ldots, 1}_{S}, 0]^{\mathsf{T}}, \quad (21)$$

$$\boldsymbol{q} := [q_1, \ldots, q_{S+1}]^{\mathsf{T}}. \quad (22)$$

The total loss is defined as the weighted sum of $\mathcal{L}_{\text{diar}}$ in (11) and $\mathcal{L}_{\text{exist}}$ in (20) with the weighting parameter $\alpha \in \mathbb{R}_+$ as

$$\mathcal{L} = \mathcal{L}_{\text{diar}} + \alpha \mathcal{L}_{\text{exist}}. \quad (23)$$

In this paper, we use $\alpha = 1$. This multi-task loss aims to optimize frame- and speaker-wise posteriors with $\mathcal{L}_{\text{diar}}$ and attractor existence probabilities with $\mathcal{L}_{\text{exist}}$.

While (23) was used for the network optimization in our previous study [15], we found that the optimization of $\mathcal{L}_{\text{exist}}$ inhibits the minimization of $\mathcal{L}_{\text{diar}}$ during the training of a model with a flexible number of speakers, which is more important for improving diarization accuracy. Therefore, when a flexible number of speakers' dataset is used for training, we use $\mathcal{L}_{\text{exist}}$ to update only the fully connected layer parameterized by $\boldsymbol{w}_{\text{exist}}$ and b_{exist} in (19). This can be implemented by cutting the graph before the fully connected layer to disable backpropagation to the preceding layers.

During inference, we cannot access the oracle number of speakers; thus, it is estimated using q_s in (19) as follows.

$$\hat{S} = \min\left\{s \mid s \in \mathbb{Z}_+ \wedge q_{s+1} < \tau\right\}, \quad (24)$$

where $\tau \in (0,1)$ is a thresholding parameter, which is set to 0.5 in this paper. We then use the first $\hat{S}$ attractors to calculate posteriors as in (18).

C. Inference Methodology

1) SAD Post-Processing: Diarization methods, especially cascaded ones, are sometimes evaluated with oracle speech segments. When evaluated in such a way, the comparison between cascaded methods and EEND-methods becomes hard, mainly because EEND-based methods perform SAD and diarization simultaneously. One reason evaluations of cascaded approaches are mainly based on oracle speech segments is to consider speaker errors and SAD errors separately. It is reasonable to use oracle speech segments to focus on reducing speaker errors. However, such segments are not accessible in real scenarios, and the existence of SAD errors may worsen the clustering performance, which directly affects the diarization accuracy. Thus, we believe that SAD errors should also be considered in the context of cascaded methods. However, it is hard to say how accurate the SAD should be for a fair comparison between cascaded and EEND-based methods. Therefore, to align with the the cascaded methods, we introduce SAD post-processing for evaluating EEND. With this method, we can conduct a fair comparison between cascaded and EEND-based methods with the same SAD. Note that it can be used to improve the diarization performance by eliminating false alarm speech and recovering missed speech when an accurate external SAD system is given.

The SAD post-processing algorithm is described in Algorithm 1. Here, we assume that we have SAD results $z_1, \ldots, z_T$ in addition to frame- and speaker-wise posteriors $\boldsymbol{p}_1, \ldots, \boldsymbol{p}_T$. We first estimate speech activities as usual by using (5) (line 1). However, this estimation is not always consistent with SAD results. Thus, we first filter false alarms (FA) by using SAD results. For each frame (line 2), if it is estimated that some speakers are active while the speech activity should be zero (line 3), we update the estimations with a zero vector (line 4). This procedure will always improve DER if $z_1, \ldots, z_T$ are the oracle speech activities. We also recover missed frames (MI) if no speaker is estimated as active while the speech activity is one (line 5). For each of such frames, we treat the speaker with the highest posterior as an active speaker (line 6–line 7). Including the oracle SAD as input will also improve the DER because missed-frame errors are replaced by correct estimation or at least speaker errors.

Algorithm 1: SAD post-processing.

Input : $(\boldsymbol{p}_1, \ldots, \boldsymbol{p}_T) \in (0,1)^{S \times T}$ // Frame-wise posteriors
$(z_1, \ldots, z_T) \in \{0,1\}^T$ // SAD results
Output: $(\hat{\boldsymbol{y}}_1, \ldots, \hat{\boldsymbol{y}}_T) \in \{0,1\}^{S \times T}$ // Speech activities

1 Compute $\hat{\boldsymbol{y}}_1, \ldots, \hat{\boldsymbol{y}}_T$ using (5) // Initial results
2 **foreach** $t \in \{1, \ldots, T\}$ **do**
3 **if** $\|\hat{\boldsymbol{y}}_t\|_1 > 0 \wedge z_t = 0$ **then** // Filter FA
4 $\hat{\boldsymbol{y}}_t \leftarrow [0, \ldots, 0]^\mathsf{T}$
5 **else if** $\|\hat{\boldsymbol{y}}_t\|_1 = 0 \wedge z_t = 1$ **then** // Recover MI
6 $s^* \leftarrow \arg\max_{s \in 1, \ldots, S} \boldsymbol{p}_t$
7 $\hat{\boldsymbol{y}}_t \leftarrow [0, \ldots, 0, \underset{\hat{s^*}}{1}, 0, \ldots, 0]^\mathsf{T} \in \{0,1\}^S$

2) Iterative Inference: Even if the model is trained to output a flexible number of speakers, the output number of speakers is empirically limited by the maximum number of speakers in a recording observed during pre-training (see Table VII). How to output the results of more than N speakers even if the model is trained on at most N-speaker mixtures is still an open question. In this paper, we propose an iterative inference method to produce results for more than N speakers by applying EEND decoding with iterative frame selection.

Preliminarily, we first reveal the characteristics of the EEND models that consist of stacked Transformer encoders and EDA. A Transformer encoder involves neither recurrence nor convolutional calculation, and we do not use positional encoding in this paper; thus, the embedding part g in (6) is an order-free transformation. EDA contains an LSTM encoder-decoder, but if the order of the input sequence to EDA is shuffled, we can say that EDA does not depend on the input order, so the EDA's classification part h in (6) is also an order-free function. Therefore, EEND-EDA does not depend on the order of the input features, which makes it possible to process features that are not extracted at equal intervals along the time axis, as in EEND as post-processing [53]. The proposed iterative inference also utilizes this characteristic.

Algorithm 2 shows the algorithm of iterative inference. In the algorithm, two processes are iteratively conducted: decoding and silence frame selection. Each process at the n-th iteration is described as follows.

1) *Decoding* (line 3): Acoustic features $\boldsymbol{x}_t$ of the selected frames $\mathcal{T}$ are fed into EEND, and the corresponding posteriors $\boldsymbol{p}_t^{(n)} \in (0,1)^{S^{(n)}}$ are obtained as

$$\left(\boldsymbol{p}_t^{(n)}\right)_{t \in \mathcal{T}} \leftarrow f_{\mathsf{EEND}}\left((\boldsymbol{x}_t)_{t \in \mathcal{T}}\right), \tag{25}$$

where $S^{(n)} \in \{0, \ldots, S_{\max}\}$ is the number of decoded speakers. The posteriors of the frames that are not in $\mathcal{T}$ are set to zero as

$$\boldsymbol{p}_t^{(n)} \leftarrow \underbrace{[0, \ldots, 0]^\mathsf{T}}_{S^{(n)}} \quad (t \in \{1, \ldots, T\} \setminus \mathcal{T}). \tag{26}$$

Algorithm 2: Iterative inference.

Input : $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$ // Acoustic features
f_{EEND} // EEND model
$S_{\max} \in \mathbb{N}$ // Max #Speakers that EEND can output
Output: $\hat{Y} \in \{0,1\}^{S \times T}$

1 $\mathcal{T} \leftarrow \{1, \ldots, T\}$ // Frame set
2 **for** $n \leftarrow 1$ **to** ∞ **do**
3 Compute $\hat{Y}^{(n)}$ by (25), (26), and (5) // Decoding
4 Update $\mathcal{T}$ by (27) // Silence frame selection
5 **if** $S^{(n)} < S_{\max} \vee |\mathcal{T}| = 0$ **then**
6 **break**
7 $\hat{Y} \leftarrow \begin{bmatrix} \hat{Y}^{(1)} \\ \vdots \\ \hat{Y}^{(n)} \end{bmatrix}$

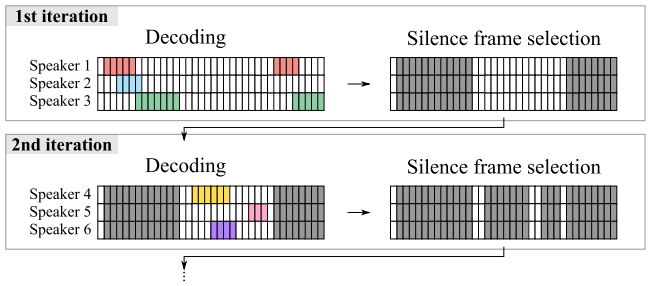


Fig. 2. Iterative inference in the case of $S_{\max} = 3$.

