Online Neural Diarization of Unlimited Numbers of Speakers Using Global and Local Attractors

Shota Horiguchi, *Member, IEEE*, Shinji Watanabe, *Fellow, IEEE*, Paola García, *Member, IEEE*, Yuki Takashima, and Yohei Kawaguchi, *Senior Member, IEEE*

***Abstract*—A method to perform offline and online speaker diarization for an unlimited number of speakers is described in this paper. End-to-end neural diarization (EEND) has achieved overlap-aware speaker diarization by formulating it as a multi-label classification problem. It has also been extended for a flexible number of speakers by introducing speaker-wise attractors. However, the output number of speakers of attractor-based EEND is empirically capped; it cannot deal with cases where the number of speakers appearing during inference is higher than that during training because its speaker counting is trained in a fully supervised manner. Our method, EEND-GLA, solves this problem by introducing unsupervised clustering into attractor-based EEND. In the method, the input audio is first divided into short blocks, then attractor-based diarization is performed for each block, and finally, the results of each block are clustered on the basis of the similarity between locally-calculated attractors. While the number of output speakers is limited within each block, the total number of speakers estimated for the entire input can be higher than the limitation. To use EEND-GLA in an online manner, our method also extends the speaker-tracing buffer, which was originally proposed to enable online inference of conventional EEND. We introduce a block-wise buffer update to make the speaker-tracing buffer compatible with EEND-GLA. Finally, to improve online diarization, our method improves the buffer update method and revisits the variable chunk-size training of EEND. The experimental results demonstrate that EEND-GLA can perform speaker diarization of an unseen number of speakers in both offline and online inferences.**

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

I. Introduction

IDENTIFYING who spoke when from the input audio is referred to as speaker diarization [1], [2]. It is a core technology of spoken language understanding of multi-talker conversations in various scenarios such as everyday conversations [3], [4], [5], doctor-patient conversations [6], meetings [7], lectures [8], and video contents [9].

While cascaded methods for diarization have been widely investigated in the literature [10], [11], [12], [13], progress on end-to-end methods has enabled highly accurate speaker diarization [14], [15], [16], [17], [18], [19], [20]. One reason for this advance is the easy handling of overlapping speech. In cascaded methods, the speaker embeddings extracted for each short segment are clustered to perform diarization. Therefore, overlap-aware speaker diarization cannot be done unless overlap detection and speaker assignment are performed as post-processing [21], [22], [23]. However, most end-to-end models can naturally handle overlapping speech because they estimate speech segments of multiple speakers simultaneously like multi-label classification. Another reason is the ease of optimization as an entire diarization system. In cascaded methods, each module (speech activity detector, speaker embedding extractor, clustering model, etc.) is trained independently, which makes it difficult to optimize the overall diarization system. In contrast, end-to-end methods use a single neural network to obtain diarization results directly from the input audio, making optimization easier than cascaded methods. This is also the same for online diarization. Online diarization with cascaded methods requires all the modules above to enable online use. On the other hand, an end-to-end method still requires a single model by simply replacing the network architecture with the one that enables online inference [24]. In other methods, an end-to-end model trained for offline use can be used for online purposes by using a buffer to store the previous input-result pairs [25], [26].

Although end-to-end methods have several advantages over cascaded methods, they still have room for improvement. The biggest challenge is in the estimation of the number of speakers. In cascaded methods, the number of speakers is estimated as the result of clustering; thus, the number of speakers can be flexibly determined and unlimited. In contrast, most end-to-end methods fix the number of output speakers due to their network architecture [14], [27]. Most methods that enable the inference of a flexible number of speakers conduct it by outputting null speech activities for absent speakers, so the maximum number of speakers is limited [18], [28]. Some methods use speaker-wise auto-regressive inference to avoid setting the maximum number of speakers by the network architecture; but in practice, the number of output speakers is still capped by the training dataset [15], [16], [29], [30]. A few studies, one of which is the basis for this paper, integrated the end-to-end approach with unsupervised clustering to solve this problem [31], [32], [33], [34]. The methods showed promising results on various

Manuscript received 1 June 2022; revised 13 December 2022; accepted 19 December 2022. Date of publication 30 December 2022; date of current version 11 January 2023. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Alberto Abad. *(Corresponding author: Shota Horiguchi.)*

Shota Horiguchi, Yuki Takashima, and Yohei Kawaguchi are with Hitachi, Ltd., Kokubunji-shi, Tokyo 185-8601, Japan (e-mail: shota.horiguchi.wk@hitachi.com; yuki.takashima.ot@hitachi.com; yohei.kawaguchi.xk@hitachi.com).

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.2022.3233237

benchmark datasets, but their online inferences have not been investigated in the literature.

In this article, we propose end-to-end neural diarization with global and local attractors (EEND-GLA), which integrates attractor-based EEND (EEND-EDA) [15], [16] with clustering to conduct speaker diarization without limiting the number of speakers. In addition to the attractors calculated from the entire recording (i.e., *global attractors*) in the same manner as in EEND-EDA, we also utilize attractors calculated from each short block (i.e., *local attractors*) to obtain block-wise diarization results. Because the set of speakers and their output order may be different among the blocks, we use clustering to find the appropriate speaker correspondence between the blocks on the basis of the similarities between the local attractors. Here, we assume that the number of speakers appearing in a short period is low, and so the number of speakers within each block can be limited and fixed with a maximum number. However, the total number of speakers is estimated as the result of clustering; it is no longer limited by the network architecture or training datasets. To enable online inference of EEND-GLA, we also propose a block-wise speaker-tracing buffer, which extends the original speaker-tracing buffer [25], [26] to update the buffer elements in a block-wise manner. With this modification, we can assume that the number of speakers within each block is limited in the buffer as well because each block stores time-consecutive elements.

This paper is organized on the basis of our previous paper [33], in which the fundamental algorithm of EEND-GLA was presented. Our contributions that differ from those of the previous paper are summarized as follows.

- We propose a block-wise speaker-tracing buffer, which enables the online inference of EEND-GLA.
- We improve the speaker-tracing buffer by introducing speaker-balanced sampling probabilities.
- We revisit variable chunk-size training to improve online diarization, especially at the very beginning of inference.
- We evaluate our method on offline and online diarization settings consistently over various prior studies.

The organization of this paper is as follows. Section II reviews offline and online diarization methods in the literature. Section III details conventional attractor-based EEND (Section III-A) and speaker-tracing buffer that enables its online inference (Section III-B). Section IV presents proposed EEND-GLA (Section IV-A) and some modifications to the speaker-tracing buffer to make it compatible with EEND-GLA and improve its performance (Sections IV-B to IV-D). Sections V and VI describe the experimental settings and results, respectively. Section VII concludes the paper.

II. Related Work

A. *Offline Diarization*

The conventional cascaded approach for speaker diarization consists of the following operations: 1) speech activity detection (SAD), 2) speaker embedding extraction from each detected speech segment, 3) clustering of the embeddings, and 4) optional overlap handling. The oracle SAD is sometimes used in the experiments, but the remaining parts are actively being studied in the literature: better speaker embedding extraction methods [35], [36], [37], [38], clustering methods [11], [13], [39], and overlap assignment methods [22], [40], [41]. The cascaded approach is based on unsupervised clustering; thus, the number of output speakers can take an arbitrary value and can be set flexibly during inference.

Different end-to-end approaches for speaker diarization have been studied, but they have drawbacks when performing speaker diarization without any restrictions. Some methods such as personal VAD [42] and VoiceFilter-Lite [43] are not suited for speaker-independent diarization because they require a target speaker's embedding vector (e.g., d-vector) for inference. Target-speaker voice activity detection (TS-VAD) accepts multiple speakers' embeddings, but they have to be obtained in advance from another diarization method such as a cascaded-based approach [17] or end-to-end neural diarization (EEND) [44]. The initial models of TS-VAD and EEND [14], [27] fix the output number of speakers with their network architectures, so they are not suited for diarization of an unknown number of speakers. The recurrent selective attention network (RSAN) [45], or some extensions of TS-VAD [28] and EEND [15], [18], [29] can deal with a flexible number of speakers. However, the TS-VAD-based methods explicitly determine the maximum number of outputs with their network architecture, except for a few recent attempts such as multi-target filter and detector (MTFAD) [46] and Transformer-based TS-VAD [47]. The EEND-based methods do not have such explicit limitations, but the maximum number of speakers is empirically known to be bounded by their training datasets. Whether or not RSAN can deal with an unlimited number of speakers is unclear because only the speaker counting accuracies on zero, one, and two-speaker conditions were reported in the paper and each was observed during training. Making the number of output speakers not only flexible but also unlimited is an important challenge for end-to-end diarization.

The combination of an end-to-end approach and clustering is a promising direction to solve the problem of the limitation of the number of speakers. For example, EEND as post-processing [23] and overlap-aware resegmentation [12] use EEND to refine the results obtained with cascaded diarization systems. Multi-scale diarization decoder [48] also employs a similar post-processing approach. In these methods, the initial results are based on the clustering of speaker embeddings; hence, the number of output speakers can be arbitrary. However, this nullifies the main advantage of the end-to-end approach, that is, simplicity. The other approach is EEND-vector clustering [31], [32], [34]. It uses EEND for shortly divided blocks and then finds the speaker corresponding between them using speaker embeddings. It is relevant to our method in this paper, but some differences exist between them. One is that EEND-vector clustering requires unique speaker identity labels *over* the recordings in the training set. This means that we must know whether or not a pair of speakers that appeared in different recordings has the same identity. Such information can be easily obtained from simulation data but is not always suitable for real recordings. EEND-GLA only requires the speaker labels *within* each recording; thus, we can use such real recordings for training. This property is also powerful when conducting, for example, unsupervised or semi-supervised domain adaptation [49]. Another difference is that EEND-vector clustering requires a somewhat long length of blocks (e.g., 30 s) to obtain reliable speaker embeddings to achieve the best performance. However, because the number of output speakers within a block is limited by the network architecture, the length would result in a limited output number of speakers in the final results. Another problem is that the length causes a severe latency if we want to use it for online inference. However, EEND-GLA splits a sequence into short blocks after generating frame-wise embeddings from acoustic features using stacked Transformer encoders. As a result, the frame-wise embeddings can capture the global context, so we can use a lower block length (5 s in this paper) than EEND-vector clustering.

B. Online Diarization

There are also cascaded and end-to-end approaches to online diarization. In cascaded approaches, of course, all modules have to work in an online manner. The most crucial part is a clustering of speaker embeddings, and many methods have been proposed for that in the literature, e.g., UIS-RNN [50], UIS-RNN-SML [51], constraint incremental clustering in overlap-aware online speaker diarization [52], and turn-to-diarize [53]. Basically, online clustering is not as good as offline clustering. In particular, VBx [13], the current state-of-the-art offline clustering method for diarization, relies on two-stage clustering to refine the results and thus is difficult to be used for online inference. In fact, even if the rest of the modules are similar between offline and online methods, replacing VBx with online clustering reportedly causes a significant drop in performance [52].

