Segment-less Continuous Speech Separation of Meetings: Training and Evaluation Criteria

Thilo von Neumann*, Keisuke Kinoshita†, Christoph Boeddeker*, Marc Delcroix†, and Reinhold Haeb-Umbach*
*Paderborn University, Germany †NTT Corporation, Japan

***Abstract*—Continuous Speech Separation (CSS) has been proposed to address speech overlaps during the analysis of realistic meeting-like conversations by eliminating any overlaps before further processing. CSS separates a recording of arbitrarily many speakers into a small number of overlap-free output channels, where each output channel may contain speech of multiple speakers. This is often done by applying a conventional separation model trained with Utterance-level Permutation Invariant Training (uPIT), which exclusively maps a speaker to an output channel, in sliding window approach called stitching. Recently, we introduced an alternative training scheme called Graph-PIT that teaches the separation network to directly produce output streams in the required format without stitching. It can handle an arbitrary number of speakers as long as never more of them overlap at the same time than the separator has output channels. In this contribution, we further investigate the Graph-PIT training scheme. We show in extended experiments that models trained with Graph-PIT also work in challenging reverberant conditions. Models trained in this way are able to perform segment-less CSS, i.e., without stitching, and achieve comparable and often better separation quality than the conventional CSS with uPIT and stitching. We simplify the training schedule for Graph-PIT with the recently proposed Source Aggregated Signal-to-Distortion Ratio (SA-SDR) loss. It eliminates unfavorable properties of the previously used A-SDR loss and thus enables training with Graph-PIT from scratch. Graph-PIT training relaxes the constraints w.r.t. the allowed numbers of speakers and speaking patterns which allows using a larger variety of training data. Furthermore, we introduce novel signal-level evaluation metrics for meeting scenarios, namely the source-aggregated scale- and convolution-invariant Signal-to-Distortion Ratio (SA-SI-SDR and SA-CI-SDR), which are generalizations of the commonly used SDR-based metrics for the CSS case.**

***Index Terms*—Continuous Speech Separation, Source Separation, Graph-PIT, Dynamic Programming, Permutation Invariant Training**

I. Introdcution

The development of a meeting transcription system, enriched with meta information about who speaks when, is currently an active field of research. The development and evaluation of such systems are equally important problems, and can be performed with similar algorithms. With meetings, we denote a conversation among a few, typically unknown number of people with intermittent speech activity. Meeting-style conversations are held everyday and speech overlaps are inevitable in such situations; an overlap of 6 % to 17 % was reported in formal meetings [1]–[4] while in other daily natural conversation it can exceed 20 % [4]–[7]. Many widely-used state-of-the-art diarization and Automatic Speech Recognition (ASR) systems, however, can only handle a single speaker speaking at any point in time, and even short overlaps have a significant impact on their performance [8]–[11]. Therefore, a "separation first" pipeline has been established. A speech separation system first creates overlap-free speech signals from the meeting recording, on which conventional diarization and ASR can be performed without modification [3], [8], [10].

In recent years, Neural Network (NN)-based separation has flourished and exceeded the separation performance of conventional techniques by a large margin. Techniques like Deep Clustering [12], [13] and Utterance-level Permutation Invariant Training (uPIT) [14]–[18] have achieved almost perfect separation of short anechoic single-channel recordings with small numbers of speakers, with Signal-to-Distortion Ratios (SDRs) exceeding 20 dB [18], [19]. These techniques assign a speaker exclusively to one output channel, so each output channel contains speech of a single speaker only. We will call this *speaker-exclusive* assignment. This constraint limits their usefulness in realistic meeting-style conversations, where the number of speakers is typically unknown a priori.

Continuous Speech Separation (CSS) has been introduced as one way to handle more realistic speaking patterns [8]. It was observed that often only a small number of the participants of a meeting overlap at any given time. Such a meeting can thus be separated into fewer overlap-free output channels, where a speaker is no longer tied to an output channel exclusively. Instead, multiple speakers can share the same output channel, and the number of output channels is limited by the number of simultaneously speaking speakers at any time point in a meeting. We will call this *speaker-shared* assignment.[1]

To achieve such a separation with the conventional speaker-exclusive techniques, a sliding-window approach has been proposed [3], [8]; the recordings are cut into short overlapping segments and passed to a separator, which may output the separated signals in an arbitrary order. The separated segments are aligned and concatenated to form continuous output signals (see Section III-B), a process that has been termed "stitching". Here, the assumption is made that the number of speakers in one segment is not larger than the number of output channels of the separator, to satisfy the requirements of the conventional speaker-exclusive assignment. Since the number of output channels should be small (typically two) [12], [14], [20], the constraint on the number of speakers in one segment forces small segment sizes. Short segment sizes increase the probability for short speech fragments in a segment, which provide insufficient context for the separator to work properly.

[1]It was called "speaker-independent" CSS when introduced in [8], but we avoid this naming due to ambiguity with separation of unknown speakers.

We proposed an alternative training scheme for the CSS approach, named "Graph-PIT" [21], that allows a separation network to directly estimate a solution for the speaker-shared CSS problem. The reference signals used for training are constructed by finding the assignment of utterances to output channels that produces the lowest loss and is overlap-free (see Fig. 1 for an example of such an assignment). We interpret the process of finding this assignment as a graph coloring problem, and present efficient algorithms for finding the optimal assignment for training (see Section IV). The constraint of CSS with uPIT, that the number of speakers active in a segment must not exceed the number of output channels, is relaxed to the much less restrictive requirement that the number of simultaneously active speakers must not exceed the number of output channels. Graph-PIT training can consequently be used to enlarge the segments used for stitching, and to even eliminate the stitching process completely, enabling segmentless CSS.

As mentioned earlier, when dealing with meeting data, evaluation of the separation performance is another issue. Many conventional source separation performance evaluation metrics cannot directly be applied. To solve this problem, we propose an extension of the commonly used Scale-Invariant SDR (SI-SDR) [15], [22] and Convolution Invariant SDR (CI-SDR) [23], [24] to meeting scenarios (see Section V-B), that is based on the Source-Aggregated Signal-to-Distortion Ratio (SA-SDR) [25] and that is well-defined both in single-speaker and in overlap regions. Further, we evaluate the systems w.r.t. Word Error Rate (WER) of a downstream speech recognizer, both with oracle utterance boundaries and in continuous evaluation mode with the Optimal Reference Combination Word Error Rate (ORC WER) [26] (see Section V-A).

This paper is built on the previously published papers [21], [25], [27], and introduces the following novelties: (1) we provide a more extensive explanation of the Graph-PIT principles compared to [21], [27]; (2) we perform new experiments on reverberated meeting-like data showing that Graph-PIT works also for such challenging conditions; (3) we show that the SA-SDR loss presented in [25] allows for simpler and more efficient training with Graph-PIT from scratch compared to the Averaged SDR (A-SDR) loss used in [21]; and (4) we propose new signal-level evaluation metrics for meeting style scenarios, namely scale- and convolution-invariant SA-SDR (SA-SI-SDR and SA-CI-SDR), based on the Graph-PIT idea.

II. Problem Formulation

A. Continuous Speech Separation

A recording of a meeting conversation consists of multiple (U) utterances uttered by multiple (K) different speakers over the course of minutes or hours. The recording can contain overlapped speech, overlap-free speech and silence parts. One way to approach speech separation in such scenarios is with Continuous Speech Separation (CSS) [28]. CSS, in general, describes the task of separating a continuous mixture signal into multiple continuous overlap-free output channels.

There are different approaches to CSS that use different assignments of utterances to output channels in order to ensure they are overlap-free. One can produce one stream per speaker (in total K streams), which are by definition overlap-free [29]–[31] (speaker-exclusive assignment). One can also argue that the number of speakers $K^{(\text{sim})}$ that overlap at any point in time is often much smaller than the total number of speakers K in the meeting and arrange the utterances in $C \geq K^{(\text{sim})}$ overlap free streams. This assignment is independent of the actual total number of speakers $K \geq K^{(\text{sim})}$ (speaker-shared assignment).

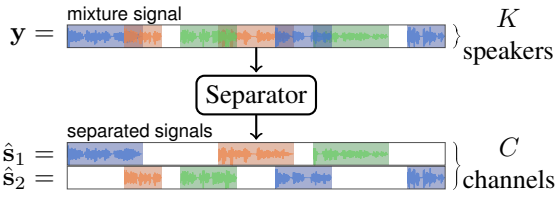

Fig. 1. Continuous Speech Separation with speaker-shared channel assignment: The overlapping input audio stream of arbitrary length is separated into multiple overlap-free channels. Different colors represent different speakers. Individual utterances from the input signal can be placed on any output as long as utterances do not overlap on any output.

We investigate the latter approach, speaker-shared assignment. An example for such a separation scheme is shown in Fig. 1, where each box represents an utterance and different colors represent different speakers. The separator aims to estimate C continuous output channels from the mixture $\mathbf{y}$, where no two utterances overlap in any output signal. The speech of one speaker can be distributed over multiple output channels, and each output channel can contain speech of multiple speakers.

CSS with speaker-shared assignment is conventionally performed with a sliding window approach, where sliding windows are separated independently and then stitched back together to obtain consistent output channels. This approach is termed "stitching", and described in detail in Section III-A. Another approach that we recently proposed is to train a neural network to directly produce continuous output channels in the CSS style, with a training method called "Graph-PIT", a natural extension of the commonly used uPIT for speaker-shared assignment [21]. We describe Graph-PIT in detail in Section IV.

B. Signal model

We assume single-microphone recordings and model the mixture signal $\mathbf{y}$ as a sum of U utterance signals $\tilde{\mathbf{s}}_u$ uttered by K speakers, and padded to the length T of the recording $\mathbf{s}_u = [0, ..., 0, \tilde{\mathbf{s}}_u, 0, ..., 0]^\mathsf{T} \in \mathbb{R}^T$. The mixture $\mathbf{y} \in \mathbb{R}^T$ is the sum of these utterance signals and a noise signal $\mathbf{n} \in \mathbb{R}^T$:

$$\mathbf{y} = \sum_{u=1}^{U} \mathbf{s}_u + \mathbf{n}. \tag{1}$$

To keep the equations simple, we here assume all signals to be anechoic. The choice of the actual reference signals for training and evaluation in the reverbant cases is described in Sections VII and VIII.

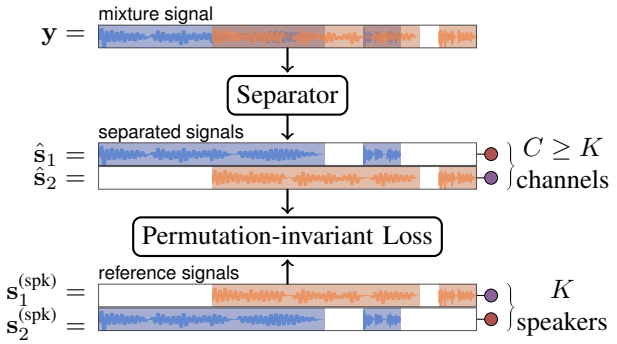

Fig. 2. Utterance-level separation with Utterance-level Permutation Invariant Training. The number of speakers that can be separated is limited by the number of output channels of the separator. Each output channel contains speech of exactly one speaker.