With the posteriors $\boldsymbol{p}_t^{(n)}$ for $t \in \{1, \ldots, T\}$, diarization results $\hat{Y}^{(n)} = (\hat{\boldsymbol{y}}_1^{(n)}, \ldots, \hat{\boldsymbol{y}}_T^{(n)})$ are computed using (5). Note that $\hat{Y}^{(n)}$ corresponds to the speech activities of the $((n-1)S_{\max}+1)$-th through $((n-1)S_{\max}+S^{(n)})$-th speakers.

2) *Silence frame selection* (line 4): Given the diarization results decoded at the n-th iteration, we select the frames in which no speaker is active to update $\mathcal{T}$ as

$$\mathcal{T} \leftarrow \left\{ t \;\middle|\; t \in \mathcal{T}, \left\|\hat{\boldsymbol{y}}_t^{(n)}\right\|_1 = 0 \right\}. \tag{27}$$

The above processes start with the initial value of $\mathcal{T}$ as the set of all frames $\{1, \ldots, T\}$ (line 1), and last until $\mathcal{T}$ becomes the empty set or when it is assumed that all the speakers are decoded (line 5–line 6). Here, we assume that all the speakers are decoded if the number of output speakers $S^{(n)}$ is smaller than the maximum output of EEND $S_{\max}$.

After the iterative process is finished, the final results $\hat{Y}$ are obtained by concatenating the results calculated at each iteration (line 7). With iterative inference, the number of speakers to be decoded is no longer limited by the training dataset. The iterative inference workflow when $S_{\max} = 3$ is also illustrated in Fig. 2.

3) Iterative Inference With DOVER-Lap (Or Iterative Inference+): Despite iterative inference being able to produce more than $S_{\max}$ speakers' speech activities, it has a potential problem in that the speech activities of two speakers decoded at different iterations never overlap. For example, the $(S_{\max}+1)$-th speaker's speech activities never overlap with those of the first

Algorithm 3: Iterative inference with DOVER-Lap (or iterative inference+).

Input : $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T$ // Acoustic features
f_{EEND} // EEND model
$S_{\max} \in \mathbb{N}$ // Max #Speakers that EEND can output
Output: $\hat{Y} \in \{0,1\}^{S \times T}$

1 **for** $S_{\text{limit}} = 1$ **to** $S_{\max}$ **do**
2 $\mathcal{T} \leftarrow \{1, \ldots, T\}$ // Frame set
3 **for** $n \leftarrow 1$ **to** ∞ **do**
4 Compute $\hat{Y}^{(n)}$ by (25), (26), (5) // Decoding
5 **if** $n = 1$ **then**
6 Limit the number of speakers in $\hat{Y}^{(n)}$ by (28)
7 Update $\mathcal{T}$ by (27) // Silence frame selection
8 **if** $S^{(n)} < S_{\max} \vee |\mathcal{T}| = 0$ **then**
9 **break**
10 $\hat{Y}_{S_{\text{limit}}} \leftarrow \begin{bmatrix} \hat{Y}^{(1)} \\ \vdots \\ \hat{Y}^{(n)} \end{bmatrix}$
11 $\hat{Y} \leftarrow \text{DOVER-Lap}\left(\hat{Y}_1, \ldots, \hat{Y}_{S_{\max}}\right)$

$S_{\max}$ speakers. This is because the frames in which the first $S_{\max}$ speakers are active will not be processed in the second iteration. To ease this problem, we introduce DOVER-Lap [54], which is the extension of DOVER [55]. Both of them are methods for combining multiple diarization results on the basis of majority voting, but unlike DOVER, DOVER-Lap take speaker overlap into account. We used a modified version of DOVER-Lap presented in [18], in which the speaker assignment strategy when multiple speakers were ranked equally was slightly different from the original DOVER-Lap [54]. Note that we did not use a hypothesis-wise weighting of DOVER-Lap, which is also introduced in [18].

The algorithm of iterative inference incorporated with DOVER-Lap is shown in Algorithm 3. In this paper, we refer to this inference as iterative inference+. The difference from the iterative inference in Algorithm 2 is that we limit the number of speakers to decode at the first iteration with $S_{\text{limit}} (\leq S_{\max})$ (line 5–line 6). After the decoding step at the first iteration using (25), (26), and (5), we choose at most the first S_{limit} speakers' speech activities from $\hat{Y}^{(1)} := (\hat{y}_{s,t})_{s,t}$ as

$$\hat{Y}^{(1)} \leftarrow (\hat{y}_{s,t})_{\substack{1 \leq s \leq \min\left(S^{(1)}, S_{\text{limit}}\right) \\ 1 \leq t \leq T}} . \tag{28}$$

The other procedures are the same as those in Algorithm 2, and finally, we obtain S_{limit}-wise diarization results $Y_{S_{\text{limit}}}$ (line 10).

In iterative inference+, S_{limit} is varied from 1 to $S_{\max}$ (line 1), which results in $S_{\max}$ diarization results for each recording. We then combine them by using DOVER-Lap to obtain the final result $\hat{Y}$ (line 11). With this procedure, the k-th speaker's speech activities can be overlapped with those of the $\max(1, (k - S_{\max} + 1))$-th to $(k + S_{\max} - 1)$-th speakers.

TABLE I
DATASETS OF SIMULATED MIXTURES

Dataset	Split	#Spk	#Mixtures	β	Overlap ratio (%)
Sim1spk	Train	1	100,000	2	0.0
	Test	1	500	2	0.0
Sim2spk	Train	2	100,000	2	34.1
	Test	2	500	2	34.4
	Test	2	500	3	27.3
	Test	2	500	5	19.1
Sim3spk	Train	3	100,000	5	34.2
	Test	3	500	5	34.7
	Test	3	500	7	27.4
	Test	3	500	11	19.2
Sim4spk	Train	4	100,000	9	31.5
	Test	4	500	9	32.0
Sim5spk	Train	5	100,000	13	30.3
	Test	5	500	13	30.7

IV. EXPERIMENTS

A. Datasets

1) Simulated Datasets: To train the EEND-EDA model, we created simulated speech mixtures from single-speaker recordings of the following corpora.

- Switchboard-2 (Phase I & II & III)
- Switchboard Cellular (Part 1 & 2)
- NIST Speaker Recognition Evaluation (2004 & 2005 & 2006 & 2008)

Note that these corpora are compatible with the Kaldi CALLHOME x-vector recipe[1].

We used the following simulation protocol to create multi-talker mixtures from single-speaker recordings:

1) Select N speakers,
2) For each speaker, randomly sample speech segments and concatenate them with silences that are interlaid between speech segments,
3) For each of the N long recordings created, randomly select a room impulse response and convolve it with the recording,
4) Mix the N long recordings and a noise signal with a randomly determined signal-to-noise ratio.

The detailed algorithm for creating simulated mixtures can be found in [12]. In the second process, we assume that the occurrence of an utterance is a Poisson process, so the duration of the silence between speech segments follows the exponential distribution $\frac{1}{\beta} \exp(-\frac{x}{\beta})$, where β is the mean value. β can be used to control the overlap ratio of the mixtures. To obtain a similar overlap ratio among various numbers of speakers, we varied β according to the number of speakers as summarized in Table I.

2) Real Datasets: For real datasets, we employed five multi-talker datasets below.