On the other hand, end-to-end approaches have also been explored in online diarization. Online diarization with end-to-end models has two directions. One is to train a model with frame-wise or block-wise inputs separately from the offline model. For example, Online RSAN [54], [55] is trained with block-wise inputs to extend the original RSAN [45] for an online manner. This method uses speaker embeddings to convey information between blocks to make the order of output speakers consistent. BW-EDA-EEND [24] replaced the Transformer encoders in EEND with Transformer-XL [56] to extend EEND-EDA [15], [16] to deal with block-wise inputs. In this method, the hidden state embeddings obtained during the processing of previous blocks are used to process the current block, thereby solving the speaker permutation ambiguity between blocks. This direction is beneficial to optimize online diarization itself, but the training cost is doubled if we need to prepare diarization systems for both offline and online inference independently. The other possibility is to divert an offline diarization model for online inference. For this purpose, speaker-tracing buffer [25], [26] has been proposed to implement online inference of EEND with no modification of the network architecture. It stores acoustic features and their corresponding diarization results of the selected past frames to solve the speaker permutation ambiguity (see Section III-B for a detailed explanation). Because it was reported that EEND-EDA with speaker-tracing buffer outperformed BW-EDA-EEND [26], we focused on this direction in this study.

III. Conventional Method

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

End-to-end neural diarization (EEND) is a framework to estimate multiple speakers' speech activities from the input audio. In particular, attractor-based EEND (EEND-EDA) [15], [16] also estimates the number of speakers simultaneously. Given T-length F-dimensional acoustic features $X \in \mathbb{R}^{F\times T}$, they are first converted to the same length of D-dimensional frame-wise embeddings $E \in \mathbb{R}^{D\times T}$ using stacked Transformer encoders:

$$E = \mathsf{TransformerEncoder}(X) \in \mathbb{R}^{D\times T}. \tag{1}$$

Then, the encoder-decoder-based attractor calculation module (EDA) calculates attractors $\boldsymbol{a}_s \in (0,1)^D$ for each speaker $s \in \mathbb{N}$ from E in (1) in a sequence-to-sequence manner as

$$\boldsymbol{a}_1, \boldsymbol{a}_2, \cdots = \mathsf{EDA}(E). \tag{2}$$

The decoder calculation in (2) continues as long as the attractor existence probability $\hat{z}_s$ calculated from $\boldsymbol{a}_s$ is not less than 0.5, and the largest s that fulfills $\hat{z}_s \geq 0.5$ is the estimated number of speakers $\hat{S}$, as follows:

$$\hat{z}_s = \sigma\left(\mathsf{Linear}\left(\boldsymbol{a}_s\right)\right) \in (0,1), \tag{3}$$

$$\hat{S} = \min\left\{s \mid s \in \mathbb{Z}_{\geq 0} \wedge \hat{z}_{s+1} < 0.5\right\}, \tag{4}$$

where $\sigma(\cdot)$ is the element-wise sigmoid function. Finally, the estimations of speech activities $\hat{Y}$ are calculated as dot products between the frame-wise embeddings and attractors with their existence probabilities greater than or equal to 0.5:

$$\hat{Y} = \sigma\left(\begin{bmatrix}\boldsymbol{a}_1 & \cdots & \boldsymbol{a}_{\hat{S}}\end{bmatrix}^{\mathsf{T}} E\right) \in (0,1)^{\hat{S}\times T}. \tag{5}$$

During training, the following loss is used for network optimization:

$$\mathcal{L}_{\text{global}} = \mathcal{L}_{\text{diar}} + \alpha \mathcal{L}_{\text{exist}}, \tag{6}$$

where α is the weighting parameter, which was set to 1 in this study. The first term $\mathcal{L}_{\text{diar}}$ is the permutation-free diarization loss, which optimizes the output speech activities, defined as

$$\mathcal{L}_{\text{diar}} = \frac{1}{TS} \mathop{\arg\min}_{\phi\in\Phi(S)} H\left(Y, P_\phi \hat{Y}\right), \tag{7}$$

where $\Phi(S)$ is a set of all the possible permutations of $(1,\ldots,S)$, $P_\phi \in \{0,1\}^{S\times S}$ is the permutation matrix that corresponds to the permutation ϕ, $H(\cdot,\cdot)$ is the sum of element-wise binary cross entropy, and S is the ground-truth number of speakers. Note that the estimation of speech activities $\hat{Y}$ is calculated using the ground-truth number of speakers during training, i.e., $\hat{Y} \in (0,1)^{S\times T}$. The second term $\mathcal{L}_{\text{exist}}$ is the attractor existence loss, which optimizes the number of output attractors, defined as

$$\mathcal{L}_{\text{exist}} = \frac{1}{S+1}\sum_{s=1}^{S+1} H\left(z_s, \hat{z}_s\right), \quad z_s = \begin{cases}1 & (s \in \{1,\ldots,S\})\\ 0 & (s = S+1)\end{cases}. \tag{8}$$

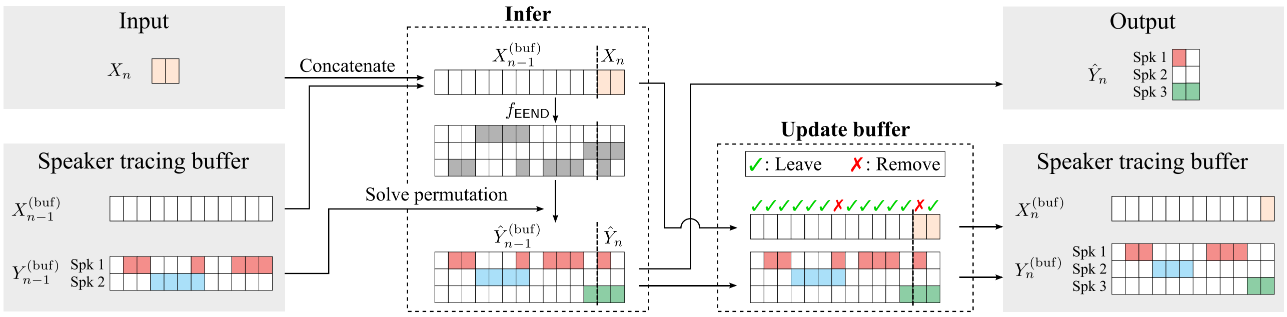


Fig. 1. Online diarization using speaker-tracing buffer proposed in [25], [26].

Following the previous study [16], the attractor existence loss is used to update only the parameters of the linear layer in (3).

B. Online Diarization With Speaker-Tracing Buffer

A speaker-tracing buffer has been proposed to enable online inference of EEND without additional training [25], [26]. The speaker-tracing buffer stores the past acoustic features and the corresponding estimation to solve the speaker permutation ambiguity. The schematic diagram of online diarization using a speaker-tracing buffer is shown in Fig. 1.

In the situation of online diarization, chunked acoustic features sequentially arrive, and the length of each chunk is ν. Suppose $X_{n-1}^{(\mathrm{buf})} \in \mathbb{R}^{F \times T_{n-1}^{(\mathrm{buf})}}$ and $Y_{n-1}^{(\mathrm{buf})} \in \mathbb{R}^{\hat{S}_{n-1} \times T_{n-1}^{(\mathrm{buf})}}$ are features and the corresponding estimations stored in the buffer just before the n-th input, respectively, where $T_{n-1}^{(\mathrm{buf})}$ is their length and $\hat{S}_{n-1}$ is the previously estimated number of speakers. Given n-th input $X_n \in \mathbb{R}^{F \times \nu}$, it is concatenated with the features in the buffer and processed by EEND f_{EEND} as

$$\begin{aligned}&\begin{bmatrix}\hat{Y}_{n-1}^{(\mathrm{buf})} & \hat{Y}_n\end{bmatrix}\\&= f_{\mathsf{EEND}}\left(\begin{bmatrix}X_{n-1}^{(\mathrm{buf})} & X_n\end{bmatrix}\right) \in (0,1)^{\hat{S}'_n \times (T_{n-1}^{(\mathrm{buf})}+\nu)},\end{aligned} \tag{9}$$

where $\hat{S}'_n$ is the newly estimated number of speakers[1], and $\hat{Y}_{n-1}^{(\mathrm{buf})}$ and $\hat{Y}_n$ are the estimated results that correspond to $X_{n-1}^{(\mathrm{buf})}$ and X_n, respectively. Here, the previously estimated number of speakers $\hat{S}_{n-1}$ and the newly estimated one $\hat{S}'_n$ may differ, e.g., $\hat{S}_{n-1} = 2$ and $\hat{S}'_n = 3$ in Fig. 1. To align them to the same number, we first update each of $\hat{Y}_{n-1}^{(\mathrm{buf})} \in (0,1)^{\hat{S}'_n \times T_{n-1}^{(\mathrm{buf})}}$ and $\hat{Y}_n \in (0,1)^{\hat{S}'_n \times \nu}$ to have $\hat{S}_n = \max(\hat{S}_{n-1}, \hat{S}'_n)$ rows by zero padding. The order of speakers is then permuted to be aligned to that of $\hat{Y}_{n-1}^{(\mathrm{buf})}$ as

$$\begin{bmatrix}\hat{Y}_{n-1}^{(\mathrm{buf})} & \hat{Y}_n\end{bmatrix} \leftarrow P_\psi \begin{bmatrix}\hat{Y}_{n-1}^{(\mathrm{buf})} & \hat{Y}_n\end{bmatrix}, \tag{10}$$

$$\psi = \underset{\phi \in \Phi(\hat{S}_n)}{\arg\max} \left\langle Y_{n-1}^{(\mathrm{buf})}, P_\phi \hat{Y}_{n-1}^{(\mathrm{buf})} \right\rangle_{\mathrm{F}}, \tag{11}$$

where $\langle A, B\rangle_{\mathrm{F}}$ denotes the Frobenius inner product between real-valued two matrices $A = [a_{ij}]$ and $B = [b_{ij}]$ defined as[2]

$$\langle A, B\rangle_{\mathrm{F}} := \sum_{i,j} a_{ij} b_{ij}. \tag{12}$$

Note that (11) is executable in polynomial time by using the well-known Hungarian algorithm. Finally, the permuted $\hat{Y}_n$ is output as the estimated result for X_n.

For the next (i.e., $(n+1)$-th) input, the buffer is updated with the current input features and the corresponding results. If the buffer length M is large enough to store all the features and results, i.e., $T_{n-1}^{(\mathrm{buf})} + \nu \leq M$, we update the buffer using

$$X_n^{(\mathrm{buf})} \leftarrow \begin{bmatrix}X_{n-1}^{(\mathrm{buf})} & X_n\end{bmatrix}, \tag{13}$$

$$Y_n^{(\mathrm{buf})} \leftarrow \begin{bmatrix}\hat{Y}_{n-1}^{(\mathrm{buf})} & \hat{Y}_n\end{bmatrix}. \tag{14}$$

If $T_{n-1}^{(\mathrm{buf})} + \nu > M$, only M frames among them are selected to be stored. The original speaker-tracing buffer mainly utilized the following two update strategies.

