Do End-to-End Neural Diarization Attractors Need to Encode Speaker Characteristic Information?

Lin Zhang[1], Themos Stafylakis[2,3], Federico Landini[4], Mireia Diez[4], Anna Silnova[4], Lukáš Burget[4]

[1]National Institute of Informatics, Tokyo, Japan [2]Omilia - Conversational Intelligence, Athens, Greece
[3]Athens University of Economics and Business, Athens, Greece
[4]Brno University of Technology, Faculty of Information Technology, Speech@FIT, Czechia

Abstract

In this paper, we apply the variational information bottleneck approach to end-to-end neural diarization with encoder-decoder attractors (EEND-EDA). This allows us to investigate what information is essential for the model. EEND-EDA utilizes attractors, vector representations of speakers in a conversation. Our analysis shows that, attractors do not necessarily have to contain speaker characteristic information. On the other hand, giving the attractors more freedom to allow them to encode some extra (possibly speaker-specific) information leads to small but consistent diarization performance improvements. Despite architectural differences in EEND systems, the notion of attractors and frame embeddings is common to most of them and not specific to EEND-EDA. We believe that the main conclusions of this work can apply to other variants of EEND. Thus, we hope this paper will be a valuable contribution to guide the community to make more informed decisions when designing new systems.

1. Introduction

End-to-end speaker diarization has gained considerable popularity in the last few years with several works exploring different frameworks [1]. Among them, end-to-end neural diarization (EEND) [2] is one of the most studied ones. It formulates diarization as a per-speaker per-frame binary classification problem where the predictions are represented by a two-dimensional matrix across time and speaker. Unlike modular diarization systems, this formulation effectively covers all aspects of the task with a single system including voice activity detection and handling of overlapped speech.

Several extensions have been proposed to this framework, among which EEND with encoder-decoder-based attractors (EEND-EDA) [3] was the first successful variant capable of dealing with a variable number of speakers. In EEND-EDA, frame embeddings are produced for the whole recording and these are encoded into a single representation from which attractors are decoded. Each attractor represents one speaker and they are compared by means of dot-product with the frame embeddings to determine which speakers are active at which frames.

EEND-EDA has become the main variant capable of handling multiple speakers and many modifications or alternatives have been proposed. In the original version, long short-term memory (LSTM) layers are used for the encoder and decoder. The decoder's initial hidden state is initialized with the final hidden state of the encoder and the decoder is fed with 0's to decode attractors one by one. Different ideas have been proposed to improve the attractor-decoding scheme. Some works [4, 5] have improved the model performance by feeding the LSTM decoder with global representations drawn from the frames rather than uninformative 0's. Other works [6, 7, 8, 9, 10, 11, 12] have proposed replacing the encoder-decoder module by an attention-based scheme using different approaches. All these methods still rely on the notion of "attractors" which are compared with frame embeddings through dot-product. Nevertheless, little analysis has been presented on the attractors and no study until now has focused on understanding what information is needed to be encoded in them. While in principle attractors are assumed to encode speaker information necessary to identify which frames were "spoken" by a certain speaker, there is no evidence of this or up to which extent they need to encode speaker characteristic information.

In information theory, the concept of an information bottleneck (IB) [13] was introduced to enforce the encoding to act as a minimal sufficient statistic of the input for predicting the target. To optimize the IB objective for deep neural networks, the deep variational information bottleneck (VIB) [14] was proposed. VIB utilizes variational inference to construct a lower bound of the IB objective, which enables optimization through stochastic gradient descent.

Given its capability to capture essential information within neural networks, we applied VIB into EEND-EDA[1] to better understand the end-to-end diarization mechanism. Specifically, we replace the point estimates of the frame embeddings and attractors of the original EEND-EDA with normal distributions with parameters estimated by VIB. Our results indicate that it is possible to obtain similar performance to the original EEND-EDA even if the attractors are regularized towards a standard normal distribution, making them less discriminative. This suggests that attractors might not need to encode specific speaker identities but rather enough information to distinguish them in a given conversation. At the same time, slight but consistent improvements can be obtained if the attractors have more freedom, suggesting that some specific speaker information is relevant to the model.

These findings could lead to more effective strategies for training speaker diarization systems and more parameter-efficient models. The analysis shows that competitive end-to-end diarization models can be trained even if the frame embeddings and attractors do not contain much speaker-specific information. While it was not the main focus of this work, the strategies we utilized can also be used if privacy concerns are relevant.

---

[1]https://github.com/BUTSpeechFIT/EENDEDA_VIB
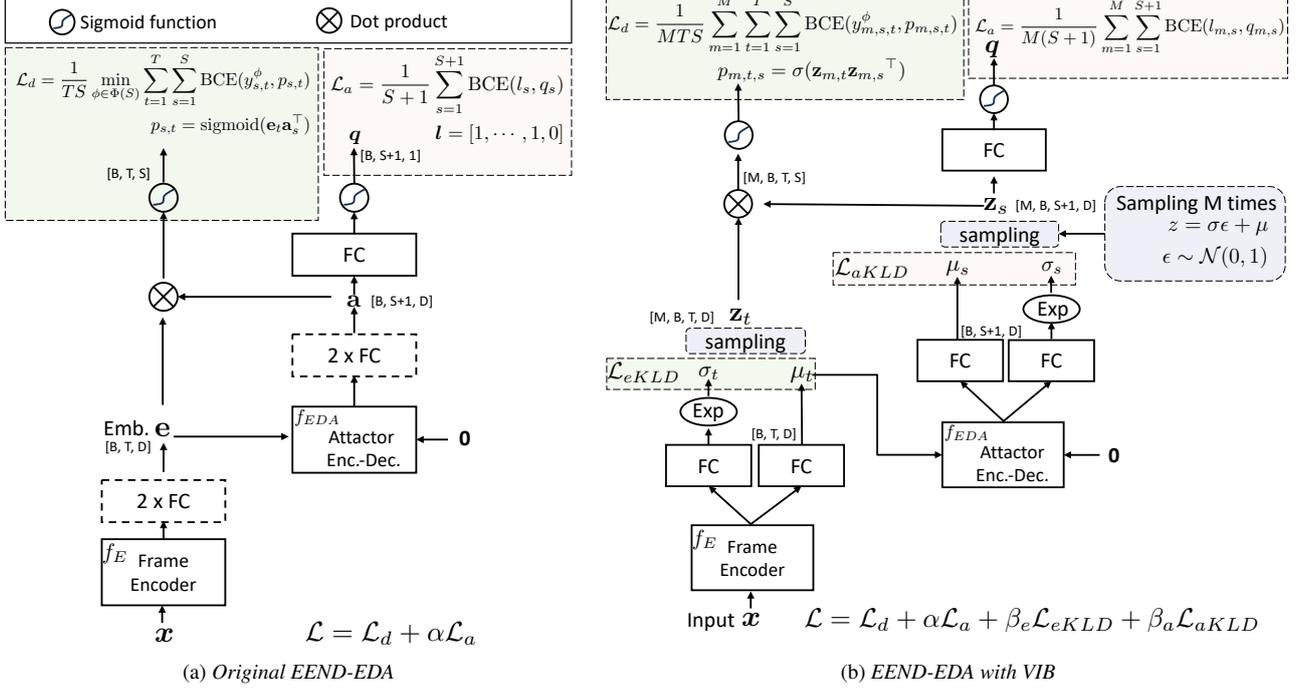
Figure 1: *Comparison between the original EEND-EDA model and EEND-EDA with VIB. Numbers within [] present dimensions. M is the sampling number, B is batch size, T is number of frames, D is feature dimension, and S is the maximum speaker number within one mini-batch. In (a), four additional FC layers in the dashed box are introduced to make the original EEND-EDA comparable with (b).*