- *CALLHOME* [56]: A dataset that consists of telephone conversations whose average duration is two minutes. We used

[1][Online]. Available: https://github.com/kaldi-asr/kaldi/tree/master/egs/callhome_diarization/v2

TABLE II
DATASETS OF REAL RECORDINGS

Dataset	Split	#Spk	#Mixtures	Overlap ratio (%)
CALLHOME-2spk [56]	Part 1	2	155	14.0
	Part 2	2	148	13.1
CSJ [57]	—	2	54	20.1
CALLHOME-3spk [56]	Part 1	3	61	19.6
	Part 2	3	74	17.0
CALLHOME [56]	Part 1	2–7	249	17.0
	Part 2	2–6	250	16.7
AMI headset mix [2]	Train	3–5	136	13.4
	Dev	4	18	14.1
	Test	3–4	16	14.6
DIHARD II [38]	Dev	1–10	192	9.8
	Test	1–9	194	8.9
DIHARD III [39]	Dev	1–10	254	10.7
	Test (Core)	1–9	184	8.8
	Test (Full)	1–9	259	9.2

the splits provided in the Kaldi x-vector recipe[1], which are denoted as Part 1 and Part 2, respectively. Two- and three-speaker subsets were used in the fixed-number-of-speakers evaluations, which are denoted as CALLHOME-2spk and CALLHOME-3spk.

- *CSJ* [57]: A dataset that consists of monologues and dialogues of Japanese speech. In this paper, we used the dialogue part of the dataset. The average duration of the recordings is about 13 minutes. Following [58], we used 54 dialogue recordings out of 58.
- *AMI headset mix* [2]: A meeting dataset that consists of 100 hours of multi-modal meeting recordings. Each meeting session is about 30 minutes. We used *headset mix* recordings, which were obtained by mixing the headset recordings of all the participants. We used the split and reference RTTMs provided in the VBx paper [35].
- *DIHARD II* [38]: A dataset used in the second DIHARD challenge. We used single-channel audio, which is used for tracks 1 and 2. The dataset consists of recordings from 11 domains (including telephone data) with an average duration of about 7 minutes.
- *DIHARD III* [39]: A dataset used in the third DIHARD challenge. It also consists of recordings from 11 domains (including telephone data) with an average duration of about 8 minutes. The test set has two evaluation conditions called *core* and *full*. The core set is a subset of the full set, in which the recordings are selected to balance the duration of each domain. In terms of the number of speakers, the full set contains more recordings of two speakers than the core set.

Their statistics are summarized in Table II. Note that the recordings in CSJ, AMI, DIHARD II, and DIHARD III were sampled at 16 kHz, so we downsampled them to 8 kHz to be aligned with those of the simulated datasets. We also note that the recordings of the CSJ corpus are in stereo, so we mixed them to create monaural recordings.

B. Training

For the embedding part g in (6) of the proposed EEND-EDA, we used four-stacked Transformer encoders with four attention heads without positional encodings, each of which outputs 256-dimensional frame-wise embeddings. The inputs for the model were log-scaled Mel-filterbank-based features. We first extracted 23-dimensional log-scaled Mel-filterbanks with a frame length of 25 ms and frame shift of 10 ms. Each of them was then concatenated with those of the preceding and following seven frames, followed by subsampling with a factor of 10. As a result, a $345\,(= 23 \times 15)$ dimensional acoustic feature was extracted for each 100 ms.

In this paper, we evaluated EEND-EDA for both fixed-numbers-of-speakers and unknown-numbers-of-speakers conditions; thus, a model was trained for each purpose. For the fixed-number-of-speakers evaluation, the model was first trained on the Sim$k$spk training set for 100 epochs and evaluated on the Sim$k$spk test set. We also adapted the model to CALLHOME-$k$spk for another 100 epochs to evaluate the model on real recordings. We used $k \in \{2, 3\}$ in this paper. For the unknown-number-of-speakers evaluation, the model that was trained on Sim2spk was finetuned by using the concatenation of Sim{1,2,3,4}spk or Sim{1,2,3,4,5}spk for 50 epochs. The model was also adapted to each target dataset for another 500 epochs.

For network training using simulated mixtures, we used the Adam optimizer [59] with the Noam scheduler [60] with 100,000 warm-up steps. For adaptation, we also used the Adam optimizer but with a fixed learning rate of 1×10^{-5}. For efficient batch processing during training, we split each recording into 500 frames when using Sim$k$spk and 2000 frames when using the adaptation sets. The batch size for training was set to 64. Note that an entire recording is fed into the network without splitting during inference.

C. Evaluation

As an evaluation metric, we used diarization error rates (DERs) defined as

$$\mathrm{DER} = \frac{T_{\mathrm{MI}} + T_{\mathrm{FA}} + T_{\mathrm{CF}}}{T_{\mathrm{Speech}}}, \tag{29}$$

where T_{Speech}, T_{MI}, T_{FA}, and T_{CF} denote the duration of total speech, missed speech, false alarm speech, and speaker confusion, respectively. Following the prior work in [12], [61], we used 0.25 sec of collar tolerance at each speech boundary for the Sim$k$spk, CALLHOME, and CSJ evaluation. For AMI, DIHARD II, and DIHARD III, we allowed no collar tolerance and used a subsampling factor of 5 during inference, which results in acoustic features extracted every 50 ms, to obtain more fine-grained results. We emphasize that speaker overlaps were NOT excluded from the evaluations.

We also report Jaccard error rates (JERs) in addition to DERs. To calculate JER, first, the optimal assignment between reference and system speakers is calculated. JER is the average score

TABLE III
DERs (%) FOR TWO-SPEAKER EVALUATIONS. 0.25 S OF COLLAR TOLERANCE WAS ALLOWED

	Simulated			Real	
Method	$\beta = 2$	$\beta = 3$	$\beta = 5$	CALLHOME-2spk	CSJ
i-vector + AHC	33.74	30.93	25.96	12.10	27.99
x-vector (TDNN) + AHC	28.77	24.46	19.78	11.53	22.96
BLSTM-EEND [12]	12.28	14.36	19.69	26.03	39.33
SA-EEND [13]	4.56	4.50	3.85	9.54	20.48
EEND-EDA (Chronol.)	3.07	2.74	3.04	8.24	18.89
EEND-EDA (Shuffled)	**2.69**	**2.44**	**2.60**	**8.07**	**16.27**

TABLE IV
DERs (%) FOR THREE-SPEAKER EVALUATIONS. 0.25 S OF COLLAR TOLERANCE WAS ALLOWED

	Simulated			Real
Method	$\beta = 2$	$\beta = 3$	$\beta = 5$	CALLHOME-3spk
x-vector (TDNN) + AHC	31.78	26.06	19.55	19.01
SA-EEND [13]	8.69	7.64	6.92	14.00
EEND-EDA (Chronol.)	13.02	11.65	10.41	15.86
EEND-EDA (Shuffled)	**8.38**	**7.06**	**6.21**	**13.92**

of each reference speaker defined as

$$\mathrm{JER} = \frac{1}{S_{\mathrm{ref}}} \sum_{s=1}^{S_{\mathrm{ref}}} \frac{T_{\mathrm{FA}}^{(s)} + T_{\mathrm{MI}}^{(s)}}{T_{\mathrm{Union}}^{(s)}}, \tag{30}$$

where S_{ref} is the number of reference speakers, and $T_{\mathrm{MI}}^{(s)}$ and $T_{\mathrm{FA}}^{(s)}$ are the duration of the missed and false alarm speech calculated between speech activities of the s-th reference speaker and the paired system speaker, respectively. $T_{\mathrm{Union}}^{(s)}$ is the time duration in which at least one of the s-th reference speakers of a paired system speaker is active.

V. RESULTS

A. Fixed Numbers of Speakers

1) Two-Speaker Experiment: First, we evaluated our method under the two-speaker condition. In this case, the model was first trained on Sim2spk and then adapted to CALLHOME-2spk Part 1. For the EEND-based methods, we used the model trained on Sim2spk to evaluate the simulated datasets and the one adapted to CALLHOME-2spk Part 1 to evaluate CALLHOME-2spk Part 2 and CSJ. For EEND-EDA, we used the first two output attractors for speech activity calculation.

Table III shows the results of the two-speaker evaluation. We observed that the proposed method with the shuffled order setting achieved the best DERs. Despite EEND-EDA being designed to deal with flexible numbers of speakers, it outperformed the conventional EENDs, i.e., BLSTM-EEND and SA-EEND, which output diarization results for fixed numbers of speakers. This is because the conventional EEND can be regarded as a fixed-attractor-based method, while EEND-EDA is an adaptive-attractor-based method as described in the last paragraph of Section III-B. This flexibility of attractors makes the proposed method more accurate even in fixed-number-of-speakers evaluations. In terms of the order of the input to EDA, shuffled sequences always performed better than chronologically ordered sequences. It indicates that the global context is more important than the temporal context to calculate attractors.