1) *First-in-first-out (FIFO):* acoustic features and results of the latest M frames are always stored in the buffer. Speakers who do not appear in the last M frames are not tracked with this strategy; thus, this strategy alone is not preferable.
2) *Sampling:* the features and results of informative M frames to solve speaker permutation ambiguity are selected among $T_{n-1}^{(\mathrm{buf})} + \nu$ and stored. In the previous studies [25], [26], sampling probabilities based on Kullback-Leibler (KL) divergence were used. The KL divergence at the t-th frame ω_t is calculated from the speaker-normalized posteriors $\bar{y}_{s,t}$ and the discrete uniform distribution with the posterior probability of $\frac{1}{S_n}$ as

$$\omega_t = \sum_{s=1}^{S_n} \bar{y}_{s,t} \log\left(\bar{y}_{s,t} S_n\right), \tag{15}$$

$$\bar{y}_{s,t} = \frac{y_{s,t}}{\sum_{s'=1}^{S_n} y_{s',t}}. \tag{16}$$

[1] Ideally, $\hat{S}'_n$ is not less than $\hat{S}_{n-1}$.

[2] In the original STB paper, mean normalization is applied for each of A and B before the calculation of the Frobenius inner product, but it does not affect the result so we omit it here.

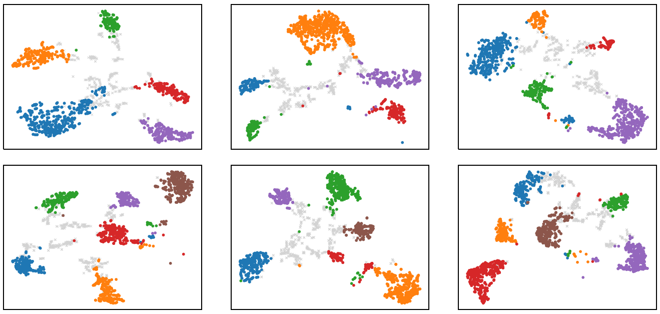


Fig. 2. t-SNE visualization of frame-wise embeddings extracted from simulated 5-speaker mixtures (top) and 6-speaker mixtures (bottom). The EEND-EDA used for extraction was trained using {1,2,3,4}-speaker mixtures. Single-speaker frames are denoted by the dots with colors corresponding to the speaker identities and overlapped frames are denoted by the crosses in light gray. Frames of silence were excluded from the visualization.

The sampling probabilities $\tilde{\omega}_t$ are defined as the normalized KL divergence so that the sum is one:

$$\tilde{\omega}_t = \frac{\omega_t}{\sum_{t'} \omega_{t'}}. \tag{17}$$

With the aforementioned speaker-tracing buffer, a trained EEND model can be used for online inference as it is. However, EEND is generally trained with a fixed length of chunks, e.g., 500 frames, so the diarization performance decreases at the very beginning of the inference where the number of frames is low. To cope with this problem, variable chunk-size training (VCT) was proposed [25]. For VCT, the length of each chunk is varied by masking the input minibatch. It has been evaluated in two-speaker conditions [25] but has not been evaluated in the flexible-number-of-speaker conditions [26]. Even the two-speaker experiments have a limited analysis of how VCT improved the diarization error rates (DERs). These aspects will be analyzed in Section VI-A.

IV. Proposed Method

A. EEND With Global and Local Attractors

While EEND-EDA can treat a flexible number of speakers, the maximum number of speakers to be output was empirically revealed to be limited by the dataset used during training. For example, if EEND-EDA is trained using mixtures, each of which contains at most four speakers, it cannot produce a valid result for the fifth or later speaker even if a mixture contains more than four speakers. To reveal which part causes this limitation, we visualized the frame-wise embeddings that were output from the last Transformer encoder using t-SNE [57] in Fig. 2. Even though EEND-EDA was trained on mixtures, each of which consists of at most four speakers, five or six speakers' speeches were clearly separated in the embedding space. The visualization revealed that EDA fails to estimate attractors for the unseen-number-of-speaker cases.

The proposed method assumes that the number of speakers that appear in a short period is bounded in practice. We first conduct attractor-based diarization for each short block and then find inter-block speaker correspondence on the basis of the similarity of the attractors. We call the attractors calculated within each block *local attractors*. Even if the number of speakers within each block is limited owing to EDA, the total number of speakers within a recording can be higher than the upper bound. Our method also utilizes global-attractor-based diarization just as EEND-EDA does.

1) Training: Fig. 3 illustrates the proposed diarization based on global and local attractors, which we call EEND-GLA. The global-attractor-based diarization is identical to EEND-EDA described in Section III-A; in this section, we introduce local-attractor-based diarization. Given frame-wise embeddings E, we first split them into short blocks, each of which has a length of λ. Here, we assume that the sequence of frame-wise embeddings is split into L blocks, i.e., $E := \begin{bmatrix} E^{(1)} & \cdots & E^{(L)} \end{bmatrix}$, where $E^{(l)} \in \mathbb{R}^{D\times\lambda}$ for $l \in \{1,\ldots,L\}$ and $L := \frac{T}{\lambda}$.[3] From the l-th block, local attractors $\boldsymbol{a}_1^{(l)},\ldots,\boldsymbol{a}_{S_l}^{(l)} =: A^{(l)}$ are calculated using (2), and the speech activities for the l-th block $\hat{Y}^{(l)} \in (0,1)^{S_l\times\lambda}$ are calculated using (5). Here, S_l is the number of speakers that appeared in the l-th block, which satisfies $0 \leq S_l \leq S$. The diarization loss $\mathcal{L}_{\text{diar}}^{(l)}$ and attractor existence loss $\mathcal{L}_{\text{exist}}^{(l)}$ for the l-th block are calculated using (7) and (8), respectively.

The local attractors are clustered to find inter-block speaker correspondence. Since the local attractors themselves are optimized to minimize the diarization error, non-parametric similarities between them are not fit for speaker clustering, like cascaded methods require a scoring model based on probabilistic linear discriminant analysis. EEND-GLA includes the scoring model equivalent that is jointly optimized with the diarization and attractor existence losses. We first convert them by using the following Transformer decoder:

$$B^{(l)} = \mathsf{TransformerDecoder}\left(A^{(l)}, E, E\right) \in \mathbb{R}^{D\times S_l}, \tag{18}$$

where the first, second, and third arguments for the Transformer decoder are query, key, and value inputs, respectively. Here, the converted attractors B are expected to be speaker discriminative within each input audio. Thus, we refer to them as relative speaker embeddings, as contrasted to general speaker embeddings with global discriminability such as x-vectors. The relative speaker embeddings from all the blocks are gathered $B = [\boldsymbol{b}_i]_i := [B^{(1)},\ldots,B^{(L)}] \in \mathbb{R}^{D\times S^*}$ and optimized to minimize the pairwise loss defined as follows:

$$\begin{aligned}\mathcal{L}_{\text{pair}} = \sum_{i,j\in\{1,\ldots,S^*\}} \frac{1}{S^2 c_i c_j} \big(\chi_{ij}\left(1-\text{sim}\left(\boldsymbol{b}_i,\boldsymbol{b}_j\right)\right) \\ + \left(1-\chi_{ij}\right)\left[\text{sim}\left(\boldsymbol{b}_i,\boldsymbol{b}_j\right)-\delta\right]_+\big),\end{aligned} \tag{19}$$

$$\chi_{ij} = \begin{cases} 1 & (\boldsymbol{b}_i \text{ and } \boldsymbol{b}_j \text{ correspond to the same speaker}) \\ 0 & (\text{otherwise}) \end{cases}, \tag{20}$$

where $S^* := \sum_{l=1}^{L} S_l$ is the total number of local attractors, $\text{sim}(\boldsymbol{b}_i,\boldsymbol{b}_j) := \frac{\boldsymbol{b}_i^\mathsf{T}\boldsymbol{b}_j}{\|\boldsymbol{b}_i\|\|\boldsymbol{b}_j\|}$ is the cosine similarity between $\boldsymbol{b}_i$ and

[3] For simplicity, we assume that the length of the sequence T is divisible by λ, but in practice, the length of the last block can be shorter than λ.

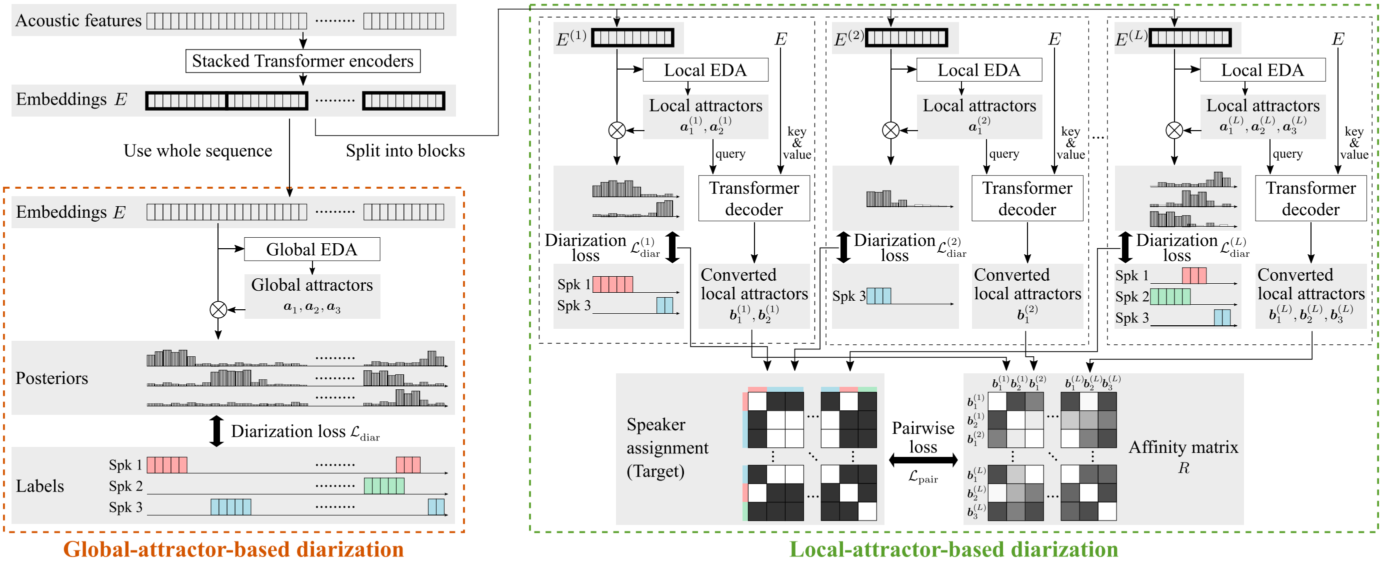

Acoustic features
Stacked Transformer encoders
Embeddings E
Use whole sequence
Split into blocks
Embeddings E
Global EDA
Global attractors
a_1, a_2, a_3
Posteriors
Diarization loss $\mathcal{L}_{\text{diar}}$
Labels
Spk 1
Spk 2
Spk 3
Global-attractor-based diarization
$E^{(1)}$
E
Local EDA
Local attractors
$a_1^{(1)}, a_2^{(1)}$
key & value
query
Transformer decoder
Diarization loss $\mathcal{L}_{\text{diar}}^{(1)}$
Spk 1
Spk 3
Converted local attractors
$b_1^{(1)}, b_2^{(1)}$
$E^{(2)}$
$a_1^{(2)}$
Diarization loss $\mathcal{L}_{\text{diar}}^{(2)}$
Spk 3
$b_1^{(2)}$
$E^{(L)}$
$a_1^{(L)}, a_2^{(L)}, a_3^{(L)}$
Diarization loss $\mathcal{L}_{\text{diar}}^{(L)}$
Spk 1
Spk 2
Spk 3
$b_1^{(L)}, b_2^{(L)}, b_3^{(L)}$
Speaker assignment (Target)
Pairwise loss
$\mathcal{L}_{\text{pair}}$
Affinity matrix
R
Local-attractor-based diarization

Fig. 3. End-to-end neural diarization with global and local attractors (EEND-GLA). The attractor existence losses are omitted from the illustration.