2. EEND-EDA with VIB

2.1. EEND with Encoder-Decoder-Based Attractors

As shown in Fig. 1a, EEND-EDA [3] can be viewed as a combination of two branches: a frame embedding branch and an attractor branch. The frame embedding branch, uses a self-attention encoder $f_E(\cdot)$ to convert the D-dimensional input acoustic feature for T frames $\boldsymbol{x}_{1:T}$ into frame embeddings:

$$\mathbf{e}_{1:T} = f_E(\mathbf{x}_{1:T}). \tag{1}$$

The attractor branch uses an encoder-decoder module $f_{EDA}(\cdot)$, usually consisting of two unidirectional LSTMs, to decode attractors (each one representing one speaker):

$$\mathbf{a}_{1:(S+1)} = f_{EDA}(\mathbf{e}_{1:T}). \tag{2}$$

Then, to determine the number of speakers present in the input recording, it estimates the existence probability of each attractor s as:

$$q_s = \text{sigmoid}(\text{FC}(\mathbf{a}_s)), s \in [1:S+1], \tag{3}$$

where FC denotes a fully connected linear layer.

During training, the attractor loss is defined:

$$\mathcal{L}_a = \frac{1}{S+1} \sum_{s=1}^{S+1} \text{BCE}(l_s, q_s), \tag{4}$$

where S is the number of valid attractors (corresponding to the number of speakers), $\boldsymbol{l}_{1:(S+1)} = [1, \cdots, 1, 0]$ is the attractor validity label and $\text{BCE}(\cdot, \cdot)$ is the binary cross entropy function. During inference, attractors are sequentially extracted until q_s is below a predefined threshold τ, usually set as 0.5.

Finally, the dot product between the per-frame embeddings and the attractors is used to produce the diarization results:

$$p_{s,t} = \text{sigmoid}(\mathbf{e}_t \mathbf{a}_s^\top), \tag{5}$$

where $(\cdot)^\top$ denotes the matrix transpose and $\boldsymbol{p} = \{p_{s,t} | s \in [1,S], t \in [1,T]\}$ is a matrix of per-frame per-speaker activity probabilities. Given the ground-truth speech activities $y_{s,t}$, where $y_{s,t} = 1$ when the s-th speaker is active in the t-th frame and $y_{s,t} = 0$ otherwise, the diarization loss is calculated using permutation invariant training [15]:

$$\mathcal{L}_d = \frac{1}{TS} \min_{\phi \in \Phi(S)} \sum_{t=1}^{T} \sum_{s=1}^{S} \text{BCE}(y_{s,t}^\phi, p_{s,t}). \tag{6}$$

where $\min_{\phi \in \Phi(S)}$ is to find the permuted sequence ϕ that has the minimum loss among all possible permutations Φ of S speakers. The model is trained using a total loss, which is a combination of both defined losses:

$$\mathcal{L} = \mathcal{L}_d + \alpha \mathcal{L}_a. \tag{7}$$

2.2. Variational Information Bottleneck

Given a distribution $p(x, y) = p(y|x)p(x)$ of inputs x and targets y, the aim of the information bottleneck method proposed in [13] is to find a stochastic encoding $p(z|x)$ that is maximally informative about the target y, and, at the same time, preserves minimum information about the input x. This is achieved by finding the encoding that maximizes the difference between mutual informations $I(y, z) - \beta I(x, z)$, where β controls the trade-off between z being expressive about y and compressive about x. Although, it is intractable to find the optimal encoding for nontrivial distributions $p(x, y)$, Alemi *et al.* [14] proposed Deep Variational Information Bottleneck (VIB) – a variational approximation to this problem, where a neural network is trained to estimate the stochastic encoding $p(z|x)$.

2.2.1. VIB model and objective

As a simple VIB example, let us first consider the classification task described in [14], where the distribution $p(\mathbf{x}, y)$ is represented by the empirical distribution of the training input observations $\mathbf{x}_n$ and the corresponding categorical labels y_n. For simplicity, the stochastic encoding will be represented by a Gaussian distribution with diagonal covariance matrix $p(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}(\mathbf{x}), \text{diag}(\boldsymbol{\sigma}(\mathbf{x})^2))$, where the mean vector $\boldsymbol{\mu}(\mathbf{x})$ and vector of standard deviations $\boldsymbol{\sigma}(\mathbf{x})$ are functions of the input $\mathbf{x}$ and these functions are parametrized by a neural network called *encoder*. To train the *encoder* network, [14] proposes to optimize the following objective

$$\frac{1}{N}\sum_{n=1}^{N}\int p(\mathbf{z}|\mathbf{x}_n)\log q(y_n|\mathbf{z}) - \beta p(\mathbf{z}|\mathbf{x}_n)\log\frac{p(\mathbf{z}|\mathbf{x}_n)}{r(\mathbf{z})}d\mathbf{z} \quad (8)$$