2) Three-Speaker Experiment: We also evaluated the method under the three-speaker condition. We first trained the model on Sim3spk and then adapted it to CALLHOME-3spk Part 1. We validated the performance on Sim3spk using the model trained on Sim3spk and that on CALLHOME-3spk Part 2 using the model adapted to CALLHOME-3spk Part 1. We used the first three attractors to evaluate EEND-EDA's performance. As shown in Table IV, EEND-EDA with sequence shuffling performed best on both simulated and real datasets.

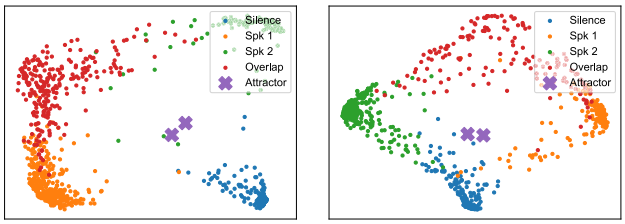
3) Effect of Input Order: For a better understanding of EDA, we tried various types of sequences as inputs to the models, each of which was trained on chronologically ordered sequences and shuffled sequences. We evaluated matched and unmatched conditions of orders, and we also evaluated the effect of reducing the sequence length by subsampling or using the last $1/N$ part of the sequences. Table V shows the results on Sim2spk ($\beta = 2$). The EEND-EDA that was trained on chronologically ordered sequences performed well on chronologically ordered sequences but did poorly on shuffled sequences. It was also affected by subsampling, while it was slightly influenced by using the last $1/N$ part. These results indicate that the length of each utterance is an important factor to decide the output attractors for the model trained on chronologically ordered sequences. On the other hand, when the model was trained on shuffled sequences, it was not that affected by the order of sequences nor subsampling. However, when the last $1/N$ of the sequences were used, its performance degradation was worse than the model trained on chronologically ordered sequences. These results indicate that EDA trained on shuffled sequences captured the distribution of embeddings; thus, subsampling did not affect the performance that much, while using the last $1/N$, i.e., biased sampling, degraded the DERs.

4) Embedding Visualization: For intuitive understanding of the behavior of EDA, we visualized the embeddings $\boldsymbol{e}_t$ and attractors $\boldsymbol{a}_s$ within a two-speaker mixture from Sim2spk ($\beta = 2$) in Fig. 3(b). They were projected to two-dimensional space by using principal component analysis (PCA). We observed that the embeddings of two speakers were well distinguished from those of silence frames, and those of overlapped frames were distributed between the areas of the two speakers. For EEND-EDA, two attractors were calculated for each of the two speakers successfully as in Fig. 3(b). In Fig. 3(a), in comparison, the fixed attractors W_{cls} of the conventional EEND were not well separated compared with the attractors calculated using EDA.

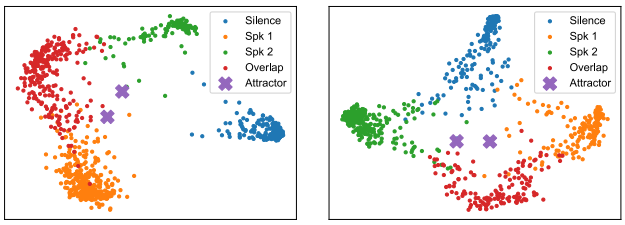
To understand the characteristics of attractors from EDA, we also visualized the inter-mixture relationship of attractors. For visualization, we first chose an anchor speaker and then selected mixtures that contained the anchor speaker. We calculated two attractors from each mixture by using EEND-EDA and mapped them onto a two-dimensional space using PCA. The speaker assignment from the calculated attractors to speaker identifiers was based on the groundtruth labels. Fig. 4 shows the attractors of two-speaker mixtures that contain the same anchor speaker.

TABLE V
DERs FOR SIM2SPK (OVERLAP RATIO: 34.4 %) USING VARIOUS TYPES OF SEQUENCES

	Using whole sequence		Subsample $1/N$					Using the last $1/N$				
Method	Chronol.	Shuffled	$N=2$	$N=4$	$N=8$	$N=16$	$N=32$	$N=2$	$N=4$	$N=8$	$N=16$	$N=32$
EEND-EDA (Train: Chronol.)	3.07	30.04	3.54	7.32	14.48	21.13	27.18	3.67	4.97	5.40	6.11	7.68
EEND-EDA (Train: Shuffled)	2.69	2.69	2.70	2.68	2.79	3.09	5.08	3.36	5.92	7.46	8.59	10.65



(a) Conventional EEND [13]



(b) EEND-EDA

Fig. 3. Visualization of embedding and attractors within each recording. For conventional EEND, weights of last fully connected layer W_{cls} were visualized instead of attractors. (a) Conventional EEND [13] (b) EEND-EDA.

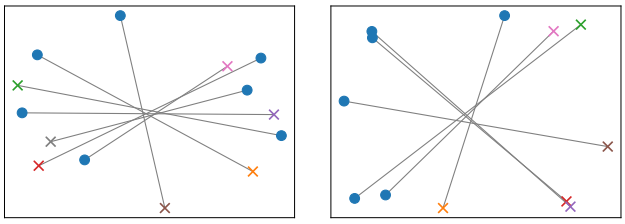


Fig. 4. Visualization of attractors across recordings. Selected speakers' attractors are marked by dots, and their interference speakers' attractors are marked by crosses. Colors of crosses correspond to speaker identities within each figure. Each pair of attractors from same mixture are connected with gray line.

It clearly shows that the each anchor speaker's attractors were not distributed near each other.

From these results, the embeddings and attractors were calculated only to separate speakers in each mixture. We can also say that the attractors were not suited for speaker identification. This also supports the idea that attractors are adaptively calculated from input embeddings. A similar observation on attractors from DANet [48] in speech separation was provided in Section V of [62] that attractors cannot be used for speaker identification or tracing.

5) Evaluation on the Mismatched Number of Speakers: We also evaluated two-speaker EEND-EDA on three-speaker datasets, and three-speaker EEND-EDA on two-speaker datasets. We used the model trained on Sim2spk or Sim3spk for the evaluation on the simulated datasets, and used the model adapted to CALLHOME-2spk or CALLHOME-3spk for the evaluation on the real datasets. The order of the embeddings is shuffled before being fed into EDA. The results are shown in Table VI. It is clearly observed that the DERs degraded when the number of speakers during training and inference was different. It is worth mentioning that three-speaker EEND-EDA did not work well on the two-speaker datasets; this indicates that the larger number of speakers during training does not serve the smaller number of speakers during inference.

TABLE VI
DERs (%) OF CROSS EVALUATIONS OF TWO- AND THREE-SPEAKER EEND-EDA. 0.25 S OF COLLAR TOLERANCE WAS ALLOWED

	Two-speaker datasets		Three-speaker datasets	
Model	Sim2spk ($\beta=2$)	CALLHOME -2spk	Sim3spk ($\beta=5$)	CALLHOME -3spk
Two-speaker EEND-EDA	2.69	8.07	28.79	20.80
Three-speaker EEND-EDA	15.12	9.95	8.38	13.92

B. Unknown Numbers of Speakers

1) Simulated Mixtures: To train EEND-EDA to output flexible numbers of speakers' results, we finetuned the model from the two-speaker model for at most 50 epochs using Sim1spk to Sim4spk or Sim1spk to Sim5spk. Table VII shows the step-by-step improvement of the model. Note that the results on the top row correspond to our previous paper [15]. First, disabling backpropagation from the attractor existence loss $\mathcal{L}_{\text{exist}}$ to update only $\boldsymbol{w}_{\text{exist}}$ and b_{exist} improved the DERs for Sim1spk to Sim4spk. However, we observed that the model still did not perform well on Sim5spk, which was not included in the training set. Adding Sim5spk to the training set solved the problem as shown in the third row, which shows DERs that improved for Sim5spk from 23.08 % to 13.70 %. This indicates that EEND-EDA's number of output speakers was empirically limited by its training datasets, even though it does not limit the number of output speakers with its network architecture. Increasing the number of training epochs further improved the DERs as shown in the last row. We also showed the DERs computed by SA-EEND [13] trained on a flexible number of speakers' dataset in the last two rows. In each case, the model's output number of speakers was set to the maximum number of speakers in the dataset, i.e., four or five, and the model was trained to output null speech activities if a recording of a fewer number of speakers was input. EEND-EDA outperformed SA-EEND in all datasets. Hereafter, we use the EEND-EDA model of the fourth row ($k \in \{1, \ldots, 5\}$, 50 epochs, using $\mathcal{L}_{\text{exist}}$ to update only $\boldsymbol{w}_{\text{exist}}$ and b_{exist} during training) and the SA-EEND model of the sixth row ($k \in \{1, \ldots, 5\}$, 50 epochs).