$\boldsymbol{b}_j$, and $[\cdot]_+$ is the hinge function. c_i (c_j) is the number of local attractors that correspond to the i-th (j-th) attractor's speaker, and this correspondence is obtained by finding the optimal speaker permutation when calculating the diarization loss using (7). This pairwise loss aims to make the angle between relative speaker embeddings of the same speaker as small as possible and those of different speakers at least $\arccos\delta$. In this paper, we used $\delta = 0.5$ during pretraining and $\delta = 0$ during adaptation. Note that this loss definition is based on the contrastive loss used for instance segmentation in computer vision [58], [59]. The process of grouping pixel-wise embeddings into instances is very similar to our problem setting of grouping local attractors into speaker identities. While x-vectors or frame-wise embeddings cannot be hardly assigned to one of the speaker identities because of overlaps, the local attractors can be divided by speaker identities because each of them corresponds to one speaker.

As a result, the loss based on local attractors is defined as

$$\mathcal{L}_{\text{local}} = \frac{1}{L}\sum_{l=1}^{L}\left(\mathcal{L}_{\text{diar}}^{(l)} + \alpha\mathcal{L}_{\text{exist}}^{(l)}\right) + \gamma\mathcal{L}_{\text{pair}}, \tag{21}$$

where γ is the weighting parameter for which we set $\gamma = 1$ in this study. The total loss of EEND-GLA is defined as a sum of global- and local-attractor-based losses:

$$\mathcal{L}_{\text{both}} = \mathcal{L}_{\text{local}} + \mathcal{L}_{\text{global}}. \tag{22}$$

2) Inference: During inference, the number of speakers within each block $\hat{S}_l \in \mathbb{Z}_{\geq 0}$ is estimated using (4), and speech activities of $\hat{S}_l$ speakers are estimated using (5). Speaker correspondence between blocks can be found by clustering the relative speaker embeddings B, and the problem here is how to determine the number of clusters.

Some conventional methods based on spectral clustering [10], [11] consist of the following steps: 1) construct an affinity matrix from frame-wise embeddings, 2) calculate its graph Laplacian, 3) use eigenvalue decomposition, and 4) determine the number of speakers as the value that maximizes the eigengap. Some tricks were used in these studies to reduce the effect of noise in the affinity matrix. In one study, an affinity matrix calculated from frame-wise d-vectors was smoothed by using Gaussian blur [10]. Another study utilized p nearest binarization to the affinity matrix to remove unreliable values [11]. In our case, local attractors are extracted not only for each block but also for each speaker within a block. Smoothing should be applied along the time axis of each speaker, but in this case, smoothing cannot be performed because the proper inter-block correspondence of the speakers has not been obtained. In our method, a few local attractors are calculated every five seconds, and hence p nearest neighbor binarization is also not suitable because it generally requires dozens of embeddings per cluster.

Therefore, in EEND-GLA, we use the unprocessed affinity matrix to estimate the number of clusters. However, if we estimate it based on the eigengaps of graph Laplacian, noises cause a lot of tiny clusters because the size of clusters is not considered in this approach. Thus, we use the affinity matrix directly instead of its graph Laplacian to penalize small clusters more. Given the positive-semidefinite affinity matrix $R = (r_{ij}) \in [-1, 1]^{S^* \times S^*}$, where $r_{ij} = \operatorname{sim}(\boldsymbol{b}_i, \boldsymbol{b}_j)$, the number of clusters $\hat{S}$ can be estimated using its eigenratios instead of eigengaps as

$$\hat{S} = \underset{1 \leq s \leq S^*-1}{\arg\min}\, \frac{\lambda_{s+1}}{\lambda_s}, \tag{23}$$

where $\lambda_1 \geq \cdots \geq \lambda_{S^*}$ are the non-negative eigenvalues of R, which are obtained with matrix decomposition:

$$R = V \operatorname{diag}\left(\lambda_1, \ldots, \lambda_{S^*}\right) V^{-1}, \tag{24}$$

where each row of $V \in \mathbb{R}^{S^* \times S^*}$ is the eigenvector that corresponds to the eigenvalues. Note that the eigenvalues indicate the number of elements of each cluster where local attractors are softly assigned.

We used the hinge function to calculate the pairwise loss in (19), and we also know that attractors from the same block correspond to different speakers. Thus, instead of R, we use the affinity matrix $R' = (r'_{ij})$ defined as

$$r'_{ij} = \begin{cases} \mathbb{1}\left(i = j\right) & (\boldsymbol{b}_i \text{ and } \boldsymbol{b}_j \text{ are from the same block}) \\ \frac{1}{1-\delta}\left[\mathrm{sim}\left(\boldsymbol{b}_i, \boldsymbol{b}_j\right) - \delta\right]_+ & (\text{otherwise}) \end{cases}, \quad (25)$$

where $\mathbb{1}(\text{cond})$ is the indicator function that returns 1 if cond is true and 0 otherwise. Matrix decomposition is then applied to R' to obtain eigenvalues $\lambda'_1 \geq \cdots \geq \lambda'_{S^*}$ in the same manner as in (24). Although R' is no longer positive-semidefinite, its eigenvalues are still good indicators of cluster size. We only use the eigenvalues greater than or equal to one to estimate the number of speakers $\hat{S}$ as follows:

$$\hat{S} = \mathop{\arg\min}_{\substack{1 \leq s \leq S^* - 1 \\ \lambda'_s \geq 1}} \frac{\lambda'_{s+1}}{\lambda'_s}. \quad (26)$$

Although we set the affinity value between a pair of local attractors from the same block to be zero in (25), naive clustering methods cannot force them to be assigned to different clusters. Thus, we utilize a clustering method that can use cannot-link constraints. COP-Kmeans clustering [60], which is used in EEND-vector clustering [31], [32] is one possible choice, but it sometimes results in failure because it cannot find the solution that fulfills the given constraints. Thus, we use the CLC-Kmeans algorithm [61], which is the modified version of the COP-Kmeans clustering, for inference of EEND-GLA. To avoid having no solution due to cannot-link constraints, we update the estimated number of speakers before applying clustering as

$$\hat{S} \leftarrow \max\left(\hat{S}, \max_{1 \leq l \leq L} \hat{S}_l\right). \quad (27)$$

EEND-GLA is optimized using both global- and local-attractor-based losses as in (22), and we can use not only local attractors but also global attractors for inference. Although local-attractor-based inference can deal with an arbitrary number of speakers, we found that global-attractor-based inference performs better when the number of speakers is low because it is trained in a fully supervised manner. Therefore, we use the results from global and local attractors depending on the estimated number of speakers. Assume that EEND-GLA is trained on mixtures each of which contains at most N speakers. If the estimated number of speakers using global attractors is less than N, we use the inference results based on global attractors. If it is equal to or larger than N, we use the inference results based on local attractors. In this paper, the value of N is set to four based on the simulated datasets we used for training, which are detailed in Section V. Even after the domain adaptation with real datasets with a larger number of speakers, we keep the value of N unchanged during inference.

B. Block-Wise Speaker-Tracing Buffer

As introduced in Section III-B, the original speaker-tracing buffer (STB) includes a frame-wise (FW) selection step to meet the requirement of the buffer length. Hereafter, for sake of distinction, we refer to it as FW-STB. When trying to use FW-STB with EEND-GLA to perform online diarization of an unlimited number of speakers, the frame-wise selection can become a problem if the selected frames are not consecutive in the whole buffer. The FIFO strategy ensures that the frames in the buffer are consecutive, but as mentioned in Section III-B, it has difficulty in capturing long context. On the other hand, while the sampling strategy can maintain long-range speaker consistency, the buffer can potentially contain non-consecutive frames of many different speakers; thus, the assumption of a limited number of speakers in a limited sequence of frames in the buffer does not hold.

To overcome this dilemma, we propose a block-wise speaker-tracing buffer (BW-STB). The core idea of BW-STB is that it guarantees that the buffer consists of blocks, and each block contains the features and the corresponding results of consecutive frames. If each block in the buffer is short enough that we can assume a limited amount of speakers, EEND-GLA can be used in the same way as the offline inference in which local attractors are obtained from the blocks formed by consecutive frames. However, a naive implementation that waits for block-length features to accumulate and then processes them would result in block-length latency. Thus, we use a frame-wise FIFO buffer and block-wise sampling buffer together to enable a low-latency online inference of EEND-GLA.

Fig. 4 shows the proposed BW-STB. For simplicity, we assume that the buffer length M is divisible by and longer than the block length λ, and λ is divisible by the online processing unit ν. M-length BW-STB is divided into blocks of length λ each. The first $\frac{M}{\lambda} - 1$ is updated via block-wise sampling, and the last one is updated in a frame-wise FIFO manner. We call them the sampling buffer and the FIFO buffer, respectively. The features in BW-STB before the n-th input $X_n \in \mathbb{R}^{F \times \nu}$ can be written as

$$X_{n-1}^{(\text{buf})} = \left[X_{n-1}^{(\text{samp})} \quad X_{n-1}^{(\text{FIFO})}\right] \in \mathbb{R}^{F \times M}, \quad (28)$$

$$X_{n-1}^{(\text{samp})} = \left[X_{n-1}^{(\text{samp})}\left[1\right] \ldots X_{n-1}^{(\text{samp})}\left[\frac{M}{\lambda} - 1\right]\right] \in \mathbb{R}^{F \times (M - \lambda)}. \quad (29)$$

Here, $X_{n-1}^{(\text{samp})}$ are the features in the sampling buffer, where each $X_{n-1}^{(\text{samp})}[k] \in \mathbb{R}^{F \times \lambda}$ $(k \in \{1, \ldots, \frac{M}{\lambda} - 1\})$ are the features of consecutive λ frames. $X_{n-1}^{(\text{FIFO})} \in \mathbb{R}^{F \times \lambda}$ is the features in the FIFO buffer, which contains those of the latest consecutive λ frames. In addition, each buffer contains the corresponding diarization results $Y_{n-1}^{(\text{samp})} \in (0, 1)^{\hat{S}_{n-1} \times (M - \lambda)}$ and $Y_{n-1}^{(\text{FIFO})} \in (0, 1)^{\hat{S}_{n-1} \times \lambda}$.