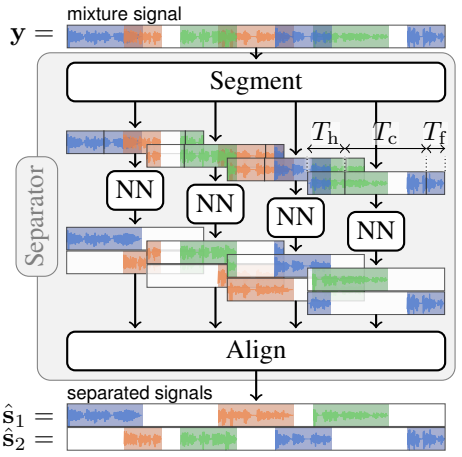

Fig. 3. The stitching approach to Continuous Speech Separation. The input mixure signal is segmented into overlapping segments, each segment is separated by a neural network, and the separated signals are aligned to obtain continuous output channels.

III. Conventional: Continuous Speech Separation with Stitching and Segment-level Separation

The conventional approach to CSS uses a speaker-exclusive NN-based separator with a sliding window stitching scheme. We first explain the Utterance-level Permutation Invariant Training (uPIT) [14] training scheme in Section III-A and elaborate on how it is applied to CSS in Section III-B.

A. Utterance-level Permutation Invariant Training (uPIT)

The traditional uPIT was designed for separation of short recordings with a small number of speakers with speaker-exclusive assignment, i.e., by uniquely mapping each speaker to an output channel of a separation network. The number of speakers K must match the number of output channels C of the separation network ($K = C$).[2] This idea is displayed in Fig. 2. The permutation problem, which arises during training from the fact that the separation network can output the separated signals in an arbitrary order, is solved by finding the permutation of reference signals that best matches the output signals. Its original formulation minimizes the average over the losses of each individual output channel:

$$\mathcal{L}^{(\text{uPIT})} = \min_{\pi \in \mathcal{P}_C} \frac{1}{C} \sum_{c=1}^{C} \mathcal{L}(\mathbf{s}^{(\text{spk})}_{\pi(c)}, \hat{\mathbf{s}}_c), \tag{2}$$

where $\mathcal{L}$ is an arbitrary signal-level loss function, $\pi \in \mathcal{P}_C$ enumerates all permutations $\pi : \{1, ..., C\} \to \{1, ..., C\}$ of length C, and $\mathbf{s}^{(\text{spk})}_c$ is the sum of all utterance signals uttered by speaker c. $\hat{\mathbf{s}}_c$ is the c-th output channel of the separation network. Back-propagation is performed only for the permutation that minimizes Eq. (2). In Fig. 2, the permutation between output channels and reference signals is indicated with the colors of the pinheads on the right of the signals.

Let us write Eq. (2) in a more general formulation to simplify further derivations. Here, $\mathcal{L}$ accepts matrices of reference signals $\mathbf{S}^{(\text{spk})} = [\mathbf{s}^{(\text{spk})}_1, ..., \mathbf{s}^{(\text{spk})}_C] \in \mathbb{R}^{T \times C}$ and estimated signals $\hat{\mathbf{S}} = [\hat{\mathbf{s}}_1, ..., \hat{\mathbf{s}}_C] \in \mathbb{R}^{T \times C}$, and the permutation is written with a permutation matrix $\mathbf{P}^{(\text{u})} \in \mathbb{R}^{C \times C}$:

$$\mathcal{L}^{(\text{uPIT})} = \min_{\mathbf{P}^{(\text{u})} \in \mathcal{P}} \mathcal{L}(\mathbf{S}^{(\text{spk})}\mathbf{P}^{(\text{u})}, \hat{\mathbf{S}}). \tag{3}$$

The matrix $\mathbf{P}^{(\text{u})}$ contains exactly one 1 in each row (one-hot vector) and each column while all other entries are 0, so that $\mathbf{S}\mathbf{P}^{(\text{u})}$ permutes the columns of $\mathbf{S}$ to construct the channel reference signals $\mathbf{S}^{(\text{chn})} = [\mathbf{s}^{(\text{chn})}_1, ..., \mathbf{s}^{(\text{chn})}_C] = \mathbf{S}^{(\text{spk})}\mathbf{P}^{(\text{u})}$.[3]

The optimal permutation that minimizes Eq. (2) or Eq. (3) can often be computed efficiently with the Hungarian algorithm [32], [33] even for large numbers of speakers.

B. Segmentation and Segment Stitching

A model trained with uPIT can be applied to CSS with a sliding window approach. It is assumed that, if the windows are chosen small enough, the number of speakers $K^{(\text{seg})}$ in one window is small, so that separation with a speaker-exclusive technique ($C \geq K^{(\text{seg})}$) is possible within each windowed segment.

The stitching procedure is visualized in Fig. 3. The input mixture signal is cut into overlapping segments of equal size, where the segmentation process is characterized by three numbers: the history context T_h, the future context T_f and the current window size T_c. The segments have a length of $T_\text{h} + T_\text{c} + T_\text{f}$ with a total overlap of $T_\text{h} + T_\text{f}$ so that the current windows of adjacent segments do not overlap. It is assumed that the number of speakers $K^{(\text{seg})}$ present in one segment is not larger than the number of channels, i.e., $K^{(\text{seg})} \leq C$. Then, the trained separator is applied to each segment independently and returns the separated signals in an arbitrary order. This poses an inter-segment speaker-permutation problem. Neighboring segments thus have to be aligned to obtain continuous and consistent output signals.

[2] When $K < C$, silent dummy speakers are introduced (i.e., $\mathbf{s} = \mathbf{0}$) so that the number of estimated outputs matches the number of targets.

[3] The superscript $^{(\text{u})}$ stands for uPIT.

The signals can be aligned based on a similarity between the overlapping parts of adjacent segments (T_h and T_f) [3], based on speaker information extracted from the separated signals [34], or with an additional tracking network [35]. In this work, we use similarity-based stitching only.

When the assumption $K^{(\mathrm{seg})} \leq C$ is violated during inference, the behavior of the separator with speaker-exclusive channel assignment is unknown in that segment. This inevitably leads to a trade-off; with larger the segment size, more context is available to the system to potentially improve the separation performance, but this also increases the probability for the model to face input signals it cannot handle. This introduces a new hyperparameter to tune, the segment size (i.e., T_h, T_c and T_f), that highly depends on the style of data being processed. Often, very short segment sizes below 2.5 s are chosen, and to avoid alignment errors, the overlap of the segments is often set to values larger than 50 % [3], [8], [36], which introduces a computation overhead proportional to the overlap $(T_\mathrm{h} + T_\mathrm{f})/(T_\mathrm{h} + T_\mathrm{c} + T_\mathrm{f})$ between the segments.

IV. Proposed: Continuous Speech Separation with Graph-PIT

Graph-PIT [21] is a natural extension of uPIT for speaker-shared assignment. It relaxes the constraint of uPIT that the number of output channels must not be smaller than the total number of speakers, i.e., $C \geq K$, to the more natural assumption that the number of concurrently speaking speakers $K^{(\mathrm{sim})}$ never exceeds the number of output channels C of the system, i.e., $C \geq K^{(\mathrm{sim})}$.

Analogous to the permutation matrices from uPIT, we model the assignment of utterances to output channels with assignment matrices $\mathbf{P} = [\mathbf{p}_1, ..., \mathbf{p}_U]^\mathsf{T} \in \{0,1\}^{U \times C}$. Each row $\mathbf{p}_u \in \mathbb{R}^C$ in $\mathbf{P}$ is a one-hot vector that describes which output channel the utterance u is assigned to. Not that compared to uPIT, the columns of the matrix can contain multiple ones since several utterances can be assigned to the same channel. The signal matrices now have the shapes $\mathbf{S} = [\mathbf{s}_1, ..., \mathbf{s}_U] \in \mathbb{R}^{T \times U}$ and $\hat{\mathbf{S}} = [\hat{\mathbf{s}}_1, ..., \hat{\mathbf{s}}_C] \in \mathbb{R}^{T \times C}$. With the set of all valid assignment matrices $\mathcal{B}$ (Section IV-A), the loss function for Graph-PIT roughly resembles Eq. (3):

$$\mathcal{L}^{(\text{Graph-PIT})} = \min_{\mathbf{P} \in \mathcal{B}} \mathcal{L}(\mathbf{SP}, \hat{\mathbf{S}}). \tag{4}$$

The channel references signals $\mathbf{S}^{(\mathrm{chn})} = \mathbf{SP}$ for training here contain sums of utterance signals instead of a permutation of speaker signals.

A. Graph-PIT as a graph coloring problem

An assignment matrix $\mathbf{P} = [\mathbf{p}_1, ..., \mathbf{p}_U]^\mathsf{T}$ is only valid if it does not map two temporally overlapping utterances to the same output channel, so the constraint

$$\mathbf{p}_u \neq \mathbf{p}_v \text{ if } u \text{ overlaps with } v, \quad \forall u, v \in \{1, ..., U\} \tag{5}$$

must be true for all assignment matrices.

Eq. (5) is equivalent to a graph coloring problem of the unweighted and undirected overlap graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Its

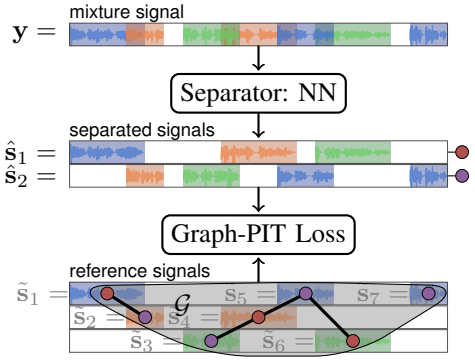

Fig. 4. The Graph-PIT approach to Continuous Speech Separation. The separation network receives the full input signal and directly produces CSS-style output streams. The overlap graph $\mathcal{G}$ is visualized in the bottom of the figure. The reference signals of the same speaker are drawn in one line for better visualization.

vertices $\mathcal{V}$ correspond to utterances and its edges $\mathcal{E}$ model overlaps between utterances:

$$\mathcal{V} = \{1, ..., U\}, \tag{6}$$

$$\mathcal{E} = \{\{v, u\} \text{ if } v \text{ overlaps with } u, \quad \forall v, u \in \mathcal{V}\}. \tag{7}$$

An edge is inserted between two vertices u and v if and only if the two utterances should not be mapped to the same output channel, i.e., if they temporally overlap. An example of an overlap graph is visualized in the bottom of Fig. 4. Finding a valid assignment of U utterances to C output channels is equivalent to finding a proper coloring of $\mathcal{G}$ with C colors.

To minimize Eq. (4), the one assignment (coloring) $\hat{\mathbf{P}}$ that minimizes the loss has to be found from $\mathcal{B}$. This can be done by first enumerating all possible assignments and selecting the best one. We will discuss efficient solutions for this minimization problem in Section IV-C. A possible coloring of the graph in Fig. 4 is indicated by the colors used to draw the nodes. The presented coloring will result in the shown mapping of utterances to output channels.