TABLE VII
STEP-BY-STEP IMPROVEMENT ON SIMULATED DATASETS. FOR SIM2SPK AND SIM3SPK, WE USED $\beta = 2$ AND $\beta = 5$, RESPECTIVELY. IN $\mathcal{L}_{\text{EXIST}}$ COLUMN, WE SHOW WHICH PARAMETERS WERE UPDATED USING $\mathcal{L}_{\text{EXIST}}$ DURING TRAINING. RESULTS ON TOP ROW CORRESPOND TO ORIGINAL SETTING [15]

Model	Training data	#Epochs	$\mathcal{L}_{\text{exist}}$	Sim$k$spk				
				$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
EEND-EDA	$k \in \{1,\dots,4\}$	25	Update all the parameters in f_{EEND}	0.39	4.33	8.94	13.76	N/A
	$k \in \{1,\dots,4\}$	25	Update only $\boldsymbol{w}_{\text{exist}}$ and b_{exist}	0.25	4.06	7.68	10.12	23.08
	$k \in \{1,\dots,5\}$	25	Update only $\boldsymbol{w}_{\text{exist}}$ and b_{exist}	0.21	4.22	8.25	10.75	13.70
	$k \in \{1,\dots,5\}$	50	Update only $\boldsymbol{w}_{\text{exist}}$ and b_{exist}	0.36	3.65	7.70	9.97	11.95
SA-EEND	$k \in \{1,\dots,4\}$	50	N/A	0.60	4.39	9.40	13.56	25.22
	$k \in \{1,\dots,5\}$	50	N/A	0.50	3.95	9.18	12.24	17.42

TABLE VIII
DERS (%) OF CALLHOME. 0.25 S OF COLLAR TOLERANCE WAS ALLOWED. TDNN-BASED X-VECTOR RESULTS WERE OBTAINED WITH KALDI RECIPE. DERS OF SINGLE-SPEAKER REGIONS ARE REPORTED IN BRACKETS. AHC: AGGLOMERATIVE HIERARCHICAL CLUSTERING, VB: VARIATIONAL BAYES RESEGMENTATION [34], VBX: VARIATIONAL BAYES HMM CLUSTERING [35]

(a) Results of cross-validation.

Method	SAD	#Speakers						Total
		2	3	4	5	6	7	
SA-EEND	-	8.51	19.84	26.16	36.82	48.52	38.24	19.82 (13.38)
EEND-EDA	-	8.18	15.05	16.54	27.29	31.40	37.23	14.81 (8.68)
X-vector (TDNN) + AHC	TDNN	14.66	18.42	20.46	31.40	32.62	46.43	19.48 (10.25)
X-vector (TDNN) + AHC + VB	TDNN	11.68	17.22	19.71	30.24	32.07	46.49	17.80 (8.29)
SA-EEND	TDNN	7.42	18.10	21.80	31.69	44.61	35.09	17.41 (10.66)
EEND-EDA	TDNN	**6.79**	**13.74**	**15.53**	**25.25**	**27.65**	**34.49**	**13.36** (**7.12**)
X-vector (TDNN) + AHC	Oracle	13.68	17.04	17.89	29.96	32.55	45.20	18.04 (8.54)
X-vector (TDNN) + AHC + VB	Oracle	10.94	15.85	17.40	29.23	33.97	42.69	16.57 (6.63)
X-vector (ResNet101) + AHC + VBx [35]	Oracle	9.83	15.23	14.29	**19.24**	**25.76**	36.25	14.21 (**4.42**)
SA-EEND	Oracle	6.02	16.28	20.26	30.42	43.51	35.09	15.90 (8.99)
EEND-EDA	Oracle	**5.50**	**12.17**	**12.86**	23.17	27.96	**34.08**	**11.72** (5.29)

(b) Results on CALLHOME Part 2.

Method	SAD	DER
SA-EEND	-	21.19
SC-EEND [16]	-	15.75
SAD-OD-fiert SC-EEND [17]	-	15.32
EEND-EDA (From [15])	-	15.29
EEND-EDA	-	**12.88**
X-vector (TDNN) + AHC	TDNN	19.43
X-vector (TDNN) + AHC + VB	TDNN	17.61
SA-EEND	TDNN	19.85
EEND-EDA	TDNN	**13.84**
X-vector (TDNN) + AHC	Oracle	17.02
X-vector (TDNN) + AHC + VB	Oracle	15.57
X-vector (ResNet101) + AHC + VBx [35]	Oracle	13.33
SA-EEND	Oracle	16.79
EEND-EDA	Oracle	**10.46**

TABLE IX
CONFUSION MATRICES FOR SPEAKER COUNTING ON CALLHOME PART 2. X-VECTOR-BASED RESULTS WERE OBTAINED WITH ORACLE SAD, WHILE EEND-BASED RESULTS WERE OBTAINED WITHOUT EXTERNAL SAD

(a) X-vector (TDNN) + AHC (Accuracy=56.4 %)

Pred. #Speakers \ Ref. #Speakers	1	2	3	4	5	6
1	**0**	2	1	0	0	0
2	0	**87**	19	3	0	0
3	0	59	**51**	14	3	2
4	0	2	4	**3**	2	1
5	0	0	0	0	**0**	0
6	0	0	0	0	0	**0**

(b) X-vector (ResNet101) + AHC + VBx [35] (Accuracy=72.0 %)

Pred. #Speakers \ Ref. #Speakers	1	2	3	4	5	6
1	**0**	21	3	0	0	0
2	0	**122**	22	2	0	0
3	0	3	**44**	7	0	0
4	0	2	5	**10**	2	1
5	0	0	0	1	**3**	0
6	0	0	0	0	0	**1**
7	0	0	0	0	0	1

(c) SC-EEND [16] (Accuracy=76.4 %)

Pred. #Speakers \ Ref. #Speakers	1	2	3	4	5	6
1	**0**	1	0	0	0	0
2	0	**134**	20	4	0	0
3	0	13	**51**	10	4	2
4	0	0	3	**6**	1	1
5	0	0	0	0	**0**	0
6	0	0	0	0	0	**0**

(d) EEND-EDA (Accuracy=84.4 %)

Pred. #Speakers \ Ref. #Speakers	1	2	3	4	5	6
1	**0**	1	0	0	0	0
2	0	**142**	7	1	0	0
3	0	5	**54**	4	0	0
4	0	0	13	**14**	4	1
5	0	0	0	1	**1**	2
6	0	0	0	0	0	**0**

2) CALLHOME: Since the CALLHOME dataset does not include an official dev/eval split, we used the split provided in the Kaldi recipe and performed cross-validation. For comparison with the prior work on EEND, we also report the results obtained for Part 2 of the dataset using the model adapted to Part 1. For

TABLE X
DERS AND JERS (%) FOR AMI HEADSET MIX. NO COLLAR TOLERANCE WAS ALLOWED

Method	SAD	Dev		Eval	
		DER	JER	DER	JER
SA-EEND	-	31.66	39.20	27.70	37.50
EEND-EDA	-	21.93	25.86	21.56	29.99
X-vector (ResNet101) + AHC	Oracle	19.61	23.90	21.43	25.50
X-vector (ResNet101) + AHC + VBx [35]	Oracle	16.33	**20.57**	18.99	**24.57**
SA-EEND	Oracle	23.95	35.64	20.88	34.38
EEND-EDA	Oracle	**15.69**	22.19	**15.80**	26.68

TABLE XI
DERS AND JERS FOR DIHARD II EVAL. NO COLLAR TOLERANCE WAS ALLOWED

Method	SAD	DER	JER
SA-EEND	-	32.14	54.32
EEND-EDA	-	29.57	51.50
EEND-EDA (Iterative inference)	-	29.41	49.61
EEND-EDA (Iterative inference+)	-	**28.52**	**49.77**
X-vector (TDNN) + AHC + VBx [52]	BUT [52]	**27.11**	**49.07**
SA-EEND	BUT [52]	32.01	54.66
EEND-EDA	BUT [52]	30.48	51.78
EEND-EDA (Iterative inference)	BUT [52]	29.80	49.99
EEND-EDA (Iterative inference+)	BUT [52]	29.09	50.45
DIHARD II baseline [38]	Oracle	28.81	50.12
X-vector (TDNN) + AHC + VBx [52]	Oracle	**18.21**	N/A
X-vector (ResNet101) + AHC [35]	Oracle	23.59	43.93
X-vector (ResNet101) + AHC + VBx [35]	Oracle	18.55	**43.91**
SA-EEND	Oracle	23.25	50.30
EEND-EDA	Oracle	20.54	46.92
EEND-EDA (Iterative inference)	Oracle	21.00	45.30
EEND-EDA (Iterative inference+)	Oracle	20.24	45.62