Given the input X_n, the features in the FIFO buffer are first updated as

$$X_n^{(\text{FIFO})} = X_{n-1}^{(\text{FIFO})} \begin{bmatrix} O_{\nu, \lambda - \nu} & O_{\nu, \nu} \\ I_{\lambda - \nu} & O_{\lambda - \nu, \nu} \end{bmatrix} + X_n \begin{bmatrix} O_{\nu, \lambda - \nu} & I_\nu \end{bmatrix}, \quad (30)$$

where I_a is an $a \times a$ identity matrix. Note that the first $\lambda - \nu$ columns of $X_n^{(\text{FIFO})}$ are identical to the last $\lambda - \nu$ columns of $X_{n-1}^{(\text{FIFO})}$, and the last ν columns of $X_n^{(\text{FIFO})}$ are identical to X_n.

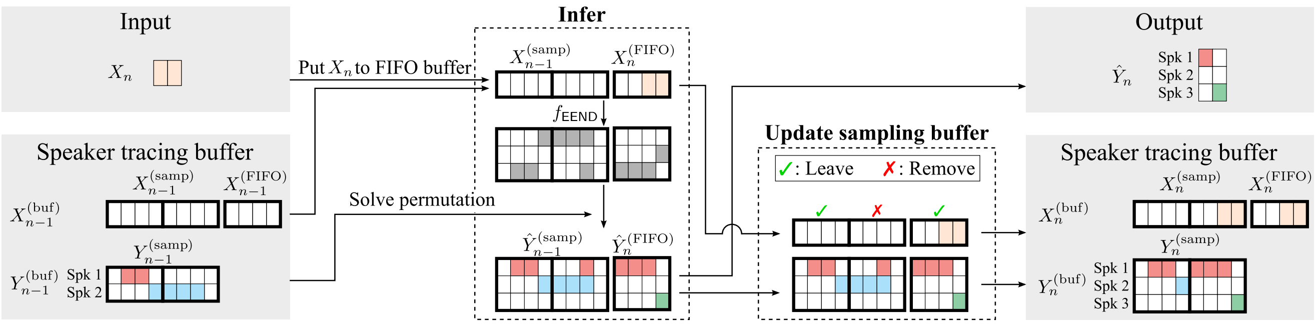
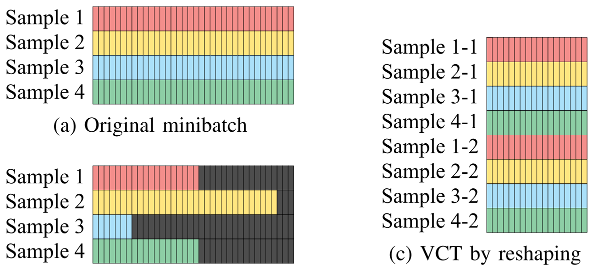


Fig. 4. Online diarization using speaker-tracing buffer with block-wise update.

TABLE I
EXAMPLE OF SAMPLING WEIGHTS DETERMINED BY (15) AND (32)

	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$	$t=8$
$y_{1,t}$	0.999	0.999	0.999	0.999	0.999	0.001	0.001	0.001
$y_{2,t}$	0.001	0.001	0.001	0.001	0.001	0.001	0.001	0.999
$\tilde{\omega}_t$ by (17)(15)	0.167	0.167	0.167	0.167	0.167	0.000	0.000	0.167
$\tilde{\omega}_t$ by (17)(32)	0.101	0.101	0.101	0.101	0.101	0.000	0.000	0.497



(a) Original minibatch

(b) VCT by masking

(c) VCT by reshaping

Fig. 5. Batch creation in the VCT.

Then, the diarization results are calculated from the concatenation of the features in the sampling and FIFO buffers as

$$\left[\hat{Y}_{n-1}^{(\mathrm{samp})} \quad \hat{Y}_n^{(\mathrm{FIFO})}\right] = f_{\mathsf{EEND}}\left(\left[X_{n-1}^{(\mathrm{samp})} \quad X_n^{(\mathrm{FIFO})}\right]\right). \tag{31}$$

With this estimation, the number of speakers is aligned via zero padding as described in Section III-B, and then the speaker order of $\hat{Y}_{n-1}^{(\mathrm{samp})}$ and $\hat{Y}_n^{(\mathrm{FIFO})}$ is aligned to that of $Y_{n-1}^{(\mathrm{samp})}$ using (10)–(12). Next, we output the last ν columns of the updated $Y_n^{(\mathrm{FIFO})}$, which correspond to the input X_n.

The sampling buffer is updated every time the FIFO buffer is fully replaced, i.e., after processing the n-th input where $n \equiv 0 \bmod \frac{\lambda}{\nu}$. During updates, $\frac{M}{\lambda} - 1$ blocks are selected from $X_{n-1}^{(\mathrm{samp})}[1] \ldots X_{n-1}^{(\mathrm{samp})}[\frac{M}{\lambda} - 1]$ and $X_n^{(\mathrm{FIFO})}$, and they are stored as $X_n^{(\mathrm{samp})}$ in the sampling buffer. The sampling probability of each block is calculated as a sum of $\tilde{\omega}_t$ of the frames in the block calculated using (17).

With the aforementioned BW-STB, the online inference having the algorithmic latency of $\nu (\ll \lambda)$ is enabled. Note that online diarization is performed using the FIFO buffer in the same way as FW-STB from the first to $\frac{\lambda}{\nu}$-th iterations because the sampling buffer is empty.

C. Speaker-Balanced Sampling Probabilities

The score in (15) is designed to weigh more on frames where a single speaker dominates the conversation; as a result, the speaker-tracing buffer becomes informative enough to solve the speaker permutation ambiguity in (10)–(11). However, in the case where some speakers dominate the conversation, the buffer contents might be biased toward those speakers, and hence the permutation ambiguity cannot be solved correctly. For example, in the two-speaker example shown in Table I, $\tilde{\omega}_t$ is maximized at $t \in \{1, 2, 3, 4, 5, 8\}$, where $(y_{1,t}, y_{2,t}) \in \{(0.001, 0.999), (0.999, 0.001)\}$. If $t = 8$ is not selected to be stored in the buffer and the third speaker emerges in the next input, we cannot distinguish between the second and third speakers.

To make the buffer unbiased, we introduce the weighting factor r_t into the sampling probability ω_t to balance the number of frames to be stored for each speaker. We propose the following alternative:

$$\omega_t = r_t \underbrace{\sum_{s=1}^{S_n} \bar{y}_{s,t} \log\left(\bar{y}_{s,t} S_n\right)}_{(15)}, \tag{32}$$

where r_t is defined as

$$r_t = \sum_{s=1}^{S_n} \frac{y_{s,t}}{\sum_{t'=1}^{T} y_{s,t'}}. \tag{33}$$

By this modification, in Table I, the sampling probability of $t = 8$ becomes about a five times larger value (0.497) than that of $t \in \{1, 2, 3, 4\}$ (0.101); thus, it is more likely to prevent the buffer from storing information that is biased toward the dominant speaker, i.e., the first speaker.

D. Variable Chunk-Size Training Via Minibatch Reshaping

The VCT described in the last paragraph of Section III-B varied the length of sequences by masking a part of each sequence (Fig. 5(b)). However, its calculation efficiency is low because the masked part does not contribute to the network optimization while still consuming GPU memory during training.

Therefore, we consider a method to use inputs of various lengths in the training process by reshaping the minibatch instead of masking. If the minibatch at an iteration has minibatch size B and input length T, we first reshape it to be a new

TABLE II
DATASETS USED IN OUR EXPERIMENTS

(a) Simulated datasets.

Dataset		#Spk	#Mixtures	Average duration	Overlap ratio
Train	Sim1spk	1	100,000	76.8 s	0.0 %
	Sim2spk	2	100,000	88.6 s	34.1 %
	Sim3spk	3	100,000	151.2 s	34.2 %
	Sim4spk	4	100,000	238.1 s	31.5 %
Adaptation	Sim1spk	1	1,000	76.0 s	0.0 %
	Sim2spk	2	1,000	89.3 s	34.5 %
	Sim3spk	3	1,000	150.3 s	34.9 %
	Sim4spk	4	1,000	238.2 s	31.4 %
Test	Sim1spk	1	500	77.2 s	0.0 %
	Sim2spk	2	500	88.2 s	34.4 %
	Sim3spk	3	500	149.7 s	34.7 %
	Sim4spk	4	500	237.4 s	32.0 %
	Sim5spk	5	500	328.8 s	30.7 %
	Sim6spk	6	500	423.4 s	29.9 %

(b) Real datasets.

Dataset		Split	#Spk	#Mixtures	Average duration	Overlap ratio
Adaptation	CALLHOME [62]	Part 1	2–7	249	125.8 s	17.0 %
	DIHARD II [63]	dev	1–10	192	444.8 s	9.8 %
	DIHARD III [64]	dev	1–10	254	483.4 s	10.7 %
Test	CALLHOME [62]	Part 2	2–6	250	123.2 s	16.7 %
	DIHARD II [63]	eval	1–9	194	416.3 s	8.9 %
	DIHARD III [64]	eval	1–9	259	458.1 s	9.2 %

TABLE III
STEP-BY-STEP IMPROVEMENT IN THE ONLINE INFERENCE OF EEND-EDA ON THE CALLHOME DATASET

			Buffer length (s)							
	VCT	ω_t	1	2	5	10	20	50	100	∞
FW-STB [26] †	None	(15)	N/A	N/A	N/A	26.6	N/A	20.0	19.5	N/A
FW-STB [26]	None	(15)	89.79	76.98	43.87	28.13	21.82	19.69	18.54	18.34
FW-STB	Mask	(15)	50.56	44.95	27.48	21.53	18.12	16.82	16.78	15.69
FW-STB	Reshape	(15)	**44.11**	**36.61**	25.41	20.54	18.11	16.20	15.74	**15.00**
FW-STB	Reshape	(32)	45.60	37.36	**24.19**	**20.10**	**16.79**	15.50	**14.93**	**15.00**
BW-STB	Reshape	(32)	N/A	N/A	N/A	24.27	16.84	**15.03**	15.06	15.42

† The values are from the original FW-STB paper [26].
VCT: variable chunk-size training.

minibatch with the size $B' = \frac{BT}{T'}$ and length T', and then use it for training. For Fig. 5, the original minibatch has the size of four (Fig. 5(a)), and the reshaped minibatch has the size of eight by setting $T' = \frac{T}{2}$ (Fig. 5(c)). In this paper, we set $B = 64$ and $T = 2000$, and in each iteration, with a probability of 50 %, we set T' to one of $\{50, 100, 200, 500, 1000\}$ to conduct VCT.