$$= \frac{1}{N}\sum_{n=1}^{N}\left(\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}\left[-\log q(y_n|\boldsymbol{\mu}(\mathbf{x}_n) + \boldsymbol{\epsilon}\odot\boldsymbol{\sigma}(\mathbf{x}_n))\right]\right.$$

$$\left. +\beta\text{KL}\left[p(\mathbf{z}|\mathbf{x}_n)||r(\mathbf{z})\right]\right) \quad (9)$$

as a lower-bound to the original information bottleneck objective $I(y, \mathbf{z}) - \beta I(\mathbf{x}, \mathbf{z})$. The terms $q(y_n|\mathbf{z})$ and $r(\mathbf{z})$ are variational approximations of $p(y_n|\mathbf{z})$ and $p(\mathbf{z})$ whose exact evaluation (given the learned $p(\mathbf{z}|\mathbf{x})$ and the empirical $p(\mathbf{x}, y)$) is intractable. The approximate distribution $q(y_n|\mathbf{z})$ is represented by another neural network called *decoder*, which maps vector $\mathbf{z}$ to a (categorical) distribution of y. This network is trained jointly with the *encoder* network using the same objective (8). The approximate marginal distribution $r(\mathbf{z})$ can also be trained, but in [14] it is fixed as a standard normal distribution as the learned $p(\mathbf{z}|\mathbf{x})$ can accommodate to it.

2.2.2. VIB training

The first term in (8) is the expected cross-entropy evaluated for the decoder output where the expectation is with respect to the encoding distribution $p(\mathbf{z}|\mathbf{x})$. This term clearly favours $p(\mathbf{z}|\mathbf{x})$ that is expressive about y as required by the IB principle. This term also encourages the distribution $p(\mathbf{z}|\mathbf{x})$ to be more certain (having lower variance) peaking around $\mathbf{z}$ for which $\log q(y_n|\mathbf{z})$ is high. The second term is the Kullback-Leibler (KL) divergence between the encoding $p(\mathbf{z}|\mathbf{x})$ and the (approximate) marginal distribution $r(\mathbf{z}) \approx p(\mathbf{z})$ favouring more uncertain and thus less expressive encoding. Clearly, in the extreme case when the encoding $p(\mathbf{z}|\mathbf{x})$ equals $p(\mathbf{z})$ for any input $\mathbf{x}$, it does not contain any information about the input. As we will see in later experiments, β controls the trade-off between the two terms in the objective where low values of β lead to more certain encoding distributions $p(\mathbf{z}|\mathbf{x})$ that are more expressive about the output y and higher values of β lead to more uncertain distributions that are less expressive about the input $\mathbf{x}$.

In order to optimize this objective during the training, the first term in (9), is rewritten using the standard reparameterization trick [16], which allows to easily back-propagate through it. Since we use Gaussian distributions for both $p(\mathbf{z}|\mathbf{x})$ and $r(\mathbf{z})$, the KL divergence can be expressed analytically and it is therefore also easy to back-propagate through it.

The *encoder* and *decoder* networks can be trained with a variant of stochastic gradient descent where, for each training example $\mathbf{x}_n$, we forward propagate through the encoder network to obtain the parameters of the encoding distribution $\boldsymbol{\mu}(\mathbf{x}_n)$ and $\boldsymbol{\sigma}(\mathbf{x}_n)$. Next, we sample from the encoding distribution using the reparameterization trick

$$\hat{\mathbf{z}}_n = \boldsymbol{\mu}(\mathbf{x}_n) + \boldsymbol{\epsilon}\odot\boldsymbol{\sigma}(\mathbf{x}_n), \quad (10)$$

where $\boldsymbol{\epsilon}$ is a sample from the standard normal distribution, and the operator $\odot$ denotes element-wise product. We forward propagate this sample through the decoder network to evaluate $q(y_n|\hat{\mathbf{z}}_n)$. This way, we obtain all the quantities necessary for evaluating the objective (9) approximating the expectation in the first term using only a single sample. This is often sufficient for the training, but multiple samples can be used to better approximate the expectation. Finally, parameters are updated by back-propagating through this objective and the *encoder* and *decoder* networks.

2.2.3. VIB regularized classifier

Once the *encoder* and *decoder* networks are trained, we can consider them in a tandem as a single classification network, where $\mathbf{z}$ functions as a probabilistic representation at the output of an internal hidden layer. The VIB training makes the hidden representation $\mathbf{z}$ forget the details of the input $\mathbf{x}$ which are irrelevant to the output, which can be seen as a specific regularization. It is shown in [14] that VIB training performs competitively with other regularization techniques and produces systems more robust to adversarial attacks. The disadvantage of such VIB trained classifier is that the uncertainty of the hidden representation $\mathbf{z}$ should be taken into account during the inference. Therefore, to correctly evaluate the output, the expected probability of $p(y|\mathbf{x})$ should be estimated by averaging the decoder output $q(y|\hat{\mathbf{z}})$ obtained from many samples of $\hat{\mathbf{z}}$. However, for practical inference only the mean $\boldsymbol{\mu}(\mathbf{x})$ of the distribution $p(\mathbf{z}|\mathbf{x})$ may be used.

2.3. EEND-EDA regularized using VIB

In our experiments, we will use the VIB principle to regularize the EEND-EDA model, to limit the information encoded in the frame embeddings and attractors. This way, we want to find what is the minimum necessary information encoded in these representations that allows for good diarization performance. For this purpose, we extend and train the EEND-EDA using the VIB as shown in Fig. 1b. Each frame embedding and each attractor are treated as stochastic encodings. Each frame embedding $\boldsymbol{e}_t$ is still evaluated as in (1), but it is further transformed by two fully connected (FC) layers, to obtain the parameters of the stochastic encodings:

$$\boldsymbol{\mu}_t = \text{FC}^{\mu_t}(\mathbf{e}_t),$$
$$\boldsymbol{\sigma}_t = \exp(\text{FC}^{\sigma_t}(\mathbf{e}_t)), \quad (11)$$

$$p(\mathbf{z}_t|\mathbf{x}_{1:T}) = \mathcal{N}(\mathbf{z}_t|\boldsymbol{\mu}_t, \text{diag}(\boldsymbol{\sigma}_t^2)). \quad (12)$$