TABLE XII
DERs AND JERs FOR DIHARD III EVAL. NO COLLAR TOLERANCE WAS ALLOWED

Method	SAD	Core		Full	
		DER	JER	DER	JER
SA-EEND	-	27.49	49.64	22.64	43.14
EEND-EDA	-	25.94	47.76	21.55	41.15
EEND-EDA (Iterative inference)	-	25.76	45.35	21.40	39.09
EEND-EDA (Iterative inference+)	-	**24.77**	**45.18**	**20.69**	**39.07**
X-vector (TDNN) + AHC + VBx [18]	Hitachi-JHU [18]	**22.99**	42.44	21.48	38.73
X-vector (TDNN) + AHC + VBx + OVL [18]	Hitachi-JHU [18]	24.58	**42.02**	21.47	**37.83**
SA-EEND	Hitachi-JHU [18]	25.79	49.20	21.29	42.68
EEND-EDA	Hitachi-JHU [18]	23.96	46.82	20.03	40.31
EEND-EDA (Iterative inference)	Hitachi-JHU [18]	24.41	44.70	20.30	38.47
EEND-EDA (Iterative inference+)	Hitachi-JHU [18]	23.43	44.93	**19.53**	38.78
DIHARD III baseline [39]	Oracle	20.65	47.74	19.25	42.45
X-vector (TDNN) + AHC + VBx [18]	Oracle	**16.89**	38.49	15.83	34.27
X-vector (TDNN) + AHC + VBx + OVL [18]	Oracle	18.20	**38.42**	15.65	**33.71**
X-vector (ResNet152) + AHC + VBx [63]	Oracle	16.56	38.72	15.79	34.46
SA-EEND	Oracle	20.21	46.17	16.19	39.44
EEND-EDA	Oracle	18.38	43.69	14.91	36.93
EEND-EDA (Iterative inference)	Oracle	18.87	41.58	15.21	35.08
EEND-EDA (Iterative inference+)	Oracle	17.86	41.69	**14.42**	35.30

TABLE XIII
BREAKDOWN RESULTS OF DIHARD III EVAL FOR EACH NUMBER OF SPEAKERS WITH ORACLE SPEECH SEGMENTS

(a) DER (%)

Method	#Speakers								
	1	2	3	4	5	6	7	8	9
X-vector (TDNN) + AHC + VBx	**1.30**	11.43	16.76	**23.09**	**44.99**	**26.43**	**25.61**	**35.57**	**2.03**
EEND-EDA	2.80	7.52	15.79	25.63	47.66	31.73	35.47	38.19	18.73
EEND-EDA (Iterative inference+)	1.47	**6.98**	**15.55**	26.32	47.48	31.44	34.79	38.26	14.99

(b) JER (%)

Method	#Speakers								
	1	2	3	4	5	6	7	8	9
X-vector (TDNN) + AHC + VBx	**2.40**	16.99	44.68	**44.70**	**66.17**	**53.32**	**56.05**	**56.71**	**8.01**
EEND-EDA	3.37	11.77	**38.70**	48.37	67.40	64.85	67.77	69.00	57.60
EEND-EDA + iterative inference+	3.31	**11.34**	39.60	48.76	68.46	62.41	62.65	65.36	41.23

SAD post-processing described in Section III-C1, we used the TDNN-based SAD provided in the Kaldi ASpIRE recipe[2] and oracle speech segments.

We show the number-of-speakers-wise results of cross-validation in Table VIII. We also show the results for only evaluated single speaker regions in brackets. For this purpose, we chose up the most probable speakers from each time frame of the EEND-EDA results for fair comparison with x-vector-based methods. EEND-EDA outperformed the state-of-the-art x-vector-based methods in total DERs. One reason is that EEND-EDA can handle speaker overlap, but it showed a competitive DER (5.29 %) even when speaker overlaps were excluded from the evaluation. Considering the number of speakers in a mixture, EEND-EDA did especially better than the x-vector-based methods with VBx clustering when the number of speakers was small (#Speakers=2,3,4), while it was worse or on par when the number of speakers was large (#Speakers=5,6,7). One reason is that the pretraining was based on mixtures with at most five speakers, and another reason is that mixtures of a larger number of speakers are rare in the CALLHOME dataset. Compared to SA-EEND, EEND-EDA achieved better DERs on all the cases. Table VIII(b) shows the results on CALLHOME Part 2. It clearly shows that EEND-EDA outperformed the other EEND-based methods [16], [17] by over two percent of absolute DER.

Table IX shows confusion matrices for the speaker counting of x-vector (TDNN) + AHC, x-vector (ResNet101) + AHC + VBx [35], SC-EEND [16], and EEND-EDA on CALLHOME Part 2. Our method achieved a higher speaker counting accuracy than the other methods by a large margin.

3) AMI Headset Mix: We next evaluated our method on the AMI headset mix, which has a different domain from the pretraining data (telephone conversation vs. meeting). We trained the model on the training set for 500 epochs and evaluated it on the dev and eval sets. The oracle speech segments were also used for SAD post-processing.

The results are shown in Table X. EEND-EDA outperformed the x-vector-based methods on both the dev and eval sets with the oracle SAD. Note that the x-vector-based methods tuned the PLDA parameters on the dev set, so the superiority of EEND-EDA was smaller on the dev set than the eval set. EEND-EDA also outperformed SA-EEND with and without the oracle SAD. We also note that the average duration of the recordings in the AMI headset mix test set is over 30 min. The performance of EEND-EDA showed that EEND-EDA generalized well to such long recordings while using 200 s segments during adaptation.

4) DIHARD II & DIHARD III: Finally, we evaluated our method on the DIHARD II and III datasets, which contain recordings from multiple domains. In this evaluation, we used iterative inference with and without DOVER-Lap, each of which are described in Section III-C2 and Section III-C3, respectively, to deal with large numbers of speakers. For SAD post-processing, we used oracle segments and the system used in the Hitachi-JHU submission to the DIHARD III challenge [18].

The results are shown in Tables XI and XII. We can see that iterative inference with DOVER-Lap (iterative inference+) consistently improved DERs. Compared with the x-vector-based

[2][Online]. Available: https://github.com/kaldi-asr/kaldi/tree/master/egs/aspire/s5

methods, EEND-EDA performed best on DIHARD III full, while the x-vector-based methods were better on DIHARD II and DIHARD III core.

We show the number-of-speakers-wise DERs and JERs on DIHARD III in Table XIII. Our method performed better when the number of speakers was small and worse when the number of speakers was large. This is why EEND-EDA performed well on DIHARD III full and worse on DIHARD II and DIHARD III eval. We also observed that the proposed iterative inference+ improved the performance, especially in terms of JERs on a large number of speaker cases, but it was still worse than the x-vector method. Handling a large number of speakers with EEND is left for future work.

VI. Conclusion

In this paper, we proposed an end-to-end speaker diarization method for unknown numbers of speakers using an encoder-decoder-based attractor calculation module called EEND-EDA. In EEND-EDA, frame-wise embeddings are firstly calculated from an input acoustic feature sequence, then speaker-wise attractors are calculated from the embeddings using EDA, and finally diarization results are obtained by the dot product of the embeddings and attractors. We also proposed to improve the performance of the diarization by shuffling the order of the embeddings before input to EDA and limiting the scope of backpropagation of the attractor existence loss. To conduct fair comparisons between EEND-based methods and cascaded methods under the same SAD condition, we introduced SAD post-processing for EEND-based methods. We also proposed iterative inference to cope with the problem of EEND-EDA's number of outputs being empirically limited by its training dataset. The evaluations on both simulated and real datasets showed that the proposed EEND-EDA performed well in both fixed-number-of-speakers and flexible-number-of-speakers evaluations.