V. EXPERIMENTAL SETTINGS

The initial training of each EEND-based model was based on the simulated mixtures shown in Table II(a). These were made with NIST SRE and Switchboard datasets as speech corpora, MUSAN [65] as a noise corpus, and simulated room impulse responses [66], following the protocol used for the original EEND [14]. Following our previous studies [16], [33], we first trained each EEND-based model using Sim2spk from scratch for 100 epochs and then finetuned it using the concatenation of Sim$\{1,2,3,4\}$spk for another 50 epochs. The Adam optimizer [67] with Noam scheduler [68] was used during the training using the simulated datasets. For online purposes, the model was adapted using the adaptation set of Sim$\{1,2,3,4\}$spk for an additional 100 epochs using variable chunk-size training (VCT). This time, the Adam optimizer with a fixed learning rate of 1×10^{-5} was used. Note that the adaptation set of each simulated dataset was the subset of the corresponding training set. The training process took about two weeks with a single NVIDIA Tesla V100 GPU.

We also used the real datasets shown in Table II(b) for evaluation. The model pretrained using Sim$\{1,2,3,4\}$spk was further adapted to the CALLHOME, DIHARD II, and DIHARD III datasets, respectively. The adaptation was conducted for another 100 epochs using the Adam optimizer with a learning rate of 1×10^{-5}. For online purposes, VCT was used instead.

For EEND-GLA, we used four- or six-stacked Transformer encoders, each outputting 256-dimensional frame-wise embeddings. We call each EEND-GLA-Small and EEND-GLA-Large, respectively. For EDA, we used an encoder-decoder based on single-layer long short-term memory with 256-dimensional hidden units. Note that the order of the input sequence is shuffled before being fed into EDA following the conventional study [16]. For the inputs to the models, 345-dimensional acoustic features extracted for each 100 ms were used, and they were obtained in the following steps: 1) extract 23-dimensional log-mel filterbanks for every 10 ms, 2) apply frame splicing (±7 frames), and 3) subsample by a factor of 10.

Unless otherwise specified, the length of an online processing unit ν was set to 1 s, and the buffer length was set to 100 s. The block length λ of the BW-STB was set to 5 s following EEND-GLA [33]; as a result, the length of sampling and FIFO buffers are 95 s and 5 s, respectively.

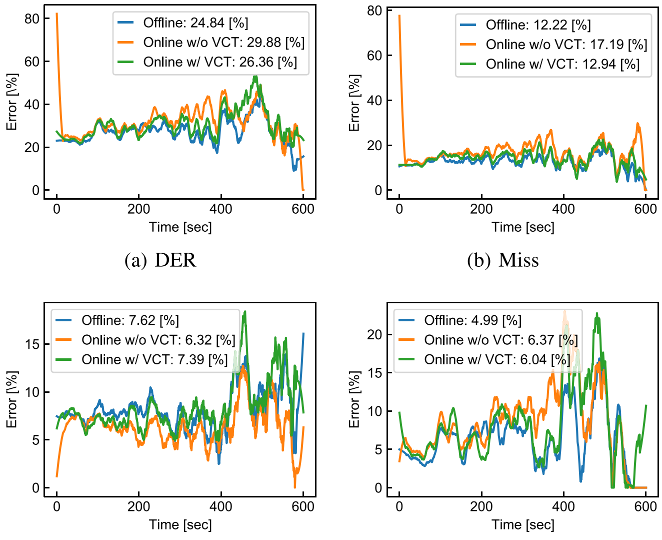
For evaluating offline diarization, we utilized several cascaded methods [12], [13], [22], [46], [69] and end-to-end methods [15], [16], [29], [32] for comparison. For evaluating online diarization, we used FW-STB with EEND-EDA based on four-stacked Transformers [26]. In addition, we referred to the results of various conventional online diarization methods [24], [26], [50], [51], [52], [70], [71] on various datasets. Some cascaded comparison methods [50], [51], [70] used the oracle SAD; for a fair comparison, we used SAD post-processing [16] for the results of EEND-based methods to recover missed speech and filter false-alarmed speech.

For the evaluation protocol, we used DERs. Following the previous studies [16], we forgave 0.25 s of its collar tolerance in the evaluations of the simulated datasets and CALLHOME, while we did not allow such a collar in the evaluations of the DIHARD II and DIHARD III datasets.

VI. RESULTS

A. *Evaluation of the Variations of Speaker-Tracing Buffer*

Before we dive into the evaluation of EEND-GLA, we evaluated the effects of each modification on the speaker-tracing buffer using EEND-EDA. Step-by-step improvement on the CALLHOME dataset is shown in Table III. The DERs were significantly reduced by using VCT. In a comparison of the



(a) DER

(b) Miss

(c) False alarm

(d) Speaker confusion

Fig. 6. Frame-wise breakdown of diarization error on CALLHOME.

TABLE IV
DERs (%) on the Simulated Datasets With 0.25 s Collar Tolerance. Unless Otherwise Specified, Each Online System Had an Algorithmic Latency of 1 s.

	# of speakers					
	seen				unseen	
	1	2	3	4	5	6
Offline						
X-vector clustering	37.42	7.74	11.46	22.45	31.00	38.62
EEND-EDA [15], [16]	0.15	**3.19**	6.60	9.26	23.11	34.97
EEND-EDA [15], [16] †	0.15	**3.19**	6.60	8.68	22.43	33.28
EEND-GLA-Small	0.25	3.53	6.79	8.98	**12.44**	**17.98**
EEND-GLA-Large	**0.09**	3.54	**5.74**	**6.79**	12.51	20.42
EEND-EDA [15], [16] ‡	0.36	3.65	7.70	9.97	11.95	22.59
Online						
BW-EDA-EEND [24]§	**1.03**	6.10	12.58	19.17	N/A	N/A
EEND-EDA [15], [16] + FW-STB	1.50	5.91	9.79	11.92	26.57	37.31
EEND-EDA [15], [16] + FW-STB †	1.50	5.91	9.79	11.85	26.63	37.25
EEND-GLA-Small + BW-STB	1.19	5.18	9.41	13.19	**16.95**	**22.55**
EEND-GLA-Large + BW-STB	1.12	**4.61**	**8.14**	**11.38**	17.27	25.77
EEND-EDA [15], [16] + FW-STB ‡	1.33	6.01	10.49	12.64	15.28	26.09

† Four attractors were used at most.
‡ Trained on Sim{1,2,3,4,5}spk. Five attractors were used at most.
§ Algorithmic latency 10 s.
Unless otherwise specified, each online system had an algorithmic latency of 1 s.

results in the third and fourth lines, VCT by reshaping outperformed that by masking in all the conditions. Introducing the speaker-balancing term in the sampling probability as in (32) improved the DERs except when the buffer length was too short to store enough information to solve the speaker permutation ambiguity, as in the fifth line. Finally, replacing FW-STB with BW-STB did not affect the diarization performance as shown in the last line, except when the buffer length was 10 s, where the sampling buffer consisted of only one block.

For the detailed error analyses, we show the frame-level breakdown of the diarization error of FW-STB with and without VCT in Fig. 6. Each graph was smoothed along the time axis using the Savizky-Golay filter [72] for visualization purposes. We clearly observed that VCT drastically decreased the error caused by missed speech at the very beginning of recordings with a slight increase in false alarms. Note that the DERs shown in Fig. 6 are different from those in Table III because the results are without a collar.

In the following experiments, we used FW-STB and BW-STB in the last two lines in Table III, i.e., VCT by reshaping and speaker-balanced sampling probabilities were utilized.

B. *Evaluation of Offline and Online Diarization for an Unlimited Number of Speakers*

1) Simulated Dataset: We first evaluated EEND-GLA on the simulated datasets. The results are shown in Table IV.

For the evaluation of offline processing, Kaldi's x-vector clustering recipe[4] was used as a baseline. The x-vector extractor was trained using the same set of datasets that were used to create the simulated datasets in Table II(a). Note that the baseline has no way to handle overlapping speech. EEND-EDA and EEND-GLA-Small performed evenly on the datasets of the seen number of speakers, while EEND-GLA-Small significantly outperformed EEND-EDA on the datasets of the unseen number of speakers. It clearly showed that EEND-GLA-Small could deal with a higher number of speakers than that observed during training by introducing clustering. It is worth mentioning that EEND-EDA sometimes outputs more than four attractors, but the results in Table IV in which ignoring the fifth and subsequent attractors improved the DERs indicate that these attractors were not correctly calculated to represent the fifth and subsequent speakers. Using EEND-GLA-Large improved the DERs for the seen number of speakers, but those for the unseen number of speakers were degraded. We considered this to be because the network was overtrained on the seen number of speakers with the larger model. For comparison, we also showed the DERs of EEND-EDA trained using mixtures, each of which contained at most five speakers. It showed a better DER on five-speaker mixtures, but the DER on six-speaker mixtures degraded rapidly. EEND-GLA achieved DERs comparable to EEND-EDA for five-speaker mixtures and significantly outperformed it for six-speaker mixtures.

In terms of online processing, STB-based methods outperformed BW-EDA-EEND [24] in all but single-speaker data even though the online processing unit was 1 s, which was ten times shorter than that of BW-EEND-EDA. Online inference of EEND-GLA-Small using BW-STB significantly improved DERs on five- and six-speaker mixtures, which were not observed during training. EEND-GLA-Large improved the DERs for the seen number of speaker conditions of EEND-GLA-Small but degraded the DERs for the unseen number of speaker conditions, the same as in offline inference.

Table V shows the offline DERs of EEND-GLA-Small obtained with various training and inference strategies. Even when only local attractors were used during both training and inference, it achieved better DERs than EEND-EDA for the unseen

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

TABLE V
OFFLINE DERS (%) OF EEND-GLA-SMALL WITH VARIOUS TRAINING AND INFERENCE STRATEGIES. LOSS: THE TRAINING OBJECTIVE USED FOR TRAINING. INFERENCE: ATTRACTORS USED DURING INFERENCE

		# of speakers					
		Seen				Unseen	
Loss	Inference	1	2	3	4	5	6
$\mathcal{L}_{\text{local}}$ (21)	Local	8.85	12.71	10.31	11.14	14.11	19.36
$\mathcal{L}_{\text{local}} + \mathcal{L}_{\text{global}}$ (22)	Local	2.84	10.21	7.54	9.08	**12.40**	18.03
$\mathcal{L}_{\text{local}} + \mathcal{L}_{\text{global}}$ (22)	Local & Global	**0.25**	**3.53**	**6.79**	**8.98**	12.44	**17.98**

Loss: the training objective used for training. Inference: attractors used during inference.