The notion of an overlap graph allows for more general definitions of overlap, where not only utterances that overlap temporally can be constrained onto different output channels, but any two utterances. As an example, utterances that are only separated by a short pause can be considered as overlapping, or a behavior similar to Group-level Permutation Invariant Training (Group-PIT) [35] is possible (see Section VI).

B. Issues with conventional loss functions for Graph-PIT training

In [21], the Graph-PIT networks were trained with a variant of the standard SDR, averaged over the output channels. Written with the standard SDR, the objective was to maximize

$$\text{A-SDR}(\mathbf{S}^{(\mathrm{chn})}, \hat{\mathbf{S}}) = \frac{10}{C} \sum_{c=1}^{C} \log_{10} \frac{\left\|\mathbf{s}_c^{(\mathrm{chn})}\right\|^2}{\left\|\mathbf{s}_c^{(\mathrm{chn})} - \hat{\mathbf{s}}_c\right\|^2}, \tag{8}$$

where $\mathbf{S}^{(\mathrm{chn})} = [\mathbf{s}_1^{(\mathrm{chn})}, ..., \mathbf{s}_C^{(\mathrm{chn})}]$ contains the reference signals $\mathbf{s}_c^{(\mathrm{chn})}$ constructed for the c-th output channel by the Graph-PIT loss, i.e., $\mathbf{S}^{(\mathrm{chn})} = \mathbf{S}\hat{\mathbf{P}}$. We call this commonly used definition

of the SDR the Averaged SDR (A-SDR). We found that the A-SDR has some unfavorable properties that were mitigated by pre-training with uPIT in [21], and in this paper by a modification of the loss function, called SA-SDR.

Firstly, the A-SDR is not defined for a completely silent output signal because $\lim_{\mathbf{s}\to\mathbf{0}} \log \frac{\|\mathbf{s}\|^2}{\|\mathbf{s}-\hat{\mathbf{s}}\|^2} = -\infty$. This never happens in typical scenarios where uPIT is applied ($K = C$), such as WSJ0-2mix [12]. However, it is likely to happen in meeting situations, especially when the model is trained on short segments randomly cut from this data.

Secondly, the A-SDR-based training leads the separator to focus its performance more on the already better separated output channel, especially when their energies differ heavily. This can be seen intuitively when looking at the value range when a reference signal is reconstructed (almost) perfectly: $\lim_{\hat{\mathbf{s}}\to\mathbf{s}} \log \frac{\|\mathbf{s}\|^2}{\|\mathbf{s}-\hat{\mathbf{s}}\|^2} = +\infty$. The loss of a well separated output channel dominates the full sum. An analysis of the gradients of A-SDR can be found in [25].

Because Graph-PIT has more freedom for placement of utterances on output channels, it favors putting many utterances on a single output channel and putting as much silence on the others when trained with A-SDR. For short training segments, which tend to contain fewer speakers and overlapping utterances, it is often possible to get a perfect reconstruction on one output channel by setting that channel to zero. The Graph-PIT models in [21] thus were taught to use both output channels by pre-training with uPIT.

Both of these problems can elegantly be mitigated by switching to the SA-SDR

$$\text{SA-SDR}(\mathbf{S}^{(\text{chn})}, \hat{\mathbf{S}}) = 10 \log_{10} \frac{\sum_{c=1}^{C} \left\|\mathbf{s}_c^{(\text{chn})}\right\|^2}{\sum_{c=1}^{C} \left\|\mathbf{s}_c^{(\text{chn})} - \hat{\mathbf{s}}_c\right\|^2} \tag{9}$$

that was proposed in [25] for training with fully overlapped speech and meeting scenarios. When we write the squared norm of a vector as a matrix operation, $\sum_c \|\mathbf{s}\|^2 = \text{Tr}(\mathbf{S}^\mathsf{T}\mathbf{S})$, it can be rewritten in a way compatible to Eqs. (3) and (4) as:

$$\text{SA-SDR}(\mathbf{S}^{(\text{chn})}, \hat{\mathbf{S}}) = 10 \log_{10} \frac{\text{Tr}(\mathbf{S}^{(\text{chn})\mathsf{T}}\mathbf{S}^{(\text{chn})})}{\text{Tr}((\mathbf{S}^{(\text{chn})} - \hat{\mathbf{S}})^\mathsf{T}(\mathbf{S}^{(\text{chn})} - \hat{\mathbf{S}}))}. \tag{10}$$

The SA-SDR is stable for a silent target signal, the gradients favor the worse output channel and its value is independent of the placement of utterances on output channels, given the separation quality is constant. A detailed discussion of the advantages of SA-SDR over A-SDR can be found in [25]. Using the SA-SDR loss allows easier training of a Graph-PIT model from scratch without pre-training with uPIT, which is shown in Section VII.

C. Efficient solutions for the Graph-PIT coloring problem

Naively searching for the best solution by brute-force testing all assignments in Eq. (4) is inefficient because of two reasons: The loss function has to be evaluated fully for each assignment and the graph coloring problem, i.e., finding one coloring for a graph given a maximum number of colors, is in general NP-hard [37]. The Graph-PIT problem is even more demanding; it is required to find not only an arbitrary coloring but the single best coloring $\hat{\mathbf{P}}$ that minimizes the loss in Eq. (4).

1) Decomposing the loss function: The first issue can be addressed by elegantly decomposing the loss function. If the Graph-PIT loss in Eq. (4) can be written as

$$\mathcal{L}^{(\text{Graph-PIT})} = f(\min_{\mathbf{P}\in\mathcal{B}} \text{Tr}(\mathbf{MP}), \mathbf{S}, \hat{\mathbf{S}}), \tag{11}$$

then we can find the best coloring based on sums of values of the matrix $\mathbf{M} \in \mathbb{R}^{C\times U}$. The matrix $\mathbf{M}$ is a score matrix computed from $\mathbf{S}$ and $\hat{\mathbf{S}}$, and $f : \mathbb{R}\times\mathbb{R}^{T\times U}\times\mathbb{R}^{T\times C} \to \mathbb{R}$ is a function strictly monotonically increasing in its first argument. The function f and the calculation of $\mathbf{S}$ depend on the actual loss function $\mathcal{L}$ used in Eq. (4). The number of times that the often expensive loss function has to be computed is reduced from $\mathcal{O}(U^C)$ to $\mathcal{O}(CU)$ to compute $\mathbf{M}$.

Two examples for decomposable loss functions are the Mean Squared Error (MSE) loss and the SA-SDR loss (see Section IV-B). The A-SDR is an example of a function that is not decomposable in this way. By using Eq. (10) as a loss function for Eq. (4), the Graph-PIT loss for SA-SDR can be rewritten as

$$\begin{aligned}\mathcal{L}^{(\text{SA-SDR})} &= \min_{\mathbf{P}\in\mathcal{B}} -10\log_{10} \frac{\text{Tr}(\mathbf{P}^\mathsf{T}\mathbf{S}^\mathsf{T}\mathbf{SP})}{\text{Tr}((\mathbf{PS}-\hat{\mathbf{S}})^\mathsf{T}(\mathbf{PS}-\hat{\mathbf{S}}))} &(12)\\ &= -10\log_{10} \frac{\text{Tr}(\mathbf{S}^\mathsf{T}\mathbf{S})}{\text{Tr}(\mathbf{S}^\mathsf{T}\mathbf{S}+\hat{\mathbf{S}}^\mathsf{T}\hat{\mathbf{S}}) + 2\min_{\mathbf{P}\in\mathcal{B}} \text{Tr}(-\hat{\mathbf{S}}^\mathsf{T}\mathbf{SP})}. &(13)\end{aligned}$$

The assignment matrix can be dropped in $\text{Tr}(\mathbf{P}^\mathsf{T}\mathbf{S}^\mathsf{T}\mathbf{SP}) = \text{Tr}(\mathbf{S}^\mathsf{T}\mathbf{S})$ because the valid assignments $\mathbf{P}$ only map non-overlapping targets to the same output channel. Comparing with Eq. (11), we can find the decomposition

$$\mathbf{M}^{(\text{SA-SDR})} = -\hat{\mathbf{S}}^\mathsf{T}\mathbf{S}, \tag{14}$$

$$f^{(\text{SA-SDR})}(x, \mathbf{S}, \hat{\mathbf{S}}) = -10\log_{10} \frac{\text{Tr}(\mathbf{S}^\mathsf{T}\mathbf{S})}{\text{Tr}(\mathbf{S}^\mathsf{T}\mathbf{S}+\hat{\mathbf{S}}^\mathsf{T}\hat{\mathbf{S}}) + 2x}. \tag{15}$$

We use this variant of the loss function with an added soft minimum in our experiments.

2) Efficient assignment algorithms: The decomposition in Eq. (11) additionally opens up new possibilities for efficiently finding the best coloring. The scores in $\mathbf{M}$ are additive so that scores for partial assignments can be combined by summing them, similar to what is done for uPIT with the Hungarian algorithm [32], [33]. Different algorithms to leverage this are presented in [27]. We here provide a more detailed explanation of the algorithm we employ in this paper, which is the fastest one among those presented in [27].[4] It takes advantage of the structure of the overlap graph $\mathcal{G}$ to solve the coloring problem with dynamic programming.

Let us assume the nodes of the graph $\mathcal{G}$ are sorted in temporal order, based on the position of $\tilde{\mathbf{s}}_u$ in $\mathbf{s}_u$ (see Section II-B). The graph $\mathcal{G}$ is a strongly chordal graph where the temporal ordering is an inverse strong perfect elimination ordering. A strongly chordal graph does not contain any induced cycles larger than 3, and the strong perfect elimination ordering

[4]The source code is available at https://github.com/fgnt/graph_pit

can be characterized as follows. Let $\mathcal{N}_u$ be the set of the utterance u and all utterances that have an earlier starting point than u. Let $\mathcal{N}_u^+ \subseteq \mathcal{N}_u$ be the set of utterances that also temporally overlap with u. The vertices in $\mathcal{N}_u^+$ form a clique, i.e., a complete sub-graph, of at most size $|\mathcal{N}_u^+| \leq C$ in $\mathcal{G}$, and the nodes in $\mathcal{N}_u^+$ appear continuous in the ordering, i.e., $!\exists\ v \in \mathcal{V} : \min(\mathcal{N}_u^+) < v < u \wedge \{u, v\} \notin \mathcal{E}$. Let us further define $\mathcal{B}_u$ and $\mathcal{B}_u^+$ as the sets of all proper colorings given $\mathcal{G}$ of $\mathcal{N}_u$ and $\mathcal{N}_u^+$, respectively. The number of proper colorings of $|\mathcal{N}_u^+|$ given $\mathcal{G}$ is $|\mathcal{B}_u^+| = \frac{C!}{(C-|\mathcal{N}_u^+|)!} \leq C!$ because it follows from the strong chordal property of $\mathcal{G}$ that the subgraph induced by $\mathcal{N}_u^+$ is complete.