One possible future direction of this research is to train EEND-EDA with simulated data of a larger number of speakers. Preparing a large amount of data in advance for training increments the storage usage. Therefore, we will need a method to prepare simulated mixtures on the fly during training as recently studied in [64]. In addition, to create a simulated mixture, we first create N recordings each of which contains one speaker, and then mix them to be an N-speaker mixture. To control the overlap ratio, we increased the value of β as the number of speakers in the mixture increased, but this leads to an increase in the duration of silence in the mixture. An investigation of a better simulation protocol is also left for future work.

Even if EEND-EDA is trained with datasets of a large number of speakers, it would still limit the maximum number of speakers by the datasets as shown in Table VII. One reason is that EEND-EDA decides the number of speakers by using a neural network trained in a fully supervised manner. One of our later works has shown that unsupervised clustering can be introduced into EEND-EDA to remove the limitation on the output number of speakers caused by the training dataset [65].

Another direction is the network architecture. Currently, EDA employs a vanilla LSTM encoder-decoder, but an attention-based LSTM or Transformer encoder-decoder may be possible alternatives. Transformer encoders to extract frame-wise embeddings from input features can be also replaced with other architectures such as Conformers [66] or time-dilated convolutional neural networks [64].

References

[1] T. J. Park, N. Kanda, D. Dimitriadis, K. J. Han, S. Watanabe, and S. Narayanan, "A review of speaker diarization: Recent advances with deep learning," *Comput. Speech Lang.*, vol. 72, 2022, Art. no. 101317.

[2] J. Carletta, "Unleashing the killer corpus: Experiences in creating the multi-everything AMI meeting corpus," *Lang. Resour. Eval.*, vol. 41, no. 2, pp. 181–190, 2007.

[3] S. Watanabe *et al.*, "CHiME-6 challenge: Tackling multispeaker speech recognition for unsegmented recordings," in *Proc. 6th Int. Workshop Speech Process. Everyday Environ.*, 2020.

[4] Z. Chen *et al.*, "Continuous speech separation: Dataset and analysis," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 7284–7288.

[5] C. Boeddeker, J. Heitkaemper, J. Schmalenstoeer, L. Drude, J. Heymann, and R. Haeb-Umbach, "Front-end processing for the CHiME-5 dinner party scenario," in *Proc. 5th Int. Workshop Speech Process. Everyday Environ.*, 2018.

[6] M. Delcroix, K. Zmolikova, T. Ochiai, K. Kinoshita, and T. Nakatani, "Speaker activity driven neural speech extraction," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 6099–6103.

[7] Q. Wang, C. Downey, L. Wan, P. Andrew Mansfield, and I. Lopez Moreno, "Speaker diarization with LSTM," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2018, pp. 5239–5243.

[8] T. J. Park, K. J. Han, M. Kumar, and S. Narayanan, "Auto-tuning spectral clustering for speaker diarization using normalized maximum eigengap," *IEEE Signal Process. Lett.*, vol. 27, pp. 381–385, 2020.

[9] G. Sell and D. Garcia-Romero, "Speaker diarization with PLDA I-vector scoring and unsupervised calibration," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2014, pp. 413–417.

[10] K. Kinoshita, L. Drude, M. Delcroix, and T. Nakatani, "Listening to each speaker one by one with recurrent selective hearing networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2018, pp. 5064–5068.

[11] T. von Neumann, K. Kinoshita, M. Delcroix, S. Araki, T. Nakatani, and R. Haeb-Umback, "All-neural online source separation, counting, and diarization for meeting analysis," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2019, pp. 91–95.

[12] Y. Fujita, N. Kanda, S. Horiguchi, K. Nagamatsu, and S. Watanabe, "End-to-end neural speaker diarization with permutation-free objectives," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 4300–4304.

[13] Y. Fujita, N. Kanda, S. Horiguchi, Y. Xue, K. Nagamatsu, and S. Watanabe, "End-to-end neural speaker diarization with self-attention," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 296–303.

[14] Y. Fujita, S. Watanabe, S. Horiguchi, Y. Xue, and K. Nagamatsu, "End-to-end neural diarization: Reformulating speaker diarization as simple multi-label classification," 2020, *arXiv:2003.02966*.

[15] S. Horiguchi, Y. Fujita, S. Wananabe, Y. Xue, and K. Nagamatsu, "End-to-end speaker diarization for an unknown number of speakers with encoder-decoder based attractors," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 269–273.

[16] Y. Fujita, S. Watanabe, S. Horiguchi, Y. Xue, J. Shi, and K. Nagamatsu, "Neural speaker diarization with speaker-wise chain rule," 2020, *arXiv:2006.01796*.

[17] Y. Takashima, Y. Fujita, S. Watanabe, S. Horiguchi, P. Garcia, and K. Nagamatsu, "End-to-end speaker diarization conditioned on speech activity and overlap detection," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2021, pp. 849–856.

[18] S. Horiguchi *et al.*, "The Hitachi-JHU DIHARD III system: Competitive end-to-end neural diarization and X-vector clustering systems combined by DOVER-Lap," in *Proc. 3rd DIHARD Speech Diarization Challenge Workshop*, 2021.

[19] E. Han, C. Lee, and A. Stolcke, "BW-EDA-EEND: Streaming end-to-end neural speaker diarization for a variable number of speakers," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 7193–7197.

[20] Y. Xue *et al.*, "Online streaming end-to-end neural diarization handling overlapping speech and flexible numbers of speakers," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3116–3120.

[21] M. À. India Massana, J. A. Rodríguez Fonollosa, and F. J. Hernando Pericás, "LSTM neural network-based speaker segmentation using acoustic and language modelling," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2017, pp. 2834–2838.

[22] T. J. Park *et al.*, "Speaker diarization with lexical information," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 391–395.

[23] 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. Audio, Speech, Lang. Process.*, vol. 22, no. 1, pp. 217–227, Jan. 2014.

[24] G. Sell *et al.*, "Diarization is hard: Some experiences and lessons learned for the JHU team in the inaugural DIHARD challenge," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2018, pp. 2808–2812.

[25] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2018, pp. 5329–5333.

[26] M. Diez, L. Burget, S. Wang, J. Rohdin, and J. Černockỳ, "Bayesian HMM based X-vector clustering for speaker diarization," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 346–350.

[27] X. Xiao *et al.*, "Microsoft speaker diarization system for the VoxCeleb speaker recognition challenge 2020," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 5824–5828.

[28] A. Zhang, Q. Wang, Z. Zhu, J. Paisley, and C. Wang, "Fully supervised speaker diarization," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2019, pp. 6301–6305.

[29] S. H. Shum, N. Dehak, R. Dehak, and J. R. Glass, "Unsupervised methods for speaker diarization: An integrated and iterative approach," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 21, no. 10, pp. 2015–2028, Oct. 2013.

[30] D. Dimitriadis and P. Fousek, "Developing on-line speaker diarization system," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2017, pp. 2739–2743.

[31] D. Garcia-Romero, D. Snyder, G. Sell, D. Povey, and A. McCree, "Speaker diarization using deep neural network embeddings," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 4930–4934.

[32] M. Maciejewski, D. Snyder, V. Manohar, N. Dehak, and S. Khudanpur, "Characterizing performance of speaker diarization systems on far-field speech using standard methods," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2018, pp. 5244–5248.

[33] D. Raj, Z. Huang, and S. Khudanpur, "Multi-class spectral clustering with overlaps for speaker diarization," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2021, pp. 582–589.

[34] M. Diez, L. Burget, F. Landini, and J. Černockỳ, "Analysis of speaker diarization based on Bayesian HMM with eigenvoice priors," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 355–368, 2020.

[35] F. Landini, J. Profant, M. Diez, and L. Burget, "Bayesian HMM clustering of X-vector sequences (VBx) in speaker diarization: Theory, implementation and analysis on standard tasks," *Comput. Speech Lang.*, vol. 71, 2022, Art. no. 101254.

[36] Q. Li, F. L. Kreyssig, C. Zhang, and P. C. Woodland, "Discriminative neural clustering for speaker diarisation," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2021, pp. 574–581.

[37] Z. Huang *et al.*, "Speaker diarization with region proposal network," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 6514–6518.

[38] N. Ryant *et al.*, "The second DIHARD diarization challenge: Dataset, task, and baselines," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 978–982.

[39] N. Ryant *et al.*, "The third DIHARD diarization challenge," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3570–3574.

[40] I. Medennikov *et al.*, "Target-speaker voice activity detection: A novel approach for multi-speaker diarization in a dinner party scenario," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 274–278.

[41] D. Yu, M. Kolbæk, Z.-H. Tan, and J. Jensen, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 241–245.