Similarly, we obtain the stochastic encodings for the attractors as:

$$\boldsymbol{\mu}_s = \text{FC}^{\mu_s}(\mathbf{a}_s),$$
$$\boldsymbol{\sigma}_s = \exp(\text{FC}^{\sigma_s}(\mathbf{a}_s)), \quad (13)$$

$$p(\mathbf{z}_s|\mathbf{x}_{1:T}) = \mathcal{N}(\mathbf{z}_s|\boldsymbol{\mu}_s, \text{diag}(\boldsymbol{\sigma}_s^2)). \quad (14)$$

To train the EEND-EDA with VIB, we proceed in a similar manner as described in Sec. 2.2: we forward propagate through the encoder network, which is represented by (14) and (12). We sample the encodings $\hat{\mathbf{z}}_s$ and $\hat{\mathbf{z}}_t$ (using the reparameterization trick, see equation (10)) for each frame and for each attractor. We pass these samples through the decoder represented by (3) and (5). In these formulas, we only substitute the deterministic attractor embeddings $\boldsymbol{a}_s$ and frame embeddings $\boldsymbol{e}_t$ with the corresponding sampled encodings $\hat{\mathbf{z}}_s$ and $\hat{\mathbf{z}}_t$, respectively.

The final loss to optimize is:

$$\mathcal{L} = \mathcal{L}_d + \beta_e \mathcal{L}_{eKLD} + \alpha \mathcal{L}_a + \beta_a \mathcal{L}_{aKLD}, \qquad (15)$$

where $\mathcal{L}_d$ and $\mathcal{L}_a$ are defined in (6) and (4), respectively. β_e and β_a control the amount of regularization imposed on the encodings by the KLD losses:

$$\mathcal{L}_{eKLD} = \frac{1}{T} \sum_{t=1}^{T} \mathrm{KL}(\mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\sigma}_t^2) \| \mathcal{N}(\mathbf{0}, \boldsymbol{I})), \qquad (16)$$

$$\mathcal{L}_{aKLD} = \frac{1}{S} \sum_{s=1}^{S+1} \mathrm{KL}(\mathcal{N}(\boldsymbol{\mu}_s, \boldsymbol{\sigma}_s^2) \| \mathcal{N}(\mathbf{0}, \boldsymbol{I})). \qquad (17)$$

In the text above, we consider that the losses $\mathcal{L}_a$ and $\mathcal{L}_d$ are evaluated using a single sample of each $\mathbf{z}_s$ and $\mathbf{z}_t$, which corresponds to approximating the expectation in (9) using only a single sample. This is the configuration used in most of our experiments, as it produced good results. However, the expectation could also be approximated by multiple samples. In such case, M samples would be generated for each $\mathbf{z}_s$ and $\mathbf{z}_t$, and the objectives $\mathcal{L}_a$ and $\mathcal{L}_d$ would be evaluated M times and averaged. We present results for these different training strategies in Sec. 4.2.

Similarly, multiple samples should be also used for the inference with the trained model. In such case, the speaker activity probabilities $p_{s,t}$ and the attractor existence probabilities q_s (see equations (5) and (3)) are evaluated and averaged for multiple samples. Alternatively, as a computationally cheaper approximation, (5) and (3) can be evaluated using the encoding means $\boldsymbol{\mu}_t$ and $\boldsymbol{\mu}_s$. This later strategy is used to report all the DER values in this paper, as it is more computationally efficient and we found it to produce similar results.

In our experiments, we also consider configurations where either the frame embeddings or the attractors in the EEND-EDA are stochastic while maintaining the other component as deterministic. This is achieved by setting the weight of the KLD loss to zero and directly using the corresponding means $\boldsymbol{\mu}_t$ or $\boldsymbol{\mu}_s$ as the deterministic embeddings. In addition, setting both β_e and β_a to zero and using only the mean vectors $\boldsymbol{\mu}_t$ and $\boldsymbol{\mu}_s$ recovers the original EEND-EDA.

3. Experimental Setup

3.1. Data

As usual with the end-to-end neural diarization framework, the models are mostly trained on synthetic data. Following [17, 18], we utilize simulated conversations (SC) to allow better performance than simulated mixtures [2]. Using statistical information derived from the development set of the conversational telephone speech domain from DIHARD-III [19], three different sets of 2480 h each were generated where recordings had 2, 3, and 4 speakers respectively. The first set, denoted "SC2" was used to train the model from scratch. The three sets were pooled to form "SC2-4", containing 7442 h of audio, which was used for a second step of training as described in the next section.

For the evaluation of the models, we use the Callhome [20] dataset consisting of telephone conversations. This dataset is commonly used to evaluate diarization systems and it is divided into two parts[2] denoted CH1 (with 8.7 h and 2-7 speakers) and CH2 (with 8.55 h and 2-6 speakers). Besides, for some of the comparisons we use the subset of CH1 with only recordings containing two speakers denoted CH1-2spk (with 3.2 h).

---

[2] https://github.com/BUTSpeechFIT/CALLHOME_sublists

3.2. Configurations

The configurations of EEND-EDA are the same as in [3, 18] and α is set as 1. Our analyses are based on a previously released PyTorch implementation[3]. The only difference with that EEND-EDA implementation is that, as mentioned in Section 2.3, four additional FC layers (each with 256 units) are introduced into the structure. Therefore, to ensure the proposed model is comparable to EEND-EDA, both versions of EEND-EDA, with and without the additional layers, are used as baselines. Specifically, when EEND-EDA has four additional FC layers, two FC layers are inserted after f_E and the other two are positioned after f_{EDA} as shown in dashed boxes of Fig. 1a.

Following standard practice when training EEND-EDA, there are three steps. (1) Initial training using SC with two speakers (using SC2) is run for 100 epochs. Then, we average the parameters of the last ten checkpoints every ten epochs, and evaluate performance on CH1-2spk to determine the optimal model. The ten checkpoints are averaged for the subsequent adaptation step. (2) Adaptation to a set with a variable number of speakers (using SC2-4) is run for 80 epochs. Then, we average the parameters of the last ten checkpoints every ten epochs, and evaluate performance on CH1 to determine the optimal model. The ten checkpoints are averaged for the subsequent fine-tuning step. (3) Fine-tuning to in-domain data (using CH1) for 100 epochs. We average the parameters of the last ten checkpoints every ten epochs, and evaluate performance on CH1 to determine the optimal model. Then, such a model is used to report results on CH2.