Let $\mathbf{P} = [\mathbf{p}_0, ..., \mathbf{p}_U]^\mathsf{T} \in \mathbb{R}^{U \times C}$ be a proper coloring matrix (see Eq. (5)) of graph $\mathcal{G}$, whose elements $\mathbf{p}_u \in \mathbb{R}^C$ are one-hot vectors that each represent one of the C colors. Let $\mathbf{P}_u \in \mathcal{B}_u$ be a coloring matrix of $\mathcal{N}_u$. Let us define that a coloring $\mathbf{P}_1$ of nodes $\mathcal{N}_1$ and a coloring $\mathbf{P}_2$ of nodes $\mathcal{N}_2$ are compatible if they color common nodes with the same colors, i.e., if $\mathbf{p}_{1,v} = \mathbf{p}_{2,v}\ \forall v \in \mathcal{N}_1 \cap \mathcal{N}_2$. Then, we can denote the score of a partial coloring $\mathbf{P}_u \in \mathcal{B}_u$ by

$$c(\mathbf{P}_u) = \sum_{v=1}^{u} \mathbf{m}_v^\mathsf{T} \mathbf{p}_v = c(\mathbf{P}_{u-1}) + \mathbf{m}_u^\mathsf{T} \mathbf{p}_u, \tag{16}$$

where $\mathbf{P}_{u-1}$ is the one coloring in $\mathcal{B}_{u-1}$ that is compatible to $\mathbf{P}_u$ and $\mathbf{m}_u$ is the u-th column of the score matrix $\mathbf{M}$.

Because $\mathcal{G}$ is a strongly chordal graph with a maximum clique size of C, no vertex u has a neighbor that appears earlier in the ordering than the vertices in $\mathcal{N}_u^+$. To find the best coloring, it is thus enough to traverse the graph in temporal order and to only keep track of the best compatible coloring $\mathbf{P}_u \in \mathcal{B}_u$ for each of the colorings of $\mathcal{N}_u^+$. We denote as $\mathbf{P}_u^{\text{opt}}(\mathbf{P}_u^+) \in \mathcal{B}_u$ the coloring of $\mathcal{N}_u$ compatible to $\mathbf{P}_u^+ \in \mathcal{B}_u^+$ that minimizes the cost:

$$\mathbf{P}_u^{\text{opt}}(\mathbf{P}_u^+) = \underset{\mathbf{P}_u \in \mathcal{B}_u \text{ compatible to } \mathbf{P}_u^+}{\arg\min} c(\mathbf{P}_u), \tag{17}$$

and

$$c^{\text{opt}}(\mathbf{P}_u^+) = c(\mathbf{P}_u^{\text{opt}}(\mathbf{P}_u^+)). \tag{18}$$

The optimal colorings for $\mathcal{N}_u^+$ can be computed from the optimal colorings of $\mathcal{N}_{u-1}^+$, resulting in a Dynamic Programming (DP) approach:

$$c^{\text{opt}}(\mathbf{P}_u^+) = \mathbf{m}_u^\mathsf{T} \mathbf{p}_u^+ + \min_{\mathbf{P}_{u-1}^+ \in \mathcal{B}_{u-1}^+ \text{ compatible to } \mathbf{P}_u^+} c^{\text{opt}}(\mathbf{P}_{u-1}^+) \tag{19}$$

The overall cost is then $c^{\text{opt}}(\mathbf{P}^{(\text{opt})}) = \min_{\mathbf{P}_U^+ \in \mathcal{B}_U^+} c^{\text{opt}}(\mathbf{P}_U^+)$.

The complexity of this algorithm is mainly given by the number of comparisons required in the $\min$ operation in Eq. (19), in addition to the number of steps U to color the full graph. It turns out that, due to the compatibility constraint, each coloring in $\mathcal{B}_{u-1}^+$ only appears in one update, so that overall only $|\mathcal{B}_{u-1}^+| \leq C!$ comparisons are required in every iteration. An upper bound on the complexity can thus be given with $\mathcal{O}(UC!)$,[5] which is linear in the number of utterances U and factorial in the number of output channels C. C is often very small, e.g., $C \in \{2, 3\}$.

[5]This upper bound is more accurate and smaller than the bound shown in [27].

D. Comparison of Graph-PIT with uPIT

When the overlap graph is complete, i.e., every vertex from $\mathcal{V}$ is connected to every other vertex, or, every utterance overlaps with every other utterance in an example, uPIT is equal to Graph-PIT. This is the case for the first connected component (set of connected vertices in a graph) in Fig. 4.

uPIT binds an output channel to a speaker exclusively, so the number of speakers the model can handle in a segment is limited by the number of output channels C. Using Graph-PIT, on the other hand, allows multiple speakers on the same channel, so the number of speakers it can handle is unlimited. It only has the much less restrictive and more natural constraint that the number of speakers that are active at the same time must not be larger than the number of output channels. The weaker constraint allows us to increase the segment size up to the point where stitching is not required to process a whole meeting. This increases the context the model sees for separation, which can be beneficial for the separation performance. As a side effect, it reduces the computational cost at test time because we can reduce or even avoid the overlapped processing required for stitching. Consequently, low-latency real-time processing without stitching is possible with Graph-PIT, as demonstrated in [38] using a causal LSTM-based network architecture.

Besides, when the training data consists of meeting-like data, Graph-PIT can be trained with more training samples than uPIT. Indeed, we have to exclude training samples for uPIT that do not meet the constraint $K \leq C$, whereas Graph-PIT's weaker constraint allows it to be trained with all training samples where $K^{(\text{sim})} \leq C$. In other words, Graph-PIT allows an NN separator to model a whole meeting with an arbitrary number of speakers.

V. Evaluation Metrics for Separation in Meeting Scenarios

Development of CSS systems has been addressed in the preceeding sections. This section will introduce techniques for evaluation of separation systems for meeting scenarios.

Speech separation systems are conventionally evaluated with signal-level metrics, such as (SI-)SDR, STOI [39], PESQ [40], or a follow-up speech recognizer and the resulting WER. These metrics require a reference signal or transcription to compare to the estimated signal or transcription. In conventional speaker-dependent cases, the references can easily be obtained by finding the best matching permutation of speaker signals or transcriptions. On the other hand, for CSS with speaker-shared channel assignment, the separation system has more freedom for assigning utterances to outputs, so that a simple permutation is not sufficient to find good references.

Other works on CSS use utterance-wise evaluation schemes [3], [8], [21], or they use complex alignment algorithms to obtain a metric for a full recording [8], also called continuous evaluation.

Utterance-wise evaluation employs the oracle utterance boundaries, known from annotations of the evaluation data, to cut segments that match the reference utterance signals from the mixture or the estimated separated signals. A conventional metric can be applied on these segments, and finding a reference is reduced to a simple selection problem. By using the oracle information, this evaluation scheme explicitly ignores regions where no matching oracle reference is found, e.g., when speech of a single utterance is wrongly output on multiple output channels. In this work, we cut the segments from the separated signals for utterance-wise evaluation.

Continuous evaluation, on the other hand, does not use such oracle utterance boundary information. Instead, it directly evaluates the full separated continuous streams without artificial segmentation. In this case, many conventional metrics cannot be directly applied because they cannot handle multiple speakers on the same output channel, or it is unclear how the references are to be constructed. In the following subsections, we discuss issues of continuous evaluation of separation systems in terms of ASR and signal-level metrics, and propose a set of appropriate metrics for them, which we believe is very important to develop separation algorithms for meeting data. We will use the discussed metrics in our experiments.

A. Continuous Evaluation with Word Error Rate (WER)

Many works use the WER for evaluation, e.g., [3], [8], [10]. It is often argued that separation systems are the front-end for further applications and that the WER reflects separation quality with respect to these follow-up systems. An existing ASR system is applied to the separated signals to obtain transcriptions. These are compared to the reference transcriptions. The WER is then the word-level Levenshtein distance between the reference (r) and estimated (e) transcriptions divided by the number of words in the reference transcriptions.

ASR systems are often built to work in batch mode on individual sentences rather than streaming mode on arbitrarily long input [9], [11]. To apply them to continuous evaluation, the continuous streams thus first have to be segmented using a Voice Activity Detection (VAD), and transcriptions are only estimated for speech parts. It is not guaranteed, and in fact often not satisfied, that the segments produced by the VAD correspond to ground-truth utterances or even contain speech of a single speaker only. Thus, to compute a WER, the ground truth reference transcriptions have to be aligned with the estimated transcriptions.

One way to find such an alignment is the ORC WER [26]. It computes a WER over the complete output channels, i.e., transcriptions obtained by concatenating transcriptions of speech segments detected with a VAD. The reference transcription for a channel is chosen as the one that minimizes the WER among the U^C possible assignments (combinations) of reference utterance transcriptions to output channels, sorted by their temporal mid-points and concatenated.

Evaluation with ASR systems has the general drawback that the reported WER depends on the used speech recognizer, and the data and details used for its training. This makes WERs only hardly comparable across different works, especially when the evaluation is performed in different environments. This calls for other metrics in addition to the WER that solely rely on the signal quality, like the SDR, described in the next section.

B. Continuous Evaluation with Signal-level SDR-based metrics

SDR-based metrics are often used to evaluate source separation systems, e.g., [12], [15], [18], [21]. Its standard definition in Eq. (8) cannot be applied to continuous evaluation directly because they assume speaker-exclusive channels, and CSS system scatter speakers across output channels. We propose to use SA-SDR (Eq. (9)) with Graph-PIT assignment also for evaluation because it can be computed for systems with speaker-shared channel assignment. Furthermore, the value of the standard SDR changes with the assignment of utterances to output channels even if the estimated utterance signals are identical. The SA-SDR produces consistent values in this case. In the extreme case of two utterances and two output channels, for example, A-SDR grows to ∞ when both utterances are placed on the same output channel, while for SA-SDR all possible assignments produce the same (finite) value. The SA-SDR, as we defined it for training in Eq. (9), can only be used as an evaluation metric in clean anechoic environments and separation systems that do not alter the scaling of the signals. We therefore present two variations of the SA-SDR in the remainder of this section.

1) SA-SI-SDR: For evaluation in anechoic environments, often a scale-invariant version of the SDR (SI-SDR)[6] is used. This is motivated by the fact that the signal quality should be judged independently of the volume of the output signal. The SI-SDR is defined for a single pair of estimation and reference as [15], [16], [18]

$$\text{SI-SDR} = 10 \log_{10} \frac{\|\alpha \mathbf{s}\|^2}{\|\alpha \mathbf{s} - \hat{\mathbf{s}}\|^2}, \tag{20}$$

where the scaling factor $\alpha = \arg\min_{\tilde{\alpha}} \|\tilde{\alpha}\mathbf{s} - \hat{\mathbf{s}}\|^2 = \frac{\mathbf{s}^\mathsf{T}\hat{\mathbf{s}}}{\mathbf{s}^\mathsf{T}\mathbf{s}}$ scales the reference signal to match the estimation. Conventionally, the permutation problem has to be solved during evaluation to find the references.