[42] Y. Luo, Z. Chen, J. R. Hershey, J. Le Roux, and N. Mesgarani, "Deep clustering and conventional networks for music separation: Stronger together," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 61–65.

[43] Y. Luo and N. Mesgarani, "TasNet: Time-domain audio separation network for real-time, single-channel speech separation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 696–700.

[44] Y. Luo and N. Mesgarani, "Conv-TasNet: Surpassing ideal time-frequency magnitude masking for speech separation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 27, no. 8, pp. 1256–1266, Aug. 2019.

[45] Y. Luo, Z. Chen, and N. Mesgarani, "Speaker-independent speech separation with deep attractor network," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 26, no. 4, pp. 787–796, Apr. 2018.

[46] N. Zeghidour and D. Grangier, "Wavesplit: End-to-end speech separation by speaker clustering," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 2840–2849, 2021.

[47] N. Takahashi, S. Parthasaarathy, N. Goswami, and Y. Mitsufuji, "Recursive speech separation for unknown number of speakers," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 1348–1352.

[48] Z. Chen, Y. Luo, and N. Mesgarani, "Deep attractor network for single-microphone speaker separation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 246–250.

[49] J. Lee, Y. Lee, J. Kim, A. R. Kosiorek, S. Choi, and Y. W. Teh, "Set Transformer: A framework for attention-based permutation-invariant neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 3744–3753.

[50] B. B. Meier, I. Elezi, M. Amirian, O. Dürr, and T. Stadelmann, "Learning neural models for end-to-end clustering," in *Proc. IAPR Workshop Artif. Neural Netw. Pattern Recognit.*, 2018, pp. 126–138.

[51] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2016, pp. 31–35.

[52] F. Landini *et al.*, "BUT system for the Second DIHARD speech diarization challenge," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 6529–6533.

[53] S. Horiguchi, P. Garcia, Y. Fujita, S. Watanabe, and K. Nagamatsu, "End-to-end speaker diarization as post-processing," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 7188–7192.

[54] D. Raj *et al.*, "DOVER-Lap: A method for combining overlap-aware diarization outputs," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2021, pp. 881–888.

[55] A. Stolcke and T. Yoshioka, "DOVER: A method for combining diarization outputs," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 757–763.

[56] 2000 NIST speaker recognition evaluation. [Online]. Available: https://catalog.ldc.upenn.edu/LDC2001S97

[57] K. Maekawa, "Corpus of spontaneous Japanese: Its design and evaluation," in *Proc. ISCA IEEE Workshop Spontaneous Speech Process. Recognit.*, 2003, pp. 7–12.

[58] N. Kanda *et al.*, "Guided source separation meets a strong ASR backend: Hitachi/Paderborn University joint investigation for dinner party scenario," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 1248–1252.

[59] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015.

[60] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[61] N. Kanda, S. Horiguchi, Y. Fujita, Y. Xue, K. Nagamatsu, and S. Watanabe, "Simultaneous speech recognition and speaker diarization for monaural dialogue recordings with target-speaker acoustic models," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2019, pp. 31–38.

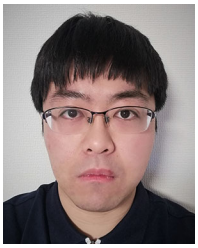
[62] L. Drude, T. von Neumann, and R. Haeb-Umbach, "Deep attractor networks for speaker re-identification and blind source separation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2018, pp. 11–15.

[63] F. Landini *et al.*, "BUT system description for the third DIHARD speech diarization challenge," in *Proc. 3rd DIHARD Speech Diarization Challenge Workshop*, 2021.

[64] S. Maiti, H. Erdogan, K. Wilson, S. Wisdom, S. Watanabe, and J. R. Hershey, "End-to-end diarization for variable number of speakers with local-global networks and discriminative speaker embeddings," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 7183–7187.

[65] S. Horiguchi, P. García, S. Watanabe, Y. Xue, Y. Takashima, and Y. Kawaguchi, "Towards neural diarization for unlimited numbers of speakers using global and local attractors," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 98–105.

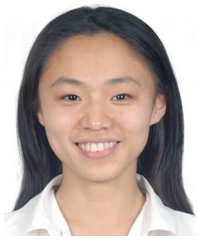
[66] Y. C. Liu, E. Han, C. Lee, and A. Stolcke, "End-to-end neural diarization: From transformer to conformer," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3081–3085.



Shota Horiguchi (Member, IEEE) received the B.S. degree in information and communication engineering and the M.S. degree in information science and technology from The University of Tokyo, Tokyo, Japan, in 2015 and 2017, respectively. He is currently a Senior Researcher with Hitachi, Ltd., Tokyo, Japan. His research interests include speech recognition, speech separation, speaker diarization, image processing, and multimodal processing. He has participated in the CHiME-5 and DIHARD III challenges as a Member of the Hitachi-JHU team.

Yusuke Fujita (Member, IEEE) received the B.S. and M.S. degrees in computer science from Waseda University, Tokyo, Japan. From 2005 to 2020, he was a Senior Researcher with Hitachi, Ltd. He is currently a Senior Research Engineer with LINE Corporation, where he develops speech recognition systems. His research interests include speech recognition, speech separation, and speaker diarization. He has been working on end-to-end speaker diarization and distant speech recognition through the CHiME-5 challenge session chair, the CHiME-6 Scientific Committee, and the participation of the CHiME and DIHARD challenges.

Shinji Watanabe (Senior Member, IEEE) received the B.S., M.S., and Ph.D. (Dr. Eng.) degrees from Waseda university, Tokyo, Japan, in 1999, 2001, and 2006, respectively. He is currently an Associate Professor with Carnegie Mellon University, Pittsburgh, PA, USA. He was a Research Scientist with NTT Communication Science Laboratories, Kyoto, Japan, from 2001 to 2011, a Visiting Scholar with the Georgia Institute of Technology, Atlanta, GA, USA, in 2009, and a Senior Principal Research Scientist with Mitsubishi Electric Research Laboratories, Cambridge, MA, USA, from 2012 to 2017. Prior to joining Carnegie Mellon University, he was an Associate Research Professor with Johns Hopkins University, Baltimore, MD, USA, from 2017 to 2020. He has authored or coauthored more than 300 papers in peer-reviewed journals and conferences. His research interests include automatic speech recognition, speech enhancement, spoken language understanding, and machine learning for speech and language processing. He was the recipient of several awards, including the Best Paper Award from the IEEE ASRU in 2019. He was an Associate Editor for the IEEE TRANSACTIONS ON AUDIO SPEECH AND LANGUAGE PROCESSING. He was/is a Member of several technical committees, including the APSIPA Speech, Language, and Audio Technical Committee, IEEE Signal Processing Society Speech and Language Technical Committee, and Machine Learning for Signal Processing Technical Committee.



Yawen Xue received the M.S. and Ph.D. degrees in information science from Japan Advanced Institute of Science and Technology, Ishikawa, Japan, in 2015 and 2018, respectively. From April 2018 to March 2019, she was a Japan Society for the Promotion of Science Research Fellow. She is currently a Researcher with Hitachi, Ltd., Tokyo, Japan. Her research interests include speech recognition, speaker diarization, emotional speech recognition, and emotional speech conversion.

Paola García (Member, IEEE) received the Ph.D. degree from the University of Zaragoza, Zaragoza, Spain, in 2014. She joined Johns Hopkins University after extensive research experience in academia and industry, including highly regarded laboratories at Agnitio and Nuance Communications. She led a team of more than 20 researchers from four of the best laboratories worldwide in far-field speech diarization and speaker recognition, under the auspices of the JHU summer workshop 2019 in Montreal, QC, Canada. She was also a Researcher with Tec de Monterrey, Campus Monterrey, Mexico for ten years. She was a Marie Curie Researcher for the Iris Project during 2015, exploring assistive technology for children with autism in Zaragoza, Spain. She was a Visiting Scholar with the Georgia Institute of Technology, Atlanta, GA, USA, in 2009 and Carnegie Mellon University, Pittsburgh, PA, USA, in 2011. She is currently working on children's speech, including child speech recognition and diarization in day-long recordings. She collaborates with DARCLE.org and CCWD that analyze child-centered speech. She is also a part of the JHU CHiME5, CHiME6, SRE18 and SRE19, SRE20, and SRE21 teams. Her research interests include diarization, speech recognition, speaker recognition, machine learning, and language processing.