We use the diarization error rate (DER) as defined by NIST [21] for evaluating the performance, considering overlaps and 0.25 collar as is standard practice for Callhome. Confidence intervals (CI) were calculated using the Interspeech official toolkit [4] with the default configuration.

4. Results and Discussions

4.1. Impact of the weight of the KLD loss

In this section, we will analyze the effect that different degrees for regularization have in the diarization performance. We individually analyze the impact of using β_e and β_a to control the degree of regularization on frame embeddings and attractors, respectively. For analyzing the effect of regularizing only frame embeddings or only attractors, we use the configurations outlined at the end of Section 2.3. That is, to analyze only the effect of β_e, we set $\beta_a = 0$, and directly use the mean $\boldsymbol{\mu}_s$ estimated by the network, and vice versa. We also explore the combination of β_a and β_e. Fig. 2 shows how diarization performance varies with the weights used for the KLD losses. We observe that the DER of the model with VIB is virtually the same as the baseline [5] for a wide range of weights. Only relatively large values of β result in performance degradation. We should not expect that the VIB would (significantly) improve the performance as the VIB regularization only limits the information contained in the frame embeddings and attractors. In section 4.3 we will analyze how the DER correlates with the amount of information that is preserved in these representations.

---

[3] https://github.com/BUTSpeechFIT/EEND

[4] https://github.com/luferrer/ConfidenceIntervals

[5] As sharp readers will spot, it is *superfluous* to have consecutive two FC layers, still, we kept them to have consistency in the number of parameters.
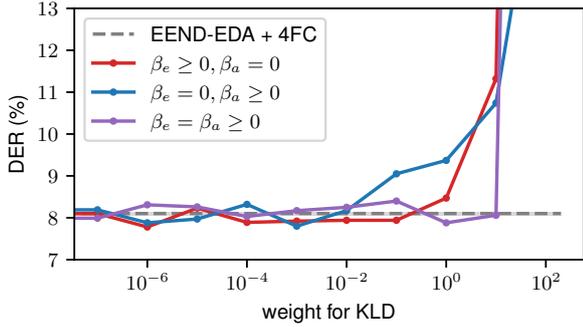
Figure 2: *DER (%) in CH1-2spk for different values of β_e and β_a. EEND-EDA + 4FC with five runs are shown as baselines within the gray shade, and the dashed line represents the mean of those five runs (overlapped due to the small variance).*

4.2. Number of samples and selection of the permutation ϕ

As commented in Sec. 2.3, for each training example the expectation in (9) can be approximated using only a single or multiple samples of each encoding z_t and z_s. The first three columns in Table 1 show the performance for 1, 12 and 100 samples, on CH1-2spk using $\beta_e = \beta_a = 10^0$ (as it was one of the best-performing configurations seen in Section 4.1). It can be seen that more samples help, however, when training with multiple samples, it is not clear how to choose the permutation ϕ for $\mathcal{L}_d$ in (6) since different samples can favor different permutations. In our case, we select the best permutation for each sample, use it to evaluate the corresponding loss, and average the loss of all samples. The problem with this solution is that it is computationally expensive, as the best permutation has to be found for each sample. At the same time, it is not clear if using potentially a different permutation for each sample can deteriorate the training process. Therefore, we also consider the variant where the best permutation is estimated using the speaker activities $p_{s,t}$ obtained from equation (5) evaluated with the mean encoding vectors $\boldsymbol{\mu}_t$ and $\boldsymbol{\mu}_s$. Such permutation should be compatible with any encoding samples. The last three columns in Table 1 show the result where this strategy is used to find permutation ϕ and different samples are used for training. Given that their results are similar to each other, we choose the relatively *cheaper* strategy—using μ to search for the best permutation and one sample to calculate losses—for the subsequent analysis and discussion.

Table 1: *Comparison of different strategies for selecting permutation ϕ and different number of training samples. DER and CI are shown on CH1-2spk.*

	Permutation based on					
	sampling z			μ		
# of samples	1	12	100	1	12	100
DER (%)	8.06	7.92	7.84	7.88	8.12	7.83
CI	±1.03	±0.97	±1.01	±1.02	±1.01	±1.00

4.3. Visualization of attractors and frame embeddings

To get insights into what key information is encoded in the frame and speaker representations of the EEND-EDA model, we visualize the posterior distribution of the attrac-

tors $p(\mathbf{z}_s|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_s, \mathrm{diag}(\boldsymbol{\sigma}_s^2))$, and the frame embeddings $p(\mathbf{z}_t|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_t, \mathrm{diag}(\boldsymbol{\sigma}_t^2))$. For the visualization, we project the attractor/embeddings into a two-dimensional space using principal component analysis (PCA) and then plot them as Gaussian ellipses, as shown in Fig. 3. Each ellipse shows the contour of the Gaussian distribution representing the corresponding stochastic enconding projected to the two dimensional space using PCA. The contour corresponds to the distance of two times the standard deviation from the mean.

To visualize the attractors, we first consider a fixed $\beta_e = 0$ and a wide range of β_a weights as shown in Fig. 3a, to get a good understanding of the effect of the regularization. We used the entire CH1-2spk to generate the plots. For each recording, three ellipses are shown in the figure, one with each category: the first speaker (spk1), the second speaker (spk2), and the invalid attractor. To visualize the frame embeddings, we equivalently consider $\beta_a = 0$ and a wide range of β_e, and randomly selected a few frames from five random audio samples from CH1-2spk (not to clutter the images). From left to right, Fig. 3 visualizes the model's behavior with varying weights from lower to higher (corresponding to lighter to stronger regularization).

As expected, Fig. 3 demonstrates the model's transition from deterministic to stochastic as the weights change. When the weight is zero (as in Fig. 3a.0 and Fig. 3b.0), the model is effectively the same as the original deterministic EEND-EDA. When the weight is larger than 0, yet close to zero, such as 10^{-6} in the Fig. 3a.1 and Fig. 3b.1, the attractor and frame embedding ellipses are close to point estimates, still similar to a deterministic model. With larger weights, more input information is forgotten, and we observe a shift towards a more regularized model characterized by the increased variance of the attractor and frame embedding distributions.