For continuous evaluation, we propose to incorporate the scaling correction into the SA-SDR, and obtain the Source-Aggregated Scale-Invariant SDR (SA-SI-SDR). We assume that the scaling factor is constant for one utterance, but can change between output channels and utterances, e.g., due to speaker movement or limited memory of the separation system. We thus have to estimate the scaling factor $\alpha_{uc} = (\mathbf{s}_u^\mathsf{T}\hat{\mathbf{s}}_c)/(\mathbf{s}_u^\mathsf{T}\mathbf{s}_u)$ for each pair of utterance u and output channel c. To simplify the equations, we now write out $\mathrm{Tr}(\mathbf{P}^\mathsf{T}\mathbf{S}^\mathsf{T}\mathbf{S}\mathbf{P}) = \sum_c \left\|\sum_u p_{uc}\mathbf{s}_u\right\|^2$ and define the SA-SI-SDR with Graph-PIT assignment as (compare Eq. (9) and Eq. (4))

$$\text{SA-SI-SDR} = \max_{\mathbf{P}\in\mathcal{B}} 10 \log_{10} \frac{\sum_c \left\|\sum_u p_{uc}\alpha_{uc}\mathbf{s}_u\right\|^2}{\sum_c \left\|\sum_u p_{uc}\alpha_{uc}\mathbf{s}_u - \hat{\mathbf{s}}_c\right\|^2}. \tag{21}$$

[6]Sometimes called Scale-invariant Source-to-Noise Ratio (SI-SNR)

A decomposition as Eq. (11) can be found (see Appendix A) as

$$\text{SA-SI-SDR} = -10\log_{10}\left(\frac{\mathrm{Tr}(\hat{\mathbf{S}}^{\mathsf{T}}\hat{\mathbf{S}})}{\max\limits_{\mathbf{P}\in\mathcal{B}}\mathrm{Tr}(\mathbf{MP})} - 1\right), \quad (22)$$

$$[\mathbf{M}]_{uc} = \alpha_{uc}\mathbf{s}_u^{\mathsf{T}}\hat{\mathbf{s}}_c, \quad (23)$$

so that the efficient assignment algorithms from Section IV-C can be used to find the best matching assignment under the overlap constraints.

Our generalization re-scales the reference signal while the conventional SI-SDR definition [22] applies the scaling correction to the estimated signal. Our approach has the drawback that the scaling of an utterance influences its contribution to the overall SDR value, e.g., setting $\hat{\mathbf{s}}_c = \mathbf{0}$ removes the c-th utterance from the evaluation. The SA-SI-SDR can thus not be used for training. This effect is, however, negligible when the separation system produces a roughly constant scaling, which is here achieved by the training loss function. Still, it should not be used as the sole evaluation metric.

2) SA-CI-SDR: For the reverberant case, the SI-SDR is extended to a Convolution Invariant SDR (CI-SDR) to allow not only scaling differences, but any distortions caused by a linear filter, similar to the SDR from the BSSEval [23], [24] toolbox. Instead of a scalar scaling factor α, now a convolution with a filter $\mathbf{a} \in \mathbb{R}^L$ of length L is used in the CI-SDR, for a single pair of estimation and reference:

$$\text{CI-SDR} = 10\log_{10}\frac{\|\mathbf{a} * \mathbf{s}\|^2}{\|\mathbf{a} * \mathbf{s} - \hat{\mathbf{s}}\|^2}, \quad (24)$$

where $*$ denotes discrete convolution and the filter is $\mathbf{a} = \arg\min_{\tilde{\mathbf{a}}} \|\tilde{\mathbf{a}} * \mathbf{s} - \hat{\mathbf{s}}\|^2$. For continuous evaluation, we propose the Source-Aggregated Convolution Invariant SDR (SA-CI-SDR):

$$\text{SA-CI-SDR} = \max_{\mathbf{P}\in\mathcal{B}} 10\log_{10}\frac{\sum_c \left\|\sum_u p_{uc}\mathbf{a}_{uc} * \mathbf{s}_u\right\|^2}{\sum_c \left\|\sum_u p_{uc}\mathbf{a}_{uc} * \mathbf{s}_u - \hat{\mathbf{s}}_c\right\|^2} \quad (25)$$

with the decomposition (see Appendix A):

$$\text{SA-CI-SDR} = -10\log_{10}\left(\frac{\mathrm{Tr}(\hat{\mathbf{S}}^{\mathsf{T}}\hat{\mathbf{S}})}{\max\limits_{\mathbf{P}\in\mathcal{B}}\mathrm{Tr}(\mathbf{MP})} - 1\right), \quad (26)$$

$$[\mathbf{M}]_{uc} = (\mathbf{a}_{uc} * \mathbf{s}_u)^{\mathsf{T}}\hat{\mathbf{s}}_c. \quad (27)$$

VI. Related Work

[35] proposes a simple way to implement a sub-set of Graph-PIT for $C = 2$, called Group-level Permutation Invariant Training (Group-PIT). They only train on groups of consecutive overlapping utterances, called "Utterance Groups", defined by a preceding and following silence of all speakers. Such an utterance group has the property that for $C = 2$ all proper colorings of the overlap graph only differ by a permutation across the output channels. Thus, only $C!$ different colorings exist so that training with the conventional uPIT is possible, where then every reference signal can contain speech of more than one speaker. An Utterance Group is equivalent to a connected component in the overlap graph $\mathcal{G}$ (see Section IV-A), and Graph-PIT and Group-PIT are identical for connected graphs with $C = 2$. When more utterance groups are present, Group-PIT cannot be applied without modification, and Graph-PIT gives more flexible results. An example for such a graph is the second connected component of $\mathcal{G}$ in Fig. 4.

The basic idea that multiple speakers can be put onto the same output stream was also used in [26] to realize a multi-speaker speech recognizer that directly outputs transcriptions in a way similar to CSS. The authors of [26] eliminated the assignment problem during training by arranging the target transcriptions on a small number of outputs in a way they term "overlap-based target arrangement": The transcription of the first utterance is put on the first output channel. Following transcriptions are concatenated on the same output channel if there is no overlap between them. The channel is switched when two consecutive utterances overlap. This assignment is one possible solution to the Graph-PIT problem. For evaluation, they introduce the ORC WER, which is used for evaluation in this work as well.

VII. Evaluation: Anechoic Meeting-like Data

We first perform a proof-of-concept experiment with anechoic meeting-like mixtures. With these experiments, we also show that SA-SDR-based metrics can be used for evaluation in meeting scenarios. We compare the performance of Graph-PIT-based and uPIT-based separators to show the advantage of Graph-PIT.

A. Data

Meeting-like data is generated based on utterance recordings taken from the WSJ database [41] with the goal to have five to eight speakers in each meeting, a total length of roughly 120 s per meeting and an overlap ratio between 20 % and 40 %. For each meeting, first the number of speakers is sampled uniformly between five and eight and the target amount of overlap is sampled uniformly between 20 % to 40 %. Utterances are sampled uniformly and start times are sampled so that the target overlap ratio is roughly fulfilled and all speakers are active for roughly the same amount of time. We generated in total 36 hours of training data based on the *train_si284* subset of WSJ, and each 1 hour of development and test data based on the *cv_dev93* and *test_eval92* subsets, respectively. We use a sample rate of 8 kHz for all experiments.

B. Model Architecture and Training Procedure

We use a Dual-Path Recurrent Neural Network (DPRNN)-TasNet separation architecture [18] with the same parameters as [18] except for the number of blocks. We use three stacked DPRNN blocks instead of six to reduce the memory footprint, each of which use two Bidirectional Long-Short-Term Networks (BLSTMs) with 128 hidden units. The number of filters in the encoder and decoder are set to 64 with a chunk length of 100 frames for the segmentation. This model achieves an SDR gain of 15.0 dB on the WSJ0-2mix benchmark database [12]. Our separator has $C = 2$ output channels.

We explicitly chose an RNN-based architecture because it has an infinitely long receptive field while the amount of information being kept is limited. The infinitely long receptive field is required to solve the assignment problem for arbitrary input data, so an architectur with a limited receptive field, such as a Convolutional Neural Network (CNN), is not well suited. The information that has to be carried between steps, though, is rather small. Only the assignment of at maximum the past few utterances has to be kept in memory.

We train the separation models with the SA-tSDR loss proposed in [25], which is the SA-SDR loss with an added soft maximum at $\text{SDR}_{\text{max}} = 30$ [42]. Training is performed with the clean utterance signals as reference. The training data are segments of the meeting-like WSJ data with a length T_{Tr} between 2 s to 64 s. If training with uPIT, parts of the training data have to be discarded that do not adhere to the number-of-speaker constraint of uPIT, i.e., in case of $C < K^{(\text{seg})}$. The amount of data that is discarded increases with larger training segment size T_{Tr}, but stays below 80 % up to $T_{\text{Tr}} = 16$ s and is indicated by the red line in the bottom of Fig. 5. The batch size is chosen so that each batch contains a total of 64 s of speech data for all configurations. We train all models with an Adam optimizer with a learning rate of 0.001 for 600000 iterations and use the checkpoint with the lowest validation loss for evaluation.

To evaluate models on full meetings of 120 s length we use a stitching approach with similarity-based alignment in the time domain [21]. We keep $T_{\text{h}} = T_{\text{f}} = 1$ s constant and only vary T_{c} between 0.4 and 14 s. In addition, to show that Graph-PIT can allow segment-less, i.e., stitcher-less, separation of meeting data by modeling a whole meeting, we also run experiments with no stitcher.

C. Evaluation Metrics

We perform both utterance-wise and continuous evaluation. For utterance-wise evaluation, we use the utterance-wise WER, i.e. the conventional way to calculate the WER [8]. For continuous evaluation, we use the ORC WER and the three SA-SDR-based metrics presented in Section V. As an ASR backend we use a factorized time-delayed neural network (TDNN-F) from the Kaldi framework [9] with the same configuration presented in [43]. It was trained on noisy WSJ data and achieves a WER of 6.8 % on clean WSJ. We use an energy-based VAD so that the backend only sees speech signals. Voice activity is detected by computing the energy within a sliding window of length 100 ms and shift of 1 sample. If the energy is larger than 20 % of the average energy of the full signal, the sample is considered to contain speech. The end result is smoothed by a moving window of length 300 ms, where a window is considered active if at least 100 ms of speech is found in that window.

To compute the SA-CI-SDR, we use a filter with length $L = 512$, which is equal to the default value in the BSSEval toolbox [23]. The reference signals for all SDR-based evaluation metrics are always the clean source signals.

D. Results

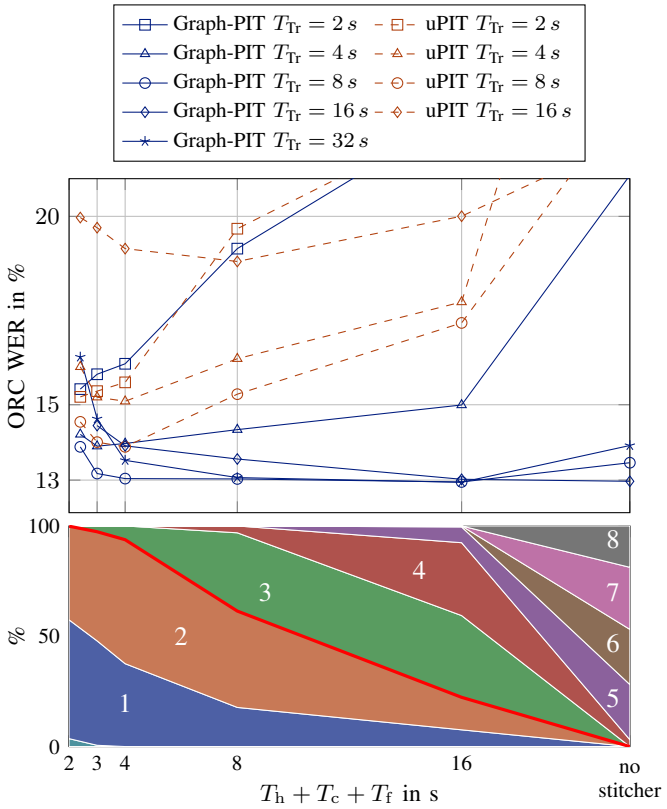

Fig. 5. *Top:* WER plotted over the segment size for stitching. Lower is better. *Bottom:* Distribution of the number of speakers in a segment. The red line represents the amount of segments that fulfill the constraints of uPIT, i.e., $K^{(\text{seg})} \leq C$.