TABLE VI
CONFUSION MATRICES FOR SPEAKER COUNTING ON THE SIMULATED DATASETS

(a) EEND-EDA (offline)

	Ref. #Speakers 1	2	3	4	5	6
Pred. #Speakers 1	**500**	0	0	0	0	0
2	0	**482**	0	0	0	0
3	0	17	**435**	5	1	0
4	0	1	65	**447**	224	139
5	0	0	0	48	**268**	337
6	0	0	0	0	7	**24**
7+	0	0	0	0	0	0

(b) EEND-GLA-Small (offline)

	Ref. #Speakers 1	2	3	4	5	6
Pred. #Speakers 1	**498**	0	0	0	0	0
2	2	**474**	0	0	0	0
3	0	25	**451**	17	2	1
4	0	1	33	**412**	78	30
5	0	0	10	62	**361**	183
6	0	0	6	7	47	**229**
7+	0	0	0	2	12	57

(c) EEND-EDA + FW-STB (online)

	Ref. #Speakers 1	2	3	4	5	6
Pred. #Speakers 1	**376**	0	0	0	0	0
2	120	**244**	0	0	0	0
3	4	249	**252**	1	0	0
4	0	7	245	**449**	271	172
5	0	0	3	50	**222**	314
6	0	0	0	0	7	**14**
7+	0	0	0	0	0	0

(d) EEND-GLA-Small + BW-STB (online)

	Ref. #Speakers 1	2	3	4	5	6
Pred. #Speakers 1	**411**	0	0	0	0	0
2	84	**343**	0	0	0	0
3	5	156	**370**	3	0	0
4	0	1	109	**302**	16	0
5	0	0	20	181	**364**	38
6	0	0	1	13	114	**385**
7+	0	0	0	1	6	77

TABLE VII
DERS (%) ON CALLHOME WITH 0.25 s COLLAR TOLERANCE. UNLESS OTHERWISE SPECIFIED, EACH ONLINE SYSTEM HAD AN ALGORITHMIC LATENCY OF 1 s

	# of speakers					
Method	2	3	4	5	6	All
Offline						
VBx [13] †	9.44	13.89	16.05	**13.87**	24.73	13.28
MTFAD [46]	N/A	N/A	N/A	N/A	N/A	14.31
SC-EEND [29]	9.57	14.00	21.14	31.07	37.06	15.75
EEND-EDA [15], [16]	7.83	12.29	17.59	27.66	37.17	13.65
EEND-vector clust. [32]	7.94	11.93	16.38	21.21	23.10	12.49
EEND-VC-iGMM [34]	8.6	12.6	16.1	27.5	26.9	13.3
EEND-GLA-Small	**6.94**	**11.42**	14.49	29.76	24.09	11.92
EEND-GLA-Large	7.11	11.88	**14.37**	25.95	**21.95**	**11.84**
Online						
BW-EDA-EEND [24] ‡	11.82	18.30	25.93	N/A	N/A	N/A
EEND-EDA [15], [16] + FW-STB §	12.70	18.40	24.30	35.83	42.21	19.51
EEND-EDA [15], [16] + FW-STB	9.08	13.33	19.36	30.09	37.21	14.93
EEND-GLA-Small + BW-STB	**9.01**	12.73	19.45	32.26	36.78	14.80
EEND-GLA-Large + BW-STB	9.20	**12.42**	**18.21**	**29.54**	**35.03**	**14.29**

† The oracle SAD was used.
‡ Algorithmic latency of 10 s.
§ The values are from the original FW-STB paper [26].
Unless otherwise specified, each online system had an algorithmic latency of 1 s

TABLE VIII
DERS (%) ON DIHARD II WITH NO COLLAR TOLERANCE. EACH ONLINE SYSTEM HAD AN ALGORITHMIC LATENCY OF 1 S

	# of speakers		
Method	≤ 4	≥ 5	All
Offline			
BUT system [22]	**21.34**	39.85	27.11
VBx + overlap-aware resegmentation [12]	21.41	**36.93**	**26.25**
EEND-EDA [15]	22.09	47.66	30.07
EEND-GLA-Small	22.24	44.92	29.31
EEND-GLA-Large	21.40	43.62	28.33
Online			
Overlap-aware speaker embeddings [52]	27.00	52.62	34.99
EEND-EDA [15], [16] + FW-STB †	28.14	53.64	36.09
EEND-EDA [15], [16] + FW-STB	25.63	50.45	33.37
EEND-GLA-Small + BW-STB	23.96	48.06	31.47
EEND-GLA-Large + BW-STB	**22.62**	**47.06**	**30.24**
Online (with oracle voice activity detection)			
UIS-RNN [50]	N/A	N/A	30.9
UIS-RNN-SML [51]	N/A	N/A	27.3
Core samples selection [71]	N/A	N/A	23.1
EEND-EDA [15], [16] + FW-STB †	17.21	43.58	25.44
EEND-EDA [15], [16] + FW-STB	16.56	42.58	24.67
EEND-GLA-Small + BW-STB	15.29	40.85	23.26
EEND-GLA-Large + BW-STB	**13.55**	**40.39**	**21.92**

† The values are from the original FW-STB paper [26].
Each online system had an algorithmic latency of 1 s.

numbers of speakers but worse ones for the seen numbers of speakers (first row). Using the global attractors jointly for training improved the performance for the seen numbers of speakers, but it was still not as good as EEND-EDA when only the local attractors were used for inference (second row), especially when the number of speakers was low (i.e., one- or two-speaker cases). This is because a small error in the number of speakers (e.g., ±1) led to a high degradation of DER. Using the results based on global attractors when the number of speakers was low resulted in good DERs for both seen and unseen numbers of speakers (third row).

We also show the confusion matrices for speaker counting on the simulated datasets in Table VI. The speaker counting accuracy of EEND-GLA-Small with BW-STB outperformed that of EEND-EDA with FW-STB, and the gaps between them were larger especially when the number of speakers was higher than four. Note that EEND-EDA with FW-STB sometimes produced the results of more than four speakers, but they did not help estimate the speech activities of more than four speakers correctly as we stated in this section.

2) CALLHOME: Table VII shows the DERs on the CALLHOME dataset. In the evaluation of offline processing, EEND-GLA-Small and EEND-GLA-Large outperformed the conventional methods with 11.92 % and 11.84 % DERs, respectively.

In online diarization, compared with the original FW-STB [26], our updates on VCT and the sampling probabilities improved the DERs from 19.51 % to 14.93 %. EEND-GLA-Small and EEND-GLA-Large with BW-STB further improved DERs to 14.80 % and 14.29 %, respectively. Our method also outperformed BW-EDA-EEND [24] by a large margin.

TABLE IX
DERs (%) on DIHARD III With No Collar Tolerance. Unless Otherwise Specified, Each Online System Had an Algorithmic Latency of 1 s

	# of speakers		
Method	≤ 4	≥ 5	All
Offline			
VBx + overlap handling [69]	16.38	42.51	21.47
VBx + overlap-aware resegmentation [12]	15.32	**35.87**	**19.33**
EEND-EDA [15], [16]	15.55	48.30	21.94
EEND-GLA-Small	14.39	44.32	20.23
EEND-GLA-Large	**13.64**	43.67	19.49
Online			
Overlap-aware speaker embeddings [52]	21.07	54.28	27.55
EEND-EDA [15], [16] + FW-STB	19.00	50.21	25.09
EEND-GLA-Small + BW-STB	15.87	47.27	22.00
EEND-GLA-Large + BW-STB	**14.81**	**45.17**	**20.73**
Online (with oracle voice activity detection)			
System by Zhang et al. [70] †	N/A	N/A	19.57
Core samples selection [71]	N/A	N/A	19.3
EEND-EDA [15], [16] + FW-STB	12.80	42.46	18.58
EEND-GLA-Small + BW-STB	9.91	40.21	15.82
EEND-GLA-Large + BW-STB	**8.85**	**38.86**	**14.70**

† Algorithmic latency of 0.5 s.
Unless otherwise specified, each online system had an algorithmic latency of 1 s.

3) DIHARD II and III: Table VIII shows the results on the DIHARD II dataset. In offline diarization, EEND-GLA-Small and EEND-GLA-Large improved the DERs from EEND-EDA, especially when the number of speakers was higher than four. Compared with the cascaded method [73] or the cascaded method incorporated with EEND for post-processing [12], EEND-GLA-Large performed on par with them when the number of speakers was low, but not when the number of speakers was high.

In online diarization, the DER of EEND-EDA was improved by using the proposed FW-STB from 36.09 % to 33.37 %, and BW-STB further improved the DERs to 31.47 % and 30.24 % with EEND-GLA-Small and EEND-GLA-Large, respectively. If we focus on the comparison methods, overlap-aware speaker embedding [12], [52] had a large gap in the DERs between offline and online inference (26.25 % vs. 34.99 %). This is because its offline performance was highly boosted by using VBx [13], which is not suited for online inference. However, the gap between the DERs of offline and online inference of EEND-GLA was only about two points and outperformed the comparable method in both cases where the number of speakers was low or high. We also show the DERs with UIS-RNN [50] and UIS-RNN-SML [51], which are based on fully supervised clustering of d-vectors extracted using a sliding window, under the condition that the oracle SAD was used. In this case, too, the EEND-GLA-based methods outperformed these comparable methods.

We also show the DERs on the DIHARD III dataset in Table IX. The results were almost the same as those of the DIHARD II dataset. EEND-GLA-Large achieved 19.49 % DER in offline diarization, which was as accurate as the best performing conventional method [12], and 20.73 % DER in online diarization, which was about seven points better than that of the conventional method [52].

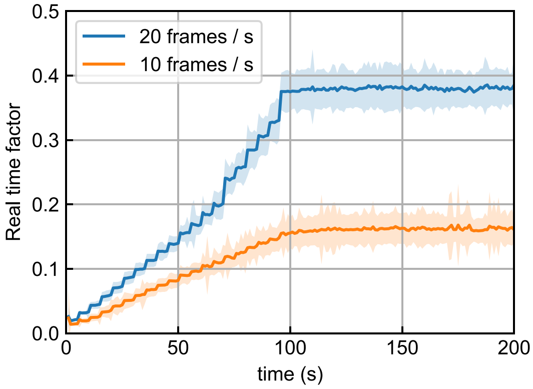


Fig. 7. Real time factor of EEND-GLA-Small with BW-STB calculated using Sim5spk. The filled areas represent the standard deviations. The DERs are 16.95 % and 18.18 % with 10 frames/s and 20 frames/s conditions, respectively.

C. Real Time Factor

To show that our method is applicable for real-time inference, we calculated the real time factor of EEND-GLA-Small with BW-STB. For the calculation, we used Sim5spk, in which clustering of relative speaker embeddings is always necessary (cf. Table VI(d)). The calculation was on an Intel Xeon Gold 6132 CPU @ 2.60 GHz using seven threads without any GPUs. Again, we used the buffer length of 100 s buffer and the online process unit length of 1 s. Fig. 7 shows the real time factor calculated as the processing time for each online process unit. The real time factor increased approximately linearly until the buffer was filled, and then it became constant. It indicates that, at least for buffer length of 100 s, the inference speed of EEND-GLA is not constrained by clustering of local attractors described in Section IV-A, which has $O(n^3)$ time complexity. The convergence value of the real time factor was about 0.16 with 10 frames per second and 0.38 with 20 frames per second. These results demonstrate that our method is fast enough for real-time inference.