For the attractors, we can see that the valid speakers (blue and orange) are always well discerned from the invalid one (gray). When lower values of regularization are used, the attractor representations are spread across one of the axes (Fig. 3a.1-3a.2). Still, the representations cluster by the corresponding (first or second) decoded attractors and nothing suggest that they would be clustered according to the specific voice of corresponding speakers (which should be different for each input conversation). As β_a increases, we observe that the distributions for all the first attractors (blue ones) significantly overlap, as seen in Fig 3a.4. And the same thing happens for the second attractors (orange ones). This has an impact on performance, degrading from 7.93% DER in 3a.0 to 10.74% DER in 3a.4. Still, the system performs reasonably well, while it is obvious that the attractors are only used for *counting* the speakers, and no longer convey any speaker information. In the last subfigure, Fig. 3a.5, we observe that the representation of both valid speakers is starting to collapse into a single mode, now significantly harming system performance.

For the frame embeddings, we observe a similar trend for frame embeddings of a single speaker. For clearer visualization, we zoomed into Fig. 3b.4 and decomposed it into two subfigures presented in Fig. 4. In Fig. 4.(i), shows the distributions of frame embeddings where only one of the speakers is active. We can observe that speakers within the same audio (in the same color) are generally distinguishable. For example, green dashed lines (corresponding to the first attractor) on the left are far from the green dotted lines (corresponding to the second attractor) on the right. However, distributions of speakers across audios are overlapped and generally clustered by the corresponding attractors. That indicates that frame embeddings only require enough

(a) *Visualizing attractor on $\beta_e = 0$*

(b) *Visualizing frame embedding on $\beta_a = 0$*
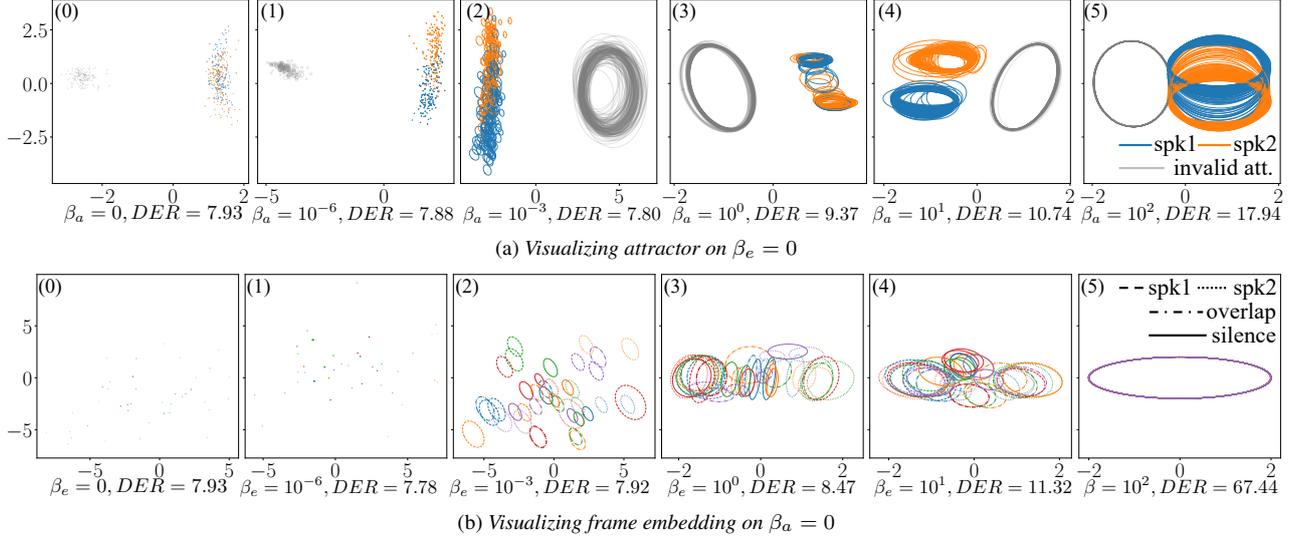
Figure 3: *Visualization of attractors and frame embeddings as Gaussian distributions, after projecting them to two dimensions using PCA (DERs are in %). In subfigure (b), colors represent different audio files; line styles denote individual speakers, overlap, and silence. Note that in subfigures (0) the attractors and frame embeddings are deterministic and are therefore represented by dots.*
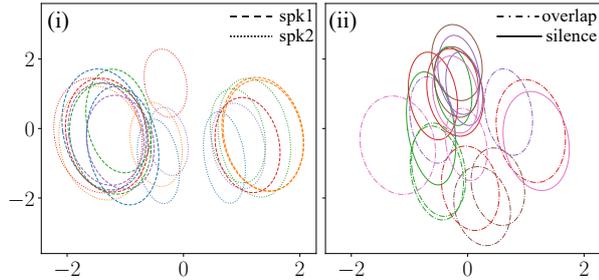
Figure 4: *Decomposed visualization of Fig. 3b.4 ($\beta_a = 0, \beta_e = 10^1$). (i) two single speakers, and (ii) overlap and silence.*
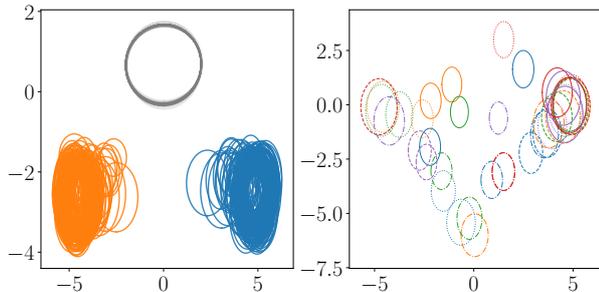
Figure 5: *Visualization of attractors (left) and frame embeddings (right) when $\beta_a = \beta_e = 10^1$.*

information to discriminate between speakers of the same audio (intra-audio), but do not need to encode individual identities across different audio (inter-audio). Besides, as shown in Fig. 4.(ii), silence across varying audios are clustered, and they are distinguishable from speech activity to some extent. This indicates that the frame-embedding requires encoding the information for speech activity. This aligns with the initial fundamental concept of the EEND model, which combines voice activity detection with diarization. Finally, we can also observe that, as expected, overlap frame representations are spread across the space between the single-speaker representations.