1) Influence of stitcher segment size: The influence of the stitcher segment size on the separation performance is shown in Fig. 5. The top part of Fig. 5 displays the ORC WER over the stitcher segment size for uPIT and Graph-PIT models, trained with an SA-tSDR loss with different training segment sizes T_{Tr}. The bottom part shows the distribution of the numbers of speakers in the test segments obtained with each stitcher segment size. The red line indicates the number of segments with $K^{(\text{seg})} > C = 2$ where models trained with uPIT have an unknown behavior. It can be seen that the performance of all uPIT models decreases with increasing numbers of speakers in the stitcher segments, and none of the models generalizes to the case where no stitcher is used. The average performance, however, increases with larger T_{Tr} up to the point where too many training examples have to be discarded, which indicates that larger contexts can be beneficial for separation.

The Graph-PIT models show a different behavior. For small stitcher segment sizes, where the number of examples with $K^{(\text{sim})} > C$ is small, they perform similarly to uPIT. The same is true for a small training segment size of $T_{\text{Tr}} = 2$ s, where the behavior of uPIT and Graph-PIT is almost identical. On the other hand, with larger training segment sizes, the Graph-PIT models outperform uPIT for larger stitcher segment sizes, where the best stitcher segment size is close to the training segment size. The same trend, i.e., Graph-PIT outperforming uPIT, was also observed in experiments where uPIT was

TABLE I
SEPARATION PERFORMANCE OF MODELS TRAINED WITH UPIT COMPARED TO GRAPH-PIT, EVALUATED WITH DIFFERENT METRICS ON ANECHOIC MEETING-LIKE MIXTURES. BEST NUMBERS PER COLUMN ARE SET IN **BOLD**, AND BEST NUMBERS WITHOUT STITCHING ARE UNDERLINED. ↑: HIGHER IS BETTER, ↓: LOWER IS BETTER

Training Scheme	T_{Tr} [s]	Stitcher $T_{\text{h}} + T_{\text{c}} + T_{\text{f}}$ [s]	SA-SDR [dB]↑	SA-SI-SDR [dB]↑	SA-CI-SDR [dB]↑	utterance-wise WER [%]↓	ORC WER [%]↓
no separation	—	—	0.0	0.0	0.0	38.1	77.7
uPIT	8	—	7.2	8.2	8.3	22.6	23.2
		1+2+1	16.7	16.9	17.2	14.9	13.9
	16	—	7.4	8.3	8.5	22.3	22.1
		1+14+1	8.0	9.0	9.2	20.0	20.0
Graph-PIT	16	—	**18.2**	18.3	18.6	14.0	13.0
		1+14+1	**18.2**	**18.4**	**18.7**	14.2	13.0
	32	—	17.7	17.8	18.0	14.3	13.9
		1+14+1	18.1	18.2	18.5	**13.6**	**12.9**

trained on data containing only two speakers so that no data had to be discarded. With large enough training segment sizes, models trained with Graph-PIT generalize to processing without a stitcher. The computational overhead introduced by the overlapping windows in the stitcher is removed, resulting in a reduction of required computational effort, which only depends on the ratio of $T_{\text{f}} + T_{\text{h}}$ to $T_{\text{f}} + T_{\text{c}} + T_{\text{h}}$, by 50 %.

2) Metrics Comparison: In Table I we show the performance of a few selected experiments with a few more metrics for deeper analysis. Let us first look at the proposed SDR metrics. We can see that SA-CI-SDR > SA-SI-SDR > SA-SDR for all experiments, which corresponds to the motivation that SA-CI-SDR allows more convolutional distortions than SA-SI-SDR and SA-SI-SDR allows scaling errors which SA-SDR does not. The SDR-based metrics roughly correlate with the WER, so we can conclude that they are reasonable metrics to evaluate separation in meeting scenarios.

Looking at the WERs, we can see that utterance-wise WER and ORC WER are often close, and sometimes the ORC WER is lower than the utterance-wise WER. It seems odd at first that the utterance-wise WER that uses oracle information is not always the lowest. But the utterance-wise WER treats every utterance independently so that some errors are seen twice (in two cut utterances) while the ORC WER only evaluates the full stream once. The ORC WER can additionally align transcriptions across utterance boundaries, which may sometimes produce lower and slightly over-optimistic WERs.

The signal-level metrics show a similar behavior to what we observed in Fig. 5. The uPIT models do not generalize to processing without a stitcher while models trained with Graph-PIT do. Larger training segment sizes improve the overall performance of the models.

3) SA-SDR loss: To show the benefits of the SA-SDR loss compared to the previously used A-SDR loss, we compare their behavior in Table II. For training with A-SDR we use the A-ε-tSDR which was also used in [21]. The results show that the SA-tSDR models surpass the performance of the A-tSDR models, especially for short segment sizes where often no overlaps are present in a training segment. For longer segment sizes, where the segments contain more overlapped utterances, the allowed outputs become more balanced so that the negative effect of the A-tSDR, focusing the already better separated output channel, becomes less severe. However, as discussed in Section IV-C, the SA-tSDR loss has advantages also for longer segments as it allows using the efficient Graph-PIT assignment algorithms, which A-tSDR does not. This speedup becomes larger with larger numbers of utterances, i.e., larger segment sizes. We observed a speedup of roughly 10 % during training when switching from A-tSDR to SA-tSDR for $T_{\text{Tr}} = 16\,\text{s}$. Additionally, the models trained with SA-tSDR consequently show a better generalization to processing without stitching. The gap between the performance with stitching and without stitching is always smaller for SA-tSDR than it is for A-tSDR.

TABLE II
COMPARISON OF SA-SDR WITH A-SDR AS A TRAINING LOSS FUNCTION WITH GRAPH-PIT

Loss	T_{Tr} [s]	Stitcher $T_{\text{h}} + T_{\text{c}} + T_{\text{f}}$ [s]	SA-CI-SDR [dB]	ORC WER [%]
A-tSDR	4	—	12.1	25.9
		1+2+1	17.7	14.9
	8	—	17.9	14.9
		1+6+1	18.1	13.3
	16	—	18.1	13.9
		1+14+1	18.1	13.3
SA-tSDR	4	—	13.8	21.1
		1+6+1	17.4	14.3
	8	—	17.8	13.5
		1+6+1	17.8	13.0
	16	—	18.6	13.0
		1+14+1	18.7	13.0

4) Training from scratch compared to Pre-Training: The performance of models pre-trained with uPIT and fine-tuned with Graph-PIT (labelled with "uPIT+Graph-PIT") is compared with models trained with only uPIT or Graph-PIT from scratch in Table III, using the SA-tSDR loss. Only models trained with $T_{\text{Tr}} = 8\,\text{s}$ are shown because the "uPIT+Graph-PIT" model achieved its best performance with this training segment size. The models were pre-trained with uPIT for 600000 iterations and fine-tuned for another 600000 iterations, resulting in twice the training time of the individual models trained from scratch. Only the best configuration is shown for

TABLE III
COMPARISON OF DIFFERENT TRAINING SCHEMES FOR $T_{\mathrm{Tr}} = 8\,\mathrm{s}$

Training Scheme	Stitcher $T_{\mathrm{h}} + T_{\mathrm{c}} + T_{\mathrm{f}}$ [s]	SA-CI-SDR [dB]	ORC WER [%]
uPIT	—	8.3	23.2
	1+2+1	17.2	13.9
uPIT + Graph-PIT	—	11.2	20.5
	1+6+1	17.9	14.6
Graph-PIT	—	17.8	13.5
	1+6+1	17.8	13.0

TABLE IV
SEPARATION PERFORMANCE FOR REVERBERATED MEETING-LIKE DATA.

Training Scheme	Stitcher $T_{\mathrm{h}} + T_{\mathrm{c}} + T_{\mathrm{f}}$ [s]	SA-CI-SDR [dB]	ORC WER [%]
no separation	—	–0.4	74.9
uPIT	—	5.3	35.6
	1+1+1	9.5	28.9
Graph-PIT	—	8.2	29.9
	1+6+1	9.8	27.7

each training scheme. The performance of the "uPIT+Graph-PIT" models lies between the separation performance of the uPIT-only and Graph-PIT-only models for separation without a stitcher. The models seem to have learned the uPIT behavior in the beginning of the training and were not able to learn the Graph-PIT generalization sufficiently well during the fine-tuning process. Training with Graph-PIT directly from scratch yields the best performance and training is much simpler and faster than when pre-training with uPIT.

VIII. EVALUATION: REVERBERATED MEETING-LIKE DATA

Next, we evaluate the Graph-PIT training scheme on more challenging reverberated data.

A. Data

We use the same meeting-like data as before (Section VII-A) with added reverberation. Speaker positions were assumed to be constant over one meeting and were sampled randomly without constraints on minimum angular distances. The room impulse responses were simulated with the image method [44], with room dimensions between 7.6 m by 5.6 m and 8.4 m by 6.4 m and a sound decay time (T60) of 200 ms to 500 ms. During training, we randomly discarded 90 % of the single-speaker training segments because it turned out that training on reverberated data requires larger amounts of overlap.

B. Model Architecture and Training Procedure

For reverberant data, we switch to a simpler Short-Time Fourier Transform (STFT)-masking-based model architecture that has shown to be more effective in these scenarios [45]. We use a BLSTM with three layers with 600 units in each direction, followed by a linear layer to map back to the STFT size. We train the model with the same time-domain thresholded SA-tSDR loss as the clean model. As reference signals for the loss computation, we convolve the clean reference signals with the first 50 ms of the room impulse response, to eliminate the barely audible but hard to reconstruct reverberation tail.

C. Results

The separation performance of the reverberant models is tabulated in Table IV. We do not apply SA-SDR or SA-SI-SDR in this case because they are not designed for reverberation, and their behavior heavily depends on the choice of the reference signals in such cases [43]. We show the performance of the best model with $T_{\mathrm{Tr}} = 4\,\mathrm{s}$, for the sake of simplicity. While the overall performance is degraded due to the more challing data, the same tendencies are visible as in the anechoic case. The Graph-PIT model generalizes to segment-less processing while the uPIT models do not. Training with Graph-PIT becomes more challenging in the reverberant case because the reverberation tail blurs the utterance boundaries which makes it harder for the model to detect them. The overall best performance can still be achieved with a Graph-PIT model.