VII. Conclusion

In this paper, we proposed EEND-GLA, a neural diarization method that can treat an unlimited number of speakers. In EEND-GLA, diarization is performed on the basis of global attractors extracted from the entire input and local attractors extracted from each chunked input, respectively. To enable online inference of EEND-GLA, we also proposed a block-wise speaker-tracing buffer; it is partitioned into blocks, and each block stores temporally continuous features and the corresponding results. The novel speaker-balanced sampling probabilities for buffer update and variable chunk-size training via minibatch reshaping were also proposed to improve online diarization.

The experimental results showed that EEND-GLA performed well on both offline and online inferences. In particular, EEND-GLA significantly outperformed the conventional methods on various datasets in online diarization. The performance of the cascaded methods heavily relies on the clustering algorithm; offline diarization can utilize two-stage clustering like VBx, while online diarization cannot. Thus, a severe gap remains between the DERs of offline and online inference of the cascaded methods. In contrast, the offline and online DERs of EEND-GLA are less far apart than those of the cascaded methods because the inference is the same for offline and online given input features.

References

[1] X. Anguera, S. Bozonnet, N. Evans, C. Fredouille, G. Friedland, and O. Vinyals, "Speaker diarization: A review of recent research," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 20, no. 2, pp. 356–370, Feb. 2012.

[2] 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, no. 7, 2022, Art. no. 101317.

[3] P. Kenny, D. Reynolds, and F. Castaldo, "Diarization of telephone conversations using factor analysis," *IEEE J. Sel. Topics Signal Process.*, vol. 4, no. 6, pp. 1059–1070, Dec. 2010.

[4] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The fifth 'CHiME' speech separation and recognition challenge: Dataset, task and baselines," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2018, pp. 1561–1565.

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

[6] G. Finley et al., "An automated assistant for medical scribes," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2018, pp. 3212–3213.

[7] S. Araki, M. Fujimoto, K. Ishizuka, H. Sawada, and S. Makino, "A DOA based speaker diarization system for real meetings," in *Proc. IEEE Hands-free Speech Commun. Microphone Arrays*, 2008, pp. 29–32.

[8] X. Zhu, C. Barras, L. Lamel, and J.-L. Gauvain, "Multi-stage speaker diarization for conference and lecture meetings," in *Multimodal Technologies for Perception of Humans*. Berlin, Germany: Springer, 2007, pp. 533–542.

[9] X. Bost, G. Linares, and S. Gueye, "Audiovisual speaker diarization of TV series," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2015, pp. 4799–4803.

[10] 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.

[11] 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.

[12] H. Bredin and A. Laurent, "End-to-end speaker segmentation for overlap-aware resegmentation," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3111–3115.

[13] 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.

[14] 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.

[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] S. Horiguchi, Y. Fujita, S. Watanabe, Y. Xue, and P. García, "Encoder-decoder based attractors for end-to-end neural diarization," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 30, pp. 1493–1507, 2022.

[17] 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.

[18] 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.

[19] 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.

[20] N. Zeghidour, O. Teboul, and D. Grangier, "DIVE: End-to-end speech diarization via iterative speaker embeddings," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 702–709.

[21] K. Boakye, B. Trueba-Hornero, O. Vinyals, and G. Friedland, "Overlapped speech detection for improved speaker diarization in multiparty meetings," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2008, pp. 4353–4356.

[22] 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.

[23] 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.

[24] 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.

[25] Y. Xue, S. Horiguchi, Y. Fujita, S. Watanabe, P. Garcia, and K. Nagamatsu, "Online end-to-end neural diarization with speaker-tracing buffer," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2021, pp. 841–848.

[26] 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.

[27] 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.

[28] M. He, D. Raj, Z. Huang, J. Du, Z. Chen, and S. Watanabe, "Target-speaker voice activity detection with improved i-vector estimation for unknown number of speakers," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3555–3559.

[29] 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*.

[30] Y. Takashima, Y. Fujita, S. Watanabe, S. Horiguchi, P. García, 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.

[31] K. Kinoshita, M. Delcroix, and N. Tawara, "Integrating end-to-end neural and clustering-based diarization: Getting the best of both worlds," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 7198–7202.

[32] K. Kinoshita, M. Delcroix, and N. Tawara, "Advances in integration of end-to-end neural and clustering-based diarization for real conversational speech," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3565–3569.

[33] S. Horiguchi, S. Watanabe, P. García, 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.

[34] K. Kinoshita, M. Delcroix, and T. Iwata, "Tight integration of neural- and clustering-based diarization through deep unfolding of infinite gaussian mixture model," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2022, pp. 8382–8386.

[35] 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.

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

[37] 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.

[38] N. Dawalatabad, M. Ravanelli, F. Grondin, J. Thienpondt, B. Desplanques, and H. Na, "ECAPA-TDNN embeddings for speaker diarization," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3560–3564.

[39] P. Singh and S. Ganapathy, "Self-supervised metric learning with graph clustering for speaker diarization," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 90–97.

[40] L. Bullock, H. Bredin, and L. P. Garcia-Perera, "Overlap-aware diarization: Resegmentation using neural end-to-end overlapped speech detection," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 7114–7118.

[41] J.-w. Jung, H.-S. Heo, Y. Kwon, J. S. Chung, and B.-J. Lee, "Three-class overlapped speech detection using a convolutional recurrent neural network," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3086–3090.

[42] S. Ding, Q. Wang, S.-Y. Chang, L. Wan, and I. L. Moreno, "Personal VAD: Speaker-conditioned voice activity detection," in *Proc. Odyssey*, 2020, pp. 433–439.

[43] Q. Wang et al., "VoiceFilter-Lite: Streaming targeted voice separation for on-device speech recognition," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 2677–2681.

[44] W. Wang and M. Li, "Incorporating end-to-end framework into target-speaker voice activity detection," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2022, pp. 8362–8366.

[45] 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.

[46] C.-Y. Cheng, H.-S. Lee, Y. Tsao, and H.-M. Wang, "Multi-target filter and detector for unknown-number speaker diarization," 2022, *arXiv:2203.16007*.

[47] D. Wang, X. Xiao, N. Kanda, T. Yoshioka, and J. Wu, "Target speaker voice activity detection with transformers and its integration with end-to-end neural diarization," 2022, *arXiv:2208.13085*.

[48] T. J. Park, N. R. Koluguri, J. Balam, and B. Ginsburg, "Multi-scale speaker diarization with dynamic scale weighting," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2022, pp. 5080–5084.

[49] Y. Takashima, Y. Fujita, S. Horiguchi, S. Watanabe, P. Garcia, and K. Nagamatsu, "Semi-supervised training with pseudo-labeling for end-to-end neural diarization," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3096–3110.

[50] 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.

[51] E. Fini and A. Brutti, "Supervised online diarization with sample mean loss for multi-domain data," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 7134–7138.

[52] J. M. Coria, H. Bredin, S. Ghannay, and R. Sophie, "Overlap-aware low-latency online speaker diarization based on end-to-end local segmentation," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 1139–1146.

[53] W. Xia et al., "Turn-to-diarize: Online speaker diarization constrained by transformer transducer speaker turn detection," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2022, pp. 8077–8081.

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

[55] K. Kinoshita, M. Delcroix, S. Araki, and T. Nakatani, "Tackling real noisy reverberant meetings with all-neural source separation, counting, and diarization system," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 381–385.

[56] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov, "Transformer-XL: Attentive language models beyond a fixed-length context," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 2978–2988.

[57] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 2579–2605, 2008.

[58] A. Fathi et al., "Semantic instance segmentation via deep metric learning," 2017, *arXiv:1703.10277*.

[59] S. Kong and C. C. Fowlkes, "Recurrent pixel embedding for instance grouping," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9018–9028.

[60] K. Wagstaff et al., "Constrained k-means clustering with background knowledge," in *Proc. Int. Conf. Mach. Learn.*, 2001, pp. 577–584.

[61] Y. Yang, T. Rutayisire, C. Lin, T. Li, and F. Teng, "An improved cop-kmeans clustering for solving constraint violation based on MapReduce framework," *Fundamenta Informaticae*, vol. 29126, no. 4, pp. 301318301–318, 2013.

[62] "2000 NIST speaker recognition evaluation," [Online]. Available: https://catalog.ldc.upenn.edu/LDC2001S97

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

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

[65] D. Snyder, G. Chen, and D. Povey, "MUSAN: A music, speech, and noise corpus," 2015, *arXiv:1510.08484*.

[66] 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. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 5220–5224.

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

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

[69] 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.

[70] Y. Zhang et al., "Low-latency online speaker diarization with graph-based label generation," in *Proc. Odyssey*, 2022, pp. 162–169.

[71] Y. Yue, J. Du, M. He, Y. Yang, and R. Wang, "Online speaker diarization with core samples selection," in *Proc. Conf. Int. Speech Commun. Assoc.*, 2022, pp. 1466–1470.

[72] A. Savitzky and M. J. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Anal. Chem.*, vol. 36, no. 8, pp. 1627–1639, 1964.

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

Shota Horiguchi (Member, IEEE) received the B.E. and M.E. degrees from the University of Tokyo, Tokyo, Japan, in 2015 and 2017, respectively. Since 2022, he has been currently working toward the Doctoral degree with the University of Tsukuba, Tsukuba, Japan. He is currently a Senior Researcher with Hitachi, Ltd., Tokyo. 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. He is also a Member of the Acoustical Society of Japan.

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 the move to 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.

Paola García (Member, IEEE) received the Ph.D. degree from the University of Zaragoza, Zaragoza, Spain, in 2014. She joined Johns Hopkins University, Baltimore, MD, USA, 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, Canada. She was also a Researcher with Tec de Monterrey, Campus Monterrey, Mexico for ten years. She was a Marie Curie Researcher of 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 in 2009 and Carnegie Mellon 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 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.

Yuki Takashima received the B.E., M.E., and Ph.D. degrees in computer science from Kobe University, Kobe, Japan, in 2015, 2017, and 2020 respectively. From 2017 to 2020, he was the recipient of the Japan Society for the Promotion of Science Research Fellowship for Young Scientists. He is currently a Researcher with Hitachi, Ltd., Tokyo, Japan. His research focuses on speech and statistical signal processing. He is a Member of IEICE and ASJ.

Yohei Kawaguchi (Senior Member, IEEE) received the B.E. and M.E. degrees from Kyoto University, Kyoto, Japan, in 2005 and 2007, respectively, and the Ph.D. (Dr. Eng.) from the University of Tsukuba, Tsukuba, Japan, in 2017. In 2007, he joined Research and Development Group, Hitachi Ltd., Japan, where he is currently a unit Manager and a Chief Researcher. He is an author or co-author of more than 60 papers in peer-reviewed journals and conferences. His research interests include acoustic signal processing, machine learning, automatic speech recognition, underwater sonar, and chemical signal processing. He has also pioneered the technical field of anomalous sound detection (ASD) by coordinating the ASD task in the DCASE Challenge for four years from 2020. He was/is a Member of the IEEE Signal Processing Society Audio and Acoustic Signal Processing Technical Committee, APSIPA Signal and Information Processing Theory and Methods Technical Committee, IEICE Signal Processing Technical Committee, and the Editorial Board of the Acoustical Society of Japan.