Equations (16) and (17) assume that the variational approximation to the marginal encoding distribution (see $r(\mathbf{z})$ in equation (9)) is fixed to be standard normal. We use this configuration in all our experiments, although, it generates a problem when both frame embeddings and attractors ($\mathbf{z}_t$ and $\mathbf{z}_s$) are simultaneously regularized using the KLD loss. When only one type of encodings (either $\mathbf{z}_t$ or $\mathbf{z}_s$) is regularized, as in Fig. 3 the encoding distributions span range around $[-2, 2]$ for higher β values. This range aligns with the standard normal Gaussian distribution that the KLD regularizes towards. However, as can be seen in Fig. 5, the encoding distributions span 3 times larger range, when regularizing both $\mathbf{z}_t$ and $\mathbf{z}_s$ simultaneously even with high regularization weights $\beta_a = \beta_e = 10^1$. Such a high dynamic range is enforced by the cross-entropy loss demanding high values of $p_{s,t}$, which can be achieved only with encodings $\mathbf{z}_t$ and $\mathbf{z}_s$ that are not close to the origin (see equation (5)) and therefore violating the assumed standard normal marginal distribution. This problem could be easily resolved by modeling the marginal distribution $r(\mathbf{z})$ by a Gaussian distribution with trainable parameters, which is left for future research.

4.4. Analysis across training, adaptation, and fine-tuning

In this section, we compare the performance of the proposed EEND-EDA with VIB in a real scenario with more speakers. For these experiments, we considered the three optimal configurations of β weights observed in Section 4.1, that is: (I) $\beta_e = 0, \beta_a = 10^{-3}$, (II) $\beta_e = 10^{-6}, \beta_a = 0$, and (III) $\beta_e = \beta_a = 10^0$. Results on CH2 are shown in Table 2. We can see that VIB maintains diarization performance closely aligned with the baseline, even for the configuration of weights (III), for which the strong regularization filters speaker information, as was shown in Fig. 5. Still, we noticed that fine-tuning the EEND-EDA model with VIB on the real data results in a smaller improvement in the performance, as compared to the fine-tuning in the original EEND-EDA.

To get some further insight into these results, in Fig. 6 we visualize the attractors of the EEND-EDA with VIB, when

(a) *Trained on SC2 (20.73)*   **(b)** *Adapted on SC2-4 (16.17)*   **(c)** *Finetuned on CH1 (15.88)*
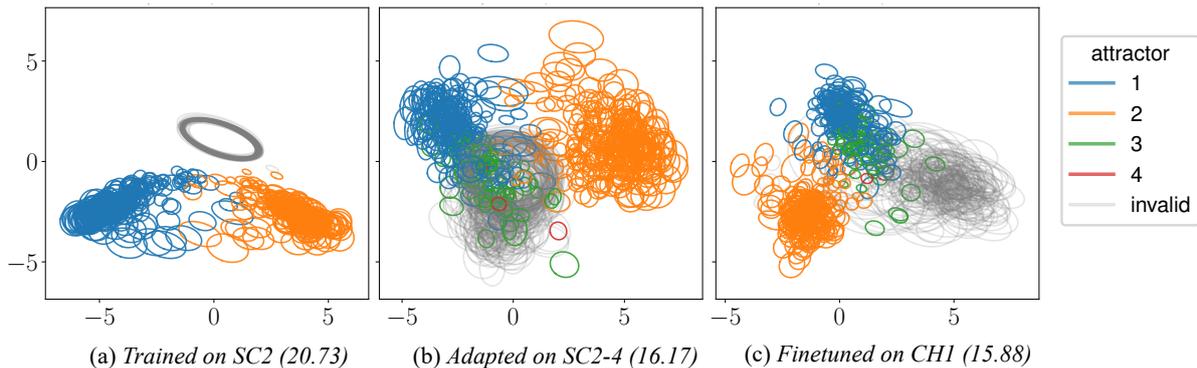
Figure 6: *Visualization of Gaussian attractors on CH2 after training, adaptation, and fine-tuning. The colors denote attractors for valid or the first invalid (gray) attractor as predicted by the model. DERs (%) are shown in brackets.*

$\beta_a = \beta_e = 10^0$, after training stage, after adaptation to SC2-4, and after fine-tuning to CH1. In the first step, the model is only trained with recordings of two speakers so it can only output two valid attractors. In this case, shown in Fig. 6.(a), the 1st, 2nd, and the invalid attractors, are clearly separated. After adapting the model to expect a variable number of speakers, in Fig. 6.(b), we can see that the 3rd and 4th valid attractors are sometimes decoded by the model. However, these attractors partially overlap with the invalid attractor (gray), and the *attractor clusters* become less compact. Given the training (and evaluation) data, we can assume that the first two attractors will almost always be valid. This is reflected in the plot: The 1st attractor (blue) and the 2nd (orange) are the most distinct ones. However, green and red ellipses overlap considerably with the gray (invalid) ones, revealing the higher uncertainty of the model about the 3rd and 4th attractors. This is intuitive since sometimes the 3rd (or 4th) attractor has to be considered invalid, thus, modeling them with closer representations to the invalid one might be beneficial. Observing plot (c), we can assume that fine-tuning could help to enhance the model's discriminative capability against invalid speakers. The resulting plot shows that fine-tuning makes the invalid speaker more clearly separated from the newly discovered third and fourth valid speakers. Nevertheless, it should be pointed out that in the plots we only use the first two dimensions after applying a PCA transformation. It is possible that, in other dimensions, the green and red ellipses do not overlap so much with the gray ones.

It is worth noting that the regions for the 1st, 2nd, 3rd and 4th attractors are fixed in the space, regardless of the input recording. This reinforces the conclusion from Section 4.3: the attractors serve as anchors for *counting* speakers and do not seem to convey (much) speaker information. Finally, the representations after the fine-tuning step, in Fig. 6.(c), are only slightly different and even if some recordings contain more than four speakers, the small development set is not enough for the model to learn that more speakers should be detected.