Training models on reverberated data appeared to be more challenging than training models for anechoic data, for both uPIT and Graph-PIT, as it was recently also observed in other works [45]. We found that more overlap is required when training for reverberated data than for anechoic data. Further tuning the model architecture may improve the separation quality for all models.

IX. CONCLUSIONS

In this paper, we introduced a generalization of Utterance-level Permutation Invariant Training (uPIT) called Graph-PIT as an alternative to stitching-based CSS with uPIT. We showed that models trained with Graph-PIT generalize to segment-less processing of long meeting-like data, and improve performance over conventional uPIT-based separation systems in both anechoic and reverberant conditions. The SA-SDR was introduced as a loss function for training Graph-PIT models from scratch, and its effectiveness was shown in terms of improvements in separation performance and training speed. We introduced new signal-level metrics, SA-SDR, SA-SI-SDR and SA-CI-SDR, for signal-level evaluation of separation systems in meeting-like scenarios. We provide source code for the Graph-PIT training criterion[7] and hope that our findings inspire further research in the separation of meeting-like conversations.

[7] https://github.com/fgnt/graph_pit

APPENDIX
DERIVATIONS OF SA-SDR-BASED EVALUATION METRICS

SA-SI-SDR

The SA-SI-SDR is defined as (compare Eq. (21))

$$\text{SA-SI-SDR} = \max_{\mathbf{P}\in\mathcal{B}} 10\log_{10} \frac{\sum_c \left\|\sum_u p_{uc}\alpha_{uc}\mathbf{s}_u\right\|^2}{\sum_c \left\|\sum_u p_{uc}\alpha_{uc}\mathbf{s}_u - \hat{\mathbf{s}}_c\right\|^2}. \tag{28}$$

with $\alpha = \arg\min_{\tilde{\alpha}} \left\|\tilde{\alpha}\mathbf{s} - \hat{\mathbf{s}}\right\|^2 = \frac{\mathbf{s}^\mathsf{T}\hat{\mathbf{s}}}{\mathbf{s}^\mathsf{T}\mathbf{s}}$.

The numerator can be simplified by writing out the squared norm

$$\sum_c \left\|\sum_u p_{uc}\alpha_{uc}\mathbf{s}_u\right\|^2 = \sum_{c,u,u'} p_{uc}p_{u'c}\alpha_{uc}\alpha_{u'c}\mathbf{s}_u^\mathsf{T}\mathbf{s}_{u'} \tag{29}$$

Using the knowledge that $\mathbf{s}_u^\mathsf{T}\mathbf{s}_{u'} = 0$ if the utterances u and u' do not overlap and that two overlapping utterances can never be mapped to the same output channel (due to the graph constraints), so $p_{uc}p_{u'c} = 0$ for overlapping utterances, we can see that $p_{uc}p_{u'c}\alpha_{uc}\alpha_{u'c}\mathbf{s}_u^\mathsf{T}\mathbf{s}_{u'} = 0$ if $u \neq u'$. From this we have

$$\sum_c \left\|\sum_u p_{uc}\alpha_{uc}\mathbf{s}_u\right\|^2 = \sum_{c,u} p_{uc}\alpha_{uc}^2\mathbf{s}_u^\mathsf{T}\mathbf{s}_u. \tag{30}$$

Writing out the squared norm also for the denominator, plugging in $\alpha_{uc}\mathbf{s}_u^\mathsf{T}\mathbf{s}_u = \mathbf{s}_u^\mathsf{T}\hat{\mathbf{s}}_c$ and using $\log(a/b) = -\log(b/a)$ gives

$$\text{SA-SI-SDR} = -10\min_{\mathbf{P}\in\mathcal{B}} \log_{10}\left(\frac{\sum_c \hat{\mathbf{s}}_c^\mathsf{T}\hat{\mathbf{s}}_c}{\sum_c\sum_u p_{uc}\alpha_{uc}\mathbf{s}_u^\mathsf{T}\hat{\mathbf{s}}_c} - 1\right). \tag{31}$$

Using $\mathbf{M} = \left[\alpha_{uc}\mathbf{s}_u^\mathsf{T}\hat{\mathbf{s}}_c\right]_{uc}$ as the score matrix, we can find

$$\text{SA-SI-SDR} = -10\log_{10}\left(\frac{\operatorname{Tr}(\hat{\mathbf{S}}^\mathsf{T}\hat{\mathbf{S}})}{\max_{\mathbf{P}\in\mathcal{B}}\operatorname{Tr}(\mathbf{MP})} - 1\right). \tag{32}$$

SA-CI-SDR

The SA-CI-SDR is defined as (compare Eq. (25))

$$\text{SA-CI-SDR} = \max_{\mathbf{P}\in\mathcal{B}} 10\log_{10} \frac{\sum_c \left\|\sum_u p_{uc}\mathbf{a}_{uc} * \mathbf{s}_u\right\|^2}{\sum_c \left\|\sum_u p_{uc}\mathbf{a}_{uc} * \mathbf{s}_u - \hat{\mathbf{s}}_c\right\|^2}, \tag{33}$$

with $\mathbf{a}_{uc} = \arg\min_{\tilde{\mathbf{a}}} \left\|\tilde{\mathbf{a}} * \mathbf{s}_u - \hat{\mathbf{s}}_c\right\|^2$. The signal parts in $\hat{\mathbf{s}}_c$ that lie outside of the boundaries of utterance u are constant and thus have no effect on the optimization.

We can write $\mathbf{a} * \mathbf{s}$ as $\tilde{\mathbf{S}}\mathbf{a}$ with $\tilde{\mathbf{S}} = \text{toeplitz}(s)$, where $\text{toeplitz}(\mathbf{s})$ creates a padded toeplitz matrix from $\mathbf{s}$. The minimization then becomes

$$\mathbf{a} = \arg\min_{\mathbf{a}} \left\|\tilde{\mathbf{S}}\mathbf{a} - \hat{\mathbf{s}}\right\|^2 \tag{34}$$

$$\frac{\mathrm{d}\left\|\tilde{\mathbf{S}}\mathbf{a} - \hat{\mathbf{s}}\right\|^2}{\mathrm{d}\mathbf{a}} = 2\tilde{\mathbf{S}}^\mathsf{T}\tilde{\mathbf{S}}\mathbf{a} - 2\tilde{\mathbf{S}}^\mathsf{T}\hat{\mathbf{s}} \stackrel{!}{=} 0 \tag{35}$$

$$\Rightarrow \tilde{\mathbf{S}}^\mathsf{T}\tilde{\mathbf{S}}\mathbf{a} = \tilde{\mathbf{S}}^\mathsf{T}\hat{\mathbf{s}}. \tag{36}$$

Plugging Eq. (36) into Eq. (25) yields a solution very similar to the SA-SI-SDR in Eq. (31), analogous to the SA-SI-SDR:

$$\text{SA-CI-SDR} = \max_{\mathbf{P}\in\mathcal{B}} 10\log_{10} \frac{\sum_c \left\|\sum_u p_{uc}\tilde{\mathbf{S}}_u\mathbf{a}_{uc}\right\|^2}{\sum_c \left\|\sum_u p_{uc}\tilde{\mathbf{S}}_u\mathbf{a}_{uc} - \hat{\mathbf{s}}_c\right\|^2} \tag{37}$$

$$= -10\log_{10}\left(\frac{\sum_c \hat{\mathbf{s}}_c^\mathsf{T}\hat{\mathbf{s}}c}{\max_{\mathbf{P}\in\mathcal{B}}\sum_{uc} p_{uc}\mathbf{a}_{uc}^\mathsf{T}\tilde{\mathbf{S}}_u^\mathsf{T}\hat{\mathbf{s}}_c} - 1\right) \tag{38}$$

$$= -10\log_{10}\left(\frac{\sum_c \hat{\mathbf{s}}_c^\mathsf{T}\hat{\mathbf{s}}c}{\max_{\mathbf{P}\in\mathcal{B}}\sum_{uc} p_{uc}(\mathbf{a}_{uc} * \mathbf{s}_u)^\mathsf{T}\hat{\mathbf{s}}_c} - 1\right) \tag{39}$$

A decomposition can be found by using Eq. (32) with $\mathbf{M} = [(\mathbf{a}_{uc} * \mathbf{s}_u)^\mathsf{T}\hat{\mathbf{s}}_c]_{uc}$.

ACKNOWLEDGMENT

Computational resources were provided by the Paderborn Center for Parallel Computing.

REFERENCES

[1] Ö. Çetin and E. Shriberg, "Analysis of Overlaps in Meetings by Dialog Factors, Hot Spots, Speakers, and Collection Site: Insights for Automatic Speech Recognition," in *Speakers, and Collection Site: Insights for Automatic Speech Recognition', Proc. Interspeech 2006*, 2006, pp. 293–296.

[2] E. Shriberg, A. Stolcke, and D. Baron, "Observations on Overlap: Findings and Implications for Automatic Processing of Multi-Party Conversation," p. 4, 2001.

[3] T. Yoshioka, H. Erdogan, Z. Chen, X. Xiao, and F. Alleva, "Recognizing Overlapped Speech in Meetings: A Multichannel Separation Approach Using Neural Networks," in *Interspeech 2018*. ISCA, Sep. 2018, pp. 3038–3042.

[4] J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal, G. Lathoud, M. Lincoln, A. Lisowska, I. McCowan, W. Post, D. Reidsma, and P. Wellner, "The AMI Meeting Corpus: A Pre-announcement," in *Machine Learning for Multimodal Interaction*, ser. Lecture Notes in Computer Science, S. Renals and S. Bengio, Eds. Berlin, Heidelberg: Springer, 2006, pp. 28–39.

[5] J. Barker, S. Watanabe, E. Vincent, and J. Trmal, "The Fifth 'CHiME' Speech Separation and Recognition Challenge: Dataset, Task and Baselines," in *Interspeech 2018*. ISCA, Sep. 2018, pp. 1561–1565.

[6] N. Kanda, C. Boeddeker, J. Heitkaemper, Y. Fujita, S. Horiguchi, K. Nagamatsu, and R. Haeb-Umbach, "Guided Source Separation Meets a Strong ASR Backend: Hitachi/Paderborn University Joint Investigation for Dinner Party ASR," in *Interspeech 2019*. ISCA, Sep. 2019, pp. 1248–1252.

[7] S. Watanabe, M. Mandel, J. Barker, E. Vincent, A. Arora, X. Chang, S. Khudanpur, V. Manohar, D. Povey, D. Raj, D. Snyder, A. Shanmugam Subramanian, J. Trmal, B. Ben Yair, C. Boeddeker, Z. Ni, S. Horiguchi, N. Kanda, T. Yoshioka, and N. Ryant, "CHiME-6 Challenge: Tackling multispeaker speech recognition for unsegmented recordings," in *6th International Workshop on Speech Processing in Everyday Environments (CHiME 2020)*, 2020.

[8] Z. Chen, T. Yoshioka, L. Lu, T. Zhou, Z. Meng, Y. Luo, J. Wu, X. Xiao, and J. Li, "Continuous Speech Separation: Dataset and Analysis," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020, pp. 7284–7288.

[9] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi Speech Recognition Toolkit," https://infoscience.epfl.ch/record/192584, 2011.

[10] D. Raj, P. Denisov, Z. Chen, H. Erdogan, Z. Huang, M. He, S. Watanabe, J. Du, T. Yoshioka, Y. Luo, N. Kanda, J. Li, S. Wisdom, and J. R. Hershey, "Integration of Speech Separation, Diarization, and Recognition for Multi-Speaker Meetings: System Description, Comparison, and Analysis," in *2021 IEEE Spoken Language Technology Workshop (SLT)*, Jan. 2021, pp. 897–904.

[11] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "ESPnet: End-to-End Speech Processing Toolkit," in *Interspeech*. ISCA, Sep. 2018, pp. 2207–2211.

[12] J. R. Hershey, Z. Chen, J. Le Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 31–35.

[13] Y. Isik, J. L. Roux, Z. Chen, S. Watanabe, and J. R. Hershey, "Single-Channel Multi-Speaker Separation Using Deep Clustering," in *Interspeech 2016*, Sep. 2016, pp. 545–549.

[14] M. Kolbæk, D. Yu, Z.-H. Tan, and J. Jensen, "Multitalker Speech Separation With Utterance-Level Permutation Invariant Training of Deep Recurrent Neural Networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 10, pp. 1901–1913, Oct. 2017.

[15] Y. Luo and N. Mesgarani, "TaSNet: Time-Domain Audio Separation Network for Real-Time, Single-Channel Speech Separation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2018, pp. 696–700.

[16] ——, "Conv-TasNet: Surpassing Ideal Time–Frequency Magnitude Masking for Speech Separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 8, pp. 1256–1266, Aug. 2019.

[17] D. Yu, M. Kolbæk, Z. Tan, and J. Jensen, "Permutation invariant training of deep models for speaker-independent multi-talker speech separation," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 2017, pp. 241–245.

[18] Y. Luo, Z. Chen, and T. Yoshioka, "Dual-Path RNN: Efficient Long Sequence Modeling for Time-Domain Single-Channel Speech Separation," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2020, pp. 46–50.

[19] J. Chen, Q. Mao, and D. Liu, "Dual-Path Transformer Network: Direct Context-Aware Modeling for End-to-End Monaural Speech Separation," in *Interspeech 2020*. ISCA, Oct. 2020, pp. 2642–2646.

[20] S. Dovrat, E. Nachmani, and L. Wolf, "Many-Speakers Single Channel Speech Separation with Optimal Permutation Training," *arXiv:2104.08955 [cs, eess]*, Apr. 2021.

[21] T. von Neumann, K. Kinoshita, C. Boeddeker, M. Delcroix, and R. Haeb-Umbach, "Graph-PIT: Generalized permutation invariant training for continuous separation of arbitrary numbers of speakers," in *Interspeech*. ISCA, 2021.

[22] J. Le Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "SDR–half-baked or well done?" in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 626–630.

[23] C. Févotte, R. Gribonval, and E. Vincent, "BSS_EVAL Toolbox User Guide – Revision 2.0," Jan. 2005.

[24] E. Vincent, R. Gribonval, and C. Fevotte, "Performance measurement in blind audio source separation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 4, pp. 1462–1469, Jul. 2006.

[25] T. von Neumann, K. Kinoshita, C. Boeddeker, M. Delcroix, and R. Haeb-Umbach, "SA-SDR: A Novel Loss Function for Separation of Meeting Style Data," in *ICASSP 2022*, 2022.

[26] I. Sklyar, A. Piunova, X. Zheng, and Y. Liu, "Multi-turn RNN-T for streaming recognition of multi-party speech," *arXiv:2112.10200 [cs, eess]*, Dec. 2021.

[27] T. von Neumann, C. Boeddeker, K. Kinoshita, M. Delcroix, and R. Haeb-Umbach, "Speeding Up Permutation Invariant Training for Source Separation," in *Speech Communication; 14th ITG-Symposium*, Sep. 2020.

[28] S. Chen, Y. Wu, Z. Chen, J. Wu, J. Li, T. Yoshioka, C. Wang, S. Liu, and M. Zhou, "Continuous Speech Separation with Conformer," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Jun. 2021, pp. 5749–5753.

[29] T. Hori, S. Araki, T. Yoshioka, M. Fujimoto, S. Watanabe, T. Oba, A. Ogawa, K. Otsuka, D. Mikami, K. Kinoshita, T. Nakatani, A. Nakamura, and J. Yamato, "Low-Latency Real-Time Meeting Recognition and Understanding Using Distant Microphones and Omni-Directional Camera," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 2, pp. 499–513, Feb. 2012.

[30] T. v 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 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2019, pp. 91–95.

[31] N. Kanda, S. Horiguchi, Y. Fujita, Y. Xue, K. Nagamatsu, and S. Watanabe, "Simultaneous Speech Recognition and Speaker Diarization for Monaural Dialogue Recordings with Target-Speaker Acoustic Models," in *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, Dec. 2019, pp. 31–38.

[32] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," in *50 Years of Integer Programming 1958-2008*.

[33] J. Munkres, "Algorithms for the assignment and transportation problems," *J. Society for Industrial and Applied Mathematics*, vol. 5, no. 1, pp. 32–38, 1957.

[34] H. Taherian and D. Wang, "Time-Domain Loss Modulation Based on Overlap Ratio for Monaural Conversational Speaker Separation," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Jun. 2021, pp. 5744–5748.

[35] W. Zhang, Z. Chen, N. Kanda, S. Liu, J. Li, S. E. Eskimez, T. Yoshioka, X. Xiao, Z. Meng, Y. Qian, and F. Wei, "Separating Long-Form Speech with Group-Wise Permutation Invariant Training," *arXiv:2110.14142 [cs, eess]*, Nov. 2021.

[36] C. Han, Y. Luo, C. Li, T. Zhou, K. Kinoshita, S. Watanabe, M. Delcroix, H. Erdogan, J. Hershey, N. Mesgarani, and Z. Chen, "Continuous Speech Separation Using Speaker Inventory for Long Multi-talker Recording," *ArXiv*, 2020.

[37] B. Bollobás, "Colouring," in *Graph Theory: An Introductory Course*, ser. Graduate Texts in Mathematics, B. Bollobás, Ed. New York, NY: Springer, 1979, pp. 88–102.

[38] C. Li, L. Yang, W. Wang, and Y. Qian, "Skim: Skipping Memory Lstm for Low-Latency Real-Time Continuous Speech Separation," in *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 681–685.

[39] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "A short-time objective intelligibility measure for time-frequency weighted noisy speech," in *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, Mar. 2010, pp. 4214–4217.

[40] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual Evaluation Of Speech Quality (pesq) – A New Method for Speech Quality Assessment of Telephone Networks and Codecs," in *J. of the Audio Engineering Society*, 2001, pp. 749–752.

[41] J. Garofalo, D. Graff, D. Paul, and D. Pallett, "Csr-i (wsj0) complete," *Linguistic Data Consortium*, 2007.

[42] S. Wisdom, E. Tzinis, H. Erdogan, R. J. Weiss, K. Wilson, and J. R. Hershey, "Unsupervised Speech Separation Using Mixtures of Mixtures," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 3846–3857.

[43] L. Drude, J. Heitkaemper, C. Boeddeker, and R. Haeb-Umbach, "SMS-WSJ: Database, performance measures, and baseline recipe for multi-channel source separation and recognition," *arXiv:1910.13934 [cs, eess]*, Oct. 2019.

[44] J. B. Allen and D. A. Berkley, "Image method for efficiently simulating small-room acoustics," *The Journal of the Acoustical Society of America*, vol. 65, no. 4, pp. 943–950, Apr. 1979.

[45] T. Cord-Landwehr, C. Boeddeker, T. Von Neumann, C. Zorilă, R. Doddipatla, and R. Haeb-Umbach, "Monaural source separation: From anechoic to reverberant environments," in *2022 International Workshop on Acoustic Signal Enhancement (IWAENC)*, 2022, pp. 1–5.

Thilo von Neumann is currently pursuing his Ph.D. degree in electrical engineering under the supervision of Prof. Reinhold Haeb-Umbach at Paderborn University. He received his bachelor's and master's degree in Computer Engineering and got awarded with the prize for the best master's degree at the department of electrical engineering and information technology in the academic year 2021. His research interests are mainly focused on meeting analysis including speech separation, diarization and speech recognition. During his studies, he completed two research internships at NTT Communication Science Labs in Kyoto, Japan, in 2018 and 2019.

Keisuke Kinoshita is a research scientist at Google Japan. He received the M. Eng. degree and Ph.D. degree from Sophia University in Tokyo in 2003 and 2010. Before joining Google, he was a research scientist at NTT Communication Science Laboratories, Kyoto, Japan, from 2003 to 2022, where he worked on the project of this paper. Throughout his career, he has been engaged in fundamental research on speech, audio, and music signal processing, including 1ch/multi-channel speech enhancement, speaker diarization, robust speech recognition, and distributed microphone array processing. He is an author or a co-author of more than 100 papers presented at peer-reviewed international conferences, more than 20 journal papers, and many patents. He has been serving as an associate editor of IEEE Transactions on Audio, Speech and Language Processing (TASLP) since 2021, as a member of IEEE Audio and Acoustic Signal Processing Technical Committee (AASP-TC) since 2019, an area chair for ICASSP since 2019, and served as an editor and an organizing committee member of several domestic and international journals/conferences.

Christoph Boeddeker (Student Member, IEEE) received the bachelor's and master's degrees in electrical engineering from Paderborn University, where he is currently working toward the Ph.D. degree, under the supervision of Reinhold Haeb-Umbach. His research interests range from multichannel speech separation, beamforming, and dereverberation to automatic speech recognition on meetings with a focus on combining statistical models and neural networks. In 2017 and 2022 he pursued a research internship with Microsoft Research, Redmond, USA and MERL, Cambridge, USA, respectively.

Marc Delcroix is a Distinguished researcher at NTT Communication Science Laboratories, NTT Corporation, Japan. He received an M.Eng. from the Free University of Brussels, Brussels, Belgium, and the Ecole Centrale Paris, Paris, France, in 2003 and a Ph.D. from the Graduate School of Information Science and Technology, Hokkaido University in 2007. His research interests cover various aspects of speech and audio processing, such as target speech and sound extraction, speech enhancement, robust speech recognition, model adaptation, and speaker diarization. He is a member of the IEEE Signal Processing Society Speech and Language Technical Committee, and an associate editor of the IEEE/ACM Transactions ASLP. He served as a member of the organizing committee of the REVERB challenge 2014, the ASRU 2017, and SLT 2022.

Reinhold Haeb-Umbach is a professor of Communications Engineering at Paderborn University, Germany. He holds a Dr.-Ing. degree from RWTH Aachen University, and he has more than 30 years of experience in speech research, which he acquired both in an industrial and academic environment. His main research interests are in speech enhancement, acoustic beamforming and source separation, as well as automatic speech recognition and unsupervised learning from speech and audio. He has more than 300 scientific publications, and his students have received several best student paper awards. From 2015 to 2020 he was member of the IEEE Signal Processing Society Speech and Language Technical Committee, and he is currently (2021 to 2023) member of the Audio and Acoustic Signal Processing Technical Committee. He is a fellow of the International Speech Communication Association (ISCA) and a fellow of the IEEE.