5. Conclusion

This study introduced VIB into EEND-EDA to understand what information needs to be stored in the frame embeddings and attractors, so that the model is effective in the diarization task. Our analyses show that attractors do not require substantial speaker characteristic information to obtain reasonable performance. Similar conclusions can be drawn from the frame embeddings: encoding unique individual identities across different audio samples (inter-audio) is not imperative; rather, the infor-

Table 2: *Comparison of DER (%) and CI on CH2.*

Model	w/o FT	w/ FT
Baselines		
EEND-EDA[3]	-	15.29
EEND-EDA (ours)	16.25 ($\pm$2.30)	15.72 ($\pm$2.34)
EEND-EDA + 4FC	16.74 ($\pm$2.36)	15.50 ($\pm$2.21)
EEND-EDA with VIB		
$\beta_a = 0$ $\beta_e = 10^{-6}$	16.06 ($\pm$2.34)	15.81 ($\pm$2.33)
$\beta_a = 10^{-3}$ $\beta_e = 0$	16.58 ($\pm$2.35)	16.04 ($\pm$2.17)
$\beta_a = 10^0$ $\beta_e = 10^0$	16.17 ($\pm$2.40)	15.88 ($\pm$2.50)

mation to discriminate speakers within the same audio (intra-audio) is enough for diarization.

Even though intuitively speaker characteristic information should not be necessary to perform diarization (models only need to "tell speakers apart" within the audio), this is, to the best of our knowledge, the first analysis in the context of end-to-end diarization models that provides evidence in this direction. We showed that models can perform reasonably well even when introducing strong regularization terms (based on VIB) for internal representations. Note, that this same strategy could be applied to produce more privacy-aware models.

6. Acknowledgements

7. References

[1] Tae Jin Park, Naoyuki Kanda, Dimitrios Dimitriadis, Kyu J Han, Shinji Watanabe, and Shrikanth Narayanan, "A review of speaker diarization: Recent advances with deep learning," *Computer Speech & Language*, vol. 72, pp. 101317, 2022.

[2] Yusuke Fujita, Naoyuki Kanda, Shota Horiguchi, Kenji Nagamatsu, and Shinji Watanabe, "End-to-End Neural Speaker Diarization with Permutation-Free Objectives," in *Proc. Interspeech 2019*, 2019, pp. 4300–4304.

[3] Shota Horiguchi, Yusuke Fujita, Shinji Watanabe, Yawen Xue, and Paola Garcia, "Encoder-Decoder Based Attractors for End-to-End Neural Diarization," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 1493–1507, 2022.

[4] Zexu Pan, Gordon Wichern, François G. Germain, Aswin Subramanian, and Jonathan Le Roux, "Towards End-to-end Speaker Diarization in the Wild," Nov. 2022, arXiv:2211.01299 [cs, eess].

[5] Samuel J. Broughton and Lahiru Samarakoon, "Improving End-to-End Neural Diarization Using Conversational Summary Representations," in *Proc. Interspeech*, 2023, pp. 3157–3161.

[6] Magdalena Rybicka, Jesus Villalba, Najim Dehak, and Konrad Kowalczyk, "End-to-End Neural Speaker Diarization with an Iterative Refinement of Non-Autoregressive Attention-based Attractors," in *Proc. Interspeech 2022*, 2022, pp. 5090–5094.

[7] Yusuke Fujita, Tatsuya Komatsu, Robin Scheibler, Yusuke Kida, and Tetsuji Ogawa, "Neural diarization with non-autoregressive intermediate attractors," in *Proc. ICASSP*, 2023, pp. 1–5.

[8] Fengyuan Hao, Xiaodong Li, and Chengshi Zheng, "End-to-end neural speaker diarization with an iterative adaptive attractor estimation," *Neural Networks*, vol. 166, pp. 566–578, 2023.

[9] Zhengyang Chen, Bing Han, Shuai Wang, and Yanmin Qian, "Attention-based Encoder-Decoder Network for End-to-End Neural Speaker Diarization with Target Speaker Attractor," in *Proc. Interspeech*, 2023, pp. 3552–3556.

[10] Zhengyang Chen, Bing Han, Shuai Wang, and Yanmin Qian, "Attention-based encoder-decoder end-to-end neural diarization with embedding enhancer," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 32, pp. 1636–1649, 2024.

[11] Federico Landini, Mireia Diez, Themos Stafylakis, and Lukáš Burget, "DiaPer: End-to-End Neural Diarization with Perceiver-Based Attractors," Dec. 2023, arXiv:2312.04324 [cs, eess].

[12] Marc Härkönen, Samuel J Broughton, and Lahiru Samarakoon, "EEND-M2F: Masked-attention mask transformers for speaker diarization," *arXiv preprint arXiv:2401.12600*, 2024.

[13] Naftali Tishby, Fernando C Pereira, and William Bialek, "The information bottleneck method," in *Proc. Allerton*, 2000, pp. 368–377.

[14] Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy, "Deep variational information bottleneck," in *Proc. ICLR*, 2017.

[15] Dong Yu, Morten Kolbæk, Zheng-Hua Tan, and Jesper Jensen, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *Proc. ICASSP*, 2017, pp. 241–245.

[16] Diederik P. Kingma and Max Welling, "Auto-encoding variational bayes," in *Proc.ICLR*, Yoshua Bengio and Yann LeCun, Eds., 2014.

[17] Federico Landini, Alicia Lozano-Diez, Mireia Diez, and Lukáš Burget, "From Simulated Mixtures to Simulated Conversations as Training Data for End-to-End Neural Diarization," in *Proc. Interspeech*, 2022, pp. 5095–5099.

[18] Federico Landini, Mireia Diez, Alicia Lozano-Diez, and Lukáš Burget, "Multi-speaker and wide-band simulated conversations as training data for end-to-end neural diarization," in *Proc. ICASSP*. IEEE, 2023, pp. 1–5.

[19] Neville Ryant, Prachi Singh, Venkat Krishnamohan, Rajat Varma, Kenneth Church, Christopher Cieri, Jun Du, Sriram Ganapathy, and Mark Liberman, "The Third DIHARD Diarization Challenge," in *Proc. Interspeech*, 2021, pp. 3570–3574.

[20] "Nist sre 2000 evaluation plan," `https://catalog.ldc.upenn.edu/LDC2001S97`, 2000.

[21] "NIST Rich Transcription Evaluations," `https://www.nist.gov/itl/iad/mig/rich-transcription-evaluation`, version: md-eval-v22.pl.